УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
сентябрь–октябрь 2020 Том 20 № 5 ISSN 2226-1494 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
September–October 2020 Vol. 20 No 5 ISSN 2226-1494 http://ntv.ifmo.ru/en/
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК УНИВЕРСИТЕТ ИТМО ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

УДК 004.75 doi: 10.17586/2226-1494-2020-20-5-739-746

МОДЕЛЬ РАСПРЕДЕЛЕННОЙ СВЕРТОЧНОЙ НЕЙРОННОЙ СЕТИ НА КЛАСТЕРЕ КОМПЬЮТЕРОВ С ОГРАНИЧЕННЫМИ ВЫЧИСЛИТЕЛЬНЫМИ РЕСУРСАМИ

Р.Р. Хайдарова[a], Д.И. Муромцев[a], М.В. Лапаев[b], В.Д. Фищенко[a]

[a] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[b] ООО «ТПП Лаб», Санкт-Петербург, 195009, Российская Федерация
Адрес для переписки: mignolowa@gmail.com

Информация о статье
Поступила в редакцию 28.05.20, принята к печати 10.09.20
Язык статьи — русский

Ссылка для цитирования: Хайдарова Р.Р., Муромцев Д.И., Лапаев М.В., Фищенко В.Д. Модель распределенной сверточной нейронной сети на кластере компьютеров с ограниченными вычислительными ресурсами // Научно-технический вестник информационных технологий, механики и оптики. 2020. Т. 20. № 5. С. 739–746. doi: 10.17586/2226-1494-2020-20-5-739-746

Аннотация

Предмет исследования. Выполнено исследование в области распределенного глубокого обучения — сверточной нейронной сети на кластере компьютеров с ограниченными вычислительными ресурсами. Рассмотрена общая архитектура и особенности сверточной нейронной сети, а также проанализированы существующие ограничения, которые возникают при их развертывании, основанные на таких архитектурах как LeNet, AlexNet, VGG-16/VGG-19. Развертывание распределенной сверточной нейронной сети на устройствах с ограниченными вычислительными ресурсами все еще является одной из трудоемких задач, где не имеется готовых решений. **Метод.** Предложен метод разделения карт признаков сверточной нейронной сети на блоки, где каждый блок соответствует определенной задаче. Представлена общая схема распределения задач для пересекающихся данных. **Основные результаты.** Разработана модель распределенной сверточной нейронной сети для кластера компьютеров с ограниченными вычислительными ресурсами, а также модель планировщика задач для пересекающихся данных при выполнении вычислений преимущественно на сверточном слое, так как данный слой является одним из самых ресурсоемких, содержащих большое количество гиперпараметров. **Практическая значимость.** Разработка распределенной системы на базе предложенных методов позволит развернуть распределенное машинное обучение, в частности, сверточную нейронную сеть на кластере из 24 одноплатных компьютеров RockPro64, где возможно выполнение различных задач в области машинного зрения, обработки естественного языка, прогнозирования и др., что может быть использовано в граничных вычислениях (edge computing).

Ключевые слова
сверточная нейронная сеть, кластер, планировщик задач, распределенная сверточная нейронная сеть, глубокое обучение, одноплатные компьютеры

doi: 10.17586/2226-1494-2020-20-5-739-746

DISTRIBUTED CONVOLUTIONAL NEURAL NETWORK MODEL ON RESOURCE-CONSTRAINED CLUSTER

R.R. Khaydarova[a], D.I. Mouromtsev[a], M.V. Lapaev[b], V.D. Fishchenko[a]

[a] ITMO University, Saint Petersburg, 197101, Russian Federation
[b] TPP Lab LTD, Saint Petersburg, 195009, Russian Federation
Corresponding author: mignolowa@gmail.com

Article info
Received 28.05.20, accepted 10.09.20
Article in Russian

For citation: Khaydarova R.R., Mouromtsev D.I., Lapaev M.V., Fishchenko V.D. Distributed convolutional neural network model on resource-constrained cluster. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2020, vol. 20, no. 5, pp. 739–746 (in Russian). doi: 10.17586/2226-1494-2020-20-5-739-746

Abstract

Subject of Research. The paper presents the distributed deep learning particularly convolutional neural network problem for resource-constrained devices. General architecture of convolutional neural network and its specificity is considered, existing constraints that appear while the deployment process on such architectures as LeNet, AlexNet, VGG-16/VGG-19 are analyzed. Deployment of convolutional neural network for resource-constrained devices is still a challenging task,

as there are no existing and widely-used solutions. **Method.** The method for distribution of feature maps into smaller pieces is proposed, where each part is a determined problem. General distribution model for overlapped tasks within the scheduler is presented. **Main Results.** Distributed convolutional neural network model for a resource-constrained cluster and a scheduler for overlapped tasks is developed while carrying out computations mostly on a convolutional layer since this layer is one of the most resource-intensive, containing a large number of hyperparameters. **Practical Relevance.** Development of distributed convolutional neural network based on proposed methods provides the deployment of the convolutional neural network on a cluster that consists of 24 RockPro64 single board computers performing tasks related to machine vision, natural language processing, and prediction and is applicable in edge computing.

