OpenVINO Deep Learning Workbench: Towards Analytical Platform for Neural Networks Inference Optimization

Alexander Demidovskij¹,²*, Artyom Tugaryov², Andrej Kashchikhin², Alexander Suvorov², Yaroslav Tarkan², Fedorov Mikhail², Gorbachev Yury²

¹Higher School of Economics, 25/12 Bolshaya Pecherskaya st., 603155, Nizhny Novgorod, Russia
²Intel Corporation, 30 Turgeneva st., 603024, Nizhny Novgorod, Russia
*E-mail: alexander.demidovskij@intel.com

Abstract. Modern neural network methods make it possible to obtain qualitative results that open the colossal potential for integrating them into industrial applications. Therefore, there is a need for systems like the OpenVINO Deep Learning Workbench to analyze and optimize neural networks’ performance on target devices. As the primary tool for inference, the OpenVINO open environment is used, which provides a wide variety of options to speed up network execution on the target hardware.

1. Introduction
Application Engineers (AEs) widely use deep learning (DL) technologies in various industrial applications. As Danny Lange, former head of machine learning at Uber, notes, “this technology [deep learning] has finally escaped the research labs and is rapidly becoming “a key element of business transformation” [1], highlighting the crucial role of neural network technologies in the modern world.

Despite the high-quality indicators of modern neural models, these models’ industrial application imposes certain limitations, such as the amount of memory consumed, power, and information processing speed. Targets can be not only high-performance platforms such as Intel® Xeon® but also low-power platforms with low memory consumption, such as the GNA co-processor on the Intel® Silver J500GNA processor or Intel® Movidius Neural Compute Stick 2. Besides power consumption, another important aspect besides is the speed of information processing, for example, the number of images processed by a neural network per second. AEs can perform optimization of models at the hardware and software levels. Accordingly, the effective execution of a model depends on both the model itself and the target device. Therefore, it becomes necessary to build platforms for analysis, optimization, quality control, and preparation of models for use in an industrial application. The purpose of this work is to demonstrate how you can optimize models and evaluate their quality using the OpenVINO Deep Learning Workbench application tool.

This paper is organized as follows. Section 2 provides a brief overview of existing software solutions for analyzing the performance and accuracy of models. Section 3 is devoted to assessing existing approaches to speeding up neural network operation: calibrating models and parallelizing hardware execution. Section 4 provides an overview of custom scripts in the OpenVINO DL Workbench for neural network optimization and quality control. The conclusion sums up the results and describes the directions for further research.
2. Overview of Existing Performance Analysis Platforms

Platforms that provide functionality for training neural networks and systems to optimize models but do not have a user interface: Visdom [2], MXBoard [3], are not considered in this work. Therefore, AEs can consider TensorBoard [4], DLProf [5], and DL Workbench [6] as existing solutions aimed at optimizing models. These systems provide complete solutions and are developed by such corporations as Google, NVIDIA, and Intel. Other solutions are either strongly inferior to the above platforms in terms of functionality (GAN Lab [7]) or are highly specialized (Netron [8]). AEs regard TensorBoard as a versatile tool for analyzing learning dynamics and visualizing various metrics. The existing extensions system enables you to add performance analytics to models and suggest hypotheses about how the model makes decisions. First and foremost, TensorBoard is being developed as a tool for analyzing models trained on the TensorFlow framework. DLProf is aimed at profiling primitive operations called at a level as close to the hardware as possible, while visualization is minimal and implemented as an extension to TensorBoard. DLProf is created to facilitate model optimization using the TensorRT framework. The DL Workbench’s critical task is to analyze model performance, control its quality, and optimize it for target devices. At the same time, the DL Workbench turns out to be the only tool in which the user can perform step-by-step optimization of models without the need to write program code. As a result, the user gets both an optimized model and a runtime environment ready to be executed on a target device. DL Workbench provides a user interface and additional analytics capabilities on top of the OpenVINO framework.

