Научная статья

УДК 005.13.11
DOI 10.25205/1818-7900-2023-21-2-51-62

Проблемы оценки качества архитектур нейронных сетей и алгоритмов поиска архитектур

Андрей Сергеевич Щербин

Новосибирский государственный научный исследовательский университет
Новосибирск, Россия

a.shsherbin@g.nsu.ru

Аннотация

В статье приведен обзор научных публикаций в области поиска архитектур нейронных сетей. Рассмотрены не только публикации, посвященные алгоритмам поиска архитектур, но и серия статьей, посвященная оценке качества алгоритмов поиска. На основе проведенного обзора обозначены актуальные проблемы в области оценки качества архитектур нейронных сетей и сравнения алгоритмов поиска.

Ключевые слова

нейронные сети, поиск архитектур нейронных сетей, оценка качества алгоритмов поиска архитектур

Для цитирования

Щербин А. С. Проблемы оценки качества архитектур нейронных сетей и алгоритмов поиска архитектур // Вестник НГУ. Серия: Информационные технологии. 2023. Т. 21, № 2. С. 51–62. DOI 10.25205/1818-7900-2023-21-2-51-62

Problems of Neural Network Architecture Benchmarking and Search

Andrey S. Shcherbin

Novosibirsk State University
Novosibirsk, Russian Federation

a.shsherbin@g.nsu.ru

Abstract

In this paper we made a survey of neural architecture search algorithms and their benchmarking. Based on our survey we highlight the current problems in the quality of neural network architecture benchmarking and in the comparison of neural architecture search algorithms.

Keywords

neural networks, neural architecture search, neural architecture search algorithms benchmarking

For citation

Shcherbin A. S. Problems of Neural Network Architecture Benchmarking and Search. *Vestnik NSU. Series: Information Technologies*, 2023, vol. 21, no. 2, pp. 51–62. DOI 10.25205/1818-7900-2023-21-2-51-62

Введение

Темпы развития алгоритмов искусственного интеллекта, основанного на искусственных нейронных сетях, показывают, что их можно считать достаточно универсальной моделью для решения разного рода практических задач. Однако вместе с универсальностью эти модели имеют и ряд недостатков, в их числе вычислительная сложность, а также объем требуемой для вычислений памяти. Эти особенности ведут к увеличению времени отклика приложений, в основе которых используются нейронные сети, а также к увеличению энергопотребления систем на их основе.

Скорость работы и потребление памяти в приложениях, основанных на нейронных сетях, чаще всего определяется архитектурой сети. Задачу нахождения оптимальной архитектуры сети с точки зрения требуемых вычислительных ресурсов, а также метрики качества решают методами поиска архитектур нейронных сетей (Neural architecture search). Для поиска применяются различные алгоритмы оптимизации: эволюционные [1], байесовские [2], дифференцируемые [3], а также алгоритмы, основанные на обучении с подкреплением [4, 5].

Задача поиска архитектуры сети является задачей оптимизации с ограничениями. Ограничения зачастую накладываются специалистами-практиками, и могут зависеть от объема памяти в целевом устройстве, требований бизнеса на время отклика системы, количества арифметико-логических устройств в разрабатываемом процессоре. То есть задача состоит в том, чтобы найти архитектуру с заданной вычислительной сложностью (возможно не ограниченной), имеющую максимальную точность для конкретной задачи.

Архитектура нейронной сети может быть представлена в виде графа, вершинами которого являются операции, а ребрами – зависимости по данным. Поиск архитектуры требуется завершить за конечное время, для этого исследователи ограничивают набор возможных операций, называя полученное множество возможных архитектур пространством поиска.

Оценка качества алгоритмов поиска архитектур является вычислительно сложной задачей, так как требует обучения всех возможных архитектур из пространства поиска. Более того, так как обучение нейронной сети является стохастическим процессом, а его результат зависит от множества гиперпараметров, проблема оценки качества алгоритмов поиска дополнительно усложняется.

1. Постановка задачи поиска архитектуры нейронных сетей

Пусть Θ – множество всех возможных архитектур, заданное исследователем; λ – функция оценки вычислительной сложности архитектуры нейронной сети; γ – верхняя граница вычислительной сложности архитектуры; μ – функция оценки качества работы архитектуры на решаемой задаче. Тогда задача поиска архитектуры может быть сформулирована следующим образом:

$$\alpha = \max(\mu(\alpha i)) \mid \lambda(\alpha i) <= \gamma \forall \alpha i \in \Theta.$$

Сложность получения $\lambda(\alpha)$ на практике зависит от того, на какие параметры накладываются ограничения. Часто необходимо, чтобы модель работала с заданной скоростью на заданном устройстве. В таком случае для оценки времени работы производится измерение этого времени на целевом устройстве с учетом всех оптимизаций программной платформы, а также особенностей устройства. Программная платформа может выбирать различные алгоритмы вычисления операций в зависимости от архитектуры модели: занимаемой ею памяти, степени параллелизма операций, количества свободных арифметико-логических устройств в каждый момент вычисления нейронной сети. В устройстве могут быть эффективно реализованы опе-

рации для конкретных размерностей тензоров входных данных или для конкретных видов операций (например свертки, количество каналов которых кратно 8).

