
Supplementary Materials for “Forward Target Propagation: A Forward-Only Approach to Global Error Credit Assignment via Local Losses”

Anonymous Author(s)

Affiliation

Address

email

1 Details of Multiply-Accumulate (MAC) Analysis

This analysis evaluates the computational complexity and training time of various learning procedures applied to different models across multiple tasks and datasets. Specifically, the study examines Backpropagation (BP), Forward-Forward (FF), PEPITA (PEP), Difference Target Propagation (DTP) and Forward Target Propagation (FTP) to assess their efficiency and scalability.

These procedures were tested on four distinct models: DS-CNN, MobileNet, ResNet, and AutoEncoder (AE). Each model was trained on a corresponding dataset—Speech Commands (SC), Visual Wake Words (VWW), CIFAR-10, and ToyADMOS—selected based on the task characteristics. The estimation process follows a similar procedure to that described in [1], from which we adopted the MAC counts for BP, FF, and PEPITA. We further extended the analysis by incorporating MAC estimations for FTP and DTP.

1.1 Methodology

1.1.1 Assumptions

The analysis is based on the following assumptions:

- **Quantization:** To simplify the calculation, Weights, biases, and activations are quantized to INT8.
- **Batch Normalization:** Batch normalization layers from the original models were excluded from the analysis.

1.1.2 MAC Estimation Procedure

As a first step, a closed-form equation was derived to compute the number of MAC required by a generic layer (e.g., fully connected, convolutional, depthwise convolutional) to perform specific operations. The operations considered include the forward pass, backward pass, weight update, and additional computations unique to forward learning procedures. These include normalization and goodness evaluation in FF, error projection in PEPITA (PEP), and target propagation in FTP/DTP.

By defining the characteristics of each layer (e.g., input channels, kernel size), the MAC required for each operation can be systematically calculated. Each layer’s specifications were recorded, and the corresponding formula was applied to determine the exact MAC needed for every operation.

The MAC count for a given operation was summed across all layers to determine the total MAC required by the entire model for that specific operation during training. Then, the total MAC for each learning procedure was obtained by aggregating the MAC of all the operations involved in the

31 training process, including forward pass, backward pass, weight update, and additional computations
32 specific to each algorithm.

33 To describe the characteristics of a layer, the following notation has been used:

- 34 • C_{in} : channels in previous layer
- 35 • C_{out} : channels in current layer
- 36 • $H_{\text{in}}, W_{\text{in}}$: height and width of input feature map
- 37 • $H_{\text{out}}, W_{\text{out}}$: height and width of output feature map
- 38 • $H_{\text{ker}}, W_{\text{ker}}$: kernel height and width
- 39 • $H_{\text{stride}}, W_{\text{stride}}$: vertical and horizontal strides

40 For fully connected layers, H_{in} and H_{out} represent the number of activations in the previous and
41 current layers, respectively.

42 The total MAC for a model was calculated by summing across all layers and operations rele-
43 vant to each learning procedure. The detailed estimations, including intermediate computations
44 and total training time for each model and learning procedure, are provided in the spreadsheet
45 `MAC_analysis_FTP.ods`, included in the shared ZIP file as part of the Supplementary Materials.

46 2 Codes

47 The code for reproducing the experiments presented in Table 1 and Table 3 is included in the folder
48 named `FTP_Codes` within the ZIP file submitted as part of the Supplementary Materials. Instructions
49 for each part of the experiment are included in `READ_ME.md` file inside the folder.

50 References

- 51 [1] Danilo Pietro Pau and Fabrizio Maria Aymone. Suitability of forward-forward and pepita learning to
52 mlcommons-tiny benchmarks. In *2023 IEEE International Conference on Omni-layer Intelligent Systems*
53 (*COINS*), pages 1–6, 2023.