ОПТИМИЗАЦИЯ АРХИТЕКТУРЫ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ НА ОСНОВЕ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

[1]Волков В.К., [2,3]Спицын В.Г.
[1]НИ ТГУ, ИПМКН, гр. 932309, студент, vikontfx@yandex.ru
[2]НИ ТГУ, ИПМКН, д.т.н., профессор каф. ТОИ
[3]НИ ТПУ, ИШИТР, д.т.н., профессор ОИТ

Аннотация

Работа посвящена исследованию объединенного процесса оптимизации архитектуры сверточной нейронной сети и ее отдельных гиперпараметров для задачи классификации при помощи генетического алгоритма. В качестве набора данных рассматривается датасет цветных изображений EuroSAT. Также в работе проведен анализ влияния механизма наследования весов на динамику схождения генетического алгоритма.

Ключевые слова: сверточная нейронная сеть, генетический алгоритм, оптимизация архитектуры, поиск архитектур нейронных сетей.

Введение

Последнее десятилетие ознаменовалось большим количеством прорывов в области глубокого обучения. Однако с увеличением разнообразия используемых архитектур и их сложности, проведение экспериментов по оптимизации или поиску новых вариантов стали трудоёмким. Поэтому начала активно развиваться дисциплина автоматизированного поиска нейросетевых архитектур (Neural Architecture Search, NAS) [1]. Идеи AutoML [2] стали применяться в глубоком обучении. Недавние исследования показывают, что подобная практика позволяет получать модели, обеспечивающие более высокое качество работы, чем альтернативы, разработанные без применения таких методов [3,4]. Помимо оптимизации архитектуры нейронной сети, некоторые методы NAS могут производить оптимизацию гиперпараметров нейронной сети (Hyperparameter Optimisation, HPO)[5]. Процесс NAS+HPO может обеспечивать получение более эффективных архитектур по сравнению с обычным NAS, однако такой поиск заметно сложнее, чем отдельный цикл HPO.

В рамках данной работы задачей было выбрано исследование процесса NAS+HPO при помощи генетического алгоритма (ГА) для сверточных нейронных сетей на датасете EuroSAT с тестированием механизма наследования весов.

Автоматизированный поиск нейросетевых архитектур

Основными элементами NAS являются – пространство поиска, метод оптимизации и метод оценки архитектуры [1]. Согласно [1], пространство поиска задает представление архитектуры и доступные для оптимизации параметры. К основным пространствам поиска относят: макроуровневое пространство поиска (macro search space), пространство поиска с цепочечной структурой (chained structured search space), пространство поиска ячеек (cell based search space), иерархическое пространство поиска (hierarchical search space).

Методы оптимизации архитектур могут в значительной степени различаться в зависимости от задачи, типа используемого пространства, доступности вычислительных ресурсов. Двумя основными группами, по которым разделяются такие методы – оптимизации методами черного ящика (black box optimization) и one-shot оптимизация. К методам оптимизации black box относятся обучение с подкреплением (reinforcement learning), нейро-эволюционные методы, байесовская оптимизация и т. д. Альтернативой black box оптимизации выступает one-shot оптимизация. Такие методы используют понятие гиперсетей (hypernetwork) [6, 7] и суперсетей (supernetwork) [8, 9].

Методы оценки архитектуры – набор различных методов, направленных на раннее определение качества архитектуры. Основными методами являются: экстраполяция кривой обучения [10], обучение на подвыборках данных, zero-cost proxies [11]. Данные методы могут ускорять работу алгоритмов NAS, позволяя избежать необходимости полного цикла обучения модели.

Реализация NAS

Для проведения исследований был разработан собственный вариант NAS, включающий в себя механизм наследования весов и модульную структуру элементов архитектуры. В качестве метода оптимизации использован адаптированный вариант генетического алгоритма, что является распространённой практикой [12-15]. Рассматриваемая реализация обеспечивает процесс NAS+HPO.

