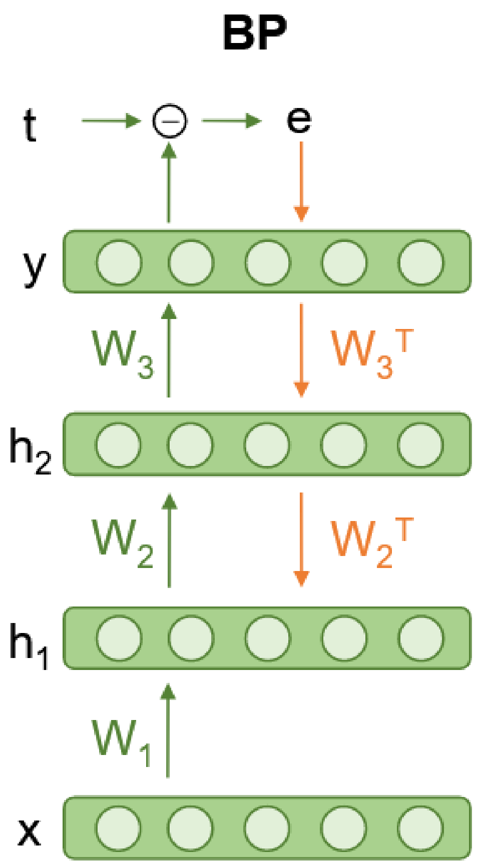
Error-driven Input Modulation: Solving the Credit Assignment Problem without a Backward Pass

Giorgia Dellaferrera & Gabriel Kreiman

ICML 2022

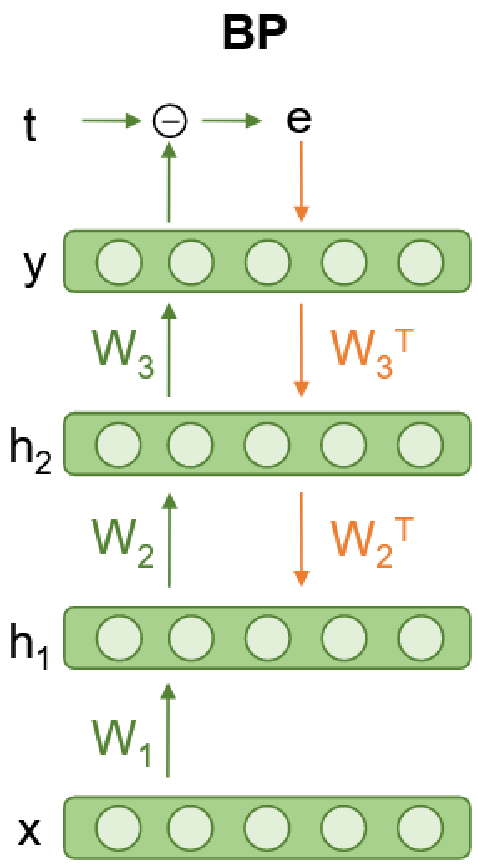
Backpropagation of the error is not biologically plausible



Rumelhart et al., 1995

Lillicrap et al., 2020

Backpropagation of the error is not biologically plausible

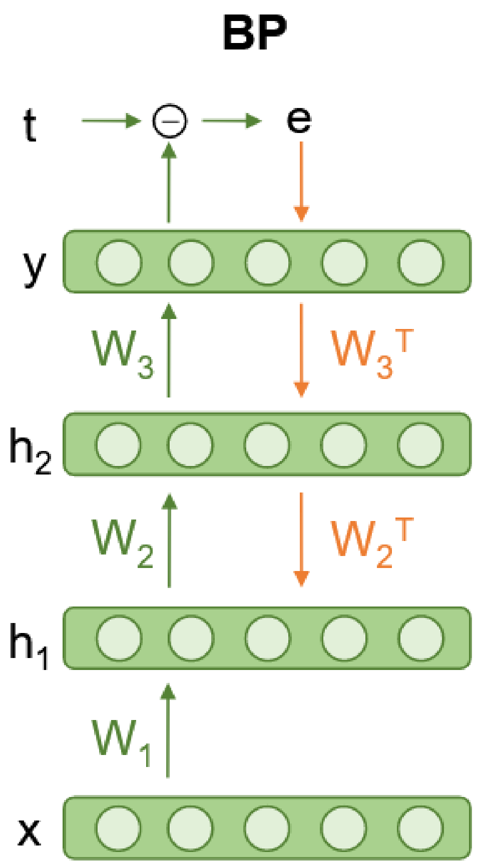


Rumelhart et al., 1995

» **Weight transport problem**
  - Symmetric weights for forward and backward computation

Lillicrap et al., 2020

Backpropagation of the error is not biologically plausible



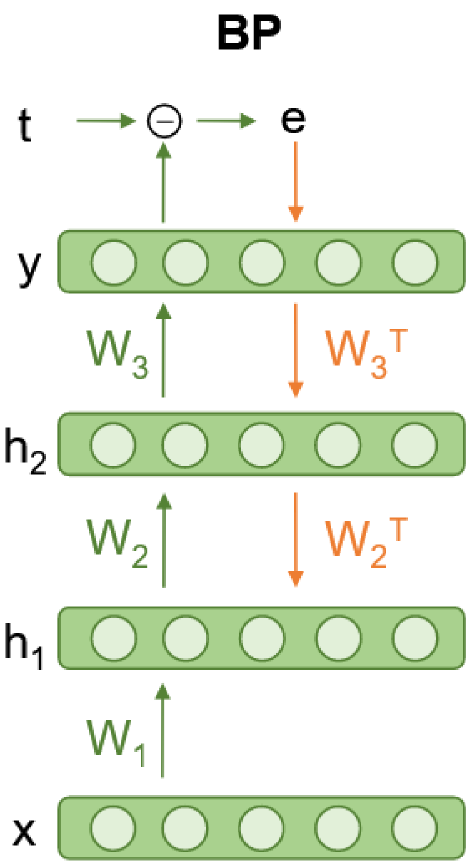
Rumelhart et al., 1995

» **Weight transport problem**
  - Symmetric weights for forward and backward computation

» **Non-local information used for the updates**
  - Global error and downstream weights needed for learning

Lillicrap et al., 2020

Backpropagation of the error is not biologically plausible



Rumelhart et al., 1995

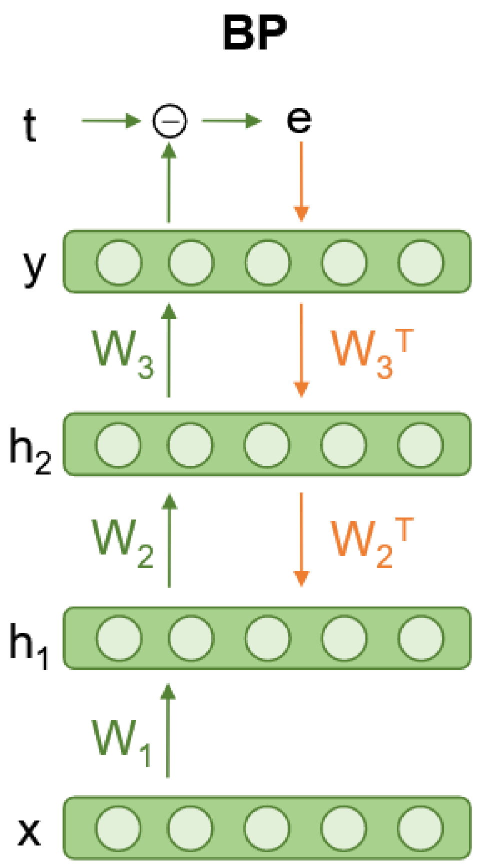
» **Weight transport problem**
  - Symmetric weights for forward and backward computation

» **Non-local information used for the updates**
  - Global error and downstream weights needed for learning

» **Frozen activity during error propagation and parameter updates**
  - Separate forward and backward computation

Lillicrap et al., 2020

Backpropagation of the error is not biologically plausible



Rumelhart et al., 1995

» **Weight transport problem**
- Symmetric weights for forward and backward computation

» **Non-local information used for the updates**
- Global error and downstream weights needed for learning

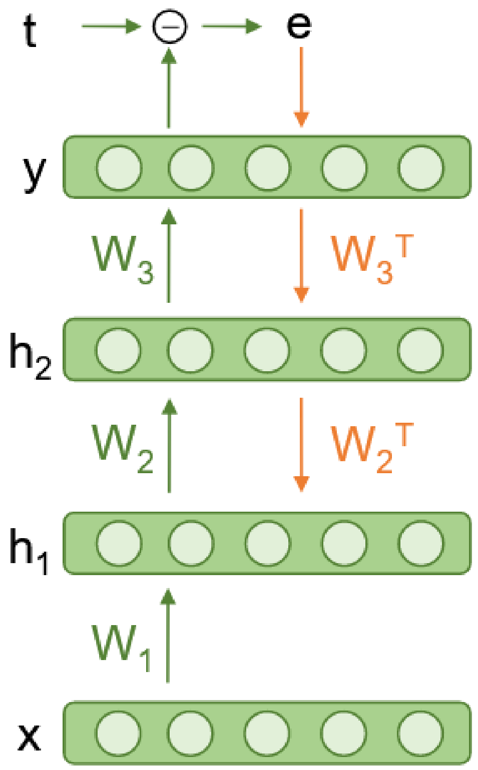
» **Frozen activity during error propagation and parameter updates**
- Separate forward and backward computation

