УДК 004.32
DOI: 10.15593/2224-9397/2020.4.07

К.Р. Ахметзянов, А.И. Тур, А.Н. Кокоулин, А.А. Южаков

Пермский национальный исследовательский политехнический университет,
Пермь, Россия

ОПТИМИЗАЦИЯ ВЫЧИСЛЕНИЙ НЕЙРОННОЙ СЕТИ

Микрокомпьютеры обладают ограниченными вычислительными мощностями (производительность процессора и количество памяти). При запуске нейронной сети на таком компьютере производительность снижается по сравнению с персональными и настольными компьютерами. Для устранения этого недостатка применяются различные методы оптимизации вычислений нейронных сетей, в частности квантование. Квантование уменьшает точность чисел, в которых хранятся весовые коэффициенты нейронных сетей, до 8-битных целых чисел или до 16-битных чисел с плавающей точкой с небольшими потерями или вовсе без потери точности нейронной сети, что приводит к уменьшению объема памяти, занимаемой моделью сети. Квантование позволяет оптимизировать вычисления под определенный вид процессора (CPU, GPU, TPU), но в этой статье будет рассмотрена оптимизация только под CPU. **Цель исследования:** уменьшение размера файла весовых коэффициентов нейронной сети. **Методы и результаты исследования:** сформулирована оптимизационная задача, заключающаяся в нахождении функции преобразования весовых коэффициентов, для которой достигается минимум суммы относительных изменений размера файла весовых коэффициентов, времени обработки изображения, точности нейронной сети. Проведены эксперименты с различными типами квантования (квантование динамического диапазона, квантование с использованием репрезентативного набора данных, целочисленное квантование с использованием репрезентативного набора данных, квантование во float16). Показаны вычислительные блоки и изменения в архитектуре нейронной сети, которые вносит каждый из рассмотренных типов квантования. Проведен сравнительный анализ типов квантования с различными характеристиками (точность нейронной сети после применения квантования, количество времени, затрачиваемое нейронной сетью на обработку одного изображения, размер файла с весовыми коэффициентами нейронной сети). Выбран метод квантования, для которого достигается минимум оптимизационной функции. **Практическая значимость:** уменьшение времени обновления нейронной сети на удаленном микрокомпьютере. В итоге выбрано квантование во float16, так как с его помощью количество занимаемой памяти уменьшается вдвое, а точность нейронной сети и производительность остаются без изменений.

Ключевые слова: нейронная сеть, оптимизация вычислений, квантование.

K.R. Akhmetzyanov, A.I. Tur, A.N. Kokoulin, A.A. Yuzhakov

Perm National Research Polytechnic University, Perm, Russian Federation

OPTIMIZATION OF NEURAL NETWORK COMPUTATION

Microcomputers have computational limitations (processor performance and amount of memory). Running a neural network on such a computer has a lower performance compared to personal and desktop computers. To eliminate this drawback, various methods for optimizing computations of neural networks, in particular, quantization are used. Quantization reduces the precision of numbers that store neural network weights to 8-bit integers or 16-bit floating point numbers with little or no reduction in neural network accuracy, but at the same time reducing the amount of memory used. Quantization makes it possible to optimize computations for a certain type of processor (CPU, GPU and TPU), but this article will only consider optimization for the CPU. **Purpose:** reducing the file size of the neural network weights. **Results:** 1) the formulated optimization problem is finding the function of transforming the weights, that reaches the minimum sum of the relative changes in the file size of the weight, the latency, and the accuracy of the neural network, 2) conducted experiments with various types of quantization (dynamic range quantization, quantization using a representative dataset, integer quantization using a representative dataset and float16 quantization), 3) showing computational units and changes in the neural network for the considered types of quantization, 4) a comparative table for these quantizations with various characteristics (the accuracy of the neural network after applying quantization, the neural network latency and the size of neural network weight file), 5) the chosen quantization method reaching the minimum of the optimization function. **Practical relevance:** reducing the time for updating the neural network on a remote microcomputer. As a result, we chose float16 quantization, since it halves the amount of memory used, and the accuracy of the neural network and performance remains unchanged.