Keywords
convolutional neural network, cluster, scheduler, distributed convolutional neural network, deep learning, single board computers

Введение

В настоящее время технологии интернета вещей (IoT, Internet of Things) все больше внедряются в нашу жизнь, а их применение становится рутинной работой. Технологии IoT заняли такие области как умный дом, умный город, здравоохранение, оптимизация промышленности и др. [1]. Большое количество данных, которые генерируются устройствами IoT, необходимо обработать с помощью средств машинного обучения. Некоторые из применений средств машинного обучения предполагают анализ данных в режиме псевдо-реального времени, где скорость обработки данных играет существенную роль, например, анализ видеозаписей, контроль работы беспилотных летательных аппаратов и т. д. В связи с ростом количества подключенных устройств IoT и повышенными требованиями к скорости обработки данных, использование технологий облачных вычислений не всегда является целесообразным. Для решения подобных задач разработана концепция «граничные вычисления» (edge computing), призванная в первую очередь приблизить обработку и хранение данных к устройствам, их генерирующим и использующим [2]. Примером оконечных устройств, функционирование которых строится в рамках концепции «граничных вычислений», являются беспилотные автомобили.

В последние годы глубокие сверточные нейронные сети (Convolutional Neural Network, CHC) являются одними из популярных нейронных сетей, которые применяются в задачах машинного зрения [3–5], обработки естественного языка [6, 7], прогнозирования [8, 9]. CHC показывают высокую точность при решении задач, однако они являются одними из самых ресурсоемких нейронных сетей, требующих значительных вычислительных ресурсов. С развитием IoT растет потребность во внедрении данной сети в устройства с ограниченными вычислительными ресурсами, с ограниченной памятью и вычислительной мощностью. В настоящее время нет готовых решений для развертывания распределенной CHC на устройствах с ограниченными вычислительными ресурсами, однако исследования ведутся [10–12]. Целью работы является разработка модели распределенной CHC для кластера компьютеров с ограниченными вычислительными ресурсами.

Сверточные нейронные сети

Существует множество архитектур CHC, которые год от года пополняются. Архитектура CHC впервые применена в LeNet [13] и была представлена пятью слоями. Выбор архитектуры в основном зависит от типа задачи и количества данных: чем больше данных, тем более глубокие сети можно спроектировать для повышения точности выполнения задач. В рамках данной работы предполагается представление общей концепции для архитектур LeNet [13], AlexNet [14], VGG-16/VGG-19 [13]. Основными слоями данных архитектур являются сверточные (convolutional), субдискретизирующие (pooling), нормализующие и полносвязные слои (fully connected), а также различные функции активации (ReLU, LeakyReLU, ELU и т. д.). Более поздние версии архитектур, например, GoogLeNet [13], не рассматриваются, так как изменяется логика построения архитектур: появляются Inception-модули, и данные типы архитектур требуют дальнейшего, более детального анализа.

Рассмотрим CHC с математической точки зрения. Пусть I — входные данные, представленные в виде матрицы размерностью $X \times Y$. Данные могут иметь различный формат: текстовые данные, временные ряды или изображения. В случае изображения матрица имеет размерность $X \times Y \times C$, где C — количество каналов (например, $C = 3$, если это RGB-изображения). Пусть $g \in \lambda = \{\omega_1, \ldots, \omega_\lambda\}$ выходные данные CHC, которые соответствуют различным задачам (например, распознавание лиц, прогнозирование и т. д.), где λ — множество классов или характеристик внутри одной задачи. Тогда CHC можно представить следующим образом:

$$g = \Phi(I),$$

$$\Phi(I) = \varphi_{\theta_l}^{(l)}(I_{l-1}),$$

$$I_i = \varphi_{\theta_j}^{(j)}(I_{j-1}),$$

$$I_1 = \varphi_{\theta_1}^{(1)}(I),$$

где выходные данные CHC (g) представлены через функцию (Φ), $j = 2, \ldots, l - 1$; l — количество слоев в CHC; $\varphi_{\theta_l}^{(l)}$ — параметризированная функция по θ_i для i-го слоя CHC ($i = 1,\ldots, l$); θ — вспомогательная переменная, называемая параметром; I_i — выходные данные (карты признаков), получаемые со слоя i, которые подаются в качестве входных данных для слоя $i + 1$; $\varphi_{\theta_i}^{(i)}$ — различные слои CHC, например, сверточный слой, субдискретизирующий и т. д. Общая архитектура CHC представлена на рис. 1, *а*.

В работе [15] уменьшена вычислительная сложность CHC, если опустить некоторые вычислительные задачи в иерархической цепочке CHC, а также уменьшено количество занимаемой памяти в битах.

