Я.И. Хамутовский,
студент 3 курса напр. «Технические науки»,
e-mail: ***yan.khamutovsky@gmail.com,***
науч. рук.: ***Н. А. Искра***,
ст. преп.,
БГУИР,
г. Минск

АКТУАЛЬНЫЕ ПРОБЛЕМЫ ИСПОЛЬЗОВАНИЯ НЕЙРОННЫХ СЕТЕЙ НА МОБИЛЬНЫХ УСТРОЙСТВАХ

Аннотация: данная статья посвящена обзору современных проблем при обучении нейронных сетей на мобильных устройствах, обзору существующих решений проблемы быстродействия, энергопотребления, размера математической модели.

Ключевые слова: нейронные сети, сверточные нейронные сети, глубокое обучение, мобильные устройства

1. Введение

В последнее время все больше и больше люди используют мобильные телефоны как основное средство коммуникаций и развлечений. Предполагается, что к 2020 году общее количество мобильных устройств составит 21% от всех сетевых устройств[12]. С увеличением количества мобильных устройств растет и популярность использования интеллектуальных мобильных приложений, которые, в свою очередь, используют технологии глубокого машинного обучения. В качестве примера можно привести известные голосовые ассистенты: Siri – для iOS, Google Assistant – для Android.

При реализации глубокого обучения в мобильных приложениях разработчики сталкиваются с различными проблемами, такими как противоречие между миниатюрной природой мобильных устройств и потребностью в ресурсах для глубоких нейронных сетей, проблемами быстродействия, энергопотребления и др. Для решения некоторых проблем за

последние несколько лет были сделаны большие шаги в этой области.

В данной статье будут рассмотрены способы решения проблемы обучения, быстродействия, энергопотребления нейронных сетей на мобильных устройствах.

2. Особенности обучения нейронных сетей

Нейронная сеть – это огромный распределенный параллельный процессор, состоящий из элементарных единиц обработки информации, накапливающих экспериментальные знания и представляющих их для последующей обработки [1].

Свёрточная нейронная сеть (CHC) – специальный вид нейронной сети для обработки данных с сеточной топологией. В таких сетях вместо общей операции умножения на матрицу по крайней мере в одном слое используется свертка [2]. CHC применяются для оптического распознавания образов [11], классификации изображений [4], детектирования предметов [5], семантической сегментации [8] и других задач [9].

Глубокие

ронных элементов [3].

Требования к математической модели

Мобильные устройства представляют собой не очень высокопроизводительные устройства по сравнению с настольными или серверными решениями, следовательно, особенные требования предъявляются к математической модели для машинного обучения.

Чаще всего для обучения CHC используются высокопроизводительные компьютеры с большим количеством графических процессоров. На такой сервер затем поступают запросы от пользователей мобильных устройств. Также возможно обучение непосредственно на мобильном устройстве - так называемое “прямое обучение”.

Прямое обучение имеет ряд преимуществ:

- конфиденциальность данных – вся информация будет храниться только на устройстве владельца;

- персонифицирование - приложения будут обучаться на основе данных использования устройства;
- экономия - отсутствие затрат на серверное оборудование.

На платформе iOS прямое обучение используется в следующих приложениях:

- системная клавиатура – анализирует тексты пользователя и предлагает “следующее слово”;
- Face ID – анализирует лицо владельца, обучается непрерывно, так как владелец может изменить свою внешность (надеть очки и тд);
- приложение “Фотографии” – автоматически сортирует фотографии с похожими лицами по папкам, также позволяет производить поиск по предметам на фотографии, например, на запрос “зонт”, приложение выведет все фотографии с предметом “зонт”;
- приложение “Здоровье” – используя датчики, например, с устройства Apple Watch, позволяет предсказать на основе данных о движении или биении сердца время пробуждения или даже прогнозировать сердечный приступ.

