On Advancements of the Forward-Forward Algorithm

Mauricio Ortiz Torres*, Markus Lange**, Arne P. Raulf***

German Aerospace Center (DLR)
Institut for AI-Safety and Security, Sankt Augustin & Ulm
* mauricio.ortiztorres@dlr.de, ** markus.lange@dlr.de, *** arne.raulf@dlr.de

ABSTRACT: The Forward-Forward algorithm has evolved in machine learning research, tackling more complex tasks that mimic real-life applications. In the last years, it has been improved by several techniques to perform better than its original version, handling a challenging dataset like CIFAR10 without losing its flexibility and low memory usage. We have shown in our results that improvements are achieved through a combination of learnable embeddings, learning rate schedules, and independent block structures during training that lead to a 20% decrease in test error percentage.

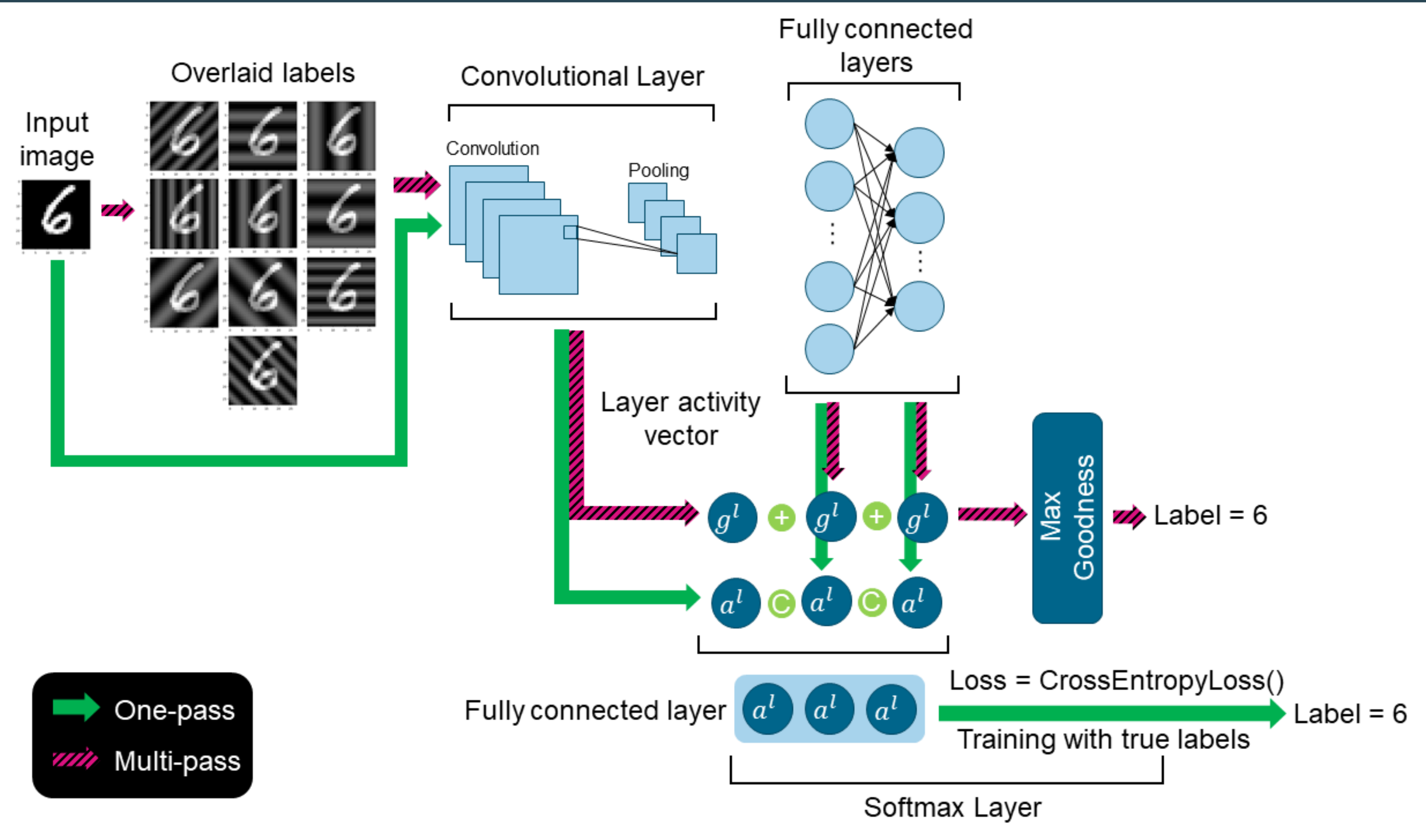

Fig. 1: Flow diagram for the one-pass and multi-pass inference step.

INTRODUCTION

The Forward-Forward Algorithm, introduced by Geoffrey Hinton in 2022, offers potential for low-power hardware in aerospace projects. Layers are trained independently using a "goodness" loss function such that the goodness is maximized with positive data. This approach reduces memory requirements during training and inference, making it suitable for low-power hardware and parallelization. The algorithm allows for flexible hypertuning and efficient inferences, with recent research leading to further improvements and lightweight models.

