Exploiting Forward-Forward based algorithm for training on device

Marco Lattuada

Fabrizio De Vita, Rawan M. A. Nawaiseh, Dario Bruneo, Valeria Tomaselli, Mirko Falchetto

STMicroelectronics

Introduction

- Training on (embedded) device is gaining interests in last years because of:
  - Customization of the model to the user/scenario
  - Privacy concerns
- Traditional backward propagation technique can be too resource-consuming to be deployed on MCUs
- We propose $\boldsymbol{\mu FF}$
  - Inspired by **Forward Forward algorithm**
  - Based on **Multivariate Ridge regression**

life.augmented

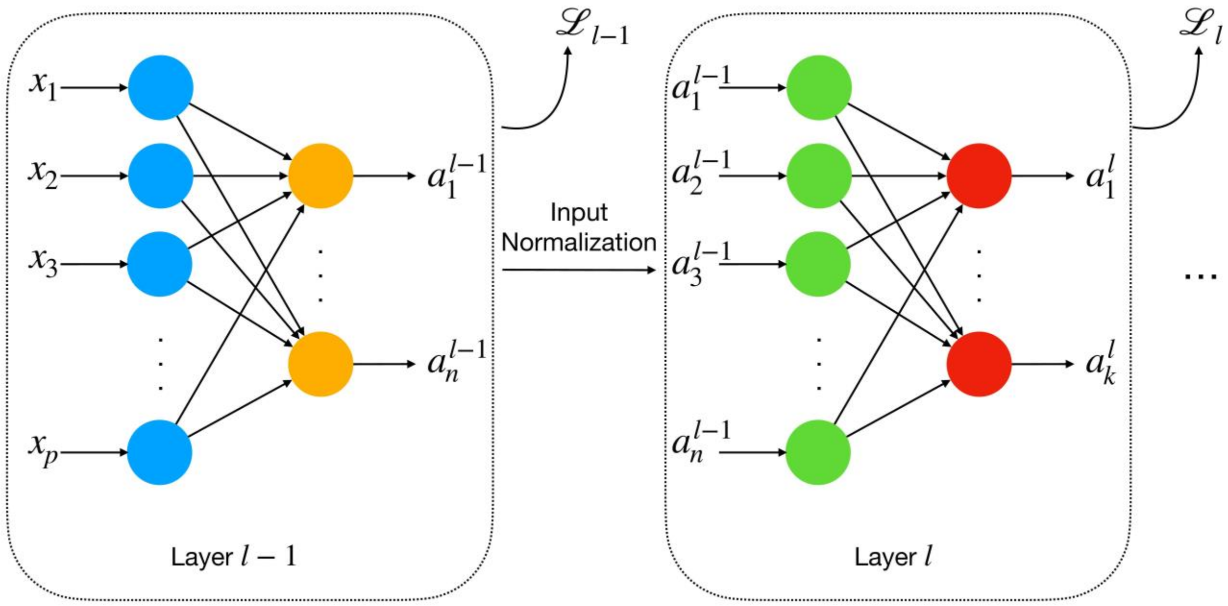
The problem

- Training on device requires to update weights on the target
- Traditional back-propagation technique requires **significant resources**:
  - Memory to store training data
  - Memory to store training intermediate results (i.e., gradients)
  - Computational power to compute gradients

- Different training techniques must be considered

The Forward-Forward algorithm: some preliminary investigations
Geoffrey Hinton (preprint – under review)

The Forward-Forward algorithm



- Each layer is considered separately
- Forward and backward passes are replaced with two forward passes
  - Positive (i.e., real) data to increase the goodness of the layers
  - Negative (i.e., corrupted) data to decrease the goodness of the layers