Во многих методах [6, 7, 8] в оптимизируемый функционал добавляются дополнительные ограничения, позволяющие находить архитектуры, удовлетворяющие требованиям не только по качеству работы модели, но и по потребляемой памяти, и времени ее вычисления. Архитектуры [8, 9], найденные для задачи ImageNet2012 [10], показывают лучшее качество при меньшем количестве операций, т. е. являются более эффективными, чем разработанные исследователями вручную с точки зрения энергопотребления. Исследователи в области поиска архитектур часто анализируют результаты своих работ в терминах Парето оптимальности. Архитектура считается оптимальной по Парето в некотором пространстве критериев, если ее улучшение по какому-либо критерию ведет к ухудшению по другим критериям.

Стоит отметить, что реализация новых идей в архитектурах нейронных сетей (остаточные связи, групповые свертки, отдельная обработка пространственных и поканальных признаков) итеративно увеличивала качество моделей на задаче ImageNet. Однако могут существовать более оптимальные (по различным критериям) модели для этой задачи, реализующие те же самые идеи. Оптимальность архитектуры достигается за счет подбора гиперпараметров под конкретную задачу.

2. Обзор алгоритмов поиска архитектур

Для понимания развития направления поиска архитектур нейронных сетей необходимо рассмотреть несколько знаковых работ. В большинстве из них основной задачей является поиск архитектуры, оптимальной для конкретной задачи, однако не производится детального анализа свойств предложенного алгоритма поиска. Некоторые из предложенных алгоритмов являются вычислительно сложными, что препятствует их применению в практических задачах.

2.1. Поиск архитектур через обучение с подкреплением

В первых работах [4], посвященных поиску архитектур нейронных сетей, поиск производился при помощи нейронной сети-контроллера, состоящей из ячеек с долгой краткосрочной памятью (Long Short Term Memory cells) [11] и алгоритма обучения с подкреплением REINFORCE [12]. Исследователи решали задачу нахождения архитектуры с максимальной точностью классификации на наборе данных CIFAR-10 [13]. Для каждой архитектуры-кандидата производилась тренировка модели с целью оценить ее точность. Для вычислительного эксперимента использовались 800 графических ускорителей. Для тренировки всех архитектур-кандидатов использовался одинаковый набор гиперпараметров обучения, предложенный в статье [14]. Гиперпараметры обучения сети-контроллера были подобраны исследователями самостоятельно.

Авторы предложили подход к поиску архитектур нейронных сетей, который получил развитие, однако при использовании одинаковых гиперпараметров обучения для оценки качества всех архитектур постановка задачи сужается. Различные архитектуры могут иметь различные оптимальные гиперпараметры обучения для одной и той же задачи, что вносит некоторый сдвиг в функцию оценки качества архитектуры при применении одинаковых гиперпараметров. Вследствие наличия сдвига утверждать об оптимальности архитектуры для задачи возможно только в контексте конкретных гиперпараметров обучения.

2.2. Алгоритм NASNet

Развитием описанных выше идей является алгоритм, который позволил найти одну из популярных по сей день сгенерированных архитектур, – NASNet [5]. В этой статье исследователи

заранее определили структуру нейронной сети: последовательность и количество блоков двух видов. Обычный блок (Normal block) не изменял пространственного разрешения карты признаков, сжимающий блок (Reduction block) уменьшал пространственное разрешение. Задача поиска архитектуры была сформулирована как поиск оптимальной структуры каждого из блоков. Поиск производился в ограниченном пространстве операций: свертки с разными размерами ядер, операции подвыборки (pooling) с разными размерами ядер, поканальные и разреженные свертки, а также тождественная операция. В качестве алгоритма поиска использовалась рекуррентная нейронная сеть, называемая контроллером. Архитектура представлялась в виде последовательности символов, что позволило авторам использовать для поиска архитектуры тот же подход, что и для генерации текста. Для оценки качества предложенного подхода авторы обращают внимание читателя на точность найденной нейронной сети на наборе данных ImageNet и утверждают, что найденная архитектура обладает лучшей точностью, чем текущие лучшие решения, при меньших вычислительных затратах.