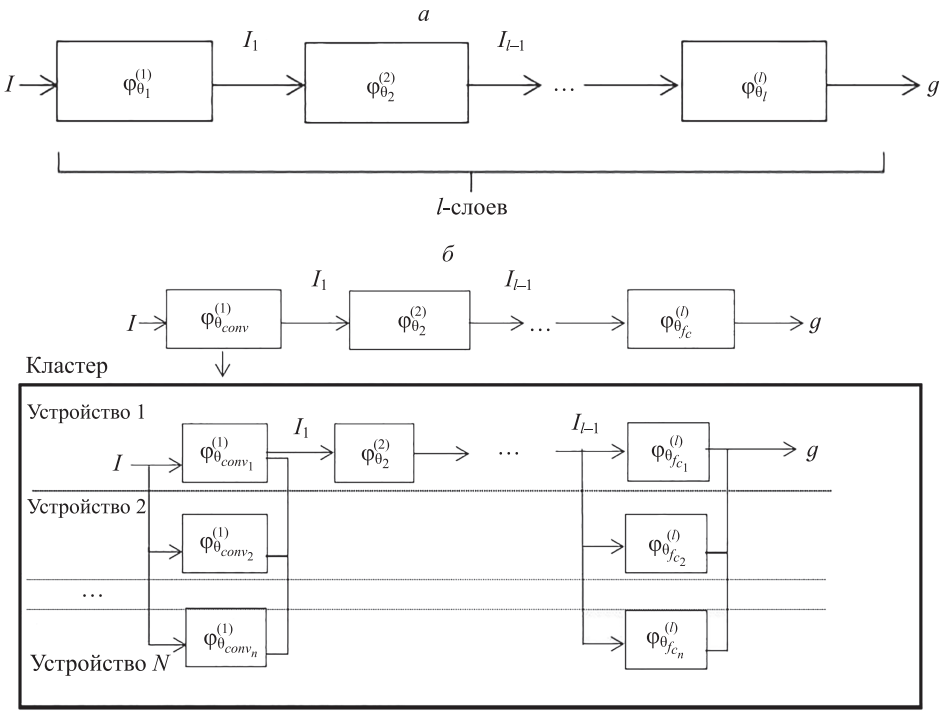

Рис. 1. Общая архитектура (*а*) и модель для кластера компьютеров с ограниченными вычислительными ресурсами (*б*) сверточной нейронной сети

Однако в данном случае архитектура СНС остается неизменной, но так как сверточные и полносвязные слои являются наиболее ресурсоемкими, поскольку содержат большое количество гиперпараметров, их необходимо распределить, разделив на более мелкие задачи между устройствами с ограниченными вычислительными ресурсами. Модель СНС для кластера компьютеров с ограниченными вычислительными ресурсами представлена на рис. 1, *б*, где сверточный слой ($\varphi_{\theta_{conv}}^{(1)}$) и полносвязный ($\varphi_{\theta_{fc}}^{(1)}$) представлены в виде ряда параллельных задач $\varphi_{\theta_{conv}} = \{\varphi_{conv_1}, \varphi_{conv_2}, \ldots, \varphi_{conv_n}\}$, $\varphi_{\theta_{fc}} = \{\varphi_{fc_1}, \varphi_{fc_2}, \ldots, \varphi_{fc_n}\}$, где n — количество подзадач в рамках одного слоя.

Существуют следующие ограничения на кластере компьютеров с ограниченными вычислительными ресурсами.

Пусть переменная $\alpha_{i,l}$ принимает следующие значения:

$$\alpha_{i,l} = \begin{cases} 1, \text{ если } i\text{-й узел кластера выполняет операцию для } l\text{-го слоя СНС} \\ 0, \text{ в противном случае.} \end{cases}$$

1. Ограничение на количество слоев (подзадач):

$$\sum_{l=1}^{L} \alpha_{i,l} \le L,$$

где L — количество слоев, которые выполняются на одном узле кластера (например, субдискретизирующие слои, функции активации). В случае разделения сверточного и полносвязного слоев на более мелкие задачи, L ограничивается количеством подзадач в рамках одного слоя.

2. Ограничение на количество памяти, необходимой для хранения параметров:

$$\sum_{l=1}^{L} \alpha_{i,l} M_l \le M_{RAM,i},$$

где $M_{RAM,i}$ — оперативная память (RAM, Random Access Memory) i-го узла кластера (в основном используется 80 % памяти); M — объем памяти, необходимый для хранения параметров на каждом слое СНС, который вычисляется:

$$M = \sum_{l=1}^{L} N_l b = \sum_{l=1}^{L} k_l k_{l-1} u_l v_l b,$$

где N — количество гиперпараметров; k — количество фильтров, $k_{l-1} = c$ (для первого слоя); u — ширина фильтра; v — высота фильтра; b — количество бит, которые необходимы для хранения весов (в данном случае $b = 4$, так как числа с плавающей запятой занимают 4 Б).

3. Ограничение на вычислительную нагрузку:

$$\sum_{l=1}^{L} \alpha_{i,l} C_l \le \overline{C}_i,$$

где C_l — вычислительная нагрузка на l-ом слое; $\overline{C}_i$ — максимальная вычислительная загрузка на i-ом узле кластера, которая зависит от типа задач. В данном случае вычислительная нагрузка для СНС — это количество вычислительных операций, необходимых для выполнения задач на сверточном и полносвязном слоях (такие слои как пулинговый, нормализационный могут быть опущены [15]):

$$C = C_{conv} + C_{fc},$$

$$C_{conv} = \sum_{i=1}^{N_{conv}} k_{i-1} k_i u_i v_i p_i q_i,$$

$$C_{fc} = \sum_{i=1}^{N_{fc}} n_{i-1} n_i,$$

где N_{conv} — количество сверточных слоев; k — количество кернелов (фильтров); $u \times v$ — размерность кернеля; $p \times q$ — размер выходной карты признаков; N_{fc} — количество полносвязных слоев; n — количество нейронов.