Keywords: neural network, computation optimization, quantization.

Введение. В проекте по созданию аппарата по сортировке твердых коммунальных отходов разработано устройство, состоящее из нескольких компонентов: микрокомпьютер, видеокамера и механическая часть. На микрокомпьютере запускается нейронная сеть, которая на основе поступающей входной информации с видеокамер принимает решение, какой вид мусора находится на изображении, затем механическая часть выполняет действия по сортировке на основании этого решения. Микрокомпьютер обладает ограниченной вычислительной мощностью. По этой причине к нейронной сети помимо высокой точности предъявляется дополнительное требование – высокое быстродействие и малый объем занимаемой памяти. Проблема заключается в преобразовании обученной нейронной сети таким образом, чтобы ее быстродействие увеличилось, размер модели нейронной сети уменьшился, а точность при этом осталась прежней или уменьшилась незначительно. Для оценки точности используется метрика accuracy, для оценки быстродействия – количество миллисекунд, затрачиваемое нейронной сетью на обработку одного изображения. Поскольку в по-

следнее время обученные нейронные сети часто запускаются на микрокомпьютерах и встраиваемых системах, например, беспилотных автомобилях [1] и роботах [2], рассматриваемая проблема является достаточно актуальной.

Постановка цели и задач исследования. Цель работы заключается в поиске такого преобразования весовых коэффициентов нейронной сети, при котором достигаются наименьшие размер файла модели нейронной сети и время обработки изображения, а точность на тестовой выборке остается как можно более близкой к точности до применения преобразования. Для достижения этой цели необходимо выполнить следующие задачи: осуществить постановку задачи оптимизации; определить множество функций преобразования весовых коэффициентов; получить значения оптимизационной функций для сформированного множества функций; выбрать функцию преобразования. Новизна исследования состоит в применении многокритериальной оптимизации для поиска функции преобразования весовых коэффициентов нейронной сети на основе методов квантования, которые обладают возможностью использования без необходимости заново обучать нейронную сеть. Предлагаемый в данной статье метод сформулирован в общем виде и является универсальным, т.е. применим не только в рамках рассматриваемой задачи выбора нейронной сети для аппарата по сортировке твердых коммунальных отходов, но и для других нейронных сетей. Метод является агностическим, т.е. подходит для нескольких видов задач: как для классификации, так и для регрессии. В данной статье описываются принципы использования предлагаемого метода для задачи классификации. При его использовании для решения задач регрессии необходимо заменить используемую метрику классификации (точность) на метрику, применяемую в регрессионном анализе (например, *MSE*).

Для решения задач, описанных выше, выполняется минимизация следующей целевой функции:

$$L(U,F) = \frac{S(U,F) - S(U_0,F)}{S(U_0,F)} + \frac{T(U,F) - T(U_0,F)}{T(U_0,F)} + \\ + \frac{A(U,F) - A(U_0,F)}{A(U_0,F)} = \Delta_r S(U,F) + \Delta_r T(U,F) + \Delta_r A(U,F), \quad (1)$$

где F – функция преобразования входных данных в распределение вероятностей принадлежности к определенному классу объекту (в этой

статье такой функцией является нейронная сеть), U – функция изменения параметров F (в этой статье такими параметрами являются весовые коэффициенты нейронной сети), U_0 – отсутствие изменения весовых коэффициентов, S – размер файла весовых коэффициентов нейронной сети, T – время обработки одного изображения нейронной сетью, A – точность нейронной сети на тестовой выборке.

Задача оптимизации (1) является безусловной и состоит в нахождении функции изменения весовых коэффициентов U, при которой значение оптимизационной функции L достигает минимума:

$$U = \operatorname{argmin}_{U \in \mathcal{U}} L(U, F), \tag{2}$$

где $\mathcal{U}$ – множество функций изменения весовых коэффициентов нейронной сети.