Однако подход, которым была найдена архитектура NASNet, является вычислительно сложным, так как производит обучение сети-контроллера. Также данный подход использует предположение о том, что существуют блоки, которые являются оптимальными на любом этапе обработки данных. Данное предположение уменьшает пространство поиска архитектур. Такой подход называется микропоиском, он противопоставляется макропоиску, когда каждый блок в сети может быть составлен из разных операций и количество используемых блоков может быть найдено алгоритмом. Макропоиск является еще более вычислительно сложным, так как количество возможных архитектур экспоненциально возрастает в сравнении с микропоиском, и для больших структур некоторые алгоритмы могут быть подвержены проклятию размерности.

2.3. PNASNet

Метод итеративного (от простого к сложному) макропоиска архитектур предложен в [15]. Идея заключается в усложнении пространства поиска в процессе оптимизации путем добавления дополнительных ячеек. Структура ячейки является фиксированной, однако набор операций в ней оптимизируется. При этом каждая ячейка может иметь собственный набор операций, оптимальный для конкретной глубины сети. В процессе оптимизации исследователи обучают модель для оценки качества архитектур. Это позволяет сократить требуемые вычислительные ресурсы, а также производить выбор операций для ячеек с учетом информации о влиянии предыдущих операций-кандидатов на точность модели.

Модель для оценки качества по архитектуре обучается на первых двух ячейках, при добавлении последующих ячеек их архитектуры ранжируются в соответствии с предсказаниями этой модели. Нейронные сети с наиболее перспективными по предсказаниям модели-оценщика ячейками дообучаются на целевой задаче, что позволяет дообучить и модель-оценщика.

В качестве модели-оценщика авторы использовали нейронную сеть с долгой краткосрочной памятью [11], архитектура и гиперпараметры модели отличались от предложенных в [5], что не позволяет сделать вывод о вкладе метода итеративного поиска в итоговую точность найденной модели и сравнить алгоритмы поиска в одинаковых условиях.

3. Развитие методов оценки качества алгоритмов поиска

Проблема воспроизводимости результатов исследований, посвященных поиску архитектур нейронных сетей, поднимается во многих публикациях [16, 17, 18, 19, 20, 21]. Исследователи отмечают, что причинами проблем с воспроизводимостью являются различия в постановке задачи, выборе пространства поиска, способах оценки качества моделей-кандидатов, гиперпараметрах обучения. Для корректного сравнения алгоритмов поиска архитектур независимо

от гиперпараметров обучения, пространства поиска и постановки задачи исследователи проводят обучение всех возможных архитектур с различными гиперпараметрами из некоторого множества конфигураций. Далее предлагается использовать в алгоритме поиска точность обученной архитектуры в качестве оценки ее качества. Далее рассмотрим реализации этой идеи, их преимущества и недостатки.

3.1. NASBench-101

Рассмотрим первой статью NasBench-101 [17] для того, чтобы более детально сформировать представление о предлагаемом подходе, а также понять мотивацию принимаемых авторами решений. Основная идея статьи состоит в создании набора данных, содержащего точности решения задачи классификации для каждой из архитектур из пространства поиска. Этот набор данных предлагается использовать как разметку, а также для оценки качества предлагаемых алгоритмом в процессе поиска архитектур. Обучение моделей производилось с одним набором гиперпараметров для всех архитектур на наборе данных CIFAR-10.

Задача поиска заключалась в оптимизации структуры блока нейронной сети, который использовался в заранее определенной структуре. То есть задача представляла собой микропоиск, что было обосновано вычислительной сложностью. Архитектура блока представлялась исследователями в виде графа, вершинами которого были операции, а ребрами – зависимости по данным. Также для ограничения пространства поиска были введены некоторые ограничения: допускалось использование только трех операций, число вершин было ограничено семью, а число ребер в графе 9.

Исследователи продемонстрировали Парето-фронт при помощи графиков, определили понятие локальности архитектур как расстояние редактирования представления графа. В результате оценки автокорреляции архитектур методом случайных блужданий было показано, что корреляция становится отличима от шума при расстоянии большем или равном шести. Также авторы заметили, что 35 % всех архитектур находятся на расстоянии редактирования не более чем 6 от лучшей выбранной архитектуры. Это означает, что 1 из 50 000 случайно выбранных архитектур является неотличимой от наилучшей архитектуры.

Для оценки качества авторы выбрали алгоритмы поиска архитектур, основанные на байесовских методах, эволюционных алгоритмах, обучении с подкреплением, оценке качества моделей в процессе их обучения, а также случайном выборе. Методы сравнивались в условиях ограниченного времени на поиск оптимальной архитектуры. Так как время на выполнение одной итерации отличается для разных методов, было произведено различное количество итераций. В качестве результатов сравнения алгоритмов авторы приводят следующие выводы:

- Эволюционный алгоритм, а также некоторые байесовские методы на данной задаче сходятся в 5 раз быстрее, чем метод случайного выбора архитектур.
- Алгоритм, использующий обучение с подкреплением хоть в начале и превосходил по качеству алгоритм случайного выбора, но в итоге уступил ему, так как требовал больших вычислительных ресурсов на итерацию.