Распределенное обучение

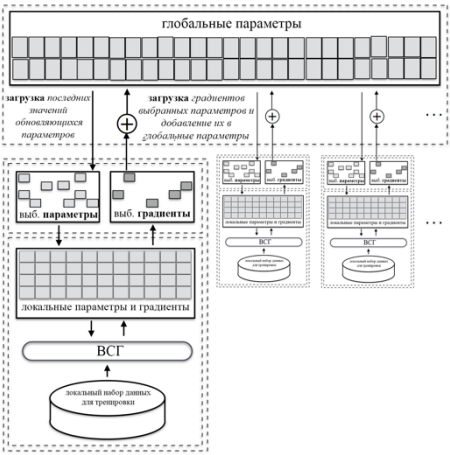
Мобильные устройства записывают большие объемы данных о своих пользователях, поэтому стадия ГНС довольно ресурсоемка. Пока обучение ГНС происходит в облачных центрах обработки.

Настройка параметров обучения в ГНС является основным в процессе обучения. Алгоритм градиентного спуска и его различные варианты являются наиболее часто используемыми алгоритмами обучения ГНС. Поэтому основной задачей распределенного обучения является разработка алгоритма распределенного градиентного спуска.

Одним из таких алгоритмов является алгоритм распределенного выборочного стохастического градиента (ВСГ)

[10], который реализует распределенное и совместное обучение с использованием данных из разных источников.

В предлагаемом алгоритме ВСГ несколько сторон занимаются одновременной тренировкой своих ГНС, а затем после каждой итерации обучения загружают градиент выбранного параметра для глобальных параметров, хранящихся на сервере, не делясь наборами входных данных. Глобальные параметры вычисляются как сумма всех параметров поставляемых от участников, затем участники загружают новые глобальные параметры для обновления собственной локальной ГНС.



Федеративное обучение

Новую схему обучения предложила компания Google, суть ее заключается в следующем: схема позволяет обучать модель, храня данные локально. Каждый участник загружает “общую” модель из облачного хранилища и улучшает модель, используя локальные данные. Затем участник вычисляет изменения между исходной моделью и обученной, а полученную разницу отправляет обратно на сервер.

Учитывая ограничения полосы пропускания Интернет соединения такая схема обучения старается максимально использовать процессоры участников, тем самым генерируя обновления для модели более высокого качества, нежели при использовании алгоритма ВСГ.

Предположим, что имеется N участников федеративной схемы обучения, которые обучают общую модель с функцией потерь L. Для участника n его градиент $g_k = grad\ L_k(\omega)$, где ω – параметры текущей модели ГНС.

В традиционном алгоритме распределенного градиентного спуска сервер глобальных параметров обновляет параметры как:

$$\omega_{t+1} = \omega_t - c\sum_{k=1}^{K}\frac{n_k}{n}g_k, \quad (1)$$

где c – скорость обучения, n_k – размер локального набора данных k -го клиента, n – сумма всех наборов данных. Можно переписать следующим образом:

$$\forall k,\ \omega_{t+1}^k = \omega_t - cg_k \quad (2)$$

Это означает, что участник вычисляет один шаг градиентного спуска по текущим параметрам, а глобальный сервер параметров аккумулирует все градиенты, загруженные участниками, для обновления текущих параметров. Загрузка градиентов после шага градиентного спуска означает большее количество итераций до сходимости. Необходимо, чтобы участник выполнял $\omega_{t+1}^k = \omega_t - cg_k$ несколько раз перед загрузкой на сервер для генерации градиентов более высокого качества. Экспериментальные результаты [22] показывают, что федеративная схема производительнее алгоритма распределенного градиентного спуска.

Федеративное обучение эффективнее простого алгоритма ВСГ, однако для мобильного устройства обучение модели – ресурсоемкая задача. Компания Google отметила, что обучение может происходить только при подключении устройства к сети питания и сети Wi-Fi, при этом оно должно находиться в режиме ожидания.

3. Проблемы машинного обучения

Быстродействие мобильного приложения, использующего ГНС должно быть высоким, соответственно время выполнения задачи должно быть минимальным.

В последних разработках был предложен фреймворк Paleo [6], который предоставляет возможность оценки метрик аппаратного обеспечения. Авторы данного фреймворка предлагают, используя аналитические методы, определять время выполнения при запуске ГНС на различных платформах.

Вычисления в нейронной сети можно выразить как ориентированный граф $N = (\{u^{(i)}\}_{i=1}^{n}, \{u^{(i)}, u^{(j)}\})$, где каждый узел $u^{(i)}$ ассоциирован с операцией $f^{(i)}$ на устройстве $d^{(i)}$. Каждая грань $(u^{(i)}, u^{(j)})$ означает, что операция $f^{(j)}$ не может быть выполнена раньше, чем операция $f^{(i)}$ закончится. $Pa(u)$ обозначает набор непосредственных родительских узлов $u^{(j)}$. Каждый слой в нейронной сети моделируется как узел, а связи между слоями – грань.