» **Update locking problem**
- Backward computation needs to be complete before the next forward pass

Lillicrap et al., 2020

Backpropagation of the error is not biologically plausible

BP



Rumelhart et al., 1995

» **Weight transport problem**
- Symmetric weights for forward and backward computation

» **Non-local information used for the updates**
- Global error and downstream weights needed for learning

» **Frozen activity during error propagation and parameter updates**
- Separate forward and backward computation

» **Update locking problem**
- Backward computation needs to be complete before the next forward pass

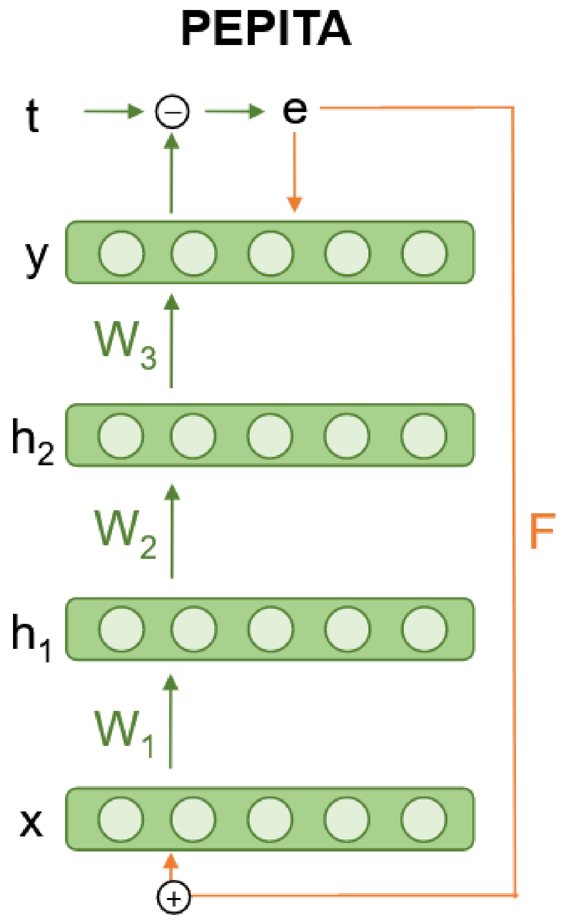
Alternative Training Schemes

Lillicrap et al., 2020

The PEPITA learning rule for Fully Connected Neural Networks

» **PEPITA** = Present the Error to Perturb the Input To modulate Activity

- Substitutes the standard Forward+Backward scheme with **two Forward Passes**

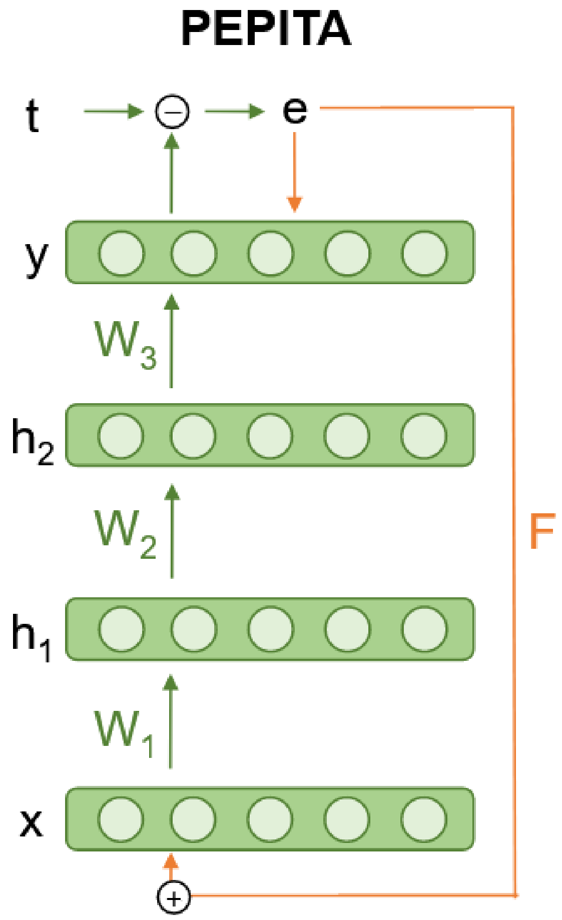


The PEPITA learning rule for Fully Connected Neural Networks

- **PEPITA** = Present the Error to Perturb the Input To modulate Activity
  - Substitutes the standard Forward+Backward scheme with **two Forward Passes**

Algorithm 1 Implementation of PEPITA

Given: Input (x) and label ($target$)
#standard forward pass
$h_0 = x$
for $\ell = 1, ..., L$
  $h_\ell = \sigma_\ell(W_\ell h_{\ell-1})$
$e = h_L - target$
#modulated forward pass
$h_0^{err} = x + Fe$
for $\ell = 1, ..., L$
  $h_\ell^{err} = \sigma_\ell(W_\ell h_{\ell-1}^{err})$
  if $\ell < L$:
    $\Delta W_\ell = (h_\ell - h_\ell^{err}) \cdot (h_{\ell-1}^{err})^T$
  else:
    $\Delta W_\ell = e \cdot (h_{\ell-1}^{err})^T$



The PEPITA learning rule for Fully Connected Neural Networks

» **PEPITA** = Present the Error to Perturb the Input To modulate Activity

- Substitutes the standard Forward+Backward scheme with **two Forward Passes**

Algorithm 1 Implementation of PEPITA

Given: Input (x) and label ($target$)

#standard forward pass

$h_0 = x$

for ℓ = 1, ..., L

$\quad h_\ell = \sigma_\ell(W_\ell h_{\ell-1})$

$e = h_L -$ $target$

#modulated forward pass

$h_0^{err} = x + Fe$

for ℓ = 1, ..., L

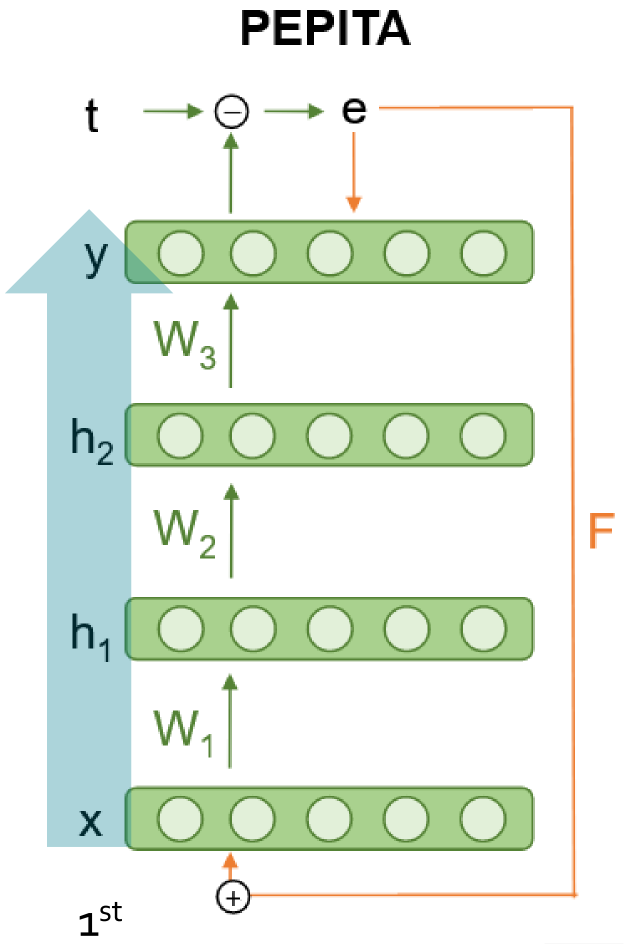
$\quad h_\ell^{err} = \sigma_\ell(W_\ell h_{\ell-1}^{err})$

$\quad$ **if** $\ell < L$:

$\quad\quad \Delta W_\ell = (h_\ell - h_\ell^{err}) \cdot (h_{\ell-1}^{err})^T$

$\quad$ **else:**

$\quad\quad \Delta W_\ell = e \cdot (h_{\ell-1}^{err})^T$



The PEPITA learning rule for Fully Connected Neural Networks

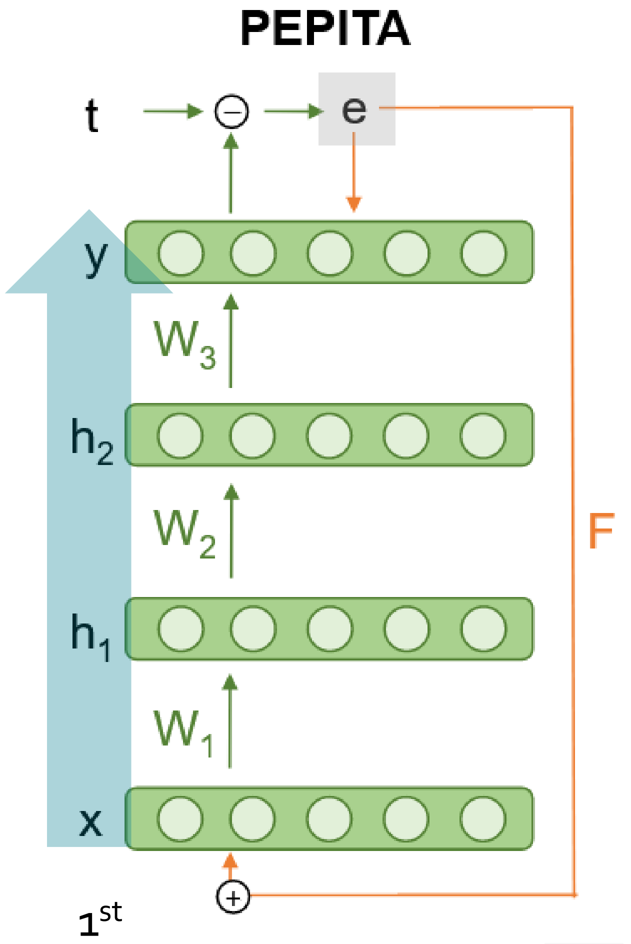
» **PEPITA** = Present the Error to Perturb the Input To modulate Activity

- Substitutes the standard Forward+Backward scheme with **two Forward Passes**

Algorithm 1 Implementation of PEPITA

Given: Input (x) and label ($target$)
#standard forward pass
$h_0 = x$
for $\ell = 1, ..., L$
  $h_\ell = \sigma_\ell(W_\ell h_{\ell-1})$
$e = h_L - target$
#modulated forward pass
$h_0^{err} = x + Fe$
for $\ell = 1, ..., L$
  $h_\ell^{err} = \sigma_\ell(W_\ell h_{\ell-1}^{err})$
  if $\ell < L$:
    $\Delta W_\ell = (h_\ell - h_\ell^{err}) \cdot (h_{\ell-1}^{err})^T$
  else:
    $\Delta W_\ell = e \cdot (h_{\ell-1}^{err})^T$

» Present the Error ...



The PEPITA learning rule for Fully Connected Neural Networks

» **PEPITA** = Present the Error to Perturb the Input To modulate Activity

- Substitutes the standard Forward+Backward scheme with **two Forward Passes**

Algorithm 1 Implementation of PEPITA

Given: Input (x) and label (*target*)

#standard forward pass

$h_0 = x$

for $\ell = 1, ..., L$

$\quad h_\ell = \sigma_\ell(W_\ell h_{\ell-1})$

$e = h_L - \textit{target}$

#modulated forward pass

$h_0^{err} = x + Fe$

for $\ell = 1, ..., L$

$\quad h_\ell^{err} = \sigma_\ell(W_\ell h_{\ell-1}^{err})$

$\quad$ **if** $\ell < L$:

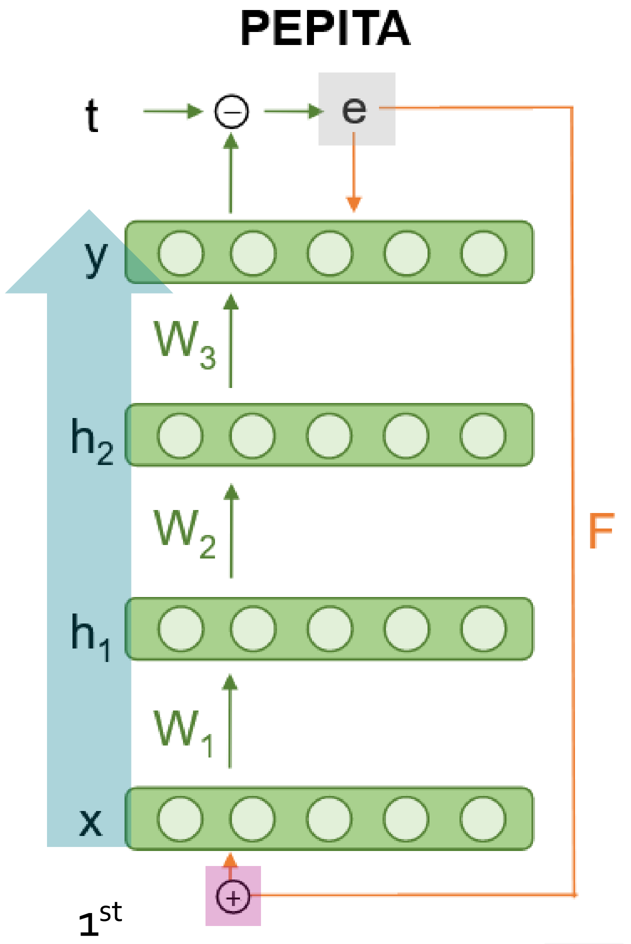
$\qquad \Delta W_\ell = (h_\ell - h_\ell^{err}) \cdot (h_{\ell-1}^{err})^T$

$\quad$ **else:**

$\qquad \Delta W_\ell = e \cdot (h_{\ell-1}^{err})^T$

» Present the Error ...

» ... to Perturb the Input...



The PEPITA learning rule for Fully Connected Neural Networks

» **PEPITA** = Present the Error to Perturb the Input To modulate Activity

- Substitutes the standard Forward+Backward scheme with **two Forward Passes**

Algorithm 1 Implementation of PEPITA

Given: Input (x) and label ($target$)

#standard forward pass

$h_0 = x$

for ℓ = 1, ..., L

$h_\ell = \sigma_\ell(W_\ell h_{\ell-1})$

$e = h_L - target$ ⟵ » Present the Error ...

#modulated forward pass

$h_0^{err} = x + Fe$ ⟵ » ... to Perturb the Input...

for ℓ = 1, ..., L

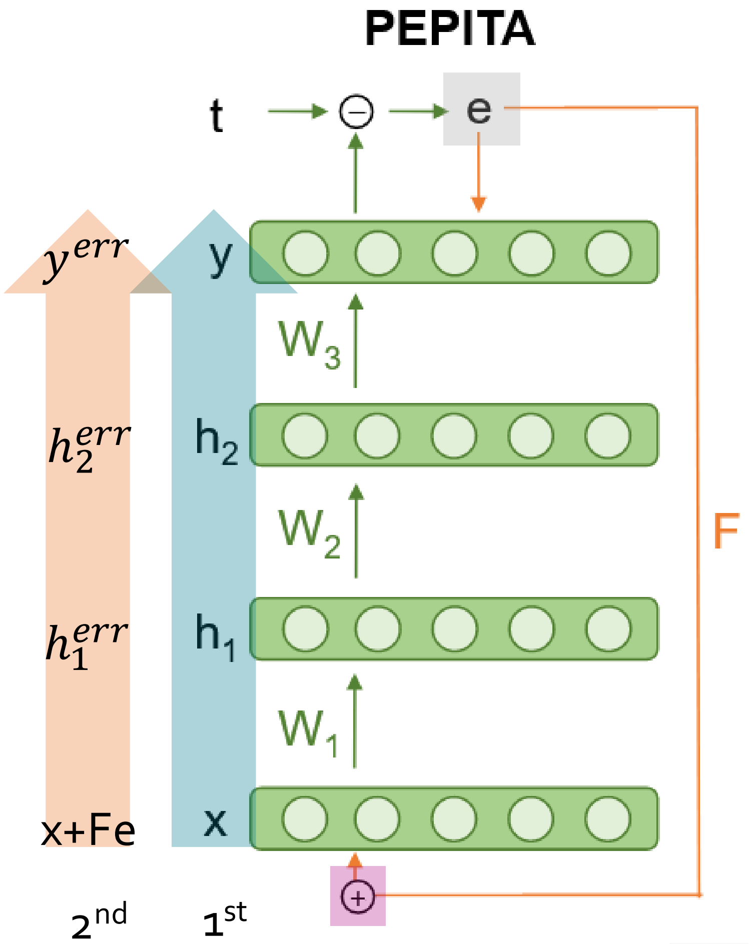
$h_\ell^{err} = \sigma_\ell(W_\ell h_{\ell-1}^{err})$ ⟵ » ... To modulate Activity

if $\ell < L$:

$\Delta W_\ell = (h_\ell - h_\ell^{err}) \cdot (h_{\ell-1}^{err})^T$

else:

$\Delta W_\ell = e \cdot (h_{\ell-1}^{err})^T$

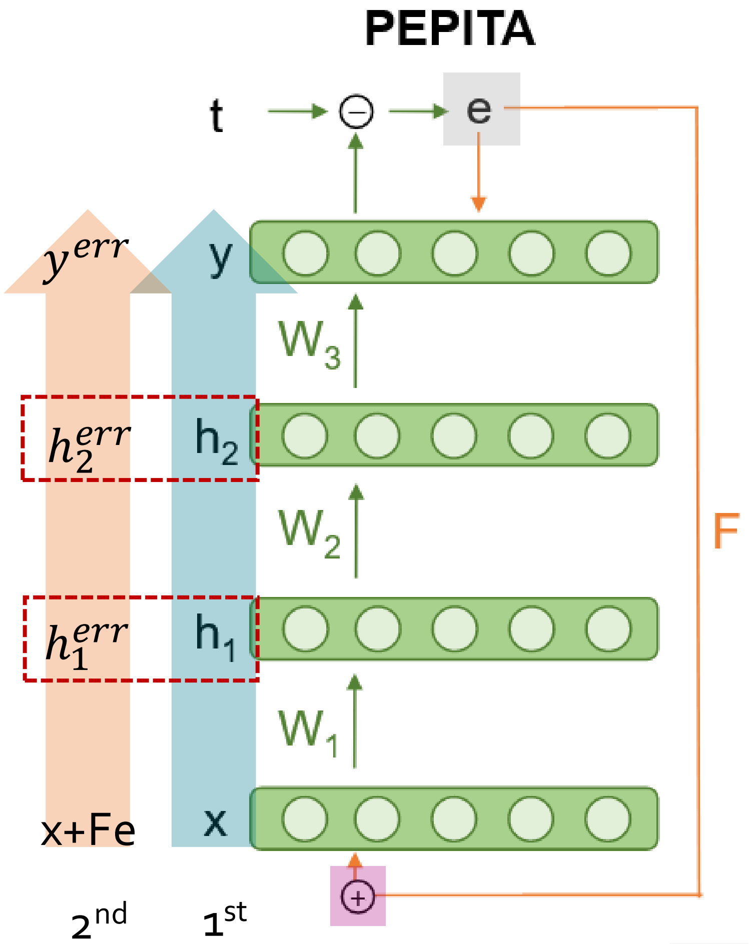


The PEPITA learning rule for Fully Connected Neural Networks

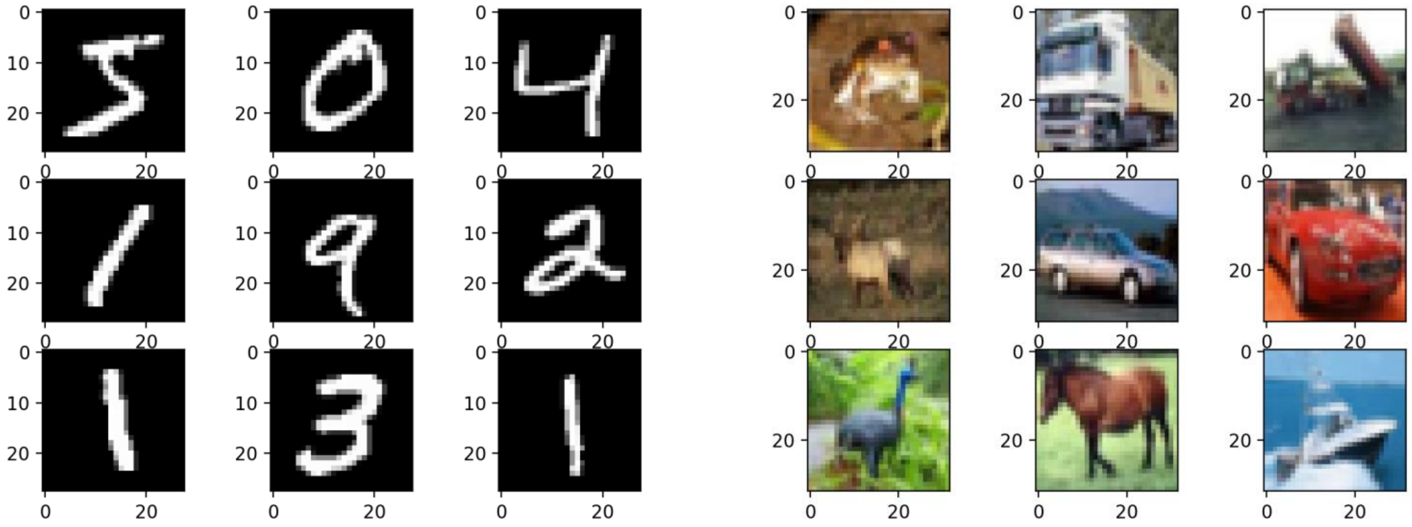
- **PEPITA** = Present the Error to Perturb the Input To modulate Activity
  - Substitutes the standard Forward+Backward scheme with **two Forward Passes**

Algorithm 1 Implementation of PEPITA

Given: Input (x) and label ($target$)
#standard forward pass
$h_0 = x$
for $\ell = 1, ..., L$
  $h_\ell = \sigma_\ell(W_\ell h_{\ell-1})$
$e = h_L - target$ ← » Present the Error ...
#modulated forward pass
$h_0^{err} = x + Fe$ ← » ... to Perturb the Input...
for $\ell = 1, ..., L$
  $h_\ell^{err} = \sigma_\ell(W_\ell h_{\ell-1}^{err})$ ← » ... To modulate Activity
  if $\ell < L$:
    $\Delta W_\ell = \boxed{(h_\ell - h_\ell^{err})} \cdot (h_{\ell-1}^{err})^T$
  else:
    $\Delta W_\ell = e \cdot (h_{\ell-1}^{err})^T$

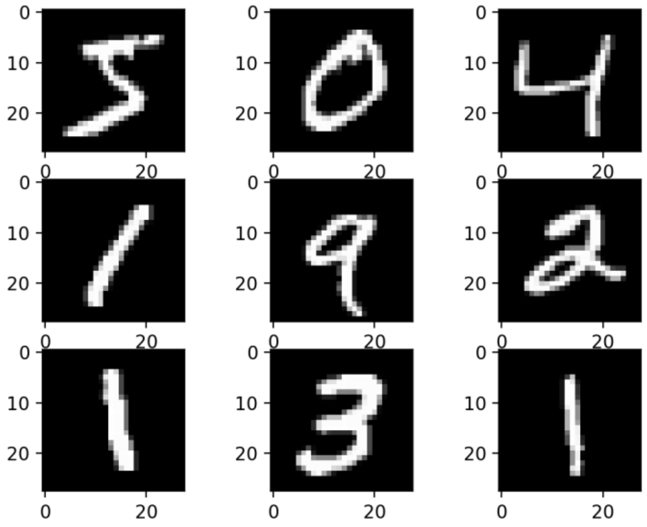


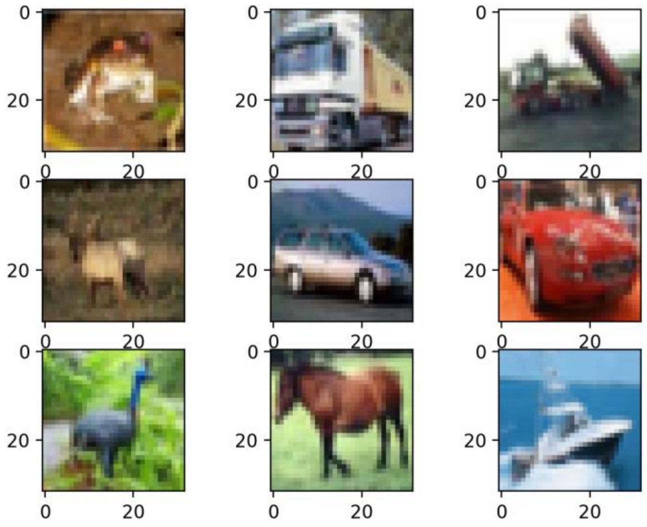
Testing PEPITA on image classification tasks - experimental results



	FULLY CONNECTED MODELS			CONVOLUTIONAL MODELS		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Testing PEPITA on image classification tasks - experimental results





Architecture:
1 hidden layer +
1 output layer

	FULLY CONNECTED MODELS			CONVOLUTIONAL MODELS		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Testing PEPITA on image classification tasks - experimental results

» Results for PEPITA are close to BP's performance

Architecture:
1 hidden layer +
1 output layer

	Fully connected models			Convolutional models		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Testing PEPITA on image classification tasks - experimental results

- Results for PEPITA are close to BP's performance
- In some tasks, PEPITA outperforms FA

Architecture:
1 hidden layer +
1 output layer

	Fully connected models			Convolutional models		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Testing PEPITA on image classification tasks - experimental results

- Results for PEPITA are close to BP's performance
- In some tasks, PEPITA outperforms FA
- PEPITA always outperforms DRTP

Architecture:
1 hidden layer +
1 output layer

	Fully connected models			Convolutional models		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Testing PEPITA on image classification tasks - experimental results

- Results for PEPITA are close to BP's performance
- In some tasks, PEPITA outperforms FA
- PEPITA always outperforms DRTP
- The PEPITA convolutional version
  - Useful 2D features

