ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

Introducing Convolutional Channel-wise Goodness in Forward-Forward Learning

Andreas Papachristodoulou
Christos Kyrkou, Stelios Timotheou, Theocharis Theocharides *

KIOS Research and Innovation Center of Excellence
University of Cyprus - Dept of Electrical and Computer Engineering
Nicosia - Cyprus

Abstract. This paper introduces a Channel-wise Goodness Function (CWG) that enhances the Forward-Forward through the use of Convolutional Neural Networks. The CWG function facilitates simultaneous feature extraction and separation, eliminating the requirement for constructing negative data and leading to faster convergence rates. The approach employs a two-component loss function that maximizes positive goodness and minimizes negative goodness. This enables the model to learn class-specific features to outperform recent non-backpropagation approaches on basic image classification datasets and shorten the gap with the well-established backpropagation methods.

1 Introduction

Optimizing neural networks is challenging due to the highly non-convex landscape. Despite its effectiveness, backpropagation faces limitations due to problems with local minima, vanishing/exploding gradients, overfitting, and slower convergence, while requiring large amounts of labeled data [1]. Additionally, its biological plausibility is doubtful since there is no evidence that the visual system propagates error derivatives or stores neural activities for a backward pass [2], [3]. Predictive Coding and Evolutionary Algorithms are non-backpropagation training schemes that can overcome some limitations in neural network training. Predictive Coding is based on the idea of the brain's predictive coding model, while Evolutionary Algorithms treat network weights as the genome to optimize them. These methods handle the vanishing/exploding gradients problem without perfect knowledge of the forward pass computation [4], [5]. Reinforcement Learning can handle black-box models, but suffers from high variance and is computationally expensive [6], [7]. Competitive Hebbian Learning involves neurons competing with each other to become more active and suppress their neighbors resulting in the selection of neurons that respond best to the input. This technique can overcome limitations of task-specific features and overfitting, but it has slow convergence and requires careful parameter tuning, [3], [8]. The Forward-Forward (FF) Algorithm [2], has been recently proposed as a promising alternative to backpropagation. This paper improves the FF-Algorithm to make it more suitable for Convolutional Neural Networks.

*This work was supported by the EU Horizon 2020 research and innovation program under grant agreement No 739551 (KIOS CoE) and from the Government of the Republic of Cyprus through the Directorate General for European Programs, Coordination and Development.

2 Advancing the Forward-Forward (FF) Algorithm

The FF Algorithm uses two forward passes, one with positive (real) data and the other with negative (fake) data. The basic positive-negative process represents "positive" data as a vector of correct labels and images, while "negative" data are created by combining incorrect labels with input images. The training is layer-wise through Gradient-Descend on a single layer that has its own objective function, which is to maximize the goodness of the model on positive data while minimizing it on negative data. In this context, goodness has been defined as the sum of squared activations in a layer for specific data. Compared to backpropagation (BP), FF learning offers a computationally efficient training process with low memory consumption, the ability to work with unknown forward computation details, and the capacity to learn while pipelining sequential data. The layer-wise training framework allows modularization, mitigation of overfitting on a layer level, independent training of each layer for a specific number of epochs, and a more transparent selection of hyperparameters and model architecture.

The goodness function and method for constructing negative data proposed initially in [2] are preliminary methods and require further investigation. The selection of the goodness function determines what the model is trying to optimize and is critical for performance. Furthermore, constructing negative data to appropriately represent the distribution of naturally occurring negative data to train and learn multi-layer representations is crucial and extremely challenging in the positive/negative framework. The task and data dependency of this approach limits its generalizability; thus, eliminating the requirement for negative data would be a significant advantage.

Despite its promise, FF is slower than backpropagation learning and also does not generalize in some simple problems, making backpropagation preferable for large models trained on large datasets. Our investigation aims to improve accuracy while also exploring the potential for faster convergence rates with the FF algorithm. Optimizing each layer directly through layer-wise training should lead to quicker convergence, as in BP the backpropagated error derivatives tune indirectly the early neural network layers.

Our approach simultaneously extracts and separates features using convolutional layers, trained with the FF Algorithm for faster convergence. Unlike Competitive Hebbian Learning, which limits competition to shared kernels of a Convolutional layer (CNN), our approach separates the feature space into Depth-wise subsets of channels that correspond to the different classes. We use a two-component loss function to maximize positive goodness and minimize negative goodness by competing locally between different representation vectors of channel blocks using the proposed Channel-Wise Goodness Function (CWG). This speeds up training, improves the ability to learn class-specific features, and eliminates the need for negative data. Our approach outperforms baseline FF implementations and recent literature approaches on simple image classification datasets, achieving better accuracy and faster convergence rates.

3 Proposed Approach

3.1 Network Architecture and Training Process

The network architecture, as demonstrated in figure 1, consists of sequential convolutional layers which act as feature extractors for the attached classifiers: (i) the FF-Goodness (Goodness CF) and (ii) the Softmax (Softmax CF).