Базовым элементом архитектуры является блок – объект, описывающий небольшую часть архитектуры нейронной сети. Общая схема блока представлена на рис. 1. «Основа блока» является универсальной и несет в себе механизмы и унифицированный набор данных, используемый для формирования отдельных слоев нейронной сети. Каждый из параметров является объектом, хранящим соответствующее вещественное число и обеспечивающий его защиту от изменения вне заданного диапазона. Отдельные значения в блоках одного типа можно заблокировать как от изменений со стороны генетического алгоритма, так и от любых изменений, которые могут возникнуть при первоначальной настройке параметров блока. «Сборщик» реализует непосредственный механизм сборки отдельных слоев. Блоки соединяются последовательно, тем самым формируя хромосому, кодирующую архитектуру нейронной сети. Каждый блок может соединяться справа только с тем блоком, который есть в списке разрешенных. Этот механизм обеспечивает корректный вид получаемых архитектур. Закодированные описанным выше образом архитектуры переводятся в последовательность слоев при помощи механизма декодирования.

Рис. 1. Структура блока

Перед процессом декодирования следует обязательный этап валидации архитектуры. При определенных значениях параметров слоя возможно несовпадение размеров конечного или промежуточных тензоров с ожидаемыми. В случае обнаружения несоответствия, архитектуре устанавливается нулевая адаптация, что ведет к ее удалению из популяции перед следующим циклом работы генетического алгоритма.

Также в рамках экспериментов проверялась работа механизма наследования весов (Weight Inheritance, WI). Изначально предложенный в работе Real et al. [16] механизм обеспечивает перенос весов отдельных слоев от родителей к потомкам. Если параметры слоя не изменяются, то его веса и смещения будут использованы в других архитектурах без изменений.

Операторы скрещивания и мутации

Оператор скрещивания (рис. 2) позволяет обмениваться центральными блоками между наборами и изменять в них значения параметров. На первом этапе происходит попытка собрать из имеющихся наборов новые. На втором этапе из центральных блоков наборов извлекаются соответствующие объекты параметров, после чего хранящиеся в них значения декодируются в коды Грэя, которые выравниваются по длине путем добавления незначащих нулей. К выравненным последовательностям применяется «однородный» оператор скрещивания, случайно распределяющий биты между двумя новыми значениями. Новые значения декодируются обратно в вещественное представление, после чего происходит проверка на принадлежность к допустимому диапазону. В случае, если значение выходит за заданный диапазон, то оно заменяется ближайшей границей диапазона.