Для выполнения преобразования весовых коэффициентов предлагается использовать квантование [3], которое применяется в работах [4–8] для сжатия нейронных сетей.

1. Описание ранее проведенных экспериментов. Учитывая результаты ранее проведенных экспериментов по выбору архитектуры нейронной сети, в работе предлагается использовать архитектуру MobileNet [9], которая относится к сетям глубокого обучения, так как она дает наибольшую точность среди других рассмотренных архитектур [10]. Результаты экспериментов по увеличению точности нейронной сети предполагают использование принципа аугментации данных [11]. Аугментация данных – это дополнение обучающей выборки изображениями с различными геометрическими и цветовыми изменениями [12–15] (например, поворот, отражение, изменение яркости, контрастности и т.д.)

2. Обучение нейронной сети. В текущих исследованиях обучающая выборка отличается от той, на которой была обучена нейронная сеть в наших предыдущих исследованиях (отличия заключаются в используемых камерах, освещении и количестве классов). Поэтому для данного исследования нейронная сеть была заново обучена. База фотографий содержит 1700 изображений бутылок и банок, сделанных в разрабатываемом аппарате по сортировке твердых коммунальных отходов. База изображений разделена по материалу предмета и категории, что в общей сложности составляет пять классов. Вся база фотографий поделена на обучающую и тестовую выборки в соотношении 80/20. Для обучения использовалось облачное вычисление Cloud TPU

[16] и фреймворк TensorFlow [17]. Для оптимизации нейронной сети использовались оптимизатор Adam [18] и циклическое обучение [19]. Нейронная сеть на тестовой выборке достигла точности 97 %.

3. Квантование нейронной сети. После обучения нейронной сети выполнено ее квантование. Квантование – это уменьшение точности чисел весовых коэффициентов моделей, при котором достигается наименьшая потеря в точности модели. Из [3] квантование вещественного числа в целочисленное производится по следующей формуле:

$$q_3^{(i,k)} = Z_3 + \sum_{i=1}^{N} a^{(i,j)} b^{(j,k)} = Z_3 + \sum_{i=1}^{N} \left(q_1^{(i,j)} - Z_1\right)\left(q_2^{(j,k)} - Z_2\right),$$

где q_3 – исходное вещественное число, a – квантованные целочисленные значения, b – весовые коэффициенты квантования, q_1 и q_2 – квантованные целочисленные значения, Z_1, Z_2 и Z_3 – смещение в нулевую точку для q_1, q_2 и q_3 соответственно.

Для квантования использовалась программа Tensor Flow Lite [20]. Проверка нейронных сетей производилась на CPU, и в качестве входных данных использовалась тестовая выборка, описанная ранее. Для визуализации архитектуры нейронной сети использовалась программа Netron [21].

Выполнена оптимизация сети по нескольким видам квантования: квантование динамического диапазона (U_1), квантование с использование репрезентативного набора данных (U_2), целочисленное квантование с использованием репрезентативного набора данных (U_3) и квантование во float16 (U_4), т.е. оптимизация целевой функции L производится среди множества $\mathcal{U} = \{U_0, U_1, U_2, U_3\}$. Далее показаны те изменения, к которым приводит определенный тип квантования.

На рис. 1 показаны типы данных для входа исходной нейронной сети до квантования. Как видно из рисунка, входным типом данных является float32, и все скрытые слои также используют его.

Проведено квантование динамического диапазона. После этого тип данных, в котором хранятся свертки нейронной сети, поменялся с float32 на int8. Также в архитектуру добавляются блоки деквантования, которые меняют тип данных int8 на float32, который используется для вычислений внутри нейронной сети. На рис. 2 показана архитектура квантованной сети с динамическим диапазоном.