3.2. NAS-Bench201

Следующая статья [18], посвященная воспроизводимости результатов и сравнимости методов в области поиска архитектур нейронных сетей продолжает идеи NasBench-101. Одним из методологических отличий является отказ от ограничений на число связей в вычислительном графе. Это позволяет расширить множество применимых алгоритмов поиска архитектур градиентными методами.

Также авторы данной статьи изменили представление вычислительного графа: в качестве вершин теперь выступают данные, а операции кодируются ребрами графа. Структура блока,

а также набор возможных операций в пространстве поиска были выбраны таким образом, чтобы в пространство поиска входила архитектура ResNet. Таким образом, исследователям удалось уменьшить количество уникальных архитектур блока в пространстве поиска за счет сокращения количества возможных операций и связей между ними.

Для более подробного изучения алгоритмов поиска в качестве одного из результатов исследователи представили историю тренировки для каждой из архитектур пространства поиска. Это позволяет оценивать качество алгоритмов, в которых оценка качества архитектуры-кандидата происходит путем ее тренировки некоторое количество шагов. Обучение моделей в рамках данной работы производилось на трех наборах данных: CIFAR-100, CIFAR-10 и ImageNet-16-120. Последний набор данных был получен из ImageNet путем выбора первых 120 классов и уменьшения разрешения изображений до 16 х 16. Возможность оценить качество модели на нескольких задачах также позволяет получить более устойчивую оценку качества алгоритма поиска. К тому же наличие нескольких задач позволяет производить поиск оптимальной архитектуры на одном наборе данных, а сравнение и оценку точности проводить на других.

Исследователи по-прежнему использовали одинаковые гиперпараметры для обучения всех архитектур. Гиперпараметры были выбраны на основе анализа научных статей, использующих наборы данных CIFAR. Для ImageNet-16-120 использовался тот же набор гиперпараметров, что и для CIFAR-100. Такой подход позволяет сократить количество обучений моделей и моделирует использование исследователем фиксированного набора гиперпараметров при изменении архитектуры модели, что имеет место в практических приложениях. Результатом исследования является набор данных, который содержит информацию о ходе каждого из трех обучений каждой модели из описанного пространства поиска на каждую из трех задач классификации.

В заключение авторы приводят ряд рекомендаций по использованию набора данных NasBench-201:

– Избегать переобучения под NasBench-201:
  - Не использовать специализированных ограничений на операции в пространстве поиска. Например, не ограничивать количество операций определенного типа.
  - Использовать при оценке кандидатов и выборе наилучшей модели представленные в наборе данных значения качества, даже если имеется другая конфигурация гиперпараметров, позволяющая получить лучшее качество.
  - В качестве итоговой метрики качества алгоритма поиска использовать результат нескольких запусков. Это делает результаты более устойчивыми, и не требует больших вычислительных затрат при использовании набора данных NasBench.
– Помнить о возможном шуме в оценке качества архитектур, вызванном использованием одинаковых конфигураций гиперпараметров для разных архитектур. Возможным решением может быть оптимизация гиперпараметров, однако это требует дополнительных вычислительных ресурсов.
– Не использовать алгоритмы поиска архитектур, устройство которых может вносить шум в оценку качества моделей. Например, алгоритмы с разделяемыми весами, в которых поиск заключается в выборе оптимальной подсети. Методики обучения таких алгоритмов могут влиять на точность всех моделей в пространстве поиска reinforce.
– Оценка генерализации алгоритмов поиска архитектур на наборе данных NasBench-201 выглядит следующим образом: эволюционный алгоритм > алгоритм reinforce > случайный выбор. Данный порядок совпадает с результатами, полученными на наборе данных NasBench-101. Порядок в оценке генерализации градиентных методов поиска: GDAS [22] > DARTS [3] > ENAS [23] совпадает с оценками, полученными в статье NasBench1shot1 [19]. Значит, результаты, полученные на NasBench-201, могут генерализироваться на другие наборы данных для оценки качества алгоритмов поиска.

3.3. HW-NAS-Bench

С целью расширения границ применимости метода оценки, предложенного авторами [18], авторы [20] расширили имеющуюся базу данных. Основная задача работы заключалась в предоставлении возможности исследователям в области поиска архитектур нейронных сетей, зачастую не имеющим компетенций в работе с устройствами на низком уровне, оценить качество разрабатываемых алгоритмов в условиях ограниченного времени исполнения найденной модели на конечном устройстве.