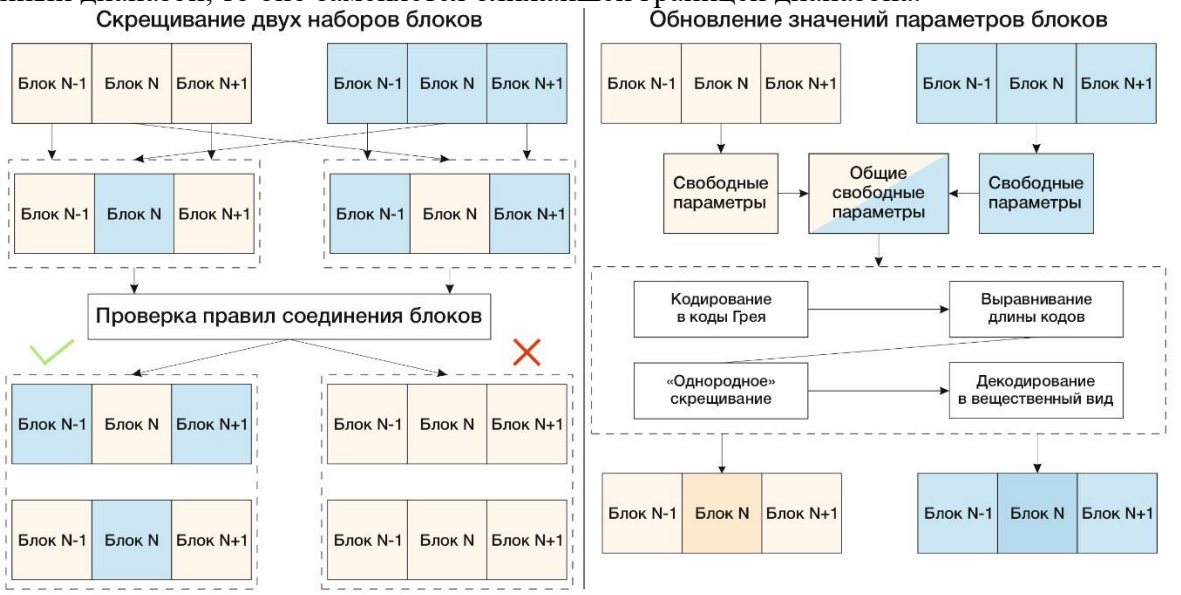

Рис. 2. Работа оператора скрещивания

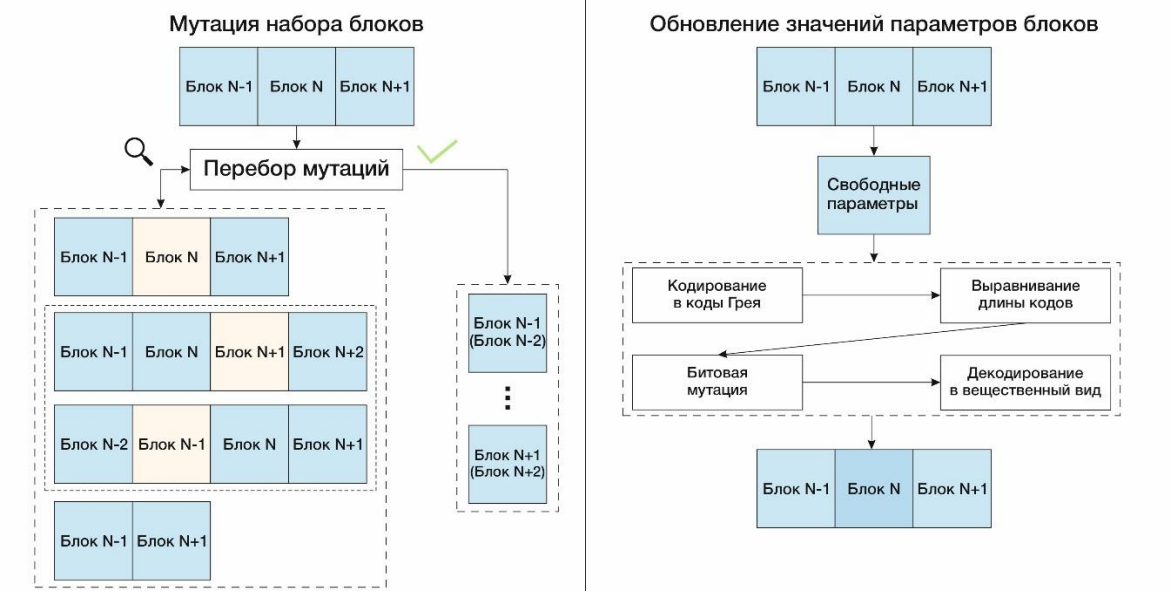

Рис. 3. Работа оператора мутации

Оператор мутации также имеет два этапа работы (рис. 3) и обладает куда более значительными возможностями. На первом этапе данный оператор может изменять тип блока на другой, добавлять новые блоки или удалять уже имеющиеся. На втором этапе оператор мутации работает схожим образом с оператором скрещивания, за исключением этапа

изменения значений - в декодированной последовательности с определенной вероятностью производится изменение отдельных битов на противоположные. Такая работа операторов скрещивания и мутации обеспечивает не только поиск оптимальной архитектуры, но и процесс HPO.

Эксперимент

В качестве набора данных для эксперимента был выбран датасет EuroSAT [17]. Этот датасет состоит из 27000 RGB изображений размера 64x64, распределенных между 10 классами. Имеющиеся исследования [17-18] демонстрируют эффективность сверточных нейронных сетей. В рамках экспериментов изображения использовались в двух формах: исходные и аугментированные. Были использованы следующие аугментации: случайный поворот изображения (до 10 градусов), горизонтальные отражения, изменение яркости (до 20 % от исходного), растяжение/сжатие/сдвиг (не более 10% от исходного размера). Применение каждого из вышеописанных преобразований при обработке изображения равновероятно. Данные были разбиты на обучающую и валидационную выборки в пропорции 80% и 20% соответственно.

