Deep Learning For Bela

Rodrigo Diaz
Queen Mary University of London
r.diazfernandez@qmul.ac.uk

Abstract

Several state-of-the-art frameworks allow the creation, training, and deployment of neural network models. Some of these also allow the deployment of models on embedded devices mainly for inference.

The models, aimed typically at recognition and classification tasks, must be heavily optimized to run under the constrained conditions of such devices. Furthermore, in several audio-related tasks, real-time inference is required.

Previous attempts at running real-time audio inference on the low-latency Bela board, have relied on the re-implementation of such models. In this work, I present a general method for the direct deployment of models using several state-of-the-art deep learning frontends. Additionally, I include a performance benchmark and examples of real-time audio inference using this method.

1. Introduction

Deep learning for embedded and mobile devices has gained popularity in recent years [2]. This is mainly due to the greater capabilities, reduced manufacturing cost, and better efficiency exemplified by devices such as the Raspberry Pi [12] or the NVIDIA Jetson [6]. While training neural networks on such devices is oftentimes prohibitive due to computational costs, the inference is possible. Several state-of-the-art frameworks [7], have introduced reduced or specialized versions to enable inference on such devices.

The performance of specialized inference pipelines is a crucial point in the development of deep learning solutions for embedded devices [5, 13]. Typically, the models used on these devices are oriented toward visual recognition and classification tasks, that require efficient implementations [4, 9].

In the audio domain, neural networks have also been mainly oriented towards similar tasks, such as keyword spotting [14] and speech recognition [10]. Recently, Devis *et al*. [3] also demonstrated neural synthesis models running on an NVIDIA Jetson Nano.[1]

The low-latency Bela platform, a BeagleBone Black-based device, offers better capabilities than others for audio and sensor processing. Solomes *et al*. [11] show that it is possible to use neural networks for inference in this device. However, their approach requires an explicit implementation of the model itself, preventing trying different models at runtime without a re-implementation. Essentially, for this approach, it is necessary to translate the operations needed for inference (e.g., matrix multiplication, non-linear activations) to highly optimized C++ and sometimes assembly code.

Frameworks such as Pytorch [8] and Tensorflow [1] implement ready-to-use optimized versions of these operations. However, compiling and setting a build environment specifically for these frameworks on the Bela is a complex and time-demanding task due to the limited resources for compilation on the device. Moreover, to perform inference, it is necessary to consider not only the former limitations but also the strict latency requirements, especially for models oriented towards real-time signal processing, such as synthesis and filtering.

The present contributions are three:

- A cross-compilation environment and a wrapper for the ArmNN, Libtorch, TFLite, and RTNeural frontends.
- Evaluation of the inference time on the Bela for each frontend.
- Diverse examples that show how to use the wrapper for real-time audio tasks.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: A system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016. 1

[1] https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[2] Jiasi Chen and Xukan Ran. Deep learning with edge computing: A review. *Proceedings of the IEEE*, 107(8):1655–1674, 2019. 1

[3] Ninon Devis and Philippe Esling. Neurorack: deep audio learning in hardware synthesizers. In *EPFL Workshop on Human factors in Digital Humanities*, number CONF, 2021. 1

[4] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 1

[5] Liangzhen Lai, Naveen Suda, and Vikas Chandra. Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus. *arXiv preprint arXiv:1801.06601*, 2018. 1

[6] Sparsh Mittal. A survey on optimized implementation of deep learning models on the nvidia jetson platform. *Journal of Systems Architecture*, 97:428–442, 2019. 1

[7] Aniruddha Parvat, Jai Chavan, Siddhesh Kadam, Souradeep Dev, and Vidhi Pathak. A survey of deep-learning frameworks. In *2017 International Conference on Inventive Systems and Control (ICISC)*, pages 1–7. IEEE, 2017. 1

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 1

[9] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 1

[10] Yuan Shangguan, Jian Li, Qiao Liang, Raziel Alvarez, and Ian McGraw. Optimizing speech recognition for the edge. *arXiv preprint arXiv:1909.12408*, 2019. 1

[11] Alexandru-Marius Solomes and Dan Stowell. Efficient bird sound detection on the bela embedded system. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 746–750. IEEE, 2020. 1

[12] Ahmet Ali Süzen, Burhan Duman, and Betül Şen. Benchmark analysis of jetson tx2, jetson nano and raspberry pi using deep-cnn. In *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–5. IEEE, 2020. 1

[13] Gaurav Verma, Yashi Gupta, Abid M Malik, and Barbara Chapman. Performance evaluation of deep learning compilers for edge inference. In *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 858–865. IEEE, 2021. 1

[14] Yundong Zhang, Naveen Suda, Liangzhen Lai, and Vikas Chandra. Hello edge: Keyword spotting on microcontrollers. *arXiv preprint arXiv:1711.07128*, 2017. 1