Для решения поставленной задачи авторы измерили время вычисления нейронных сетей из пространства поиска [18] на устройствах разных классов:

- Компактный графический ускоритель (Nvidia Jetson TX2);
- Одноплатный компьютер для интернета вещей (Raspberry Pi 4);
- Тензорный процессор, разработанный специально для вычисления нейронных сетей (Google TPU);
- Мобильный телефон, имеющий специализированный сопроцессор (Google Pixel 4);
- Интегральная схема специального назначения (ASIC-Eyeriss).

По результатам измерений времени вычисления различных операций исследователи обнаружили, что для разных программно-аппаратных платформ оптимальными являются разные архитектуры. На основе полученного авторами результата можно сделать вывод, что косвенные метрики сложности вычисления модели, например, количество операций сложения и умножения, являются не репрезентативными, если задача состоит в ускорении модели на конкретном устройстве.

Также особенностью методологии, описанной в статье, является использование, помимо пространства поиска для блока нейронной сети, и пространства поиска, которое позволяет находить различные блоки на разных этапах обработки данных. Авторы использовали пространство поиска, описанное в статье [7], которое в оригинальной реализации содержит порядка 10^{23} архитектур. Для этого пространства поиска авторы измерили время вычисления операций, и воспользовались предположением, что сумма времен вычисления операций равна времени вычисления модели. Данное предположение было проверено на 100 случайных архитектурах из пространства [7], и подтверждено высокими коэффициентами корреляции.

Таким образом, авторы привнесли в методологию разработки метода оценки качества алгоритмов поиска привязку эффективности найденных архитектур нейронных сетей к конкретным программно-аппаратным комплексам и создали возможность производить такую оценку эффективности моделей для устройств разных классов. Это может стать фундаментом для появления и развития алгоритмов поиска архитектур нейронных сетей, специализированных под конкретные программно-аппаратные решения.

Однако результаты, описанные в статье, трудно воспроизводимы, так как авторы не зафиксировали версии программного обеспечения, которое было установлено на устройствах во время выполнения измерений.

3.4. NAS-HPO-BENCH

В работе [21] исследователи обращают внимание на гиперпараметры обучения моделей. Основная задача авторов – создание базы данных, в которой сохранена информация об обучении моделей разных архитектур с различными гиперпараметрами. Такая база данных может использоваться другими исследователями для оценки качества собственных алгоритмов поиска гиперпараметров и архитектур моделей. Также на основе данных, собранных в базе, возможен анализ важности гиперпараметров, относящихся к обучению и архитектуре моделей.

Обучение моделей производилось на четырех табличных наборах данных, различных по количеству примеров и количеству признаков в таблице. Однако для всех наборов дан-

ных решалась задача регрессии. В качестве архитектурных гиперпараметров выступали: количество нейронов, вероятность случайного отключения нейронов (dropout), а также функция активации для каждого из двух слоев нейронной сети. В качестве параметров обучения были выбраны: начальный шаг обучения (learning rate), размер пакета для оптимизации (batch size), расписание шага обучения (learning rate scheduler). В общей сложности было обучено 995328 модели, по 62208 моделей для каждого из четырех наборов данных. При этом каждая конфигурация обучалась четыре раза для возможности оценки шума. Каждая модель обучалась в течение 100 эпох при помощи алгоритма оптимизации Adam [24].

В результате анализа собранной в базе данных информации исследователи выяснили, что шум в значениях средней квадратичной ошибки уменьшается с увеличением количества эпох обучения. Для большинства моделей основным параметром по оценке важности методом Functional ANOVA оказался начальный шаг обучения. Однако оценка всех конфигураций не является информативной, так как в практических экспериментах специалистов чаще всего интересуют модели с наименьшей ошибкой. Поэтому был проведен анализ важности гиперпараметров для 1 % и 10 % моделей с наименьшей ошибкой. Также с целью учесть возможные взаимосвязи между гиперпараметрами была проведена оценка важности пар гиперпараметров. По результатам оценки оказалось, что для 1 % моделей с наименьшей ошибкой наиболее важными параметрами является начальный шаг обучения и количество нейронов в первом слое нейронной сети. А для 10 % наиболее точных моделей – начальный шаг обучения и функция активации первого слоя.

Следующим шагом исследователи провели сравнение существующих методов оптимизации гиперпараметров на полученной базе данных. Методы, основанные на байесовской оптимизации, а именно SMAC, TPE и Bohamiann на первых итерациях показывали качество, сравнимое со случайным выбором значений. Однако после накопления статистики, которое произошло примерно в одно время, стали превосходить его. Перечисленные методы при этом достигли разных оптимумов, что объясняется различными моделями (априорными распределениями), используемыми в методах.

