Optymalizacja pamięci w procesie uczenia sieci neuronowych i propagacja gradientu macierzami losowymi

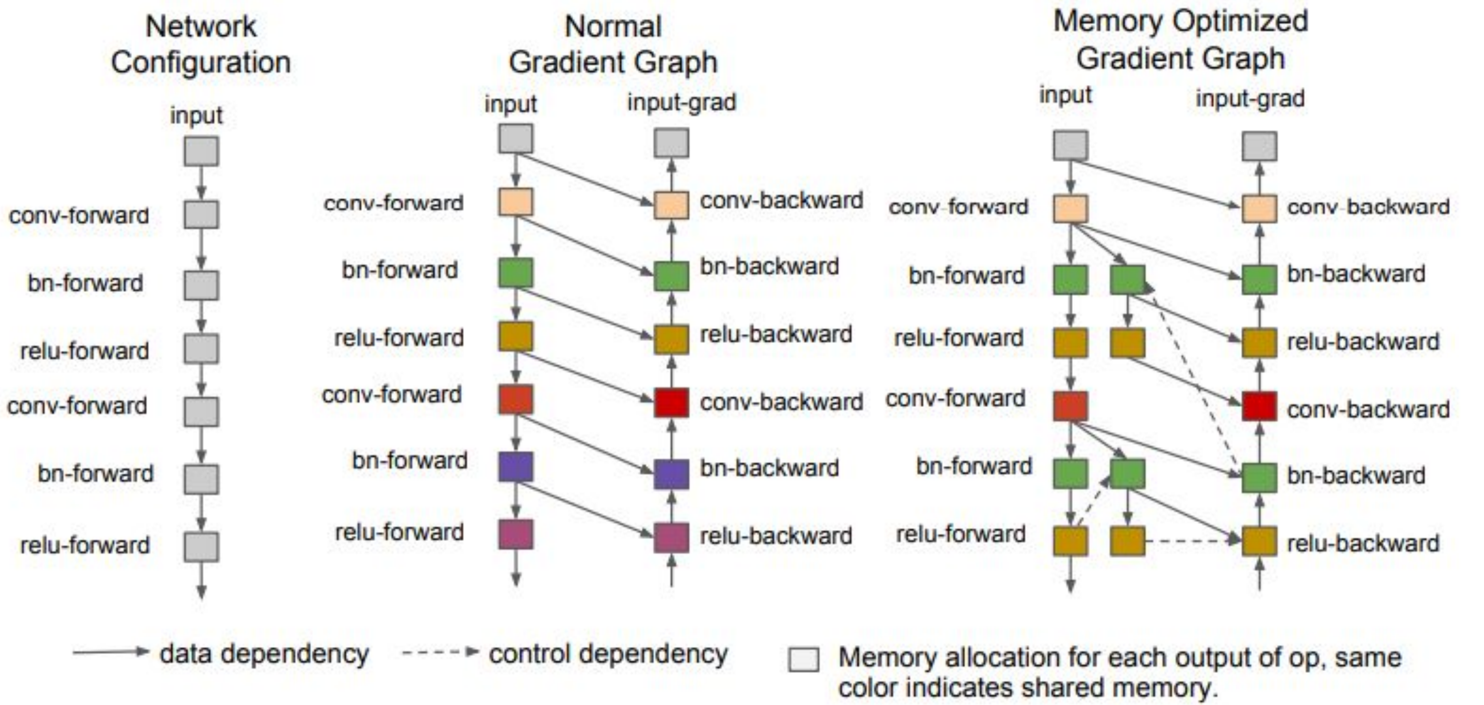
Koszt pamięciowy sieci

-Koszt modelu

-Koszt pamiętania wektorów aktywacji na potrzeby backpropagacji

Czemu musimy przetrzymywać wektory aktywacji?