Разделение карт признаков

Основной идеей использования сверточного слоя является применение математической операции свертки (фильтра) к входным данным. Фильтр — это набор кернелов. Кернел — матрица с числами, называемыми весами, размерностью, например, 3 × 3. Входные данные можно представить в виде матрицы, где каждый элемент матрицы умножается на соответствующий элемент кернеля, полученные произведения суммируются и записываются в определенный нейрон сверточного слоя. Далее фильтр сдвигается с определенным шагом (stride). Вычисления повторяются до последнего элемента матрицы. Совокупность этих значений формирует карту признаков. Карты признаков могут быть разделены на более мелкие фрагменты в зависимости от гранулярности. Несколько задач итерационно могут выполняться на одном устройстве, где задается потребляемая память для каждой итерации, вместо выполнения операции над полной картой признаков. Однако необходимо учесть, что ввиду особенности СНС, некоторые области с данными будут пересекаться (рис. 2).

Существует два решения для таких проблем:

1) необходимо заново выполнять вычисления, когда они необходимы;
2) сохранить в кэше данные и переиспользовать промежуточные результаты.

Первое решение самое простое, но оно добавляет дополнительные арифметические операции. Кэширование промежуточных результатов позволяет уменьшить количество операций, однако в некоторых ситуациях вычисления будут неравномерными. Первый случай можно использовать, если вычислений немного, однако СНС не обладает такой особенностью, она известна как одна из самых ресурсоемких нейронных

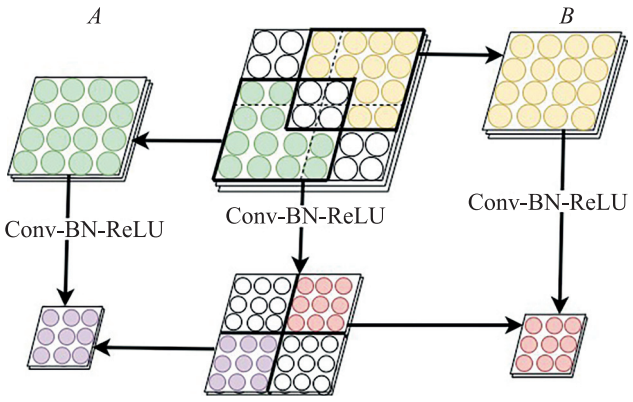

Рис. 2. Пересечение областей карт признаков в сверточной нейронной сети (Conv-BN-ReLU — «Сверточный слой–Пакетная нормализация–Функция активации»)

сетей. Исходя из этого, в данной работе рассматривается второй случай.

Допустим, области A и B распределены между двумя устройствами. Области, которые пересекаются, должны быть продублированы и в каждом устройстве. Требуется хранение данных о пересекающейся области. Но такая схема повторного использования создает зависимости между смежными областями и уменьшает параллелизм. Наиболее применяемый способ — это сбор всех пересекающихся данных и их перераспределение между устройствами. Но в этом случае необходимо постоянно синхронизировать данные слой за слоем, что ограничивает производительность и добавляет дополнительные межкомпонентные взаимодействия. Для организации параллельного решения задач с максимальным переиспользованием данных, когда требуется минимизация синхронизации в динамической распределенной среде, представлен новый метод распределения задач в СНС для кластера компьютеров с ограниченными вычислительными ресурсами (рис. 3).

При распределении заданий в СНС планировщик задач выполняет главную роль, так как он включает в себя очереди задач, стратегии распределения задач и рабочие потоки, которые фактически выполняют задачи. Существует два основных подхода планировщика задач: work sharing, когда планировщик «делится» задачами, и work stealing — «крадет» задачи. Теоретически доказано [16], что стратегия «кражи» задачи дает распределение задач, близкое к оптимальному, поэтому при распределении задач СНС используется стратегия work-stealing. На рис. 3 представлена схема «кражи» задач между устройствами.

Задачи A, C, G, I распределяются по разным узлам устройств и выполняются параллельно, так как они не имеют пересечений в используемых ресурсах. Задачи B, D, F, H выполнятся только после того, как смежные задачи (A–E) распределятся и выполнятся локально, чтобы можно было переиспользовать полученные результаты, находящиеся в пересекающихся областях. На рис. 4 представлена общая концепция планировщика задач СНС для пересекающихся данных.

Модуль *сборщик пересекающихся данных* хранит в себе задачи, которые необходимо переиспользовать, и передает в *пул пересекающихся данных*, откуда данные поступают в *обработчик запросов*. Если данные не поступают с устройства, тогда вычисления выполняются локально на ведущем узле, где и находится *планировщик задач*. Полученные результаты используются в модуле *CHC*, где происходят вычисления, связанные с СНС.

Экспериментальная часть

Описание кластера. Кластер состоит из 24 одноплатных компьютеров RockPro64. В эксперименте использованы 20 узлов. RockPro64 построен на 64-битном процессоре Rockchip RK3399, где система на чипе объединяет двухъядерный ARM Cortex-A72 и четырехъядерный Cortex-A53, имеющий графический процессор Mali-T860MP4 GPU, 4 ГБ ОЗУ. Также имеется встроенная eMMC-память объемом 64 ГБ.