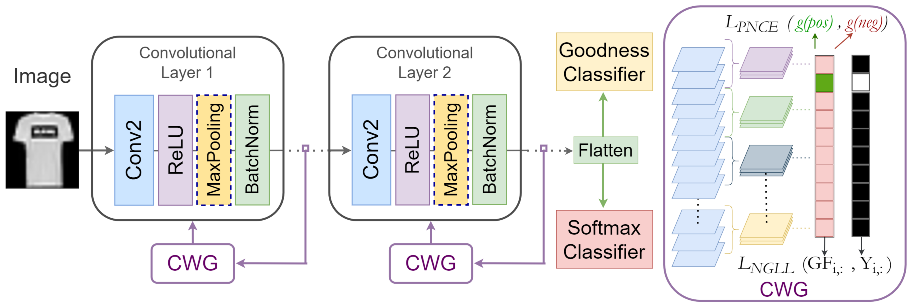

Fig. 1: Network architecture and CWG function

The model learns to extract features through the convolution operation on the total number of channels of the joint inter-class features. Through the proposed objective function, we achieve feature space separation via the channel dimension by associating specific subsets of the input channels to represent each class enabling the model to learn compositional intra-class features. The layers are trained independently, i.e., in a layer-wise manner, with the Adam Optimizer [9]. The output features of each layer are passed through ReLU activations, followed by Max-Pooling and Batch Normalisation (BatchNorm) [10].

The outputs of the last Convolutional Layer are flattened and used as input to the fully connected classifiers. The Goodness Classifier follows the basis of the FF Algorithm [2], overlaying the True and False labels as positive and negative features with the original goodness function. The Softmax Classifiers are standard softmax layers trained using the Cross Entropy Loss function.

We employ two strategies to train each layer with a proper initialization and give successive layers a headstart to achieve better and faster convergence. The first strategy is progressive and involves the complete training of a layer before progressing to the successive layer, while the second strategy involves the scheduled training of each layer so that each layer is trained in parallel for some epochs with their predecessors.

3.2 Channel-wise Goodness Function

We define the Goodness Factor, **GF**, as a matrix of size (N, C) where N is the number of samples in the batch, and C the number of classes. It involves the splitting of the feature matrix activations that are output from the convolutional layer into subsets of channels each representing a different class. A class goodness vector, of size C is calculated as the spatial and channel-wise mean of the squared activations that correspond to each subset of channels. The Goodness Factor is simply the concatenation of the different class goodnesses of the batch.

The binary label mask for a mini-batch of N examples with C classes is converted into a matrix $\mathbf{Y} \in \{0,1\}^{N\times C}$, where each row corresponds to an example and each column corresponds to a class.

The Positive Goodness, g^+ is a vector of size N, representing the goodness score of the subset of channels associated with the true class of the input features. It is computed by taking the dot product, $(\cdot)$ of the goodness matrix map, **GF**. and the binary label mask transpose, $\mathbf{Y^T}$.

The Negative Goodness g^- is a vector of size N, that represents the activations of the other subsets of channels which are associated with the false classes and is computed by taking the dot product of **GF**, and the complement of the binary label mask, **1-Y**, divided by the number of false classes, (number of classes - 1). This gives the mean of the squared activations for false classes.

$$g^+ = \mathbf{GF} \cdot \mathbf{Y}^T, \qquad g^- = \frac{1}{(C-1)} \mathbf{GF} \cdot (1 - \mathbf{Y}^T) \tag{1}$$

The CWG loss function is proposed for achieving local feature separation in addition to feature extraction. It is constructed as a weighted two-component loss function, L_{CWG}. The first component is a Positive vs Negative Cross Entropy Function, L_{PNCE} that maximizes the positive goodness, g^+, and minimizes the negative goodness, g^-. In this context, the symbol "$\|$" represents the concatenation of the positive and negative components. With this Objective, we treat the problem as a binary 'one vs all' classification task evaluating whether the input was predicted to be the correct or the false class using the threshold, θ, as the addend hyperparameter that determines the boundary between the correct and false class.

$$L_{PNCE} = \frac{1}{N} \sum_{n=1}^{N} \log\left(1 + \exp\left((-g_n^+ + \theta) \| (g_n^- - \theta)\right)\right) \tag{2}$$

The second component, computes the Negative Goodness Log-Likelihood of the true class probabilities, which are obtained by applying the softmax function to the respective class goodness score $GF_{n,c}$ where $y_{n,c}$ is the target class for the n^{th} example and the c^{th} class.

$$L_{NGLL} = -\frac{1}{N} \frac{1}{C} \sum_{n=1}^{N} \sum_{c=1}^{C} \log\left(\frac{\exp(GF_{n,c})}{\sum_{j=1}^{C} \exp(GF_{n,j})}\right) y_{n,c} \tag{3}$$

$$L_{CWG} = \alpha \times L_{PNCE} + (1-\alpha) \times L_{NGLL} \tag{4}$$

4 Experiments and Results

This section outlines our experimental methodology and results for evaluating the proposed model using established protocols from the relevant literature.