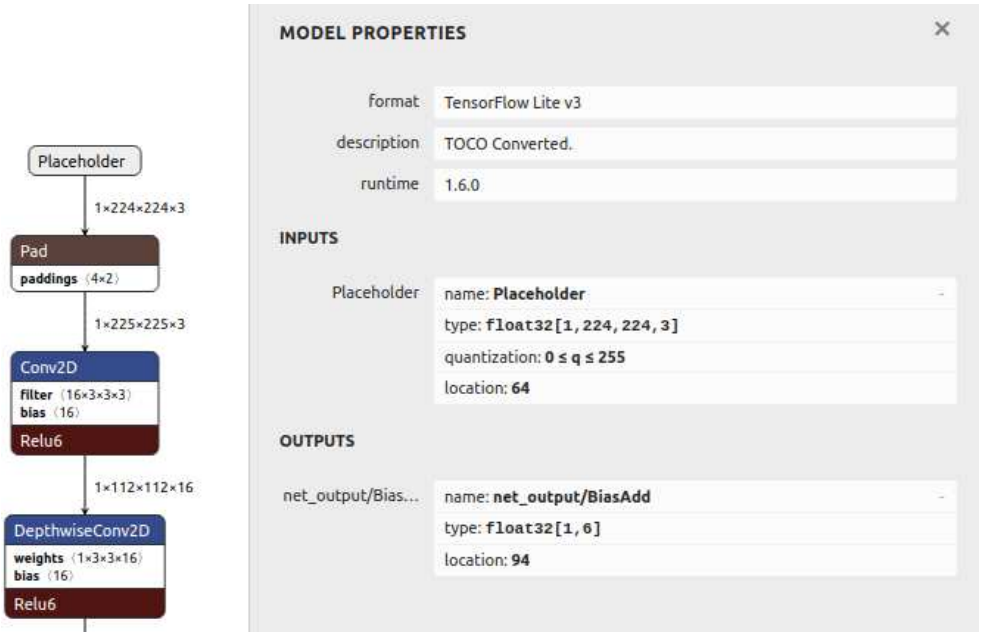

Рис. 1. Вход исходной нейронной сети

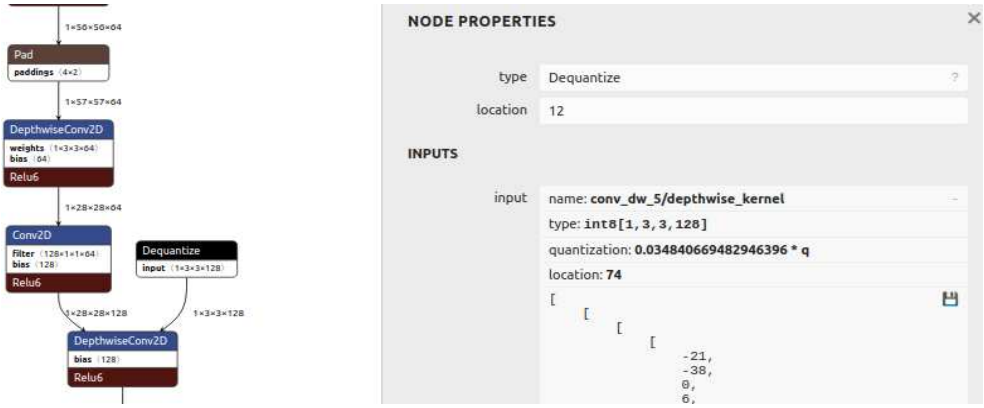

Рис. 2. Добавление блока деквантования для сверток
после квантования динамического диапазона

После применения квантования с использованием репрезентативного набора данных внутри нейронной сети тип данных – int8, а в начале архитектуры добавляется блок квантования. На рис. 3 показаны свойства свертки после применения этого типа квантования.

После применения целочисленного квантования с использованием репрезентативного набора данных внутри нейронной сети тот же тип данных – int8, что и в предыдущем случае, но отличие заключается

в используемом входном типе данных uint8 (только этот тип данных допустим в качестве входа в TPU и микроконтроллеров [22]). На рис. 4 показана характеристика для входного узла нейронной сети.