Table 1. Types of experiments on the considered models.

| Model number | Model precision | Calibration algorithm type |
|--------------|-----------------|---------------------------|
| 1            | FP16\(^1\)      | No algorithm applied (original model) |
| 2            | INT8\(^2\)      | Default algorithm, Mixed calibration pre-set |
| 3            | INT8            | Default algorithm, Performance calibration pre-set |
| 4            | INT8            | AccuracyAware algorithm, Mixed calibration preset, accuracy drop threshold 1% |
| 5            | INT8            | AccuracyAware algorithm, Mixed calibration preset, accuracy drop threshold 5% |
| 6            | INT8            | AccuracyAware algorithm, Performance calibration preset, accuracy drop threshold 1% |
| 7            | INT8            | AccuracyAware algorithm, Performance calibration preset, accuracy drop threshold 5% |

3. Evaluation of Model Optimization Methods in the OpenVINO Deep Learning Workbench

OpenVINO framework provides DL Workbench with the necessary toolset and optimized runtime, dedicated to the efficient execution of neural networks on Intel architectures. As noted in [9], the critical areas of neural network optimization are:

1) Simplification of a network structure (merging layers)
2) Batch processing of input data (batch)
3) Parallelization at the network execution level (stream)
4) Reduction of the precision of calculations (calibration)

\(^1\) FP16 – 16-bit float numbers
\(^2\) INT8 – 8-bit integer numbers
Table 2. Results for EfficientNet on Intel® Core™ i7-9700K 3.6GHz.

| Experiment number | Metric | Settings | Quality metric (top-5) |
|-------------------|--------|----------|------------------------|
|                   |        | {# of streams in OpenVINO, batch} |                        |
|                   |        | {1; 1}   | {1; 4} | {2; 8} | {4; 8} |                        |
| 1                 | FPS\(^3\) | 15.74    | 11.44 | 11.34 | 11.25 | 91.47                  |
|                   | Lat.\(^4\), ms. | 63.12   | 347.88 | 1403.46 | 2840.78 |                        |
| 2                 | FPS | 14.48    | 11.47 | 10.99 | 11.70 | 90.62                  |
|                   | Lat., ms. | 68.64   | 347.10 | 1440.98 | 2727.00 |                        |
| 3                 | FPS | **23.09** | 23.13 | **22.69** | **22.51** | 91.26                  |
|                   | Lat., ms. | 43.06   | 171.92 | 701.23 | 1409.21 |                        |
| 4                 | FPS | 14.46    | 11.18 | 11.00 | 11.63 | 90.62                  |
|                   | Lat., ms. | 68.79   | 357.14 | 1451.13 | 2728.33 |                        |
| 5                 | FPS | 14.47    | 11.52 | 11.12 | 11.81 | 90.62                  |
|                   | Lat., ms. | 68.72   | 345.60 | 1434.02 | 2678.86 |                        |
| 6                 | FPS | 23.01    | 23.15 | 22.76 | 22.42 | 91.26                  |
|                   | Lat., ms. | 43.19   | 171.77 | 694.00 | 1414.57 |                        |
| 7                 | FPS | 22.80    | 23.11 | 22.13 | 22.40 | 91.26                  |
|                   | Lat., ms. | 43.61   | 172.04 | 726.23 | 1414.69 |                        |

Network simplification was considered in detail in [9]. Therefore, further, we will consider the experimental results of applying optimizations # 2 and # 3 for models executed in 32-bit floating-point numbers and a set of derived models in 8-bit integers. It is important to note that for each 32-bit model, two algorithms, Default and AccuracyAware, were used to obtain 8-bit models. Each algorithm was considered with different calibration schemes: Mixed and Performance. For AccuracyAware, two options were also considered: the maximum drop in accuracy by 1% and 5%. The Default algorithm, unlike AccuracyAware, by design cannot guarantee the accuracy of the resulting 8-bit model. These algorithms are described in more detail in [10], and specific experiments are presented in Table 1.

Platforms that provide functionality for training neural networks, as well as systems aimed at optimizing models, but do not have a user interface: Visdom [2], MXBoard [3], are not considered in this work.