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

Dataset and Training Setup. Our model was trained and evaluated on three datasets: the Grayscale-image MNIST [11] and Fashion-MNIST [12], and the RGB-image CIFAR-10 [13]. We compared the Softmax Classifier variant of our proposed model, CWG+SF, against the performance of the original Forward-Forward algorithm as stated in [2], the Reproduced FF Model FF(Rep*), and with three FF-based models found in the literature namely, CaFo+CE, CaFo+SL [14], and PFF-RNN [5]. The Cascaded Forward (CaFo), [14], alters the original FF Algorithm by directly outputting label distributions for each cascaded convolutional block, while also eliminating the need for negative sampling. The Predictive Forward-Forward (PFF-RNN) approach utilizes a dynamic recurrent neural system that integrates lateral competition, noise injection, and predictive coding to conduct credit assignments in neural systems. The architecture used for the reported set of experiments consists of 3 Convolutional Layers of 20, 80, and 320 channels respectively, with different start and stopping training schedules. In addition, we compare our model, CWG+SF, with a Backpropagation Convolutional Model, BP (rep*), that has the same CNN + Softmax architecture with CWG+SF, and the BP implemented in [2]. Our model was trained for 10, 15, and 20 epochs on MNIST, F-MNIST, and CIFAR-10 respectively, using a 0.01 learning rate for all convolutional and fully-connected layers.

	MNIST		**F-MNIST**		**CIFAR-10**	
	Test Error (%)	**Training Epochs**	**Test Error** (%)	**Training Epochs**	**Test Error** (%)	**Training Epochs**
BP (*)	1.4	40	10.73	40	39	40
BP (rep*)	0.8	20	8.4	15	20.1	20
FF (*)	1.37	60	-	-	41	N/R
FF (rep*)	2.01	60	10.81	60	46.03	60
CaFo+CE	1.3	5000	-	-	32.57	5000
CaFo+SL	1.2	5000	-	-	33.74	5000
PFF-RNN	1.3	60	10.41	60	-	-
CWG+SF	1.3	**10**	**9.53**	**15**	**28.22**	**20**

Table 1: Testing Error and Training Epochs on MNIST, F-MNIST, CIFAR-10

Quantitative Results. The results of our experiments on MNIST, F-MNIST, and CIFAR-10 datasets are presented in Table 1. On MNIST, the proposed CWG+SF achieved a testing error of 1.3%, which is comparable to the other FF-inspired models such as CaFo+CE and CaFo+SL which also utilize convolutional layers. However, their model required training for 5000 epochs which is far longer than our approach which required only 10. On F-MNIST, CWG achieved a testing error of 9.53%, which is the lowest among all the models apart from the reproduced BP. The next best model, PFF-RNN, achieved a testing error of 10.41% but was trained for 60 epochs. On CIFAR-10, the difference in performance is magnified: CWG+SF achieved a testing error of 28.22%, which is significantly better than all the other Non-BackPropagation (non-BP) models. The second-best non-BP model, CaFo+CE, achieved a significantly lower performance at 32.57% while being trained again for 5000 epochs in comparison with our CWG+SF that required 20. The BP (rep*) model is the only model that outperforms CWG+SF while being trained for similar epochs. However, with layer-wise training, our approach consumes memory equivalent to its largest

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

layer, as there is no need to store neural activities and gradients. On CIFAR-10 input dimensions, our model consumes 0.82M parameters, whereas an identical BP (rep*) architecture requires 1.06M parameters.

Discussion. Our research is a proof of concept that the introduction of the Convolutional Channel-wise Goodness in the Forward-Forward Algorithm enables the algorithm to handle complex problems better through simultaneous feature extraction and separation. Our results demonstrate that CWG outperforms significantly other non-backpropagation approaches in both testing accuracy and convergence rate, especially in comparison with the other CNN-based FF implementations supporting our claim that Channel-wise Goodness enables the FF algorithm to work better with CNNs. This is a significant step forward for the Forward-Forward Algorithm since the gap observed between the original FF algorithm and the well-established backpropagation techniques is narrowed while eliminating the need for the construction of negative data. This was achieved without the utilization of high-level regularization and architecture techniques that appear in state-of-the-art backpropagation implementations and signifies the potential for further improvements in this direction.

References

[1] Steve Lawrence and C Lee Giles. Overfitting and neural networks: conjugate gradient and backpropagation. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 1, pages 114–119. IEEE, 2000.

[2] Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations, 2022.

[3] Takashi Shinozaki. Biologically-motivated deep learning method using hierarchical competitive learning, 2020.

[4] Rajesh P.N. Rao and Dana H. Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87, 1999.

[5] Alexander Ororbia and Ankur Mali. The predictive forward-forward algorithm, 2023.

[6] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, 2018.

[7] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning, 2017.

[8] Thomas Miconi. Hebbian learning with gradients: Hebbian convolutional neural networks with modern deep learning frameworks, 2021.

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[10] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[13] Alex Krizhevsky, Geoffrey Hinton, and et al. Learning multiple layers of features from tiny images. In *2009*, 2009.

[14] Gongpei Zhao, Tao Wang, Yidong Li, Yi Jin, Congyan Lang, and Haibin Ling. The cascaded forward algorithm for neural network training, 2023.