Всего было проведено 4 основных эксперимента: с исходными изображениями и выключенным наследованием весов, с аугментированными изображениями и выключенным наследованием весов, с исходными изображениями и включенным наследованием весов.

Чтобы ускорить проведение экспериментов размер популяции был ограничен 20 особями, число циклов работы генетического алгоритма также было ограничено до 20. Для селекции отбиралось 5 архитектур при помощи турнирного отбора с размером турнира равного 2. Все эксперименты проводились при фиксированном значении параметра вероятности мутации 0.15, вероятность добавления нового блока в результате мутации: 0.2, вероятность изменения блока в результате мутации: 0.2, вероятность скрещивания участков хромосомы и соответствующих параметров: 0.5. Каждая из рассматриваемых моделей обучалась в течение 20 эпох. Раз в 5 эпох обучения, модель проверялась на валидационном наборе. Наибольшее значение выбиралось как итоговый уровень точности. Для ускорения сходимости генетического алгоритма используется перенос лучшей модели текущего поколения в следующее.

Виды использованных блоков: LinearReLU – блок, реализующий линейный слой вместе с функцией активации ReLU. Данный блок лежит в основе полносвязной части нейронной сети. В качестве опционального элемента при сборке нейронной сети можно добавить Dropout слой. Диапазон изменения числа нейронов в слое: [1,512]; LinearSoftmax – аналог LinearReLU блока с функцией активации Softmax; ConvReLU – блок, реализующий сверточный слой с функцией активации ReLU. В качестве опциональных элементов выступают слои Dropout и BatchNorm2d. Из данных блоков собрана сверточная часть нейронной сети. Диапазон изменения числа ядер свертки: [1, 128], диапазон изменения размера ядра свертки: [2,5], диапазон изменения padding: [1,1], диапазон изменения stride: [1,2]; Flatten – блок, используемый для соединения сверточной и полносвязной части нейронной сети путем добавления Flatten слоя. Для повышения качества результатов, в каждом из экспериментов использовались слои Dropout и BatchNorm2d.

Одной из задач в проводимых экспериментах является дополнительное изучение работы механизма наследования весов, описанного в [16]. В отличии от оригинальной работы, в экспериментах используется иное пространство поиска, представление архитектуры и иные возможности ее изменения.

Результаты

На рис. 4 представлены данные о максимальной и средней адаптации по популяции в экспериментах без использования аугментации изображений. В таком сценарии обучения

различия между обычным NAS и NAS с наследованием весов слабо выражено. В случае работы механизма WI наблюдается избыточного увеличения размера архитектуры. Такое увеличение не приносит заметного улучшения в точности.