Рис. 3. Изменение точности чисел сверток после квантования с репрезентативным набором данных

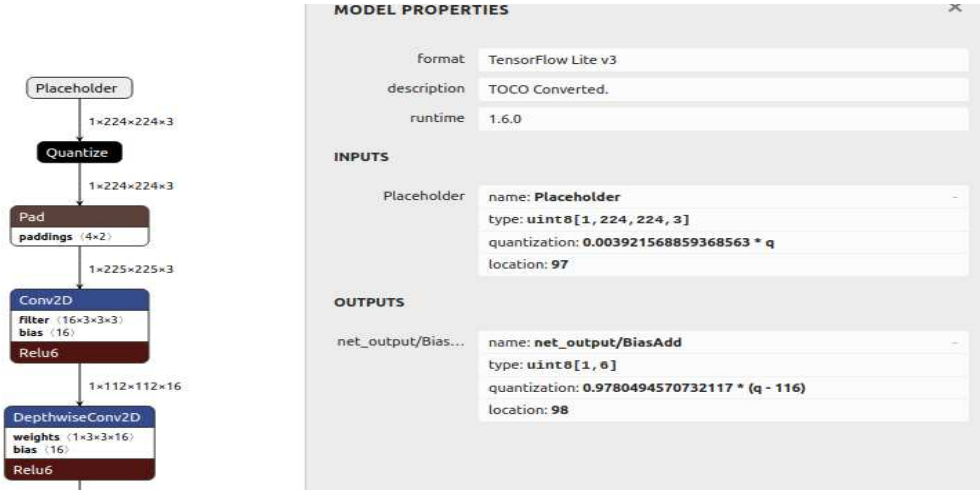

Рис. 4. Изменение точности чисел входа нейронной сети после целочисленного квантования с репрезентативным набором данных

При использовании квантования float16 в нейронную сеть добавляются блоки деквантования как для сверток, так и для смещений (bias), а сами весовые коэффициенты хранятся во float16. Использование float16 позволяет увеличить скорость нейронной сети на GPU, но

на CPU скорость остается прежней, так как на CPU float16 преобразуется до float32 [23] (рис. 5).

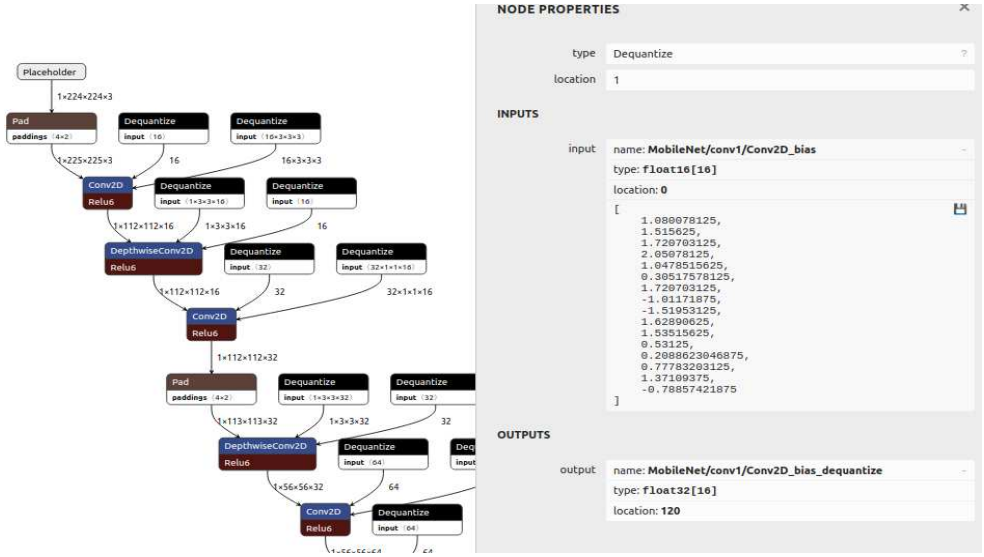

Рис. 5. Добавление блоков деквантования для весовых коэффициентов нейронной сети перед сверточными слоями после квантования во float16

Полученные после проведения квантования нейронные сети проверены на тестовой и обучающей выборках. Для этих сетей получены следующие характеристики: точность на тестовой выборке, точность на обучающей выборке, среднее время обработки одного изображения и размер файла модели, а также вычислены значения функции L по формуле (1) относительно характеристик первоначальной нейронной сети до квантования. Использование квантования влияет на различные характеристики получаемой нейронной сети. В таблице показано сравнение исходной нейронной сети со всеми типами квантования, описанными выше.