Table 3. Results for EfficientNet on Intel® Xeon® Platinum 8280 2.7GH.

| Experiment number | Metric | Settings | Quality metric (top-5) |
|-------------------|--------|----------|------------------------|
|                   |        | {# of streams in OpenVINO, batch} |                        |
|                   |        | {1; 1}   | {1; 4} | {2; 8} | {4; 8} |                        |
| 1                 | FPS | 31.14    | 48.63 | 71.79 | 72.38 | 91.47                  |
|                   | Lat., ms. | 31.66   | 80.35 | 221.32 | 438.14 |                        |
| 2                 | FPS | 33.72    | 63.53 | 93.37 | 93.4 | 90.62                  |
|                   | Lat., ms. | 29.31   | 62.26 | 170.62 | 341.59 |                        |
| 3                 | FPS | **45.74** | 134.93 | **247.62** | 269.35 | 91.26                  |
|                   | Lat., ms. | 21.57   | 29.29 | 63.99 | 118.22 |                        |
| 4                 | FPS | 34.32    | 63.02 | 94.33 | 93.43 | 90.62                  |
|                   | Lat., ms. | 28.83   | 62.74 | 168.9 | 341.18 |                        |
| 5                 | FPS | 34.22    | 65.18 | 93.12 | 93.18 | 90.62                  |
|                   | Lat., ms. | 28.9    | 60.59 | 170.73 | 342.16 |                        |

\(^3\) FPS (Frames Per Second) – the number of images processed by a neural network per second. The higher this value, the more efficient the neural network.

\(^4\) Latency – time required to process one image. The lower the value, the better.
The experiments were carried out on machines with the following computational characteristics: Intel® Core™ i7-9700K 3.6GHz (Table 2 and 4), Intel® Xeon® Platinum 8280 2.7GHz (Table 3, 5). OpenVINO version 2020.4. The frequency was not tracked. EfficientNet and VGGNet-SSD300 were used as models of interest.

### Table 4. Results for VGGNet-SSD300 on Intel® Core™ i7-9700K 3.6GHz.

| Experiment number | Metric | Settings {# of streams in OpenVINO, batch} | Quality metric (top-5) |
|-------------------|--------|------------------------------------------|------------------------|
| 1                 | FPS    | 13.89, 14.09                             | 91.47                  |
|                   | Lat., ms. | 71.51, 113.07                           | 2318.11                |
| 2                 | FPS    | 24.03, 25.00                             | 90.62                  |
|                   | Lat., ms. | 40.93, 636.26                           | 1254.08                |
| 3                 | FPS    | 24.80, 25.03                             | 91.26                  |
|                   | Lat., ms. | 40.05, 637.20                           | 1249.91                |
| 4                 | FPS    | 24.60, 25.00                             | 90.62                  |
|                   | Lat., ms. | 40.35, 637.79                           | 1253.24                |
| 5                 | FPS    | 24.63, 24.97                             | 90.62                  |
|                   | Lat., ms. | 40.31, 637.68                           | 1251.80                |
| 6                 | FPS    | 24.68, 24.98                             | 91.26                  |
|                   | Lat., ms. | 40.24, 638.52                           | 1252.19                |
| 7                 | FPS    | 24.57, 24.94                             | 91.26                  |
|                   | Lat., ms. | 40.40, 638.23                           | 1255.35                |

Some patterns can be noticed from the results obtained. First, quite expectedly, the use of more powerful hardware, namely the Xeon platform instead of the Core platform, allows accelerating the classification model by 1.9 times and the detection model by 4.46 times. Considering these platforms separately, calibrating models to 8-bit integers can improve performance in all scenarios. On the Core platform, the average acceleration on 8-bit models is 1.19 and 1.77 times for the classification and detection models, respectively. On the Xeon platform, the acceleration from calibration alone is 1.27 and 2.05 times for the classification and detection models, respectively.