Architecture:
1 hidden layer +
1 output layer

	Fully connected models			Convolutional models		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Testing PEPITA on image classification tasks - experimental results

- Results for PEPITA are close to BP's performance
- In some tasks, PEPITA outperforms FA
- PEPITA always outperforms DRTP
- The PEPITA convolutional version
  - Useful 2D features
- Learning speed
  - Between BP (the fastest) and FA (the slowest)

Architecture:
1 hidden layer +
1 output layer

	FULLY CONNECTED MODELS			CONVOLUTIONAL MODELS		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Testing PEPITA on image classification tasks - experimental results

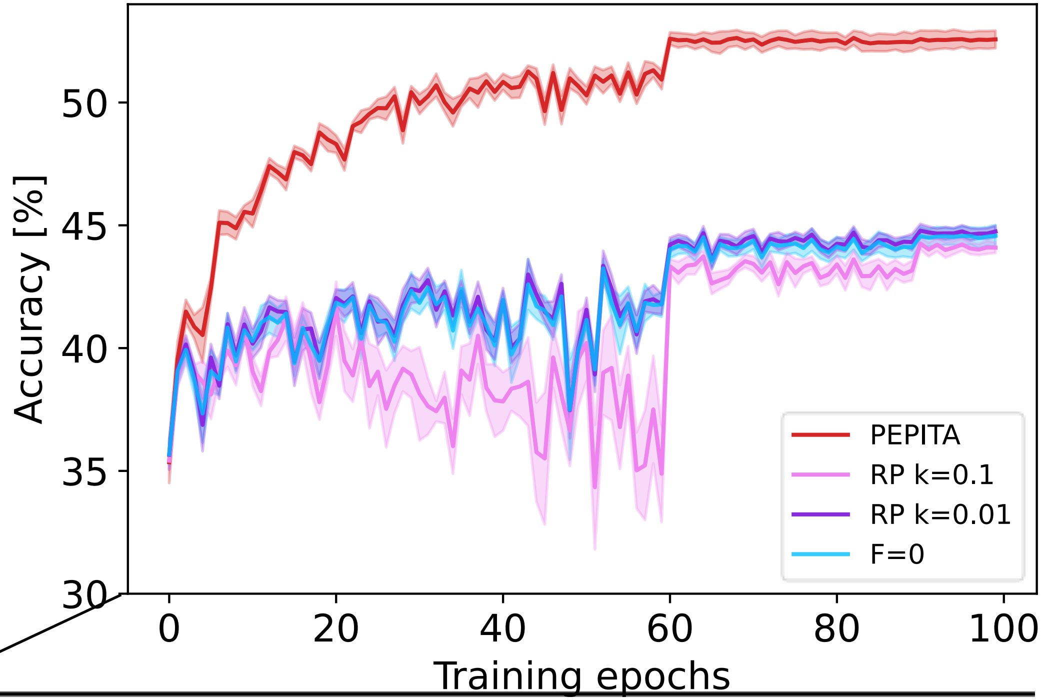
» Results for PEPITA are close to BP's performance

» In some tasks, PEPITA outperforms FA

» PEPITA always outperforms DRTP

» The PEPITA convolutional version
  - Useful 2D features

» Learning speed
  - Between BP (the fastest) and FA (the slowest)

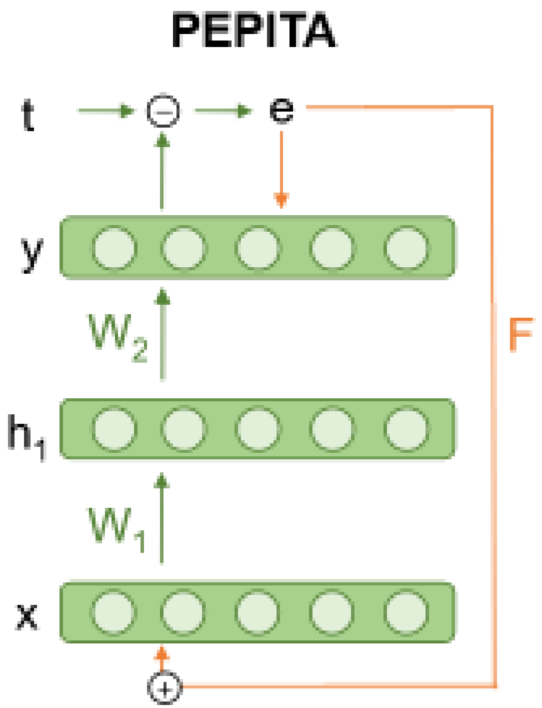


Architecture:
1 hidden layer +
1 output layer

	Fully connected models			Convolutional models		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