Как видно из таблицы, по сравнению с исходной моделью использование квантования динамического диапазона уменьшает точность на тестовой выборке примерно на 9 %, на обучающей выборке – на 7 %, увеличивает время обработки изображения примерно в два раза и уменьшает размер файла модели на 74 %. Использование репрезентативного набора данных оставляет точность без изменений, увеличивая время обработки на 2000 % и уменьшая размер файла модели на 69 %. Использование репрезентативного набора данных с целочисленным

входом уменьшает точность на тестовой выборке на 0,1 % с сохранением примерно той же точности на обучающей выборке, увеличением времени обработки на 2000 % и уменьшением размера файла модели на 69 %. Использование float16 оставляет точность и время на обработку изображений без изменений. Размер файла модели меньше на 50 %.

Характеристики нейронной сети после каждого типа квантования

Данные	Точность на тестовой выборке, %	Точность на обучающей выборке, %	Время/изображение, мс	Размер файла, КБ	L
Исходная модель (U_0)	99,18	99,92	23	3200	0,00
Динамический диапазон (U_1)	90,26	92,12	47	834	0,21
Использование набора данных (U_2)	99,18	99,92	483	999	19,31
Использование набора данных (целочисленный вход) (U_3)	99,08	99,94	486	1000	19,44
Использование float16 (U_4)	99,18	99,92	23	1612	–0,50

Таким образом, выбрано квантование во float16 (U_4), при котором достигается наименьшее значение (–0,50) оптимизируемой функции L среди множества $\mathcal{U}$.

Выводы. Проведены эксперименты по выбору метода квантования нейронной сети для уменьшения занимаемой памяти с минимальными потерями точности и скорости. Среди нескольких типов квантования (квантование динамического диапазона, квантование с репрезентативным набором данных, целочисленное квантование с репрезентативным набором данных, квантование во float16) выбран тип квантования float16, так, с его помощью файл весовых коэффициентов уменьшается на 50 %, а точность нейронной сети и скорость остаются прежними. Таким образом, практической значимостью результата является уменьшение файла модели нейронной сети на 50 % и, как следствие, уменьшение времени обновления нейронной при передаче этого файла через Интернет в аппарат по сортировке твердых коммунальных

отходов, а также значимой является возможность применения выбранного метода квантования для других обученных нейронных сетей.

В перспективе предполагаются следующие направления дальнейших исследований в рамках рассматриваемой тематики:

1. Применение другого метода квантования – квантование во время обучения.

2. Использование других методов оптимизации вычислений нейронной сети – кластеризация весовых коэффициентов и сокращение избыточных нейронов.

Исследование выполнено при финансовой поддержке правительства Пермского края в рамках научного проекта № С26/174.6.

Библиографический список

1. End to end learning for self-driving cars / M. Bojarski [et al.] // arXiv preprint arXiv:1604.07316. – 2016.

2. Bekey G.A., Goldberg K.Y. (eds.). Neural networks in robotics // Springer Science & Business Media. – 2012. – Vol. 202.

3. Quantization and training of neural networks for efficient integer-arithmetic-only inference / B. Jacob [et al.] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. – 2018. – P. 2704–2713.

4. Han S., Mao H., Dally W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding // arXiv preprint arXiv:1510.00149. – 2015.

5. Low-bit quantization of neural networks for efficient inference / Y. Choukroun [et al.] // 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW). IEEE. – 2019. – P. 3009–3018.

6. Haq: Hardware-aware automated quantization with mixed precision / K. Wang [et al.] // Proceedings of the IEEE conference on computer vision and pattern recognition. – 2019. – P. 8612–8620.