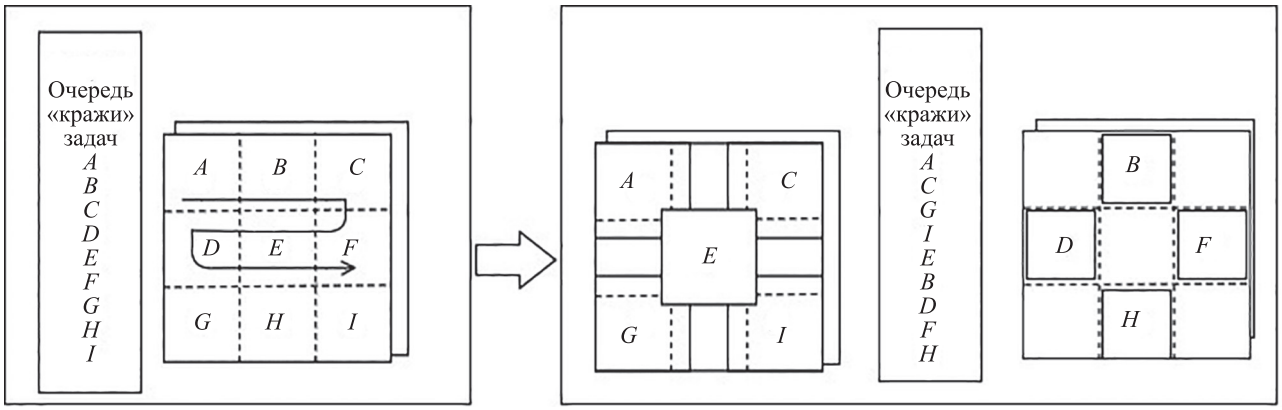

Рис. 3. Распределение задач в сверточной нейронной сети

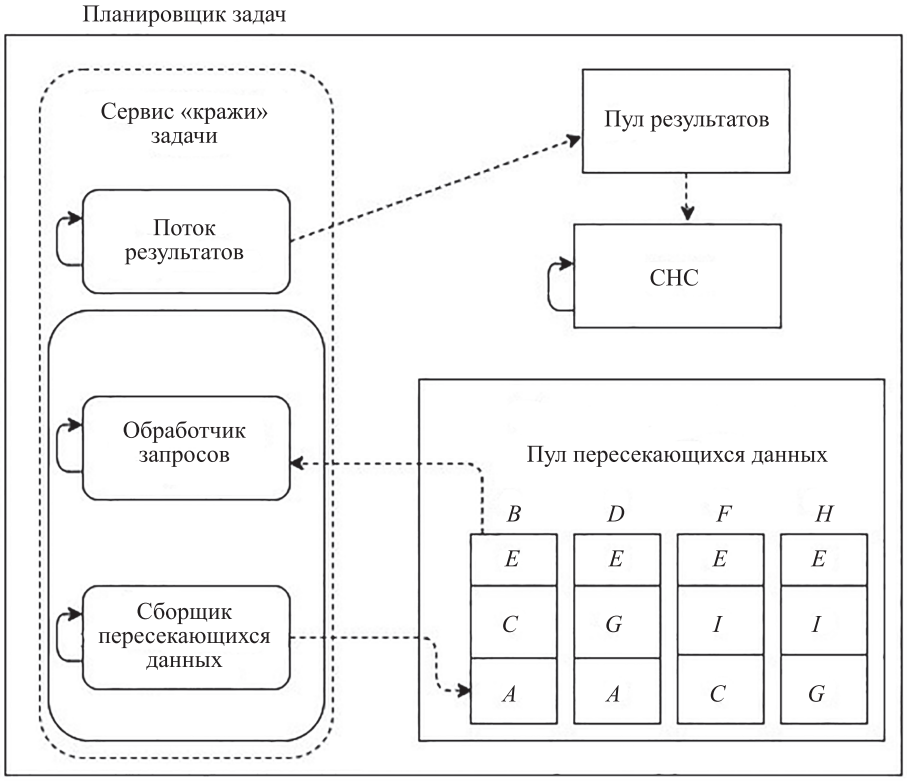

Рис. 4. Общая концепция планировщика задач сверточной нейронной сети для пересекающихся данных

Для определения ограничений при работе с изображениями, проведено распределенное распознавание лиц на кластере из одноплатных компьютеров. В качестве эксперимента выбран набор небольших цветных изображений из датасета FFHQ Faces Data Set [17] размером 128 × 128 пикселей и размером 1000 × 500 из датасета WIDER FACE: A Face Detection Benchmark [18]. Для распознавания лиц использована модель MMOD-CNN из библиотеки dlib [19]. Результаты эксперимента представлены в таблице. Из-за ограничения оперативной памяти на кластере удалось запустить лишь 2000 изображений размером 1000 × 500, что заняло приблизительно 49 мин, далее был превышен объем оперативной памяти. Данное время почти соизмеримо для 50 000 изображений размером 128 × 128 на всех узлах кластера. Для 100 изображений размером 128 × 128 нецелесообразно проведение эксперимента, так как время выполнения существенно не отличается. Распознавание лиц для 50 000 изображений на пяти узлах кластера заняло 2 ч 44 мин.

Для верификации модели и планировщика задач распределенной СНС на кластере из одноплатных компьютеров выполнено тестирование архитектур VGG-16/VGG-19 и AlexNet в задаче классификации изображений, и измерено потребление оперативной памяти на каждом узле кластера (рис. 5). Тестирование проведено на 21-м узле кластера, который является главным узлом, чем обусловлено наибольшее потребление памяти. Однако потребление памяти на кластере является неравномерным, поскольку отсутствует балансировщик нагрузки.