Checkpointing



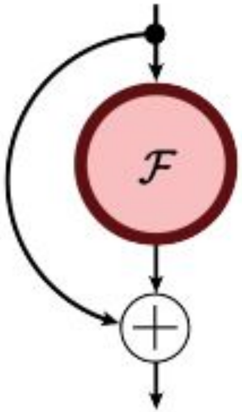
Odwracalność warstw

Szybka odwracalność warstw rozwiązuje problem przetrzymywania wektorów aktywacji

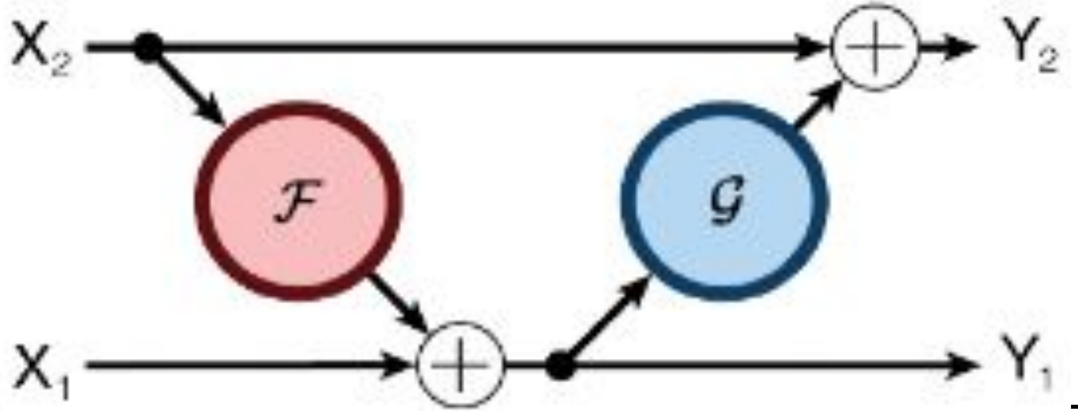
Architektura RevNet

Bloki RevNet są odwracalnym odpowiednikiem bloków ResNet

Blok ResNet



Blok RevNet



Architektura RevNet

Forward pass

$$y_1 = x_1 + \mathcal{F}(x_2)$$
$$y_2 = x_2 + \mathcal{G}(y_1)$$

Backward pass

$$x_2 = y_2 - \mathcal{G}(y_1)$$
$$x_1 = y_1 - \mathcal{F}(x_2)$$

Architecture	CIFAR-10 [15]		CIFAR-100 [15]	
	ResNet	RevNet	ResNet	RevNet
32 (38)	**7.14%**	7.24%	29.95%	**28.96%**
110	**5.74%**	5.76%	26.44%	**25.40%**
164	5.24%	**5.17%**	**23.37%**	23.69%