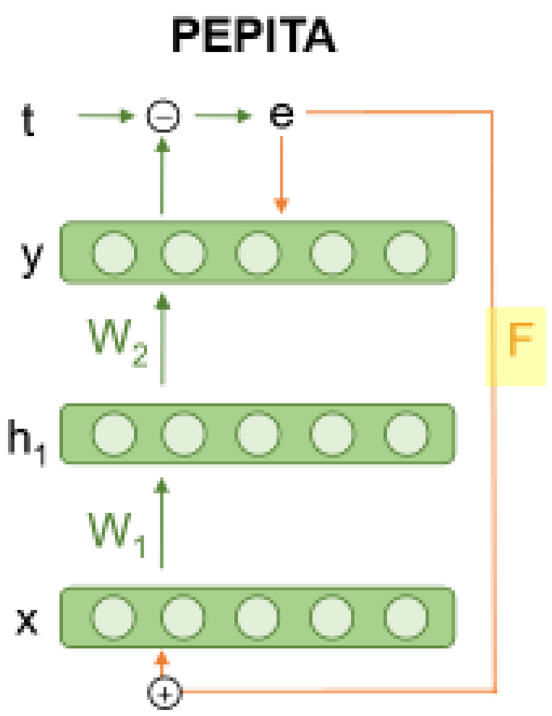
Why it works: soft-antialignment

» Soft-antialignment



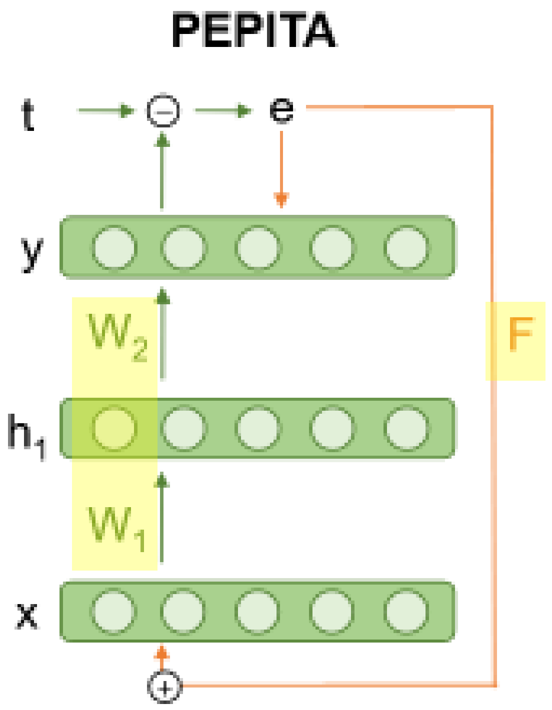
Why it works: soft-antialignment

» **Soft-antialignment**
- Angle between
  - projection matrix F and



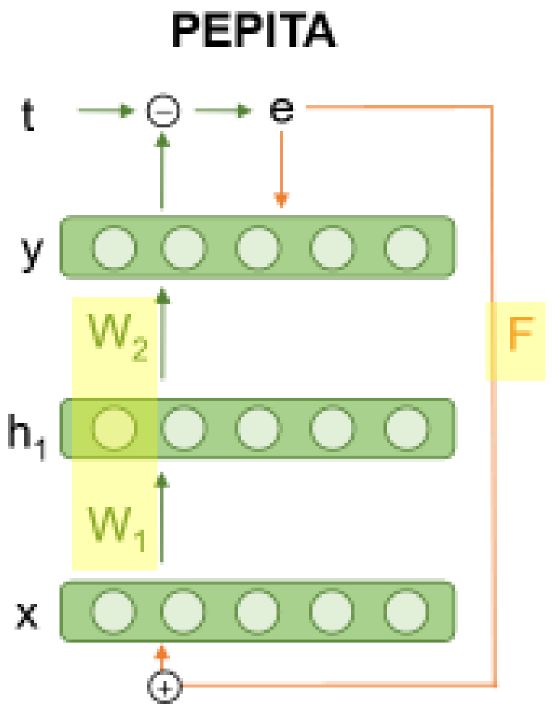
Why it works: soft-antialignment

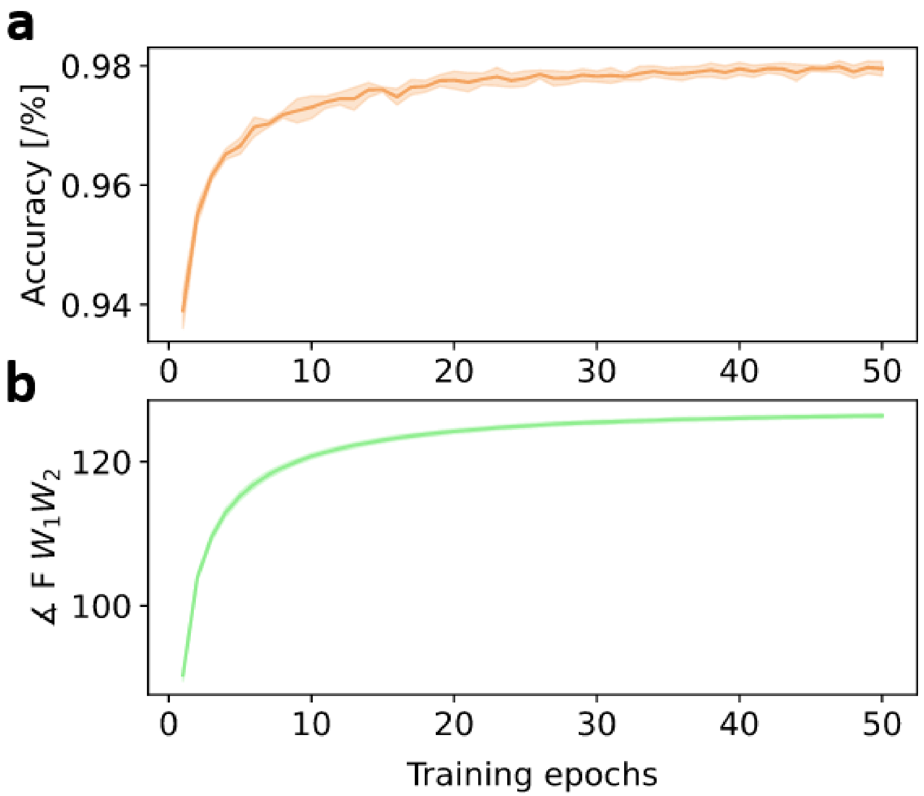
» **Soft-antialignment**
- Angle between
  - projection matrix F and
  - product between the forward weight matrices



Why it works: soft-antialignment

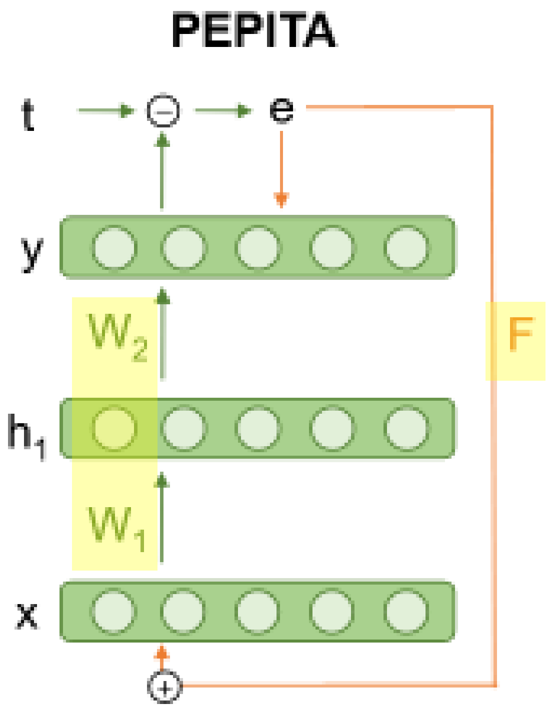
» **Soft-antialignment**
- Angle between
  - projection matrix F and
  - product between the forward weight matrices
- Evolution during learning → soft antialignment

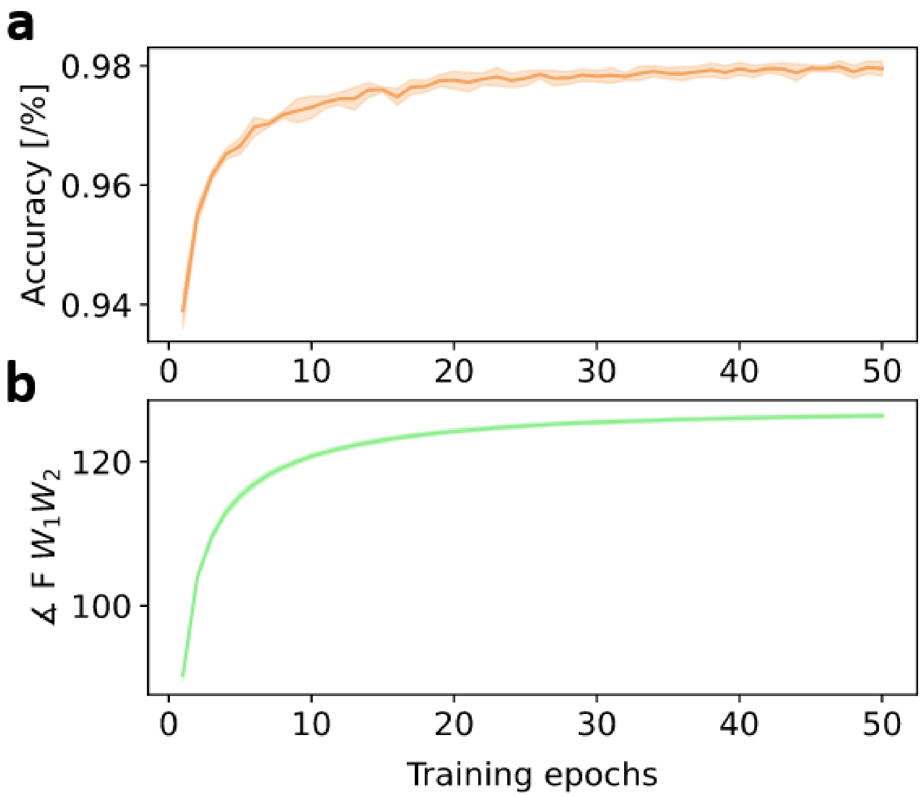




Why it works: soft-antialignment

» **Soft-antialignment**
- Angle between
  - projection matrix F and
  - product between the forward weight matrices
- Evolution during learning → soft antialignment
- <u>Analytically proven</u> for one-hidden layer linear network





Summary and Outlook

» **PEPITA**

- Is a novel training scheme relying only on **forward computations**
- Solves weight transport, freezing of neural activity, non-local weight updates and backward locking
- Achieves performance on-par with FA on simple image classification tasks

Summary and Outlook

» **PEPITA**

- Is a novel training scheme relying only on **forward computations**
- Solves weight transport, freezing of neural activity, non-local weight updates and backward locking
- Achieves performance on-par with FA on simple image classification tasks

» **Challenges**

- Performance **does not improve with depth**
- Different non-linearities
- Residual connection
- Training the F matrix

Thank you for your attention!

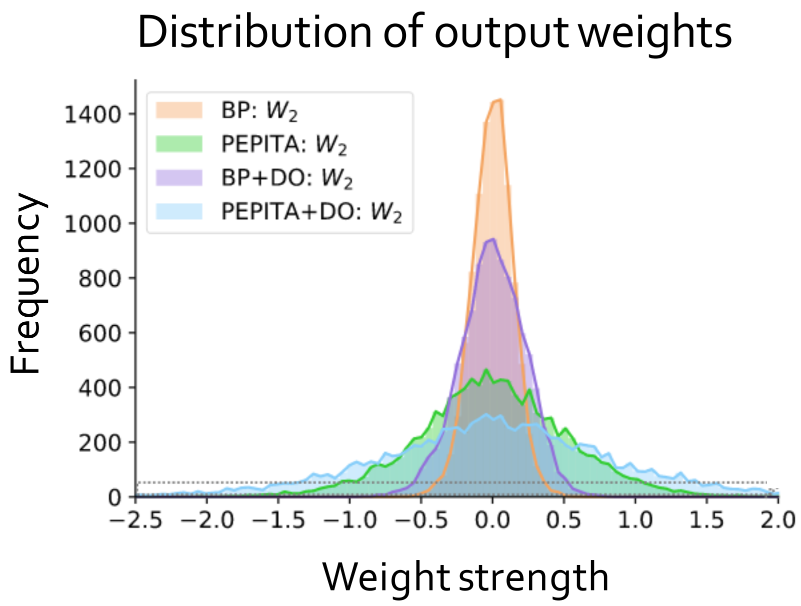
» Questions?
» Ideas?
» Suggestions?