Для сравнения производительности работы, распознавание лиц для 50 000 изображений размером 128 × 128 пикселей было запущено на персональном

Таблица. Результаты тестирования на кластере из одноплатных компьютеров RockPro64

Размер изображений, пиксель	Количество изображений	Количество узлов						
		3	6	9	12	15	18	20
128 × 128	100	Время работы, с						
		36	19	15	12	11	10	9
	1000	328	166	116	91	77	63	56
	5000	1639	825	571	439	355	301	271
	10 000	3240	1643	1138	873	707	597	542
		Количество узлов						
		5		10		15	20	
		Время работы, с						
	50 000	9840		4972		3791	2959	
1000 × 500		Количество узлов						
		20						
		Время работы, с						
	100	177						
	1000	1508						
	2000	2954						

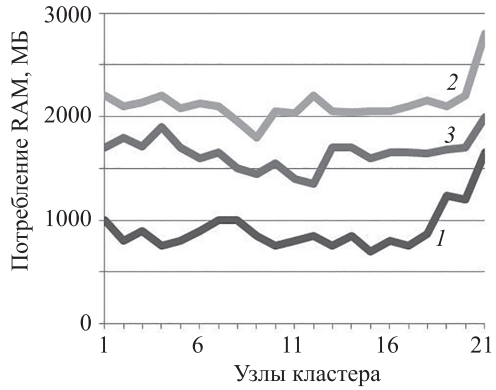

Рис. 5. Потребление оперативной памяти (RAM) на кластере для архитектур AlexNet (кривая *1*), VGG-19 (кривая *2*), VGG-16 (кривая *3*)

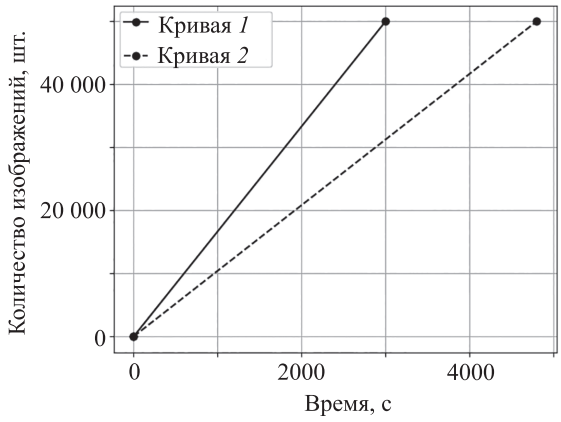

Рис. 6. Сравнение времени работы распознавания лиц на кластере RockPro64 (кривая *1*) и на персональном компьютере «Intel(R) Core(TM) i7-7700» (кривая *2*)

компьютере (ПК) Intel(R) Core(TM) i7-7700 CPU с частотой 3,6 ГГц, имеющем 4 ядра, а также 16 ГБ оперативной памяти и 111 ГБ постоянной памяти, и на кластере из одноплатных компьютеров RockPro64, состоящем из 22 узлов. На ПК распознавание лиц заняло 1 ч 20 мин, а на кластере RockPro64 — 50 мин (рис. 6).

Заключение

Представлено математическое описание сверточной нейронной сети, разработана модель распределенной сверточной нейронной сети для кластера компьютеров с ограниченными вычислительными ресурсами, описаны существующие ограничения на кластере, а также представлен метод разделения карт признаков, что является частью как сверточного слоя, так и других слоев сверточной нейронной сети. Разработана общая концепция планировщика задач для развертывания распределенной сверточной нейронной сети с учетом пересекающихся областей. Для верификации разработанных моделей проведено тестирование на кластере из одноплатных компьютеров RockPro64. По результатам первого эксперимента отмечено существенное сокращение времени при работе с большим количеством изображений, а именно, распознавание лиц для 50 000 изображений размером 128 × 128 пикселей на 20 узлах заняло 49 мин, а на пяти — 2 ч 44 мин, при этом наблюдается незначительная разница распределенного распознавания лиц на 100 изображениях. Для изображений в количестве 1000 и 5000 отмечена небольшая разница времени распределенного распознавания между 12 и 15, а также 15 и 18 узлами, чем больше изображений, тем разница существенней. Однако существует ограничение на количество изображений большего размера, чем больше размер, тем меньшее количество изображений можно распознать. По результатам второго эксперимента наблюдалось неравно-

мерное потребление памяти на разных узлах кластера, однако в рамках архитектур AlexNet, VGG-16/VGG-19 не выявлено нехватки памяти. Представлено сравнение производительности в задаче распознавании лиц для персонального компьютера, а также для кластера из одноплатных компьютеров, что показало значительное преимущество в случае обработки на кластере. В ходе дальнейшей работы предполагается проведение более детальных экспериментов для выявления ограничений, разработки модели балансировщика нагрузки, а также экспериментальная оценка полученных результатов на различных типах данных [20].