MemCNN - PyTorch Framework

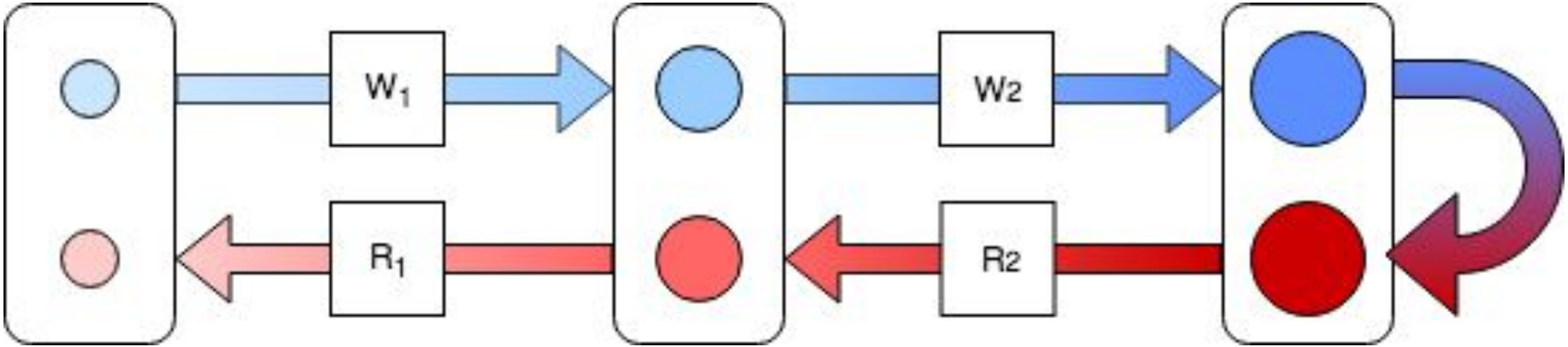
MEMCNN: A FRAMEWORK FOR DEVELOPING MEMORY EFFICIENT DEEP INVERTIBLE NETWORKS

Propagacje gradientu z wykorzystaniem macierzy losowych

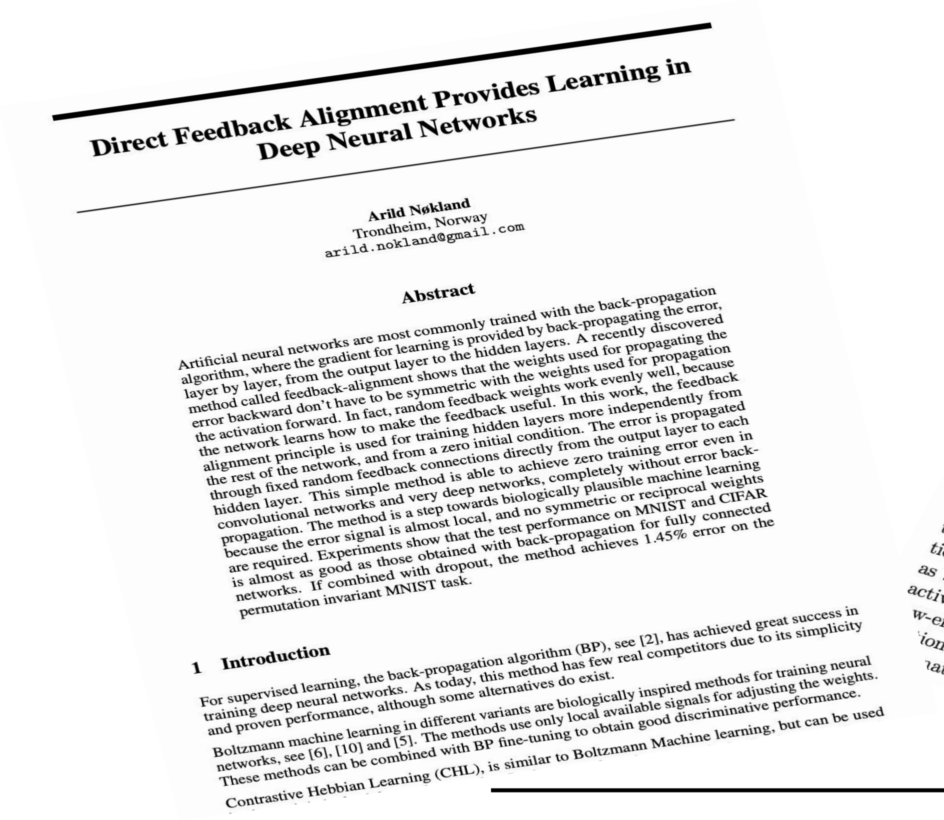
Backpropagacja jest wysoko zorganizowanym i precyzyjnym procesem, zbyt skomplikowanym jak na “architekturę” mózgu. W ostatnich latach powstało wiele algorytmów zmniejszających bardziej biologicznie prawdopodobnych.

Random Feedback Alignment

Algorytm FA polega na zastąpieniu mnożenia



Prace wprowadzające DFA



Direct Feedback Alignment Provides Learning in Deep Neural Networks

Arild Nøkland
Trondheim, Norway
arild.nokland@gmail.com

Abstract

Artificial neural networks are most commonly trained with the back-propagation algorithm, where the gradient for learning is provided by back-propagating the error, layer by layer, from the output layer to the hidden layers. A recently discovered method called feedback-alignment shows that the weights used for propagating the error backward don't have to be symmetric with the weights used for propagation the activation forward. In fact, random feedback weights work evenly well, because the network learns how to make the feedback useful. In this work, the feedback alignment principle is used for training hidden layers more independently from the rest of the network, and from a zero initial condition. The error is propagated through fixed random feedback connections directly from the output layer to each hidden layer. This simple method is able to achieve zero training error even in convolutional networks and very deep networks, completely without error back-propagation. The method is a step towards biologically plausible machine learning because the error signal is almost local, and no symmetric or reciprocal weights are required. Experiments show that the test performance on MNIST and CIFAR is almost as good as those obtained with back-propagation for fully connected networks. If combined with dropout, the method achieves 1.45% error on the permutation invariant MNIST task.

1 Introduction

For supervised learning, the back-propagation algorithm (BP), see [2], has achieved great success in training deep neural networks. As today, this method has few real competitors due to its simplicity and proven performance, although some alternatives do exist.

Boltzmann machine learning in different variants are biologically inspired methods for training neural networks, see [6], [10] and [5]. The methods use only local available signals for adjusting the weights. These methods can be combined with BP fine-tuning to obtain good discriminative performance.