7. Quantized convolutional neural networks for mobile devices / J. Wu [et al.] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. – 2016. – P. 4820–4828.

8. Krishnamoorthi R. Quantizing deep convolutional networks for efficient inference: A whitepaper: arXiv preprint arXiv:1806.08342. – 2018.

9. Mobilenets: Efficient convolutional neural networks for mobile vision applications / A.G. Howard [et al.] // arXiv preprint arXiv:1704.04861. – 2017.

10. Ахметзянов К.Р., Южаков А.А. Сравнение сверточных нейронных сетей для задач сортировки мусорных отходов // Известия СПбГЭТУ ЛЭТИ. – 2018. – № 6. – С. 27–32.

11. Ахметзянов К.Р., Южаков А.А. Увеличение точности сверточной нейронной сети за счет возрастания количества данных // Нейрокомпьютеры: разработка, применение. – 2018. – № 7. – С. 14–19.

12. Perez L., Wang J. The effectiveness of data augmentation in image classification using deep learning // arXiv preprint arXiv:1712.04621. – 2017.

13. Vasconcelos C.N., Vasconcelos B.N. Convolutional neural network committees for melanoma classification with classical and expert knowledge based image transforms data augmentation // arXiv preprint arXiv:1702.07025. – 2017.

14. Random Erasing Data Augmentation / Z. Zhong [et al.] // AAAI. – 2020. – P. 13001–13008.

15. Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition // arXiv preprint arXiv:1409.1556. – 2014.

16. Cloud TPU [Электронный ресурс]. – URL: https://cloud.google.com/tpu (дата обращения: 05.09.2020).

17. TensorFlow [Электронный ресурс]. – URL: https://www.tensorflow.org/ (дата обращения: 05.09.2020).

18. Kingma D.P., Ba J. Adam: A method for stochastic optimization // arXiv preprint arXiv:1412.6980. – 2014.

19. Smith L.N. Cyclical learning rates for training neural networks // IEEE Winter Conference on Applications of Computer Vision (WACV). – IEEE. – 2017. – P. 464–472.

20. ML for Mobile and Edge Devices – TensorFlow Lite [Электронный ресурс]. – URL: https://www.tensorflow.org/lite (дата обращения: 05.09.2020).

21. Netron [Электронный ресурс]. – URL: https://lutzroeder.github.io/netron/ (дата обращения: 05.09.2020).

22. Post-training integer quantization [Электронный ресурс]. – URL: https://www.tensorflow.org/lite/performance/post_training_integer_quant (дата обращения: 05.09.2020).

23. Post-training float16 quantization [Электронный ресурс]. – URL: https://www.tensorflow.org/lite/performance/post_training_float16_quant (дата обращения: 05.09.2020).

References

1. Bojarski M. et al. End to end learning for self-driving cars. a*rXiv preprint arXiv:1604.07316*, 2016.

2. Bekey G.A., Goldberg K.Y. (eds.). Neural networks in robotics. *Springer Science & Business Media*, 2012, vol. 202.

3. Jacob B. et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2704-2713.

4. Han S., Mao H., Dally W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

5. Choukroun Y. et al. Low-bit quantization of neural networks for efficient inference. *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 2019, pp. 3009-3018.

6. Wang K. et al. Haq: Hardware-aware automated quantization with mixed precision. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 8612-8620.

7. Wu J. et al. Quantized convolutional neural networks for mobile devices. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4820-4828.

8. Krishnamoorthi R. Quantizing deep convolutional networks for efficient inference: A whitepaper: arXiv preprint arXiv:1806.08342, 2018.

9. Howard A.G. et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

10. Akhmetzianov K.R., Iuzhakov A.A. Sravnenie svertochnykh neironnykh setei dlia zadach sortirovki musornykh otkhodov [Convolutional neural networks comparison for waste sorting tasks]. *Izvestiia Sankt-Peterburgskogo gosudarstvennogo elektrotekhnicheskogo universitета "ЛЭТИ"*, 2018, no. 6, pp. 27-32.