Литература

1. Shafique K., Khawaja B.A., Sabir F., Qazi S., Mustaqim M. Internet of Things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios // IEEE Access. 2020. V. 8. P. 23022–23040. doi: 10.1109/ACCESS.2020.2970118
2. Shi W., Cao J., Zhang Q., Li Y., Xu L. Edge computing: Vision and challenges // IEEE Internet of Things Journal. 2016. V. 3. N 5. P. 637–646. doi: 10.1109/JIOT.2016.2579198
3. Tarasenko A.O., Yakimov Y.V., Soloviev V.N. Convolutional neural networks for image classification // CEUR Workshop Proceedings. 2019. V. 2546. P. 101–114.
4. Zangeneh E., Rahmati M., Mohsenzadeh Y. Low resolution face recognition using a two-branch deep convolutional neural network architecture // Expert Systems with Applications. 2020. V. 139. P. 112854. doi: 10.1016/j.eswa.2019.112854
5. Solovyev R., Kustov A., Telpukhov D., Rukhlov V., Kalinin A. Fixed-point convolutional neural network for real-time video processing in FPGA // Proc. of the 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). 2019. P. 1605–1611. doi: 10.1109/EIConRus.2019.8656778
6. Widiastuti N.I. Convolution neural network for text mining and natural language processing // IOP Conference Series: Materials Science and Engineering. 2019. V. 662. N 5. P. 052010. doi: 10.1088/1757-899X/662/5/052010
7. Giménez M., Palanca J., Botti V. Semantic-based padding in convolutional neural networks for improving the performance in natural language processing. A case of study in sentiment analysis // Neurocomputing. 2020. V. 378. P. 315–323. doi: 10.1016/j.neucom.2019.08.096
8. Sim H.S., Kim H.I., Ahn J.J. Is deep learning for image recognition applicable to stock market prediction? // Complexity. 2019. P. 4324878. doi: 10.1155/2019/4324878
9. Borovykh A., Bohte S., Oosterlee C.W. Conditional time series forecasting with convolutional neural networks // arXiv. arXiv:1703.04691. 2018.
10. Mao J., Chen X., Nixon K.W., Krieger C., Chen Y. MoDNN: Local distributed mobile computing system for deep neural network // Proc. 20th Design, Automation and Test in Europe Conference and Exhibition (DATE 2017). 2017. P. 1396–1401. doi: 10.23919/DATE.2017.7927211
11. Wang S., Tuor T., Salonidis T., Leung K.K., Makaya C., He T., Chan K. When edge meets learning: Adaptive control for resource-constrained distributed machine learning // Proc. of the IEEE Conference on Computer Communications (INFOCOM 2018). 2018. P. 63–71. doi: 10.1109/INFOCOM.2018.8486403
12. Motamedi M., Fong D., Ghiasi S. Machine intelligence on resource-constrained IoT devices: The case of thread granularity optimization for CNN inference // ACM Transactions on Embedded Computing Systems. 2017. V. 16. N 5s. P. 151. doi: 10.1145/3126555
13. Khan A., Sohail A., Zahoora U., Qureshi A.S. A survey of the recent architectures of deep convolutional neural networks // Artificial Intelligence Review. 2020. V. 53. N 8. P. 5455–5516. doi: 10.1007/s10462-020-09825-6
14. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks // Advances in Neural Information Processing Systems. 2012. V. 2. P. 1097–1105.
15. Alippi C., Disabato S., Roveri M. Moving convolutional neural networks to embedded systems: the alexnet and VGG-16 case // Proc. 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN). 2018. P. 212–223. doi: 10.1109/IPSN.2018.00049
16. Кучумов Р.И. Реализация и анализ work-stealing планировщика задач // Стохастическая оптимизация в информатике. 2016. Т. 12. № 1. С. 20–39.

References

1. Shafique K., Khawaja B.A., Sabir F., Qazi S., Mustaqim M. Internet of Things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios. *IEEE Access*, 2020, vol. 8, pp. 23022–23040. doi: 10.1109/ACCESS.2020.2970118
2. Shi W., Cao J., Zhang Q., Li Y., Xu L. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 2016, vol. 3, no. 5, pp. 637–646. doi: 10.1109/JIOT.2016.2579198
3. Tarasenko A.O., Yakimov Y.V., Soloviev V.N. Convolutional neural networks for image classification. *CEUR Workshop Proceedings*, 2019, vol. 2546, pp. 101–114.
4. Zangeneh E., Rahmati M., Mohsenzadeh Y. Low resolution face recognition using a two-branch deep convolutional neural network architecture. *Expert Systems with Applications*, 2020, vol. 139, pp. 112854. doi: 10.1016/j.eswa.2019.112854
5. Solovyev R., Kustov A., Telpukhov D., Rukhlov V., Kalinin A. Fixed-point convolutional neural network for real-time video processing in FPGA. *Proc. of the 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, 2019, pp. 1605–1611. doi: 10.1109/EIConRus.2019.8656778
6. Widiastuti N.I. Convolution neural network for text mining and natural language processing. *IOP Conference Series: Materials Science and Engineering*, 2019, vol. 662, no. 5, pp. 052010. doi: 10.1088/1757-899X/662/5/052010
7. Giménez M., Palanca J., Botti V. Semantic-based padding in convolutional neural networks for improving the performance in natural language processing. A case of study in sentiment analysis. *Neurocomputing*, 2020, vol. 378, pp. 315–323. doi: 10.1016/j.neucom.2019.08.096
8. Sim H.S., Kim H.I., Ahn J.J. Is deep learning for image recognition applicable to stock market prediction? *Complexity*, 2019, pp. 4324878. doi: 10.1155/2019/4324878
9. Borovykh A., Bohte S., Oosterlee C.W. Conditional time series forecasting with convolutional neural networks. *arXiv*, arXiv:1703.04691, 2018.
10. Mao J., Chen X., Nixon K.W., Krieger C., Chen Y. MoDNN: Local distributed mobile computing system for deep neural network. *Proc. 20th Design, Automation and Test in Europe Conference and Exhibition (DATE 2017)*, 2017, pp. 1396–1401. doi: 10.23919/DATE.2017.7927211
11. Wang S., Tuor T., Salonidis T., Leung K.K., Makaya C., He T., Chan K. When edge meets learning: Adaptive control for resource-constrained distributed machine learning. *Proc. of the IEEE Conference on Computer Communications (INFOCOM 2018)*, 2018, pp. 63–71. doi: 10.1109/INFOCOM.2018.8486403
12. Motamedi M., Fong D., Ghiasi S. Machine intelligence on resource-constrained IoT devices: The case of thread granularity optimization for CNN inference. *ACM Transactions on Embedded Computing Systems*, 2017, vol. 16, no. 5s, pp. 151. doi: 10.1145/3126555
13. Khan A., Sohail A., Zahoora U., Qureshi A.S. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 2020, vol. 53, no. 8, pp. 5455–5516. doi: 10.1007/s10462-020-09825-6
14. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 2012, vol. 2, pp. 1097–1105.
15. Alippi C., Disabato S., Roveri M. Moving convolutional neural networks to embedded systems: the alexnet and VGG-16 case. *Proc. 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2018, pp. 212–223. doi: 10.1109/IPSN.2018.00049
16. Kuchumov R.I. Implementation and analysis of work-stealing task scheduler. *Stohasticheskaja optimizacija v informatike*, 2016, vol. 12, no. 1, pp. 20–39. (in Russian)