Contrastive Hebbian Learning (CHL), is similar to Boltzmann Machine learning, but can be used

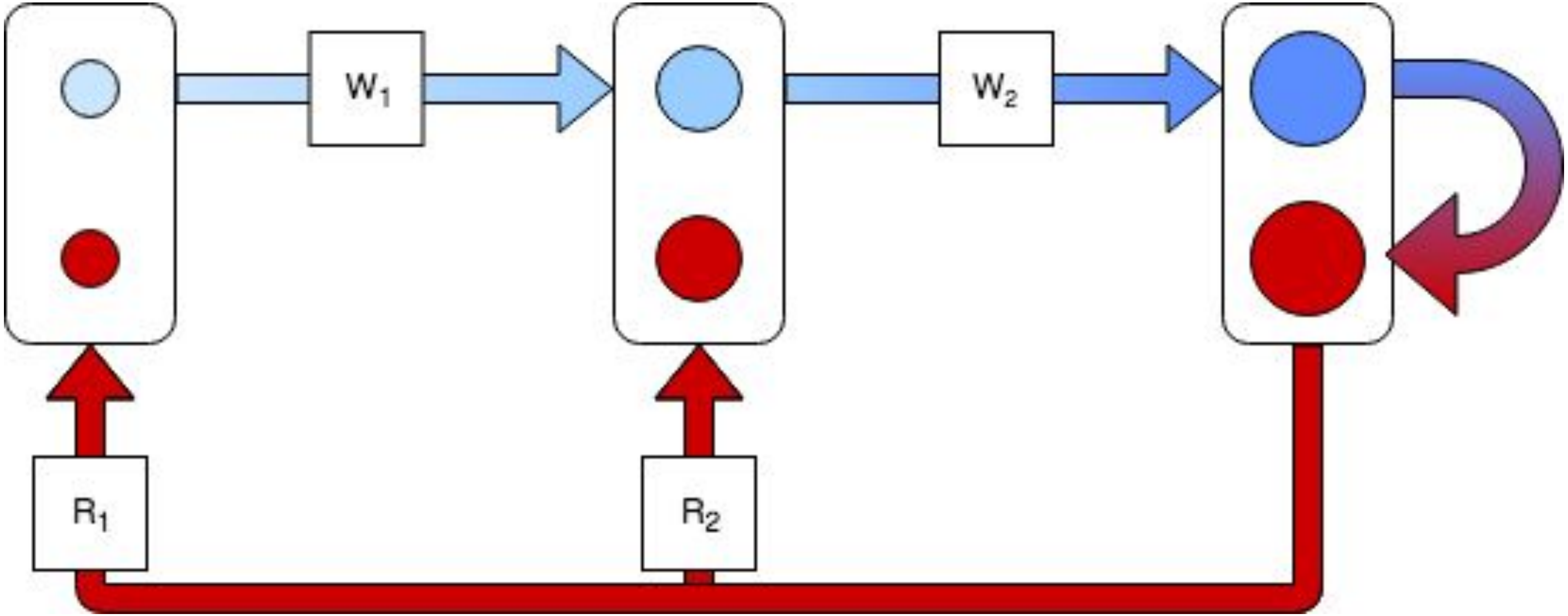
Learning in the Machine: Random Backpropagation and the Deep Learning Channel

Pierre Baldi[1,*], Peter Sadowski[1], and Zhiqin Lu[2]