✉ *gde@zurich.ibm.com*

Weight distribution after training

» **Wider weight distribution**

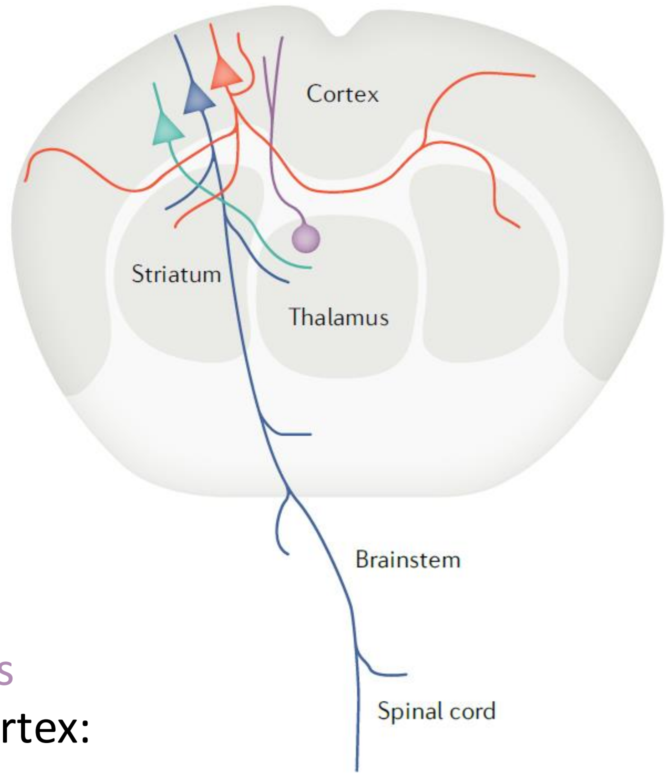
- PEPITA learns different solutions compared to BP



Analysis of PEPITA from a biological standpoint

PEPITA solves the BP's issues of biological plausibility, but it introduces additional elements:

» Projection of the error onto the input through a fixed random matrix
  - Reminiscent of cortico-thalamo-cortical loops



In the thalamus:
- Thalamocortical (TC) neurons

Excitatory neurons in the neocortex:
- Intratelencephalic (IT)
- Pyramidal tract (PT)
- Corticothalamic (CT) neurons

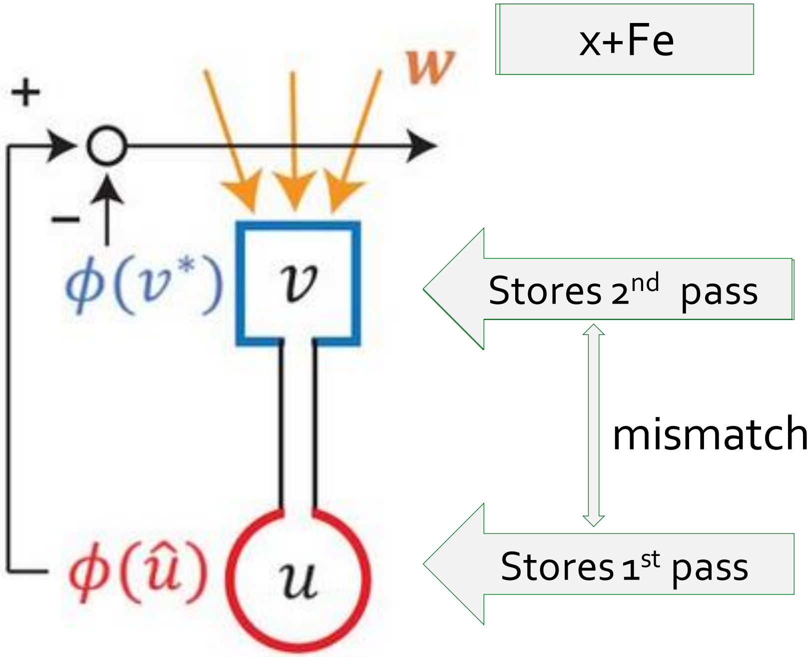
Shepherd & Yamawaki, 2021

Analysis of PEPITA from a biological standpoint

PEPITA solves the BP's issues of biological plausibility, but it introduces additional elements:

» Projection of the error onto the input through a fixed random matrix
  - Reminiscent of cortico-thalamo-cortical loops

» Storing of the activation of the *Standard pass* until the *Modulated pass*
  - Can be implemented in biological neurons through mismatch between dendritic and somatic activity



Asabuki & Fukai, 2020

The PEPITA learning rule

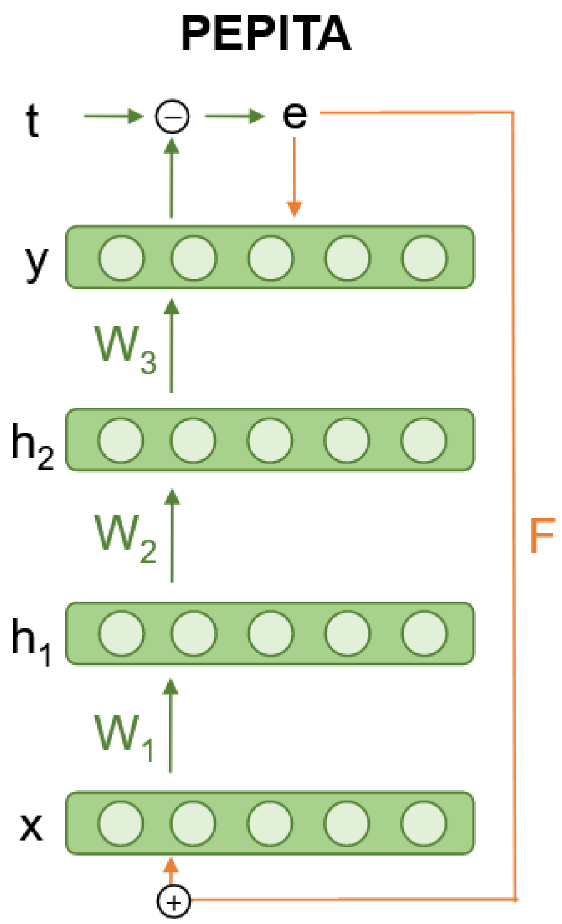
» **PEPITA** = Present the Error to Perturb the Input To modulate Activity

» Substitutes the standard Forward+Backward scheme with **two Forward Passes**

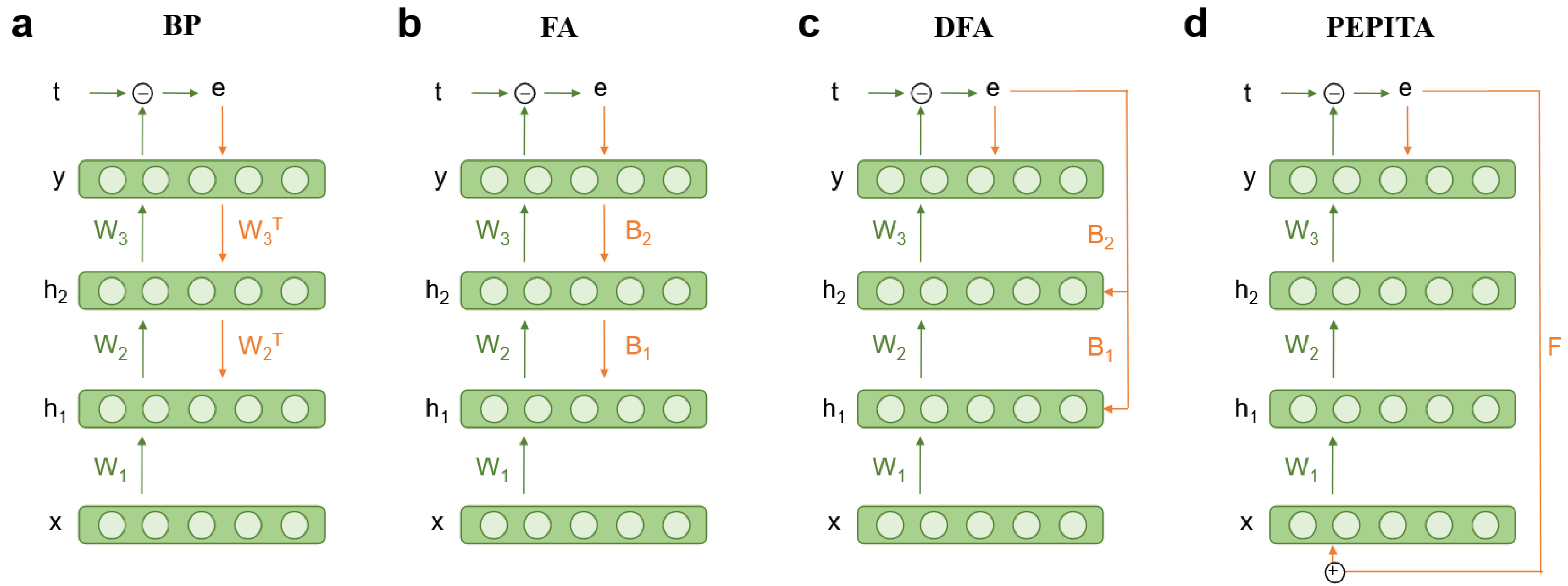
- *Standard Forward pass* → same as for standard algorithms
- *Modulated Forward pass* → input is modulated by the error

» F = **projection matrix** to add the error onto the input

» Update relies **on difference of activations** between *Standard* and *Modulated pass*