THE FORWARD-FORWARD (FF) ALGORITHM

ORIGINAL APPROACH

The Forward-Forward algorithm's architecture is similar to a traditional neural network, but its training procedure differs significantly. It uses two input spaces: positive and negative samples created from the dataset by overlaying images with correct or incorrect labels. For example, in the MNIST dataset, an image is overlaid with its correct label to create a positive sample, and with an incorrect label to create a negative sample.

The algorithm operates with two forward passes: one positive pass to increase the "goodness" of each layer for positive samples, and one negative pass to decrease it for negative samples. The goodness loss function of each layer is measured by the logistic function of the sum of the squares of its activity vectors minus some threshold θ, i.e.,

$$C_{FF} = C^l_{pos} + C^l_{neg} =$$

$$\ln\left[1 + \exp\left(\theta - g^l(x_{pos})\right)\right] + \ln[1 + \exp(g^l(x_{neg}) - \theta)]$$

During training, the output of each hidden layer is normalized to transmit relative information. After training, the inference phase can be initiated using two methods: one-pass inference and multi-pass inference. One-pass inference involves training a SoftMax/Sigmoid layer with a general multi-classification loss function, using the concatenated activity vectors of the original network as input. Multi-pass inference uses the trained network with multiple generated overlaid images as input, collecting goodness values for each layer and determining the final label using an argmax function (see Figure 1).

VARIATIONS TO THE ALGORITHM

Variations of the Forward-Forward algorithm have been explored to address its weaknesses. These include creating better correlated input data, improving training loss functions, and developing more effective training routines and inference schemes. Techniques such as spatially extended labeling, learnable embeddings, and Noise Contrastive Estimation have been used to create positive and negative samples. One approach substitutes the positive and negative samples with convolutional group channels. New loss functions based on convolutional channel-wise competitive learning have been proposed to measure discrepancies between input patterns and labels. Additionally, parallel training architectures and lightweight inference methods have been investigated to improve efficiency.