17. Dang H., Liu F., Stehouwer J., Liu X., Jain A. On the detection of digital face manipulation // arXiv. arXiv:1910.01717. 2019.
18. Yang S., Luo P., Loy C.C., Tang X. WIDER FACE: A face detection benchmark // Proc. 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016). 2016. P. 5525–5533. doi: 10.1109/CVPR.2016.596
19. Boyko N., Basystiuk O., Shakhovska N. Performance evaluation and comparison of software for face recognition, based on dlib and opencv library // Proc. 2nd IEEE International Conference on Data Stream Mining and Processing (DSMP). 2018. P. 478–482. doi: 10.1109/DSMP.2018.8478556
20. Khaydarova R., Fishchenko V., Mouromtsev D., Shmatkov V., Lapaev M. ROCK-CNN: a distributed RockPro64-based convolutional neural network cluster for IoT. Verification and performance analysis // Proc. 26th Conference of Open Innovations Association (FRUCT). 2020. P. 174–181. doi: 10.23919/FRUCT48808.2020.9087457

17. Dang H., Liu F., Stehouwer J., Liu X., Jain A. On the detection of digital face manipulation. *arXiv*, arXiv:1910.01717, 2019.
18. Yang S., Luo P., Loy C.C., Tang X. WIDER FACE: A face detection benchmark. *Proc. 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, 2016, pp. 5525–5533. doi: 10.1109/CVPR.2016.596
19. Boyko N., Basystiuk O., Shakhovska N. Performance evaluation and comparison of software for face recognition, based on dlib and opencv library. *Proc. 2nd IEEE International Conference on Data Stream Mining and Processing (DSMP)*, 2018, pp. 478–482. doi: 10.1109/DSMP.2018.8478556
20. Khaydarova R., Fishchenko V., Mouromtsev D., Shmatkov V., Lapaev M. ROCK-CNN: a distributed RockPro64-based convolutional neural network cluster for IoT. Verification and performance analysis. *Proc. 26th Conference of Open Innovations Association (FRUCT)*, 2020, pp. 174–181. doi: 10.23919/FRUCT48808.2020.9087457

Авторы

Хайдарова Резеда Раитовна — инженер-исследователь, аспирант, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 57031874500, ORCID ID: 0000-0001-8270-9192, mignolowa@gmail.com
Муромцев Дмитрий Ильич — кандидат технических наук, доцент, доцент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 55575780100, ORCID ID: 0000-0002-0644-9242, d.muromtsev@gmail.com
Лапаев Максим Владимирович — кандидат технических наук, ведущий разработчик, ООО «ТПП Лаб», Санкт-Петербург, 195009, Российская Федерация, Scopus ID: 56338242800, ORCID ID: 0000-0002-1043-1352, max.wproject@gmail.com
Фищенко Владислав Дмитриевич — студент, инженер-исследователь, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, ORCID ID: 0000-0003-1129-1301, fishenko.vlad@gmail.com

Authors

Rezeda R. Khaydarova — Research Engineer, Postgraduate, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 57031874500, ORCID ID: 0000-0001-8270-9192, mignolowa@gmail.com

Dmitry I. Mouromtsev — PhD, Associate Professor, Associate Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 55575780100, ORCID ID: 0000-0002-0644-9242, d.muromtsev@gmail.com
Maxim V. Lapaev — PhD, Senior developer, TPP Lab LTD, Saint Petersburg, 195009, Russian Federation, Scopus ID: 56338242800, ORCID ID: 0000-0002-1043-1352, max.wproject@gmail.com

Vladislav D. Fishenko — Student, Research Engineer, ITMO University, Saint Petersburg, 197101, Russian Federation, ORCID ID: 0000-0003-1129-1301, fishenko.vlad@gmail.com