Алгоритмы с возможностью оптимальным образом использовать вычислительные ресурсы: Hyperband (HB), Bayesian optimization hyperband (BOHB) показали такое же поведение, как и байесовские методы. В BOHB для оценки полезности примеров использовалась функция, учитывающая взаимосвязи между конфигурациями гиперпараметров, в отличие от TPE. Однако TPE превзошел BOHB и HB по качеству, что исследователи связывают с использованием в BOHB параметров по умолчанию, которые могут быть не оптимальны в данной конкретной задаче, однако превосходят TPE на большом наборе задач [25].

Эволюционному алгоритму потребовалось еще больше итераций, чтобы превзойти качество случайного поиска. Но именно он достиг наилучших результатов из всех оцениваемых алгоритмов. Алгоритм, основанный на обучении с подкреплением, оказался слишком неэффективным по количеству семплирований и лишь немного лучше случайного поиска.

3.5. NAS-HPO-BENCH II

В следующей статье, посвященной оценке алгоритмов поиска архитектур и гиперпараметров обучения моделей [26], авторы используют набор данных CIFAR-10 и поиск оптимальной архитектуры сверточной нейронной сети. В качестве гиперпараметров обучения в данной работе исследователи используют размер пакета (batch size) для оптимизации и шаг обучения (learning rate). При этом производится поиск оптимального блока сети в пространстве из 4 000 вариантов. Авторы выполнили обучение каждой конфигурации 3 раза в течение 12 эпох и обучили модель предсказывать качество после 200 эпох обучения. Итого набор данных об обучениях моделей, который предлагается использовать для оценки качества алгорит-

мов поиска, содержит 192 000 комбинаций гиперпараметров и архитектур и данные о каждом из трех таких обучений.

Для предсказания качества моделей после 200 эпох тренировки авторы обучили еще 4 800 конфигураций. Эти конфигурации использовались как набор данных для обучения суррогатной модели. Структура блока нейронной сети превращалась в вектор при помощи сети изоморфизма графа [27], а представление выбранных параметров обучения как бинарных векторов – при помощи многослойного перцептрона. Полученные векторы параметров обучения и архитектуры модели конкатинировались и подавались на вход другому многослойному персептрону.

Суррогатная модель являлась объединением 10 многослойных перцептронов, обученных на разных подмножествах конфигураций. Также производилась кросс-оценка качества модели на 5 непересекающихся подмножествах. В результате была получена модель, которая решает задачу предсказания качества по архитектуре, параметрам обучения и качеству на двенадцатой эпохе через 200 эпох обучения с коэффициентом детерминации 0,876. Исследователи рекомендуют использовать для предсказания точности после 200 эпох ансамбль моделей, обученных на пяти используемых подмножествах.

В качестве алгоритмов поиска гиперпараметров для тестирования использовалось 6 алгоритмов. Метрикой для оценки качества являлась ошибка на тестовом множестве найденной модели (выбранной архитектуры, обученной с выбранными гиперпараметрами). Приведем алгоритмы в порядке улучшения качества:

1. Байесовский поиск параметров обучения с остановкой обучения наименее перспективных конфигураций и поиск архитектуры через регуляризованную эволюцию;
2. Случайный поиск параметров обучения и поиск архитектуры через регуляризованную эволюцию;
3. Случайных поиск;
4. Алгоритм, основанный на обучении с подкреплением reinforce [12];
5. Байесовский поиск архитектуры и параметров обучения с остановкой обучения наименее перспективных конфигураций;
6. Регуляризованная эволюция.

С этой работы начинаются значительные изменения в методологии оценки качества алгоритмов поиска архитектур (и подбора гиперпараметров), так как здесь появляется идея использования модели для предсказания точности конфигурации. Использование этой идеи позволяет сократить вычислительную сложность разработки наборов данных с информацией о качестве моделей, но в то же время может приводить к неточностям в оценках качества самих алгоритмов поиска.

4. Проблемы оценки качества алгоритмов поиска архитектур нейронных сетей

Современные методы поиска архитектур [6, 7, 28] предполагают извлечение оптимальной подсети из большой обученной модели. Это позволяет снизить стоимость поиска, так как позволяет использовать одну обученную модель для извлечения подсетей с разными характеристиками. При этом происходит глобальный поиск архитектуры сети, т. е. в разных частях модели могут выбраться блоки разной архитектуры. Для оценки качества таких алгоритмов требуются свои наборы данных, отличные от рассмотренных выше. Однако вычислительная сложность создания таких наборов на порядок выше даже для небольших пространств поиска и небольших наборов обучающих данных. Поэтому набирающая популярность идея замены обучения модели на предсказание ее качества в данном случае заслуживает особого внимания. Однако ошибка, вносимая использованием такого метода, может быть слишком большой, и тогда оценка, полученная на таком наборе данных, не является репрезентативной.