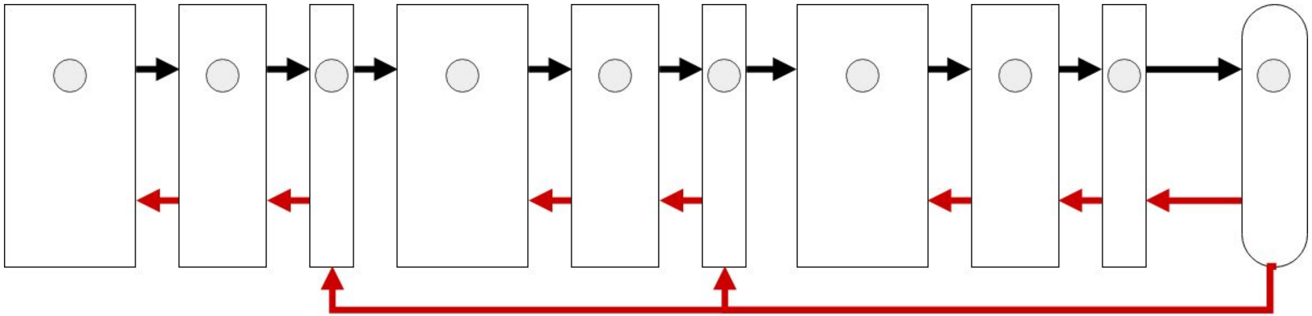
Abstract

Abstract: Random backpropagation (RBP) is a variant of the backpropagation algorithm for training neural networks, where the transpose of the forward matrices are replaced by fixed random matrices in the calculation of the weight updates. It is remarkable both because of its effectiveness, in spite of using random matrices to communicate error information, and because it completely removes the taxing requirement of maintaining symmetric weights in a physical neural system. To better understand random backpropagation, we first connect it to the notions of local learning and learning channels. Through this connection, we derive several alternatives to RBP, including skipped RBP (SRPB), adaptive RBP (ARBP), sparse RBP, and their combinations (e.g. ASRBP) and analyze their computational complexity. We then study their behavior through simulations using the MNIST and CIFAR-10 bechmark datasets. These simulations show that most of these variants work robustly, almost as well as backpropagation, and that multiplication by the derivatives of the activation functions is important. As a follow-up, we study also the ...w-end of the number of bits required to communicate error information over the learning channel. We then provide partial intuitive ...ations for some of the remarkable properties of RBP and its