11. Akhmetzianov K.R., Iuzhakov A.A. Uvelichenie tochnosti svertochnoi neironnoi seti za schet vozrastaniia kolichestva dannykh [Increasing the accuracy of convolutional neural networks by using data augmentation techniques]. *Neirokomp'iutery: razrabotka, primenenie*, 2018, no. 7, pp. 14-19.

12. Perez L., Wang J. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.

13. Vasconcelos C.N., Vasconcelos B.N. Convolutional neural network committees for melanoma classification with classical and expert

knowledge based image transforms data augmentation. *arXiv preprint arXiv:1702.07025*, 2017.

14. Zhong Z. et al. Random Erasing Data Augmentation. *AAAI*, 2020, pp. 13001-13008.

15. Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

16. Cloud TPU, available at: https://cloud.google.com/tpu (accessed 5 September 2020).

17. TensorFlow, available at: https://www.tensorflow.org/ (accessed 5 September 2020).

18. Kingma D.P., Ba J. Adam: A method for stochastic optimization. a*rXiv preprint arXiv:1412.6980*, 2014.

19. Smith L.N. Cyclical learning rates for training neural networks. *IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 464-472.

20. ML for Mobile and Edge Devices - TensorFlow Lite, available at: https://www.tensorflow.org/lite (accessed 5 September 2020).

21. Netron, available at: https://lutzroeder.github.io/netron/ (accessed 5 September 2020).

22. Post-training integer quantization, available at: https://www.tensorflow.org/lite/performance/post_training_integer_quant (accessed 5 September 2020).

23. Post-training float16 quantization, available at: https://www.tensorflow.org/lite/performance/post_training_float16_quant (accessed 5 September 2020).

Сведения об авторах

Ахметзянов Кирилл Раисович (Пермь, Россия) – аспирант кафедры «Автоматика и телемеханика» Пермского национального исследовательского политехнического университета (614990, Пермь, Комсомольский пр., 29, e-mail: kirill94a@mail.ru).

Тур Александр Игоревич (Пермь, Россия) – аспирант кафедры «Автоматика и телемеханика» Пермского национального исследовательского политехнического университета (614990, Пермь, Комсомольский пр., 29, e-mail: tur.aleksandr93@mail.ru).

Кокоулин Андрей Николаевич (Пермь, Россия) – кандидат технических наук, доцент кафедры «Автоматика и телемеханика» Пермского национального исследовательского политехнического университета (614990, Пермь, Комсомольский пр., 29, e-mail: a.n.kokoulin@at.pstu.ru).

Южаков Александр Анатольевич (Пермь, Россия) – доктор технических наук, профессор, заведующий кафедрой «Автоматика и телемеханика» Пермского национального исследовательского политехнического университета (614990, Пермь, Комсомольский пр., 29, e-mail: uz@at.pstu.ru).

About the authors

Akhmetzyanov Kirill Raisovich (Perm, Russian Federation) is a Graduate Student of the Department Automatic and Telemechanic Perm National Research Polytechnic University (614990, Perm, 29, Komsomolsky pr., e-mail: kirill94a@mail.ru).

Tur Alexander Igorevich (Perm, Russian Federation) is a Graduate Student of the Department Automatic and Telemechanic Perm National Research Polytechnic University (614990, Perm, 29, Komsomolsky pr., e-mail: tur.aleksandr93@mail.ru).

Kokoulin Andrey Nikolaevich (Perm, Russian Federation) is a Ph. D. in Technical Sciences, Associate Professor of the Department Automatic and Telemechanic Perm National Research Polytechnic University (614990, Perm, 29, Komsomolsky pr., e-mail: a.n.kokoulin@at.pstu.ru).

Yuzhakov Aleksandr Anatolyevich (Perm, Russian Federation) is a Doctor of Technical Sciences, Professor, Head of Department Automatic and Telemechanic Perm National Research Polytechnic University (614990, Perm, 29, Komsomolsky pr., e-mail: uz@at.pstu.ru).

Получено 07.10.2020