Volodymyr Rusinov, Oleksii Cherevatenko

METHOD OF NEURAL NETWORK TRAINING FOR EDGE ARCHITECTURE

The article analyzes the issue of heterogeneous CPU-GPU system use in accelerating applied tasks, related to the neural network learning process.

Keywords: edge computing, neural networks, machine learning, CPU, GPU.

Fig.: 3. Tab.: 0. Bibl.: 7.

Relevance of the research topic. The current state of embedded computing and IoT presence creates an opening for the development of intelligent systems based on the idea of edge computing. Thanks to the recent advances in wireless technology, multiple devices that can work independently can form a single computing unit, not so different from supercomputers, albeit on a smaller scale. [1] Being small, power-efficient devices, some problems should be addressed, for example, such devices are not very powerful, latency can be disruptive, a specific setup is required and all of this directly influences the user experience.

Target setting. After the advent of neural networks, AI-enabled applications are becoming more widespread every day. At the same time, IoT devices are becoming more powerful and power-efficient. New research shows that when coupled IoT devices can reach high performance, allowing them to train machine learning models and run them more seamlessly than by using dedicated servers.

Actual scientific researches and issues analysis. Numerous studies have been published exploring how to run machine learning-enabled tasks on embedded devices and edge nodes. Approaches range from using one designated device to train a model and distribute the finished model over the network to creating SDN-powered nodes to distribute the learning process over the devices. This subject is quite novel, therefore no ubiquitous solution has yet been proposed.

Uninvestigated parts of general matters defining. Edge computing and machine learning are both relatively new technologies that are constantly evolving. Many of the issues are still unresolved or require additional investigation. This article presents one of the methods to resolve the issue of consumption of time and power during the training process, thereby allowing to deploy AI models faster by distributing work across multiple devices.

The research objective. The goal of this article is to develop a method that speeds up the training process significantly by distributing the workload across edge

devices. Also, the “cut” model will be tested to determine whether it is feasible to use only part of the model for faster deployment of AI models.

The statement of basic materials. Deep Neural Networks are the most commonly used way to adopt machine learning. Since their first introduction, as a perceptron, DNNs have developed into a versatile tool for absorbing patterns in massive amounts of data. [2] There is a range of different ways to compose a DNN, some of them have names like AlexNet or GoogleNet. Unfortunately, DNNs demand high computational power, and embedded devices are generally tuned for relatively simple tasks. The most prominent example of DNNs is convolutional neural networks (CNN) known for their high level of performance when working with visual input. [3]

Recent developments in IoT and ARM architecture saw IoT devices such as Raspberry Pi 4 run desktop OSes and applications on par with desktop PCs. On top of that, there are IoT devices that sport a GPU or a TPU onboard for tasks like machine learning or fast processing of video input. Recently, ARM-based CPUs have been able to perform on par and better than existing x86-based CPUs in various tests, among which is video processing.

However, to be power efficient, chips must have a TDP of around 5 to 15 W at a high load, therefore limiting their capabilities. To overcome problems presented by low-power SoCs on IoT devices, connecting several devices can be useful in increasing overall productivity. This approach is called edge computing, and it aims to exploit the growing amount of IoT devices. Coupled with recent developments in SDN technology it is possible to achieve low latency and fast data delivery at a low cost.

Edge computing has advantages over cloud computing or using specialized equipment, making it at least an alternative to these approaches. Edge nodes are usually close to their target datalakes. This leads to two important points of consideration: secure access to the data and decreased latency. Also, edge provides services at an affordable cost. As mentioned earlier, IoT devices can run desktop OSes, which in turn provide access to many third-party libraries as their code need not be translated for a specific device, therefore creating a level of abstraction increasing both flexibility and sustainability.

There are multiple ways to construct an AI-enabled architecture. [4] With many articles devoted to the creation of NN models in resource-constraint environments, effort goes into NN design for embedded devices. Some of the ideas like pruning and quantization proved to be efficient when used in training on low-powered devices. [5] However NN models have to be small thereby limiting their flexibility.