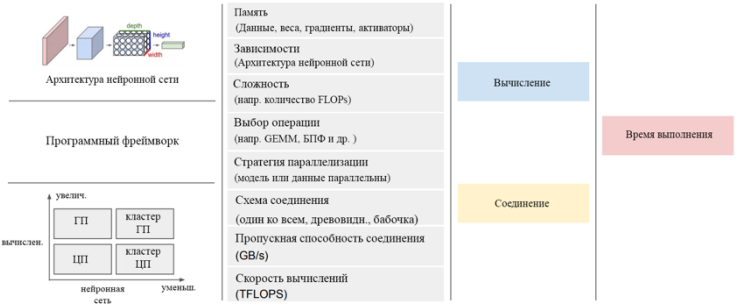


Рисунок 2 – Обзор подхода к моделированию в Paleo [6]

Paleo делит время выполнения на 2 части: время вычисления и время связи. Для слоя u и операции f на устройстве d полное время выполнения может быть определено как:

$$T(u) = R(Pa(u)) + C(f, d) + W(f, d), \quad (3)$$

где $R(Pa(u))$ – время для получения входных данных, созданных родительскими слоями, $C(f, d)$ – полное время вычисления операции f на устройстве d, а $W(f, d)$ – время записи вывода в локальную память устройства.

Интерес представляют метрики для устройства d. Пусть скорость вычислений определена как средняя, соответственно, что время вычисления может быть определено как $C(f, d) = FLOPs(f) / speed(d)$. FLOPs (операция с плавающей запятой). Время ввода $R(Pa(u))$ и вывода $W(u)$ рассчитывается

как размер элементов памяти, участвующих в вычислениях, деленный на ширину полосы ввода вывода устройства.

Одной из центральных проблем машинного обучения являются требования к энергоэффективности. Мобильные устройства, как правило, комплектуются аккумуляторами небольшой емкости, что в свою очередь не позволяет ГНС быть энергозатратной.

Для прогнозирования потребляемой энергии, времени работы был внедрен фреймворк NeuralPower, который базируется на многослойной разреженной полиномиальной регрессии для определения энергопотребления сверточных нейронных сетей.

Послойно– уровневый подход обеспечивает точность модели для запуска на определенном слое конкретной платформы. NeuralPower изучает глубоко обученную модель путем запуска на конкретной платформе. Например, время выполнения может быть определено как:

$$F(x_T) = \mathrm{c} \cdot \prod_{i=1}^{D_T} x_i^{q_i} + \sum_s l_s^{'} F_s(x_T), \quad (4)$$

где $x_T \in R^{D_T}; q_i \in N; \forall j; \sum_{i=1}^{D_T} q_i \le K_T$.

Выражение состоит из двух частей.

Первая часть – регулярные члены полинома степени K_T, которые являются функциями входного вектора x_T. q_i – показатель для x_i в полиномиальном члене, а c – скорость обучения. Вектор D^T содержит гиперпараметры конфигурации слоев. Для различных слоев размерность D^T варьируется.

Вторая часть состоит из специальных полиномиальных термов F, которые представляют физические операции, связанные с каждым слоем (общее количество обращений к памяти и др.). Число специальных термов отличается от слоя к слою. А $l_s^{'}$ – коэффициент s - го специального термина для изучения.

NeuralPower в настоящее время является самым точным методом с точки зрения ошибки прогнозирования при

определении времени работы и потребления энергии для различных сверточных нейронных сетей, со средней квадратичной ошибкой менее 5%.

Кроме того, точность NeuralPower на 70% выше по сравнению с Paleo.

Еще одной немаловажной проблемой является размер обучаемой модели, поскольку размер хранилища мобильного устройства ограничен.