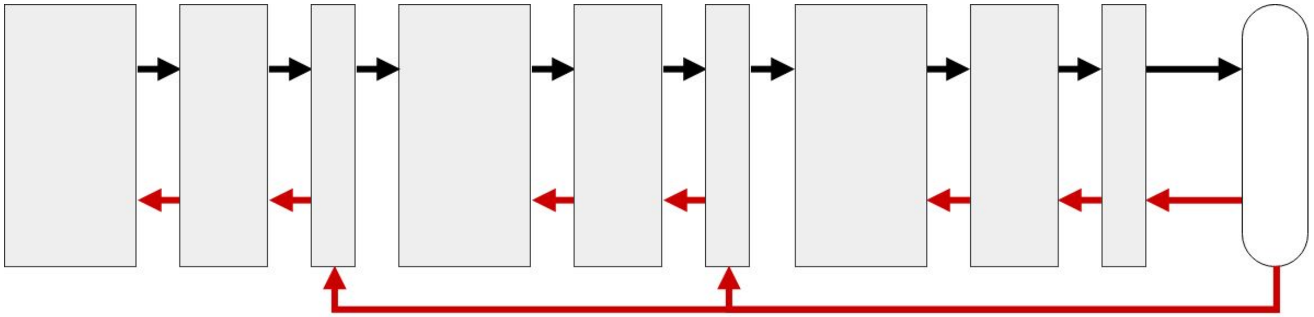
Direct Feedback Alignment



Zwykła implementacja DFA

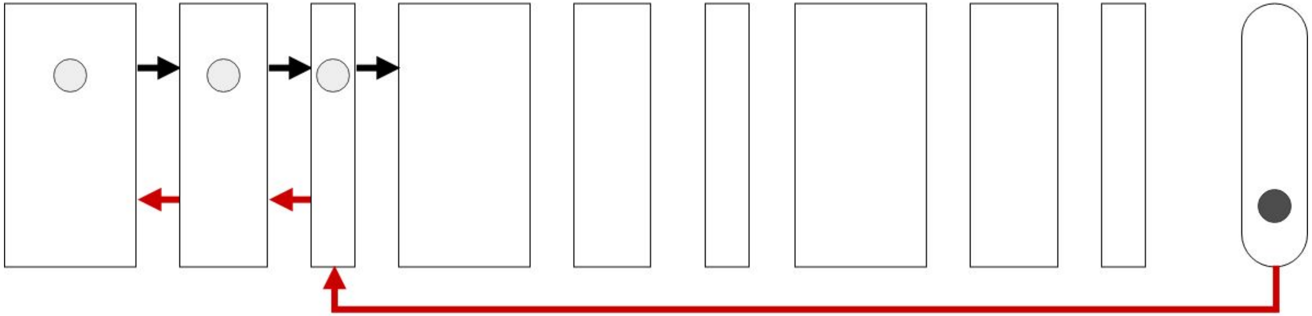


Zwykła implementacja DFA



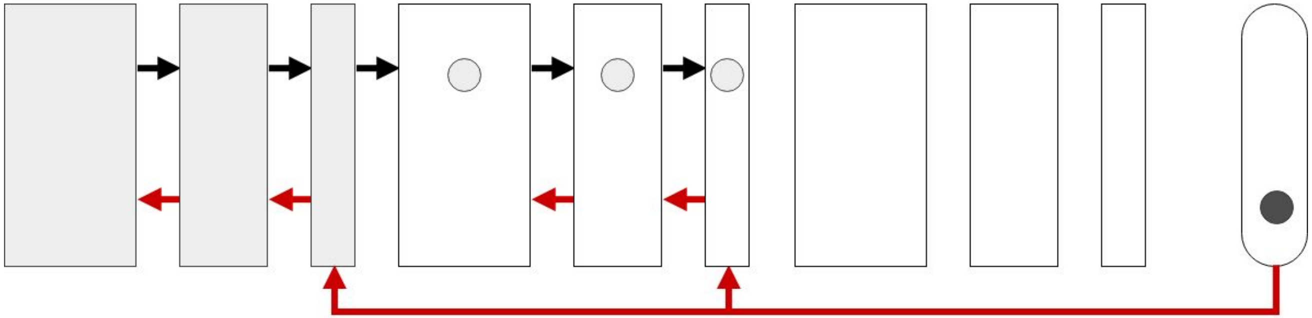
Oszczędna pamięciowo implementacja DFA

W implementacji DFA zaproponowanej przez nas w naszej pracy zastąpiliśmy forward i backward pass dwoma forward passami. Podczas pierwszego nie zapamiętujemy żadnej aktywacji poza ostatnią.

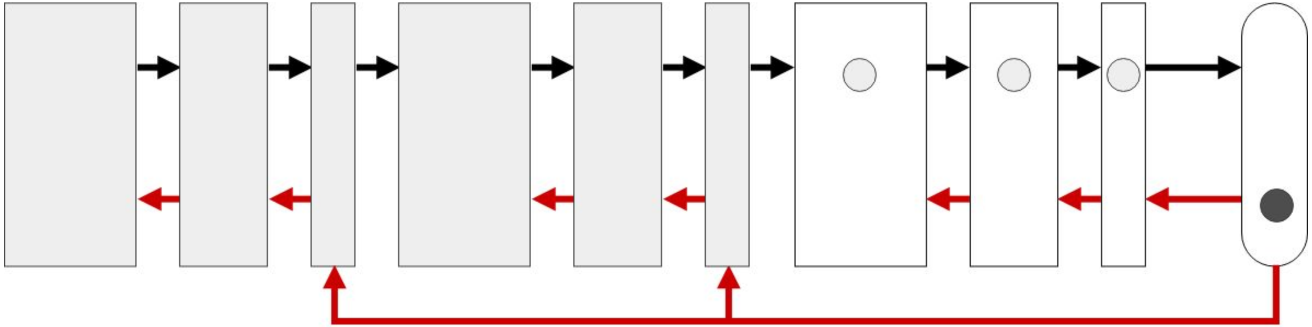


Oszczędna pamięciowo implementacja DFA

Podczas drugiego forward pass używamy zapamiętanej ostatniej aktywacji do obliczania gradientu w każdej warstwie.



Oszczędna pamięciowo implementacja DFA



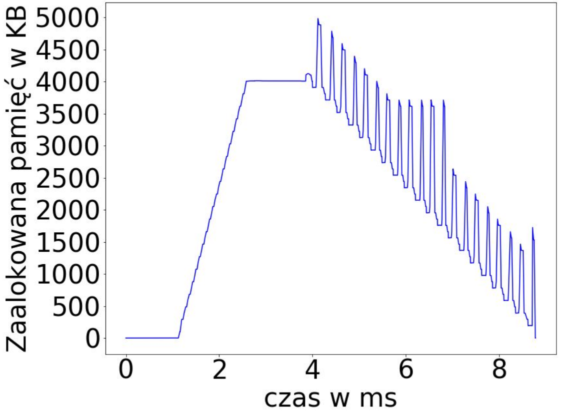
Drobne problemy techniczne

Profilowanie zużycia pamięci

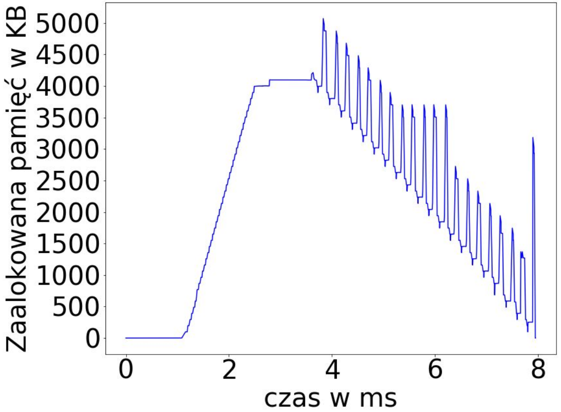
Inwazyjna paralelizacja TensorFlowa

Wyniki pomiarów pamięci (20 warstw Conv)