The suggested solution is to distribute the model over the edge devices. This will enable fast delivery of AI on the edge device, by letting the other parts of the edge train on the output of the model present. Suppose we have a random fully-connected, feedforward DNN, if we take the output of any layer, we can use it later on the next layer with the only rule being that the next layer accepts the output from the former layer.

To distribute the learning process over the devices, we use SDN-powered nodes. Using of SDN technology allows us to increase network capabilities such as traffic, capacity, number of nodes. This would come handy in computer processing when we need to transfer big amounts of data between devices. [6]

To test the mentioned solution we first need to choose a tested model. For that purpose, we take MobileNet, which performs very well on IoT devices. Architecturally it is also very streamlined, having multiple blocks consisting of convolution layer, depthwise convolution, and pointwise convolution layers, together forming potent depthwise separable convolution blocks. By adding a dense layer at the end of such block it can be used to train separately from the rest of the model. [7]

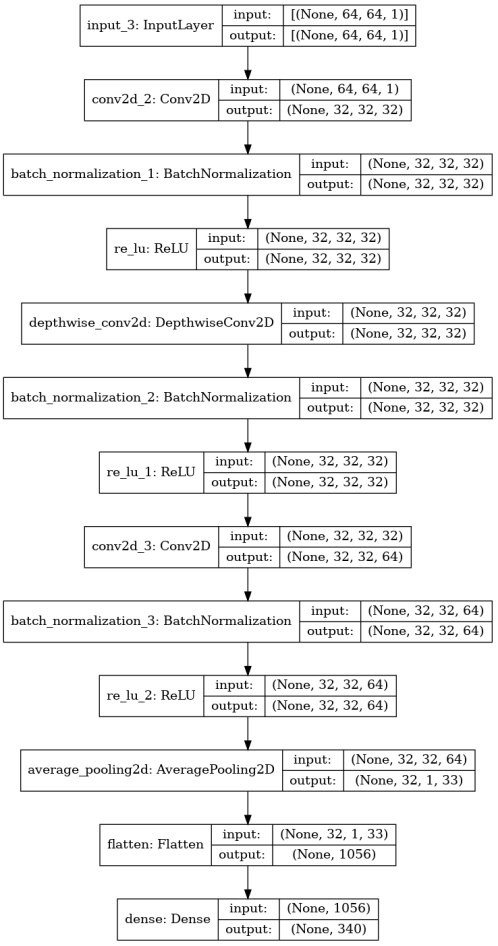

Fig 1. The architecture of the MobileNet-based model

Dataset is made up of over 40 million pictures of doodles provided by Google, of which only 100,000 were used during training. An app gave users one of the 340 prompts to which they have a minute to draw the best representation of that prompt drew these 64x64 doodles. During training, these doodles were also split into mini-batches of 256 to speed up the training process.

Fig 2. Doodles from the dataset

Testing is performed on Raspberry Pi 3 with following hardware:

Cortex-A53 ARM 64-bit SoC at 1.4 GHz, 1 GB LPDDR2 RAM, Bluetooth 4.2 BLE, Wi-Fi 802.11.b/g/n/ac with microSD card acting as a permanent storage.