Сверточные нейронные сети обычно состоят из сверточных слоев и полностью связанных слоев, которые доминируют в вычислении стоимости и памяти соответственно. В работе [14] была показана возможность устранения избыточности нейронных сетей, было предложено несколько методов сжатия сверточных нейронных сетей. Исследование [15] показало, что весовую матрицу полностью связанного слоя можно сжать путем применения усеченного сингулярного разложения (УСР) без значительного снижения точности предсказания. Несколько лет назад были предложены различные методы, основанные на векторном квантовании [16] и тензорном разложении [17]. Они показали лучшую способность к сжатию чем УСР. Для ускорения сверточных слоев были предложены несколько методов, основанных на низкоразрядном разложении тензора сверточного ядра [15, 19, 21], но они сжимают не всю сеть, а только некоторые её слои.

В [18] было представлено «асимметричное (3d) разложение» для ускорения всех сверточных слоев, где исходная свертка DЧD разложена до DЧ1, 1ЧD и свертки 1Ч1. Кроме того, также был представлен метод оптимизации, который минимизирует ошибки нелинейных ответов.

4. Практическое применение нейронных сетей

Нейронные сети применяются при разработке мобильных ассистентов, таких как Cortana Microsoft, Siri Apple, Алиса Яндекс.

Нейронные сети широко применяются также и в медицине, так, например, приложения, следящие за сердечным ритмом человека могут предсказывать ухудшения состояния здоровья. Причем, диапазон весьма широк: нейронные сети

могут предсказывать и образование раковых опухолей, и заболевания верхних дыхательных путей, например, астму.

Для определения астмы [21] нейронные сети используют медицинские отчеты, содержащие различную информацию: тесты функций легких (например, результаты спирометрии), отчеты компьютерной томографии грудной клетки, результаты импульсной осциллометрии и др. Затем результаты анализируются с помощью специальных алгоритмов: контекстно-зависимой автоассоциативной памяти [21], C4.5 алгоритм с байесовскими сетями [21], модель обратного распространения [21], метод оптимизации роевых частиц [21].

Рисунок 3 – Алгоритм контекстно- зависимой автоассоциативной памяти [21]

Этот алгоритм хорошо подходит для медицинской сферы, особенно в диагностике и прогнозировании, так как он вычисляет выходные данные после сопоставления с большой базой данных, а затем добавляет связанные выходные данные в ранее существовавшую базу данных о заболеваниях. Лучший способ представить входные данные - через векторы и / или матрицы. Векторы связываются с новой информацией, добавленной пользователем, и набором возможных прогнозов в зависимости от ранее загруженной базы данных.

Согласно исследованию [21] алгоритм контекстно-зависимой автоассоциативной памяти показал наилучшую точность (86 ± 3%) по сравнению с другими алгоритмами.

5. Выводы

Нейронные сети предоставляют огромный потенциал для решения сложных задач.

В данной статье рассмотрены современные проблемы в обучении и применении нейронных сетей на мобильных устройствах, такие как повышенные требования к энергопотреблению, быстродействие, размер математической модели.

Несмотря на достигнутые успехи в минимизации размера математической модели, оптимизации быстродействия нейронных сетей, наиболее сложным является обучение нейронной сети непосредственно на мобильном устройстве. Данная задача оставляет открытыми ряд вопросов для дальнейшего поиска наиболее эффективных решений.

Литература и примечания:

[1] Хайкин С. Нейронные сети: полный курс, 2-е издание: учеб. пособие. – М. : Издательский дом “Вильямс”, 2006 – 1104 с.

[2] Гудфеллоу Я. Глубокое обучение 2- издание: учеб. пособие. – М.: ДМК Пресс, 2018. – 652 с.

росетевые технологии обработки данных: учеб. пособие – Минск: БГУ. –2017. – 263 с.

[4] Russakovsky O. ImageNet Large Scale Visual Recognition Challenge // International Journal of Computer Vision (IJCV). – 2015. – vol. 115, no. 3.

[5] Girshick R. Region-based Convolutional Networks for Accurate Object Detection and Segmentation // The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2014. – pp. 580-587.

[6] Stamoulis D. Paleo: a performance model for deep neural networks // Proceedings of the International Conference on Computer-Aided Design (ICCAD). – 2018. – Article No. 23

[7] HyperPower: Power- and memory-constrained hyper-parameter optimization for neural networks
– // Design, Automation & Test in Europe Conference & Exhibition. – 2018. – pp. 19 – 24.

[8] Long J. Fully Convolutional Networks for Semantic Segmentation // The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2015. – pp. 3431 - 3440.