The Forward-Forward Algorithm

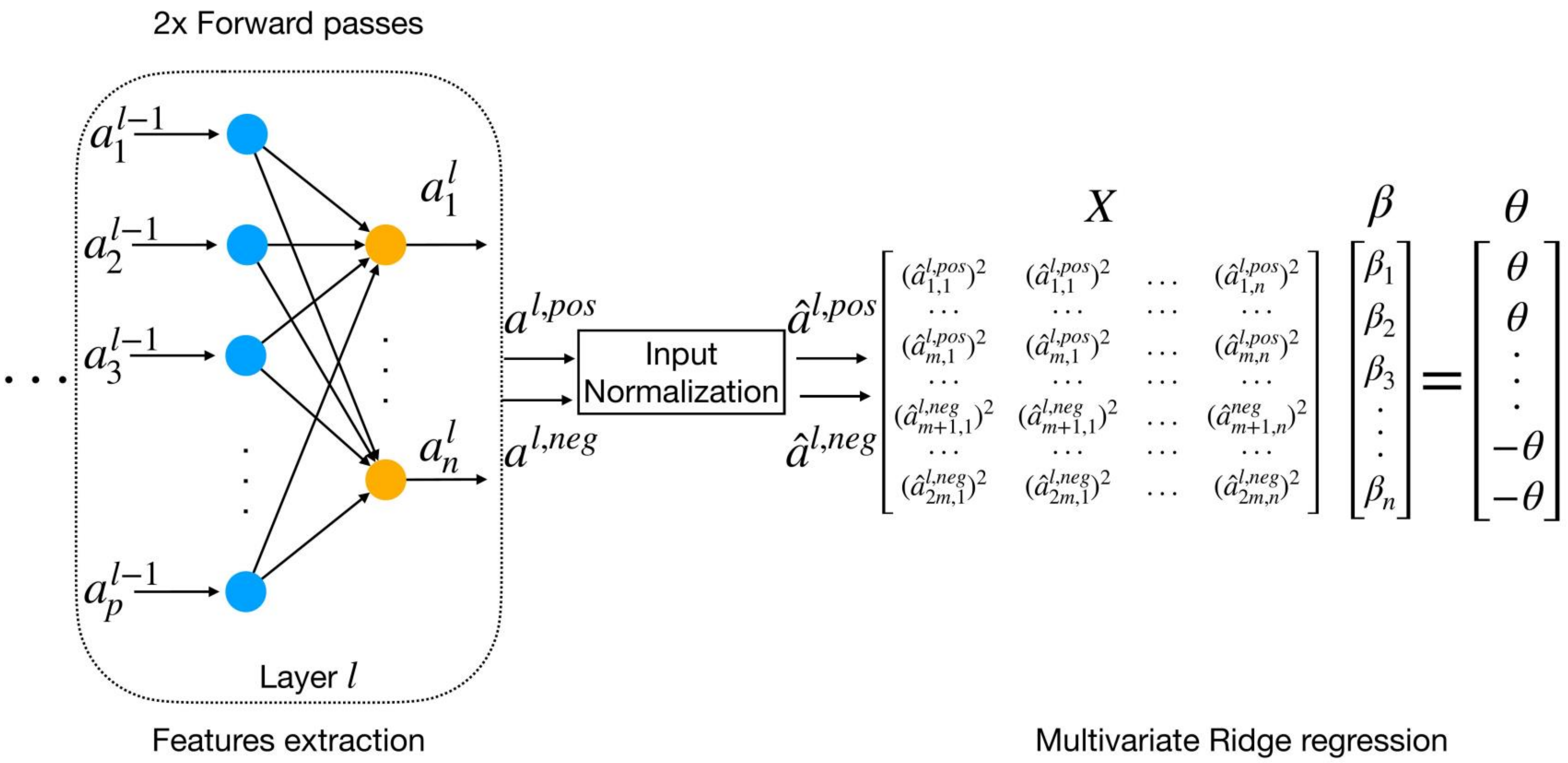
PROS

- Layers are updated independently
- Use only forward function
  - Layers can be considered as black box
- Memory usage and computation power saving

CONS

- Good accuracy only for some types of networks
- Gradient computation still needed
- Required memory and computation power can be not suitable for MCUs