Интерес к подбору архитектур сетей специально под вычислительное устройство (программно-аппаратную платформу) может добавлять смещение в процесс оценки алгоритма. Это смещение может быть вызвано процедурой оценки производительности или ограничениями в поддержке операций на целевом устройстве. Разработка набора данных для оценки алгоритмов поиска под специализированное устройство сталкивается с проблемами воспроизводимости, так как устройства могут иметь различия как в программном обеспечении, так и в физической реализации логики на кристаллах процессоров.

Заключение

Анализируя методы оценки качества алгоритмов поиска архитектур нейронных сетей, можно заметить, что появляется тенденция на снижение вычислительной сложности оценки качества. Данная идея реализуется через обучение моделей для предсказания качества архитектур. Проблема снижения вычислительной сложности, а вместе с этим и экологичности машинного обучения, поднимается и в статье [6] и является, безусловно, важной. Возможным решением этой проблемы в задаче оценки качества алгоритмов поиска является использование байесовских методов [25, 29, 30] для выбора архитектур или подбора гиперпараметров.

Гиперпараметры обучения моделей во время выбора архитектуры, а также дообучения выбранной модели играют важную роль и привлекают внимание исследователей. Если пространство возможных архитектур может являться конечным и основываться на возможностях аппаратных вычислителей, то задача поиска гиперпараметров имеет континуальное количество вариантов, что часто требует иных подходов. Дискретизация пространства гиперпараметров является одним из решений данной проблемы, однако при этом теряется общность полученного результата, так как при другой дискретизации выводы исследователей могут измениться.

Задачи, решаемые при помощи нейронных сетей, не ограничиваются обработкой изображений и анализом табличных данных, однако для разных направлений часто используются разные архитектуры. Современным требованием в задачах компьютерного зрения является использование архитектуры трансформеров, изначально разработанной для анализа текстов. Применение поиска нейросетевых архитектур к трансформерам может привести к появлению в некоторой мере универсальных моделей, применимых для данных разной природы. В контексте оценки качества таких алгоритмов поиска стоит отметить, что ввиду большей вычислительной сложности самих моделей потребуются методы, использующие ограниченное количество сравнений архитектур для оценки качества алгоритма поиска.

Список литературы / References

1. **Real E., Moore S., Selle A., Saxena S., Suematsu Y. L., Tan J., Le Q. V., Kurakin A.** Large-Scale Evolution of Image Classifiers [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1703.01041.pdf (дата обращения: 27.12.22).
2. **Kandasamy K., Neiswanger W., Schneider J., Poczos B., Xing E. P.** Neural Architecture Search with Bayesian Optimisation and Optimal Transport [Электронный ресурс]. Режим доступа: https://proceedings.neurips.cc/paper/2018/file/f33ba15effa5c10e873bf3842afb46a6-Paper.pdf (дата обращения: 27.12.22).
3. **Liu H., Simonyan K., Yang Y.** DARTS: DIFFERENTIABLE ARCHITECTURE SEARCH [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1806.09055.pdf (дата обращения: 27.12.22).
4. **Zoph B., Le Q. V.** NEURAL ARCHITECTURE SEARCH WITH REINFORCEMENT LEARNING [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1611.01578.pdf (дата обращения: 27.12.22).