[9] Gu J. Non-autoregressive neural machine translation // ICLR 2015. – 2015. – pp. 245-258.

[10] Shokri R. Privacy-preserving deep learning // 22Nd ACM SIGSAC Conference on Computer and Communications Security (CCS). – 2015. – pp. 345 – 357.

[11] LeCun Gradient based learning applied to document recognition // Proceedings of the IEEE. –1998. – pp. 2278 - 2324.

[12] Cisco Cisco visual networking index: forecast and methodology 2015- 2020: White Paper [электронный ресурс] // CISCO.COM: Cisco networking. 2016 г. – Электрон. данные. URL: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html (дата обращения 01.12.2018 г.).

[13] Marculescu D. Hardware-Aware Machine Learning: Modeling and Optimization [электронный ресурс] // ARXIV.ORG: Cornell University Library. 2018 г. – Электрон. данные. URL: https://arxiv.org/abs/1809.05476v1 (дата обращения 01.12.2018 г.).

[14] Denil M. Predicting Parameters in Deep Learning [электронный ресурс] // PAPERS.NIPS.CC: Electronic Proceedings of the Neural Information Processing Systems Conference Neural Information Processing Systems Conference. 2013 г. – Электрон. данные. URL: http://papers.nips.cc/paper/5025-predicting-parameters-in-deep-learning.pdf (дата обращения 03.12.2018 г.).

[15] Denton E.L. Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation[электронный ресурс] // PAPERS.NIPS.CC: Electronic Proceedings of the Neural Information Processing Systems Conference Neural Information Processing Systems Conference. 2014 г. – Электрон. данные. URL: https://papers.nips.cc/paper/5544-exploiting-linear-structure-within-convolutional-networks-for-efficient-evaluation.pdf (дата обращения 03.12.2018 г.).

[16] Gong Y. Compressing Deep Convolutional Networks using Vector Quantization[электронный ресурс] // ARXIV.ORG: Cornell University Library. 2014 г. – Электрон. данные. URL: https://arxiv.org/abs/1412.6115 (дата обращения 04.12.2018 г.).

[17] Novikov A. Tensorizing Neural Networks [электронный ресурс] // PAPERS.NIPS.CC: Electronic Proceedings of the Neural Information Processing Systems Conference Neural Information Processing Systems Conference. 2015 г. – Электрон. данные. URL: https://papers.nips.cc/paper/5787-tensorizing-neural-networks (дата обращения 04.12.2018 г.).

[18] Zhang X. Character-level Convolutional Networks for Text Classification [электронный ресурс] // PAPERS.NIPS.CC: Electronic Proceedings of the Neural Information Processing Systems Conference Neural Information Processing Systems Conference. 2015 г. – Электрон. данные. URL: https://papers.nips.cc/paper/5782-character-level-convolutional-networks-for-text-classification (дата обращения 05.12.2018 г.).

[19] Jaderberg M. Spatial Transformer Networks Part [электронный ресурс] // PAPERS.NIPS.CC: Electronic Proceedings of the Neural Information Processing Systems Conference Neural Information Processing Systems Conference. 2015 г. – Электрон. данные. URL: https://papers.nips.cc/paper/5854-spatial-transformer-networks (дата обращения 06.12.2018 г.).

[20] Lebedev V. Speeding-up Convolutional Neural Networks Using Fine-tuned CP-Decomposition [электронный ресурс] // ARXIV.ORG: Cornell University Library. 2018 г. – Электрон. данные. URL: https://arxiv.org/abs/1412.6553 (дата обращения 09.12.2018 г.).

[21] Kukreja S. A Comprehensive Study on the Applications of Machine Learning for the Medical Diagnosis and Prognosis of Asthma [электронный ресурс] // ARXIV.ORG: Cornell University Library. 2018 г. – Электрон. данные. URL: https://arxiv.org/abs/1804.04612v1 (дата обращения 09.12.2018 г.).

[22] McMahan H. B. Federated Learning: Collaborative Machine Learning without Centralized Training Data [электронный ресурс] // AI.GOOGLEBLOG.COM:. Google AI Blog 2017 г. – Электрон. данные. URL:

https://ai.googleblog.com/2017/04/federated-learning-collaborative.html (дата обращения 07.12.2018 г.).