### Table 5. Results for VGGNet-SSD300 on Intel® Xeon® Platinum 8280 2.7GHz.

| Experiment number | Metric | Settings {# of streams in OpenVINO, batch} | Quality metric (top-5) |
|-------------------|--------|------------------------------------------|------------------------|
| 1                 | FPS    | 61.96, 100.09                            | 91.47                  |
|                   | Lat., ms. | 15.98, 159.23                           | 306.97                 |
| 2                 | FPS    | 155.31, 316.05                           | 90.62                  |
|                   | Lat., ms. | 6.34, 50.10                             | 94.46                  |
| 3                 | FPS    | 154.64, 317.38                           | 91.26                  |
|                   | Lat., ms. | 6.35, 49.91                             | 94.55                  |
| 4                 | FPS    | 146.36, 315.23                           | 90.62                  |
|                   | Lat., ms. | 6.69, 50.31                             | 94.72                  |
Additional tuning of hyperparameters, such as the batch size of the model and the number of OpenVINO streams, allow AEs to get 2.04 and 2.86 times increase on Xeon for the classification and detection models, respectively. The cumulative effect of applying optimization to the model and further hyperparameter tuning leads to an acceleration of 1.3 and 3.17 times on Core, 2.59, and 5.88 times on the Xeon platform for classification and detection models, respectively. Finally, we would like to draw reader’s attention to best results. There is 21.7- and 24.3-times speedup for classification and detection models respectively when the platform is switched from Core to Xeon, 8-bit integer inference is used instead of 32-bit floating-point one and best inference hyperparameters are found. Best results are demonstrated in Figure 1. Simultaneously, it is possible to significantly expand the search space for hyperparameters, which vary depending on different hardware platforms. For example, there is a certain dependence of performance on the correspondence of the number of streams to the number of microprocessor cores.

Figure 1. Summary of best results for the considered models. FPS – the higher is better, Latency – the lower is better. EfficientNet: ■ – Precision: FP32, Platform: Core, {Stream: 1, Batch: 1}; ★ – FP32, Xeon, {4, 8}; ● – INT8 (Default algorithm, Performance preset), Core, {1, 4}; ● – INT8 (AccuracyAware with 5% drop, Performance preset), Xeon, {4, 8}. VGGNet-SSD300: ▼ – FP32, Core, {2, 8}; ● – INT8 (AccuracyAware with 5% drop, Mixed preset), Core, {4, 8}; ▲ – FP32, Xeon, {4, 8}; ♦ – INT8 (Default, Performance preset), Xeon, {4, 8}.

4. Neural Network Optimization in OpenVINO DL Workbench

Previously, we considered various platforms for performance and accuracy analysis of neural networks and the positioning of these tools. The OpenVINO DL Workbench offers the following optimization approach:

1) evaluation of such parameters as latency and the number of frames processed per second;
2) detection of the level at which network optimization is required: the level of primitive functions, the structure of a model, precision of calculations. The DL Workbench’s general recommendation is to translate the original model to at least 16-bit execution. For a DL
Workbench user translation happens during the desired model registration for further work. Similarly, at the registration step, the model structure is automatically optimized using the OpenVINO Model Optimizer.

3) conversion of model precision into 8-bit integers. For this, DL Workbench uses a specialized OpenVINO Post-Training Optimization Tool [9]. Within the framework of this tool, two algorithms for model calibration are implemented: AccuracyAware and Default. Each of these algorithms attempts to translate most of the model execution into 8-bit numbers. A complete translation of the model in practice is almost impossible due to the architectural features of models or the lack of optimized primitive functions for complex, unpopular, or custom layers.

4) evaluation of model quality. DL Workbench offers a flexible toolkit for assessing model’s accuracy, using the OpenVINO Accuracy Checker tool as a basis [11]. It is also possible to compare the performance of two models on different devices and display the results the execution of models visually.

5) optimization of model execution through parallelization and batch processing of input data. The additional speedup can be obtained through the use of parameters such as stream and batch discussed earlier. DL Workbench provides a convenient form for finding the best combination of these execution hyperparameters and analyzing the results. In particular, there is an execution exploration technique based on a comparison of the original model in the format of OpenVINO IR and the execution graph. It is important to note that different accelerators such as CPU or iGPU execute one model differently. It is very helpful for AEs to to look inside the practical accelerator representation of the model for further optimization. Therefore, DL Workbench as a platform for optimization and performance analysis allows users to render both IR and Execution graphs simultaneously, visually compare them, identify execution hotspots with the help of color encoding of heavy layers (Figure 2).