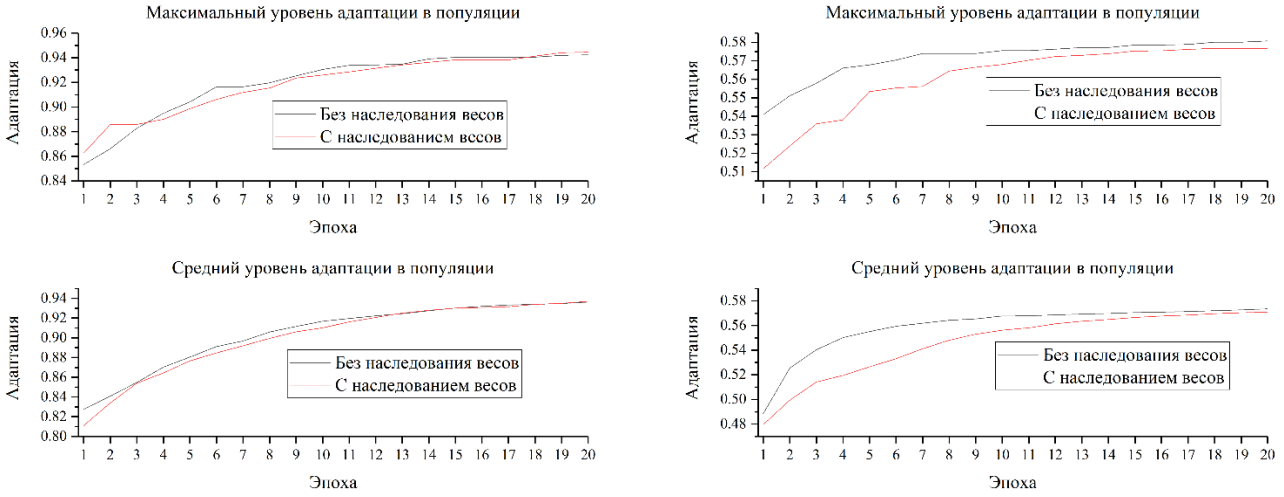

Рис. 4. Динамика изменения адаптации в популяции, слева без аугментации изображений, справа с аугментацией

В случае использования аугментаций динамика заметно изменяется. Механизм WI вызывает снижение скорости схождения генетического алгоритма (рис. 4). Особенно ярко этот эффект выражен для показателя среднего уровня адаптации в популяции. В отличии от предыдущих экспериментов избыточное увеличение архитектуры при таких параметрах не наблюдалось. Среднеквадратичное отклонение (СКО) среднего уровня без использования аугментированных изображений (рис. 5) ниже при использовании механизма WI. При использовании аугментированных изображений наблюдается быстрое уменьшение СКО. Спустя 20 эпох среднеквадратичное отклонение при обычном NAS уменьшилось на ~59%, в то же время при наследовании весов уменьшение составило ~65%. Использование механизма WI замедляет рост точности в популяции, что может быть сигналом снижения эффективности операторов ГА и увеличении шанса преждевременного схождения.

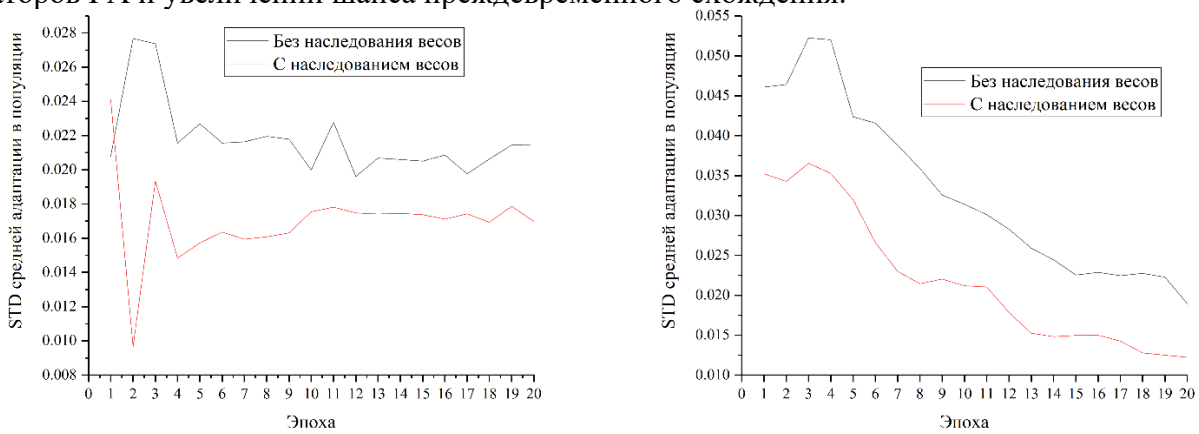

Рис. 5. Динамика изменения среднеквадратичного отклонения средней адаптации в популяции. Слева без аугментации изображений, справа с аугментацией

Заключение

Осуществлена реализация автоматизированного поиска нейросетевых архитектур (NAS) на датасете EuroSAT и проанализирована динамика параметров популяций в различных сценариях работы, в частности с механизмом наследования весов. Полученные

данные свидетельствуют об успешной работе данного вида NAS, однако также в процессе работы были получены данные, свидетельствующие о ряде особенностей работы механизма WI, часть из которых не отмечена в оригинальной работе [16]. Из-за различия в используемых видах пространства поиска, данных и параметрах эксперимента невозможно однозначно определить их источник, данный вопрос нуждается в дальнейшем исследовании.