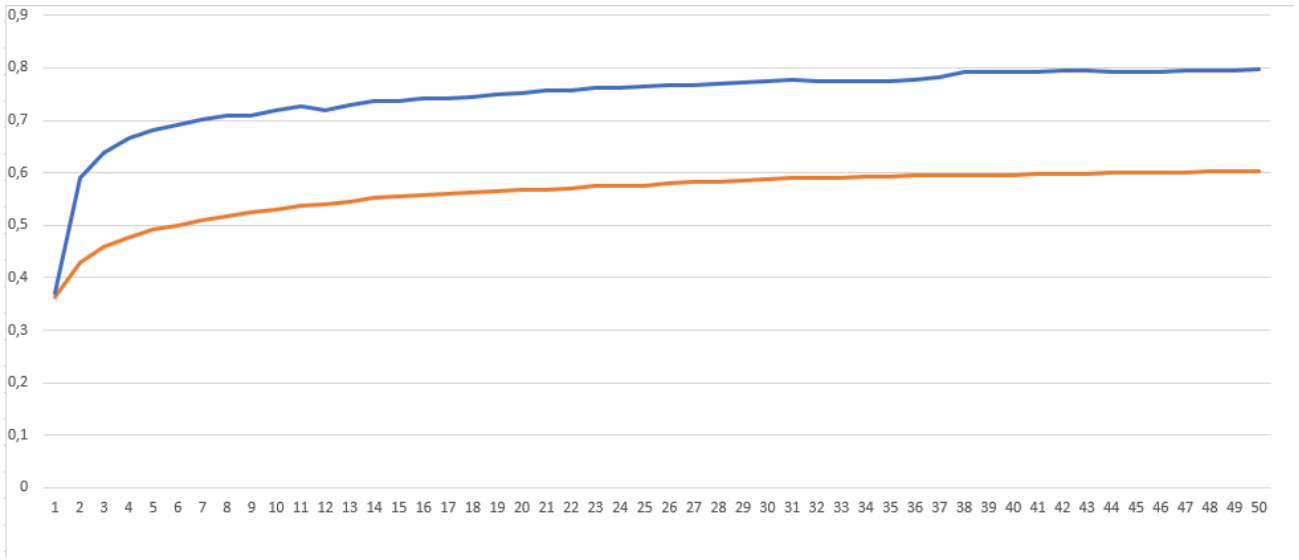

Fig. 3. Test accuracy of the full MobileNet (blue) and first block of the MobileNet (orange)

Conclusion. In this article, we explored the idea of distributing learning on Edge architecture, creating on-site AI presence. Results show that accuracy of the MobileNet for this dataset reaches 79.74% and our MobileNet-based model gets up to 60.03% on 50th epoch. As for speed, the MobileNet model is much slower on the

Raspberry Pi, however MobileNet-based test model is approximately as fast on Raspberry Pi as the whole model is on dedicated machine, sporting an NVidia Tesla GPU.

Further areas of exploration for this approach may include using specialized accelerators such as Intel Compute Stick, or NVidia Jetson and using real-time video feed for testing unsupervised classification.

References

1. Zhao, Z., Barijough, K. M., & Gerstlauer, A. (2018). Deepthings: Distributed adaptive deep learning inference on resource-constrained iot edge clusters. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 37(11), 2348-2359.
2. Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. Insights into imaging, 9(4), 611-629.
3. Li, E., Zeng, L., Zhou, Z., & Chen, X. (2019). Edge AI: On-demand accelerating deep neural network inference via edge computing. IEEE Transactions on Wireless Communications, 19(1), 447-457.
4. Xu, X., Ding, Y., Hu, S. X., Niemier, M., Cong, J., Hu, Y., & Shi, Y. (2018). Scaling for edge inference of deep neural networks. Nature Electronics, 1(4), 216-222.
5. Merenda, M., Porcaro, C., & Iero, D. (2020). Edge machine learning for ai-enabled iot devices: A review. Sensors, 20(9), 2533.
6. Lemeshko A. V., Evseeva O. Yu., Garkusha S. V. (2014). Research on Tensor Model of Multipath Routing in Telecommunication Network with Support of Service Quality by Greate Number of Indices. Telecommunications and Radio Engineering, 73(15), 1339—1360.
7. Abich, G., Reis, R., & Ost, L. (2021, February). The impact of precision bitwidth on the soft error reliability of the MobileNet network. In 2021 IEEE 12th Latin America Symposium on Circuits and System (LASCAS) (pp. 1-4). IEEE.

AUTHORS

Volodymyr Rusinov – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: VRusinovIO51@office365.fiot.kpi.ua

Oleksii Cherevatenko – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: chereva@ukr.net