Propagacja wsteczna

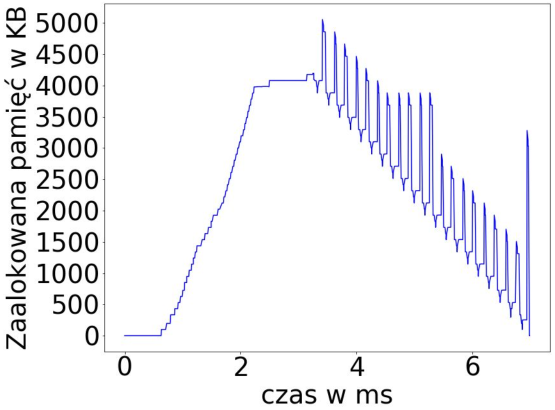


Random Feedback Alignment

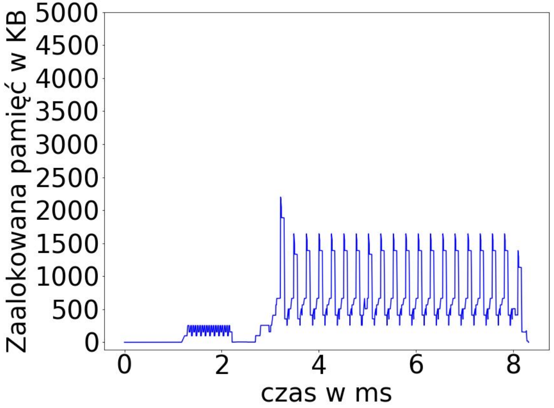


Wyniki pomiarów pamięci (20 warstw Conv)

Direct Feedback Alignment

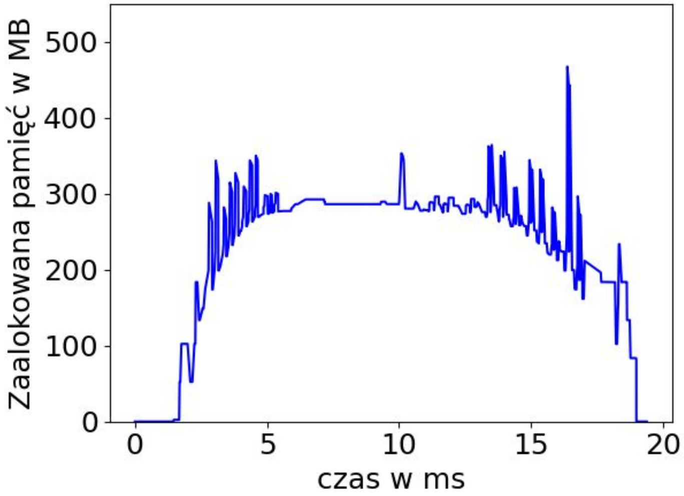


MEM-DFA

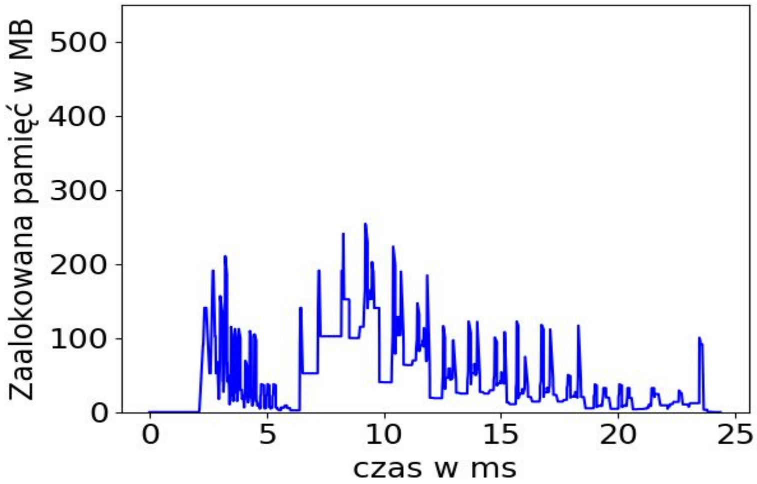


Bardziej praktyczna architektura (VGG-16)