As a demonstration of the instrumental richness of the OpenVINO Deep Learning Workbench, we can refer to the fact that the experimental data presented in the previous section is fully collected by this tool. Although a calibrated model is generally less accurate than the original model, the accuracy drop can be negligible. For different tasks, the appropriate accuracy drop can vary. During the calibration process, accuracy drop can be controllable if the AccuracyAware algorithm is used. However, even when there is an accuracy drop, the model can still be very accurate and produce indistinguishable results from the original model. In order for users to visually see the difference, DL Workbench provides
capabilities of visualizing model inference results on arbitrary images. Currently, various use cases are supported, such as Object Detection (Figure 3), Instance Segmentation (Figure 4), Semantic Segmentation (Figure 5). In practice, it often turns out that the drop in accuracy of an 8-bit model in comparison with its 32-bit counterpart turns out to be indistinguishable to the eye and acceptable from the point of view of quality indicators. Simultaneously, this results in a significant acceleration of the model operation, as well as its footprint on the file system is greatly reduced, on average four-fold.

Figure 3. Displaying model results in DL Workbench as an additional way to verify the model quality. Object Detection use case.

Figure 4. Displaying model results in DL Workbench. Instance segmentation use case.

Figure 5. Displaying model results in DL Workbench. Semantic segmentation use case.

5. Conclusions
This paper discusses neural network performance optimization options and how model’s quality monitoring strategies proposed in the OpenVINO Deep Learning Workbench. Experimental results of launching neural networks are analyzed, and the functionality of the tool under consideration is
demonstrated. It is worth noting that systems such as the OpenVINO Deep Learning Workbench make it easier to find the optimal combination of available optimizations and compare the performance of models on different target devices. As a further direction of this study, the introduction of a block for advanced quality control of a model is considered, which should include the analytics of the data on which a modified model performs better or worse than a reference model and advanced visualization of quality metrics.

Acknowledgments
The authors would like to express their deep and sincere gratitude to Alina Alborova for comments that significantly improved the manuscript.

References
[1] Shaeffer E, Sovie D 2019 Reinventing the Product: How to Transform your Business and Create Value in the digital edge (Kogan Page), pp. 132
[2] Facebook 2018 Visdom: Rich and Live Visualizations (accessed June 27, 2020). [Online]. Available: https://ai.facebook.com/tools/visdom/
[3] A. W. S. Labs, MXBoard, 2018 (accessed June 27, 2020). [Online]. Available: https://github.com/awslabs/mxboard
[4] Google 2020 TensorBoard: TensorFlow’s visualization toolkit, 2016 (accessed June 27, 2020). [Online]. Available: https://www.tensorflow.org/tensorboard
[5] NVIDIA, Deep Learning Profiler (DLProf), 2019 (accessed June 27, 2020). [Online]. Available: https://docs.nvidia.com/deeplearning/ frameworks/dlprof- user- guide/
[6] Demidovskij A, Gorbachev Y, Fedorov M, Slavutin I, Tugarev A, Fatekhov M, Tarkan Y 2019 OpenVINO deep learning workbench: Comprehensive analysis and tuning of neural networks inference in Proceedings of the IEEE International Conference on Computer Vision Workshops
[7] Kahng M, Thorat N, Chau D H P, Vie gas F B, and Wattenberg M 2018 Gan lab: Understanding complex deep generative models using interactive visual experimentation IEEE Transactions on Visualization and Computer Graphics 25(1), 1–11
[8] Roeder L, Netron - a viewer for neural network, deep learning and machine learning models, 2017 (accessed June 27, 2020). [Online]. Available: https://github.com/lutzroeder/netron
[9] Demidovskij A et al 2020 OpenVINO Deep Learning Workbench: A Platform for Model Optimization, Analysis and Deployment. 2020 IEEE 32th International Conference on Tools with Artificial Intelligence (ICTAI)
[10] OpenVINO, Post-Training Optimization Toolkit, 2020 (accessed June 27, 2020). [Online]. Available: https://docs.openvino-toolkit.org/latest/ README.html
[11] OpenVINO, Deep Learning Accuracy Validation Framework, 2018 (accessed June 27, 2020). [Online]. Available: https://docs. openvino-toolkit.org/latest/ tools accuracy checker README.html