Proposed solution architecture

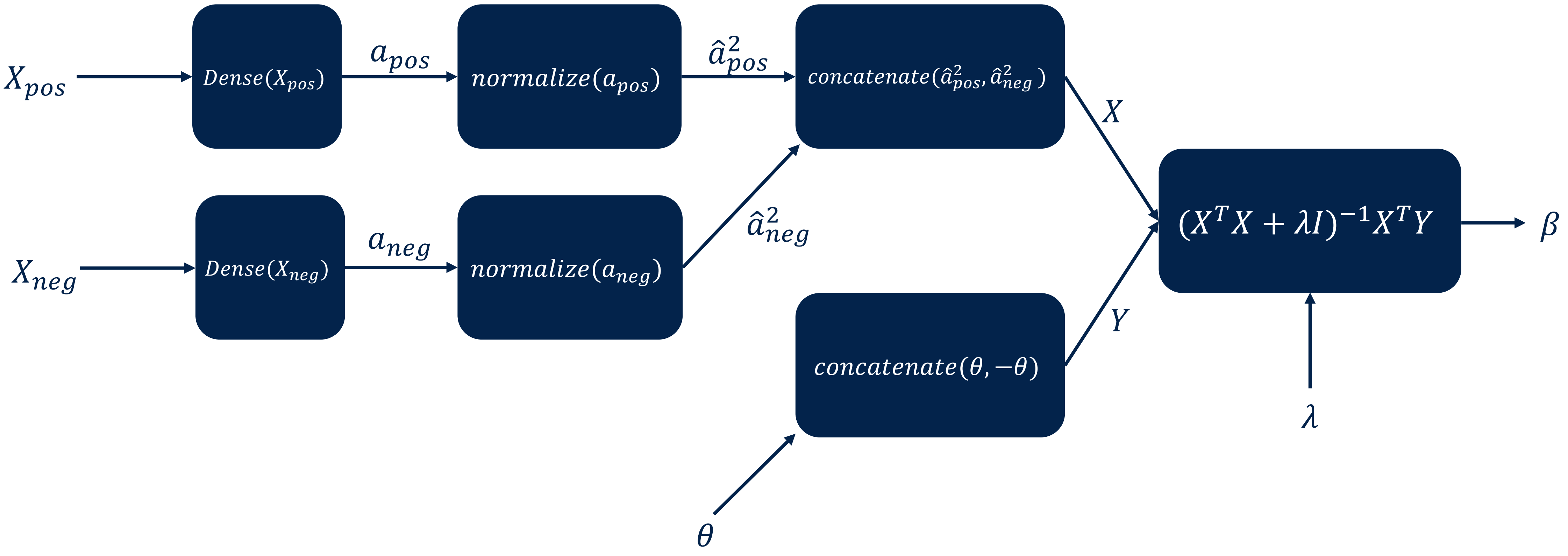


- Different loss function: Mean Square Error
- Training process becomes a multivariate Ridge Regression Problem

  → No iteration nor recursion is required to compute the weights

  → Predictable and fixed training time

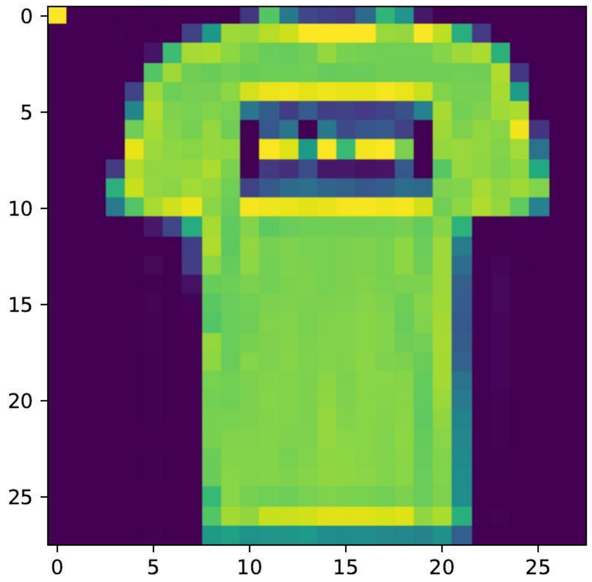
Proposed solution algorithm



life.augmented

Experimental setup

- Use case: dress classification on fashion-MNIST
- Classification task: fully connected (500 neurons) + ReLU
- Dataset is composed of 70,000 28x28 gray scale images
- Nine experiments:
  - 3, 4, and 5 classes
  - 100, 200, and 300 samples for each class
- Labels are embedded inside images in the corners
- Negative data are created embedding wrong labels
- Target device: STM32H741I-DISCO



life.augmented

Experimental results - accuracy

Back propagation	100 training samples	200 training samples	300 training samples
3 classes	0.977	0.981	0.983
4 classes	0.960	0.965	0.966
5 classes	0.851	0.863	0.869

μFF	100 training samples	200 training samples	300 training samples
3 classes	0.936	0.942	0.945
4 classes	0.791	0.822	0.840
5 classes	0.679	0.732	0.770

Experimental results - training time

μFF training time [ms]	100 training samples	200 training samples	300 training samples
3 classes	6,012	12,004	17,994
4 classes	8,009	16,000	23,989
5 classes	10,007	19,994	29,981

life.augmented

Conclusions

- Resource usage of backward-propagation technique can prevent its adoption on embedded design because of power, time, and memory constraints
- Proposed μFF combines the use of **Positive** and **Negative data**, **MSE**, and multivariate **Ridge Regression** to allow training on device with limited and predictable resources
- Future works will focus on extending proposed approach to different types of network topologies

Our technology starts with You

Find out more at www.st.com

life.augmented