Propagacja wsteczna



MEM-DFA



Problem z skalowaniem na większe zbiory danych

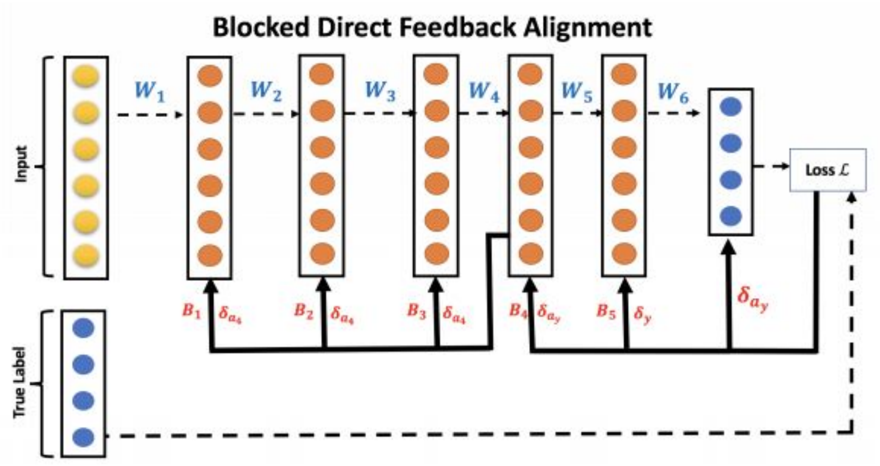
Rozmiar macierzy stosowanych do propagacji gradientu jest wprost proporcjonalny do ilości predykowanych klas.

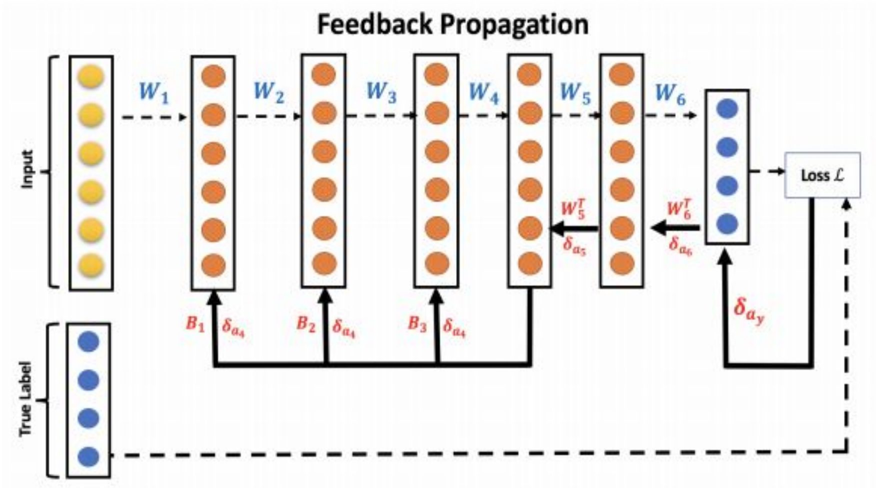
Direct Feedback Alignment with Sparse Connections for Local Learning

Można z powodzeniem stosować macierze rzadkie do propagowania błędu.

Benchmark	BP	DFA	SSDFA
MNIST	99.1	99.1	99.0
CIFAR10	77.1	77.8	78.0
CIFAR100	48.2	49.0	48.2
ImageNet (Alexnet)	49.0	48.8	46.3
ImageNet (VGG)	65.8	65.3	64.5

Blocked Direct Feedback Alignment: Exploring the Benefits of Direct Feedback Alignment





Część ciekawych prac

Dokładna analiza algorytmu FA : *Random feedback weights support learning in deep neural networks*

Jedna z prac wprowadzających DFA (tu nazwane SRBP) i dająca dużo intuicji: *Learning in the Machine: Random Backpropagation and the Deep Learning Channel*

Praca o wykorzystywaniu rzadkich macierzy: *Direct Feedback Alignment with Sparse Connections for Local Learning*

Praca o RevNetach: *Memcnn: a framework for developing memory efficient deep invertible networks*, *The reversible residual network: Backpropagation without storing activation*

Część ciekawych prac

Prace o checkpointingu: *Training deep nets with sublinear memory cost*, Memory-efficient backpropagation through time, *Cutting down training memory by re-fowarding*

Wydajne swapowanie pamięci RAM i VRAM: *Training deeper models by GPU memory optimization on TensorFlow*

Skrypty do profilowania pamięci:
`https://github.com/openai/gradient-checkpointing`

No i oczywiście nasze repo :) *https://github.com/chutien/zpp-mem*