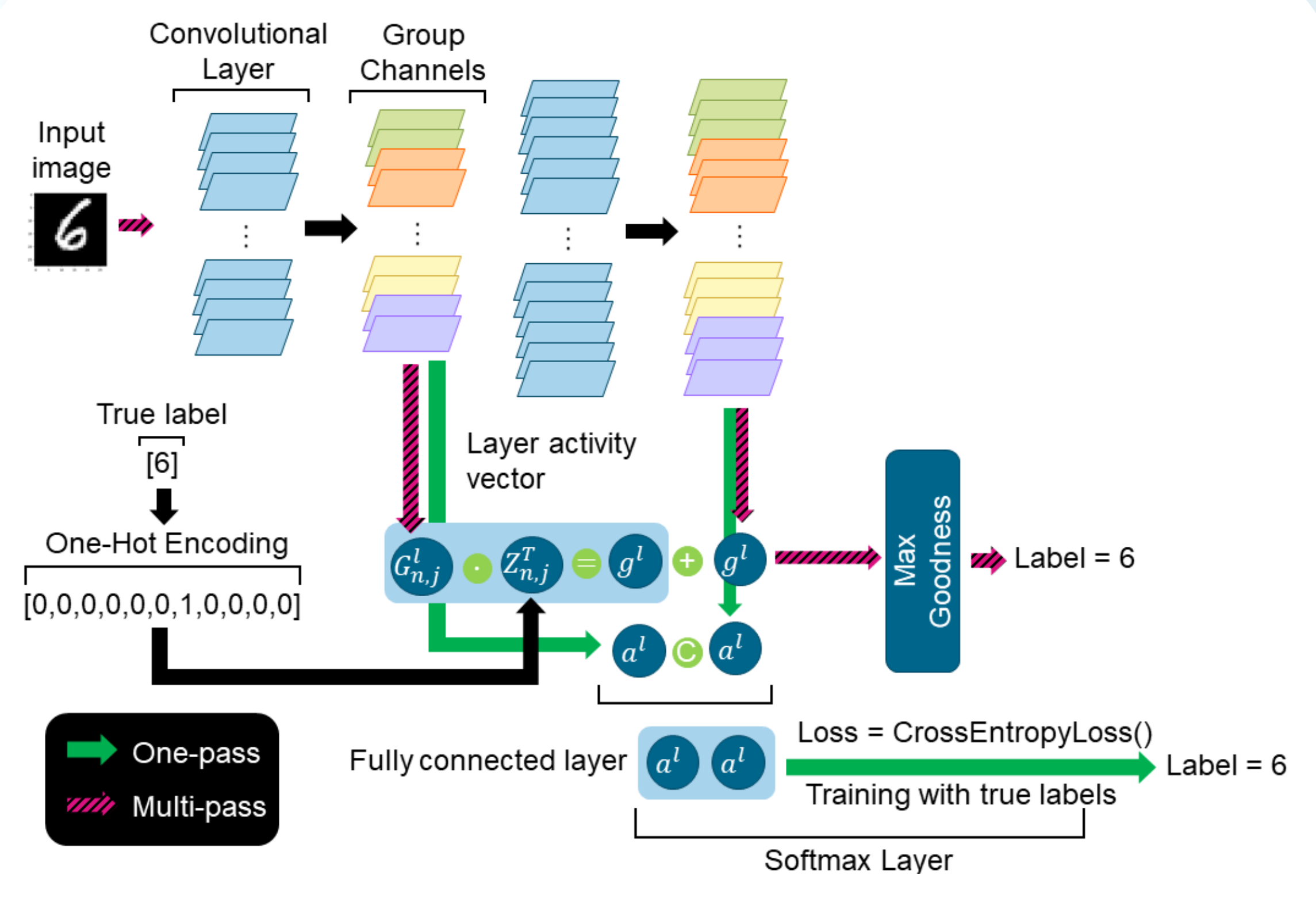

Fig. 2: Flow diagram for the inference step in our improved algorithm.

Table 1: Test results of FF algorithm, for MNIST and CIFAR10 datasets.

Dataset	Layers	Inference	Training Error [%]	Test Error [%]
MNIST	MLP	One-pass	8.6	8.0
	MLP	Multi-pass	7.3	7.2
	CNN	One-pass	4.5	4.4
	CNN	Multi-pass	4.9	5.0
CIFAR10	MLP	One-pass	39.2	39.3
	MLP	Multi-pass	41.1	41.2
	CNN	One-pass	20.5	44.3
	CNN	Multi-pass	44.8	48.3

RESULTS & ANALYSIS

We have developed with an improved algorithm that has the following characteristics:

- Creation of convolutional group channels for positive/negative sampling (see Figure 2).
- Training through chunked local updates that alternate between iterations.
- Realization of inference based on the activity vectors from the last two layers of the model.
- Use of the channel-wise loss function:

$$C^l = -\frac{1}{N}\sum_{n=1}^{N}\log\left(\frac{\exp\left(\left(g^l_{pos}\right)_n\right)}{\sum_{l=1}^{L}\exp\left(G^l_{n,j}\right)}\right),$$

where, $G^l_{n,j} = \frac{1}{S\times H\times W}\sum_{s=1}^{S}\sum_{h=1}^{H}\sum_{w=1}^{W}\left(\hat{Y}^l_{n,j,s,h,w}\right)^2$, and $\hat{Y}^l_j \in \mathbb{R}^{N\times S\times H\times W}$, where $S = C/J$, and J corresponds to the number of classes present in the classification problem.[1] Our results in Table 1 show that the algorithm works correctly as a classifier and has similar results when employing either of the two inference options. In Table 2, we show a list of the studied light FF models, their respective architectures, and their test results. The models has a test error of (21±3)% and offer a different range of trainable parameters. Our smallest network, *FF_tiny,* has 96% less parameters compared to *FF_deep*, but is subject to the highest test error of 24.1%. In contrast, *FF_optimal* conserves a test error below 20% and a parameter reduction of 81% in comparison to our largest model.

CONCLUSION

The FF algorithm continues to evolve, showing potential for low-power hardware. Improved versions perform better than the original, with a 20% decrease in test error percentage. Lighter FF models achieve low test error percentages (21±3%) with fewer parameters (164,706-754,386), comparable to state-of-the-art models but smaller in size. These models offer high performance while keeping a smaller size. Further investigations will focus on verifying and validating these neural networks.

Table 2: Architectures and test results from the light FF models trained under the CIFAR10 datasets with the improved FF algorithm.

Model name	Channels; Kernel Size	Trainable Parameters	Inference	Training Error [%]	Test Error [%]
FF_tiny	[3, 50, 50, 50, 50]; [$\underline{3}$, 3, $\underline{3}$, 4]	164,706	One-pass	15.3	24.4
			Multi-pass	17.9	24.1
FF_small	[3, 50, 50, 70, 70]; [$\underline{3}$, 3, $\underline{3}$, 4]	239,266	One-pass	12.3	23.4
			Multi-pass	14.8	23.1
FF_medium	[3, 50, 50, 100, 150]; [$\underline{3}$, 3, $\underline{3}$, 4]	484,506	One-pass	8.0	21.5
			Multi-pass	10.1	20.7
FF_optimal	[3, 50, 50, 100, 160, 160]; [$\underline{3}$, 3, $\underline{3}$, 4, 3]	754,386	One-pass	2.0	19.6
			Multi-pass	2.4	19.3
FF_deep	[3, 130, 130, 260, 260, 260, 510]; [$\underline{3}$, 3, $\underline{3}$, 5, 3, $\underline{3}$]	4,131,346	One-pass	1.2	18.8
			Multi-pass	1.8	18.2

[1] *We work with the NCHW format, where N : Number of data samples in the batch, C : Image channels, H : Image height, W : Image width.*