5. **Zoph B., Vasudevan V., Shlens J., Le Q. V.** Learning Transferable Architectures for Scalable Image Recognition [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1707.07012.pdf%27 (дата обращения: 27.12.22).
6. **Cai H., Gan C., Wang T., Zhang Z., Han S.** ONCE-FOR-ALL: TRAIN ONE NETWORK AND SPECIALIZE IT FOR EFFICIENT DEPLOYMENT [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1908.09791.pdf (дата обращения: 27.12.22).
7. **Bichen Wu B., Dai X., Zhang P., Wang Y., Sun F., Wu Y., Tian Y., Vajda P., Jia Y., Keutzer K.** FBNet: Hardware-Aware Efficient ConvNet Designvia Differentiable Neural Architecture Search [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1812.03443.pdf (дата обращения: 27.12.22).
8. **Tan M., Le Q. V.** EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1905.11946.pdf (дата обращения: 27.12.22).
9. **Zoph B., Vasudevan V., Shlens J., Le Q. V.** Learning Transferable Architectures for Scalable Image Recognition [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1707.07012.pdf (дата обращения: 27.12.22).
10. **Krizhevsky A., Sutskever I., Hinton G. E.** ImageNet Classification with Deep Convolutional Neural Networks [Электронный ресурс]. Режим доступа: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (дата обращения: 27.12.22).
11. **Hochreiter S., Schmidhuber J.** LONG SHORT-TERM MEMORY [Электронный ресурс]. Режим доступа: http://www.bioinf.jku.at/publications/older/2604.pdf (дата обращения: 27.12.22).
12. **Williams R. J.** Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning [Электронный ресурс]. Режим доступа: https://link.springer.com/content/pdf/10.1007/BF00992696.pdf?pdf=button (дата обращения: 27.12.22).
13. Dataset Website [Электронный ресурс]. Режим доступа: https://www.cs.toronto.edu/~kriz/cifar.html (дата обращения: 27.12.22).
14. **Huang G., Liu Z., van der Maaten L., Weinberger K. Q.** Densely Connected Convolutional Networks [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1608.06993.pdf (дата обращения: 27.12.22).
15. **Liu C., Zoph B., Neumann M., Shlens J., Hua W., Li L.-J., Fei-Fei L., Yuille A., Huang J., Murphy K.** Progressive Neural Architecture Search [Электронный ресурс]. Режим доступа: https://download.arxiv.org/pdf/1712.00559v3.pdf (дата обращения: 27.12.22).
16. **Sekanina L.** Neural Architecture Search and Hardware Accelerator Co-Search: A Survey [Электронный ресурс]. Режим доступа: https://ieeexplore.ieee.org/document/9606893 (дата обращения: 27.12.22).
17. **Ying C., Klein A., Real E., Christiansen E., Murphy K., Hutter F.** NAS-Bench-101: Towards Reproducible Neural Architecture Search [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1902.09635.pdf (дата обращения: 27.12.22).
18. **Dong X., Yang Y.** NAS-BENCH-201: EXTENDING THE SCOPE OF REPRODUCIBLE NEURAL ARCHITECTURE SEARCH [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/2001.00326.pdf (дата обращения: 27.12.22).
19. **Zela A., Siems J., Hutter F.** NAS-BENCH-1SHOT1: BENCHMARKING AND DISSECTIN ONE-SHOT NEURAL ARCHITECTURE SEARCH [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/2001.10422.pdf (дата обращения: 27.12.22).
20. **Li C., Yu Z., Fu Y., Zhang Y., Zhao Y., You H., Yu Q., Wang Y., Lin Y.** HW-NAS-BENCH: HARDWARE-AWARE NEURAL ARCHITECTURE SEARCH BENCHMARK [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/2103.10584.pdf (дата обращения: 27.12.22).

21. **Klein A., Hutter F.** Tabular Benchmarks for Joint Architecture and Hyperparameter Optimization [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1905.04970.pdf (дата обращения: 27.12.22).
22. **Dong X., Yang Y.** Searching for A Robust Neural Architecture in Four GPU Hours [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1910.04465.pdf (дата обращения: 27.12.22).
23. **Pham H., Guan M. Y., Zoph B., Le Q. V., Dean J.** Efficient Neural Architecture Search via Parameter Sharing [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1802.03268.pdf (дата обращения: 27.12.22).
24. **Kingma D. P., Ba J. L.** ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1412.6980.pdf (дата обращения: 27.12.22).
25. **Falkner S., Klein A., Hutter F.** BOHB: Robust and Efficient Hyperparameter Optimization at Scale [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1807.01774.pdf (дата обращения: 27.12.22).
26. **Hirose Y., Yoshinari N., Shirakawa S.** NAS-HPO-Bench-II: A Benchmark Dataset on Joint Optimization of Convolutional Neural Network Architecture and Training Hyperparameters [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/2110.10165.pdf (дата обращения: 27.12.22).
27. **Xu K., Hu W., Leskovec J., Jegelka S.** HOW POWERFUL ARE GRAPH NEURAL NETWORKS? [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1810.00826.pdf (дата обращения: 27.12.22).
28. **Yu J., Jin P., Liu H., Bender G., Kindermans P.-L. Tan M., Huang T., Song X., Pang R., Le Q.** BigNAS: Scaling Up Neural Architecture Search with Big Single-Stage Models [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/2003.11142.pdf (дата обращения: 27.12.22).
29. **Bergstra J., Bardenet R., Bengio Y., Kegl B.** Algorithms for Hyper-Parameter Optimization [Электронный ресурс]. Режим доступа: https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf (дата обращения: 27.12.22).
30. **Li L., Jamieson K., DeSalvo G., Rostamizadeh A., Talwalkar A.** Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization [Электронный ресурс]. Режим доступа: https://arxiv.org/pdf/1603.06560.pdf (дата обращения: 27.12.22).

Информация об авторе

Щербин Андрей Сергеевич, аспирант кафедры общей информатики факультета информационных технологий Новосибирского государственного университета
SPIN 3542-1054

Information about the Author

Shcherbin S Andrey, Phd student at Common Informatics Chair of the Information Technologies Department of Novosibirsk State University
SPIN 3542-1054

Статья поступила в редакцию 26.03.2023;
одобрена после рецензирования 30.05.2023; принята к публикации 30.05.2023

The article was submitted 09.12.2022;
approved after reviewing 30.05.2023; accepted for publication 30.05.2023