Список использованных источников

1. White C. и др. Neural Architecture Search: Insights from 1000 Papers // arXiv. – 2023. [Электронный ресурс] – URL: arxiv.org/abs/2301.08727 (дата обращения: 14.11.2024).
2. Salehin I. и др. AutoML: A systematic review on automated machine learning with neural architecture search // Journal of Information and Intelligence. – 2024. – Т. 2, № 1. – С. 52–81.
3. Hu H. и др. Efficient Forward Architecture Search // arXiv. – 2019. – [Электронный ресурс] – URL: arxiv.org/abs/1905.13360 (дата обращения: 14.11.2024).
4. Chen B. и др. MnasFPN: Learning Latency-aware Pyramid Architecture for Object Detection on Mobile Devices // arXiv. – 2020. – [Электронный ресурс] – URL: arxiv.org/abs/1912.01106 (дата обращения: 14.11.2024).
5. Yu T., Zhu H. Hyper-Parameter Optimization: A Review of Algorithms and Applications // arXiv. – 2020. [Электронный ресурс] – URL: arxiv.org/abs/2003.05689 (дата обращения: 14.11.2024).
6. Chauhan V.K. и др. A Brief Review of Hypernetworks in Deep Learning // Artificial Intelligence Review. – 2024. – Т. 57, № 9. – С. 250.
7. Zhang C., Ren M., Urtasun R. Graph HyperNetworks for Neural Architecture Search // arXiv. – 2020. – [Электронный ресурс]. – URL: arxiv.org/abs/1810.05749 (дата обращения: 14.11.2024).
8. Saxena S., Verbeek J. Convolutional Neural Fabrics // arXiv. – 2017. – [Электронный ресурс]. – URL: arxiv.org/abs/1606.02492 (дата обращения: 14.11.2024).
9. Bender G. и др. Understanding and Simplifying One-Shot Architecture Search. – [Электронный ресурс]. – URL: proceedings.mlr.press/v80/bender18a/bender18a.pdf (дата обращения: 14.11.2024).
10. Domhan T., Springenberg J.T., Hutter F. Speeding up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves. – [Электронный ресурс]. – URL: ml.informatik.uni-freiburg.de/wp-content/uploads/papers/15-IJCAI-Extrapolation_of_Learning_Curves.pdf (дата обращения: 14.11.2024).
11. Mellor J. и др. Neural Architecture Search without Training // arXiv. – 2021. – [Электронный ресурс]. – URL: arxiv.org/abs/2006.04647 (дата обращения: 14.11.2024).
12. Gibb S., La H.M., Louis S. A Genetic Algorithm for Convolutional Network Structure Optimization for Concrete Crack Detection // 2018 IEEE Congress on Evolutionary Computation (CEC). – Rio de Janeiro: IEEE, 2018. – С. 1-8.
13. Suganuma M., Shirakawa S., Nagao T. A Genetic Programming Approach to Designing Convolutional Neural Network Architectures // arXiv. – 2017. – [Электронный ресурс]. – URL: arxiv.org/abs/1704.00764 (дата обращения: 14.11.2024).
14. Sun Y. и др. Automatically designing CNN architectures using genetic algorithm for image classification // IEEE Trans. Cybern. – 2020. – Т. 50, № 9. – С. 3840-3854.
15. Xie L., Yuille A. Genetic CNN // 2017 IEEE International Conference on Computer Vision (ICCV). – Venice: IEEE, 2017. – С. 1388-1397.
16. Real E. и др. Large-Scale Evolution of Image Classifiers // arXiv. – 2017. – [Электронный ресурс]. – URL: arxiv.org/abs/1703.01041 (дата обращения: 14.11.2024).
17. Helber P. и др. EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification [Электронный ресурс] // arXiv. – 2019. – [Электронный ресурс]. – URL: arxiv.org/abs/1709.00029 (дата обращения: 14.11.2024).
18. Verma M. и др. Explainable Custom CNN Architecture for Land Use Classification using Satellite Images // 2021 Sixth International Conference on Image Information Processing (ICIIP). – Shimla, India: IEEE, 2021. – С. 304-309.