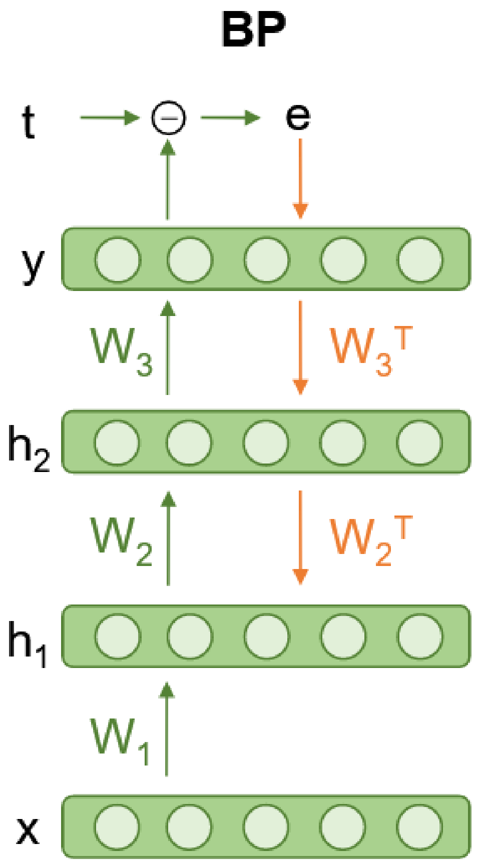


PEPITA solves the biologically implausible aspects of BP



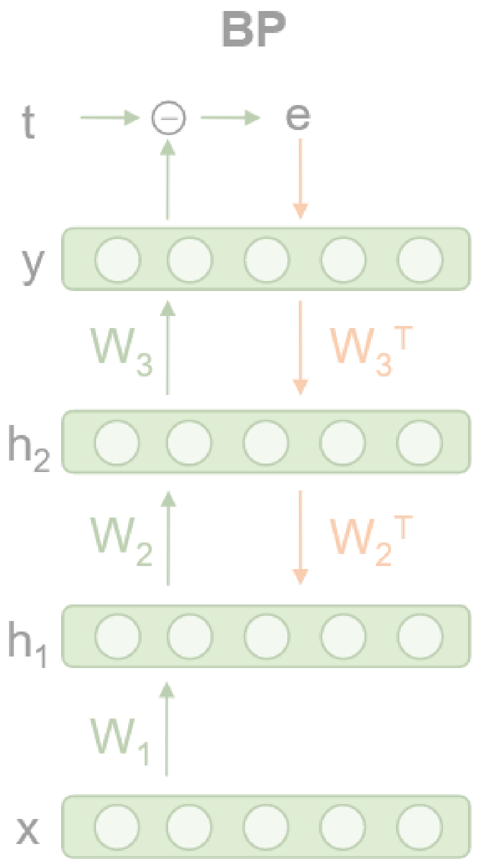
	BP	FA	DFA	PEPITA
$\Delta W_\ell =$	$-(W_{\ell+1}^T \delta a_{\ell+1}) \odot f'(a_\ell) h_{\ell-1}^T$	$-(B_\ell^T \delta a_{\ell+1}) \odot f'(a_\ell) h_{\ell-1}^T$	$-(B_\ell^T e) \odot f'(a_\ell) h_{\ell-1}^T$	$(h_\ell - h_\ell^{err}) \cdot (h_{\ell-1}^{err\,T})$
WEIGHT-TRANSPORT-FREE	✗	✓	✓	✓
LOCAL RULE	✗	✗	✗	✓
FREEZING OF ACTIVITY	✗	✗	✗	✓
UPDATE-UNLOCKED	✗	✗	PARTIALLY	PARTIALLY

Alternatives to BP: relaxing symmetry requirements

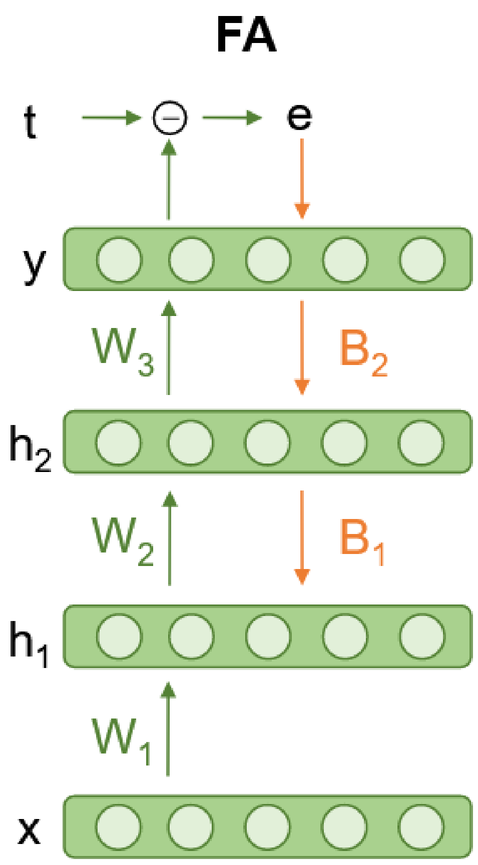


Rumelhart et al., 1995

Alternatives to BP: relaxing symmetry requirements

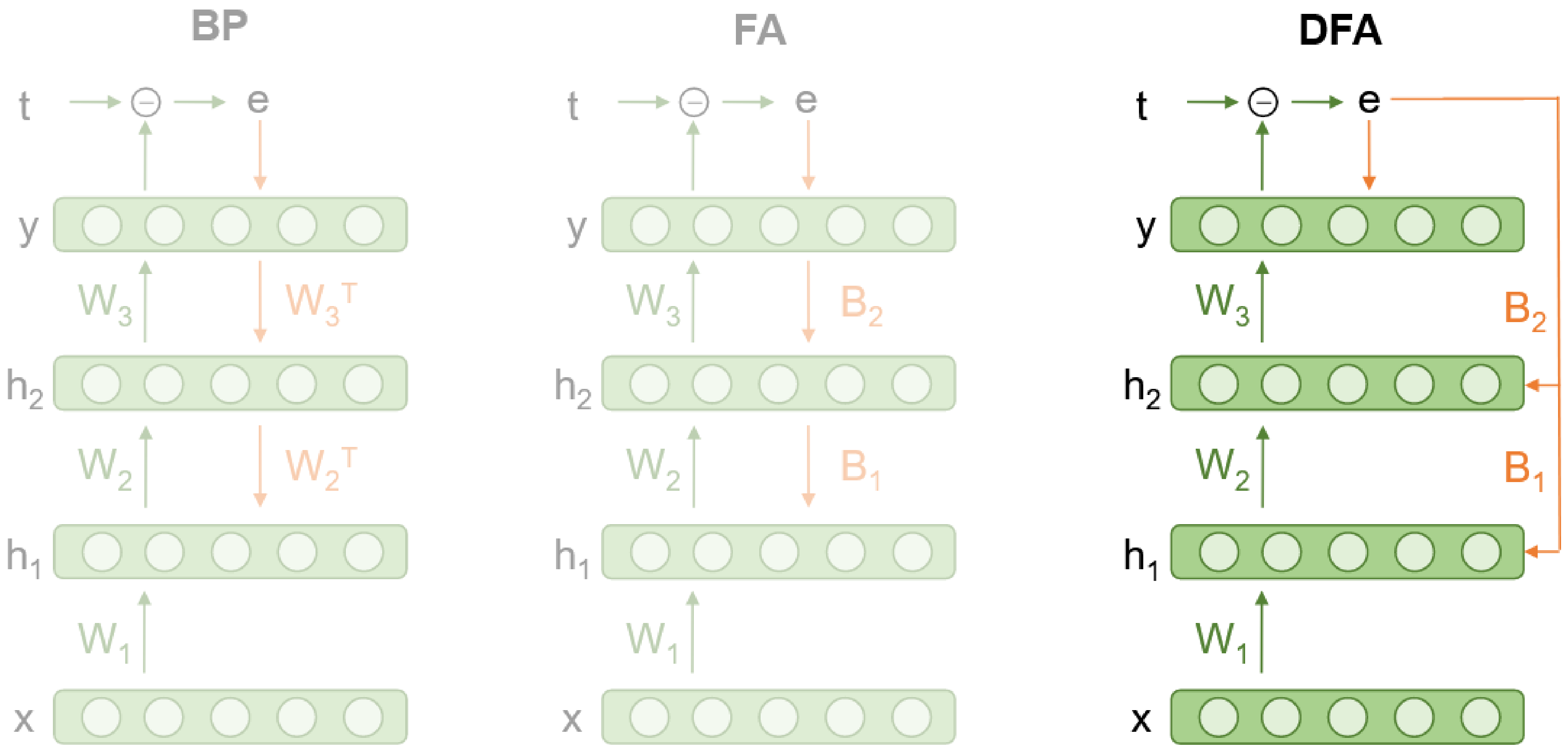


Rumelhart et al., 1995



Lillicrap et al., 2016

Alternatives to BP: relaxing symmetry requirements

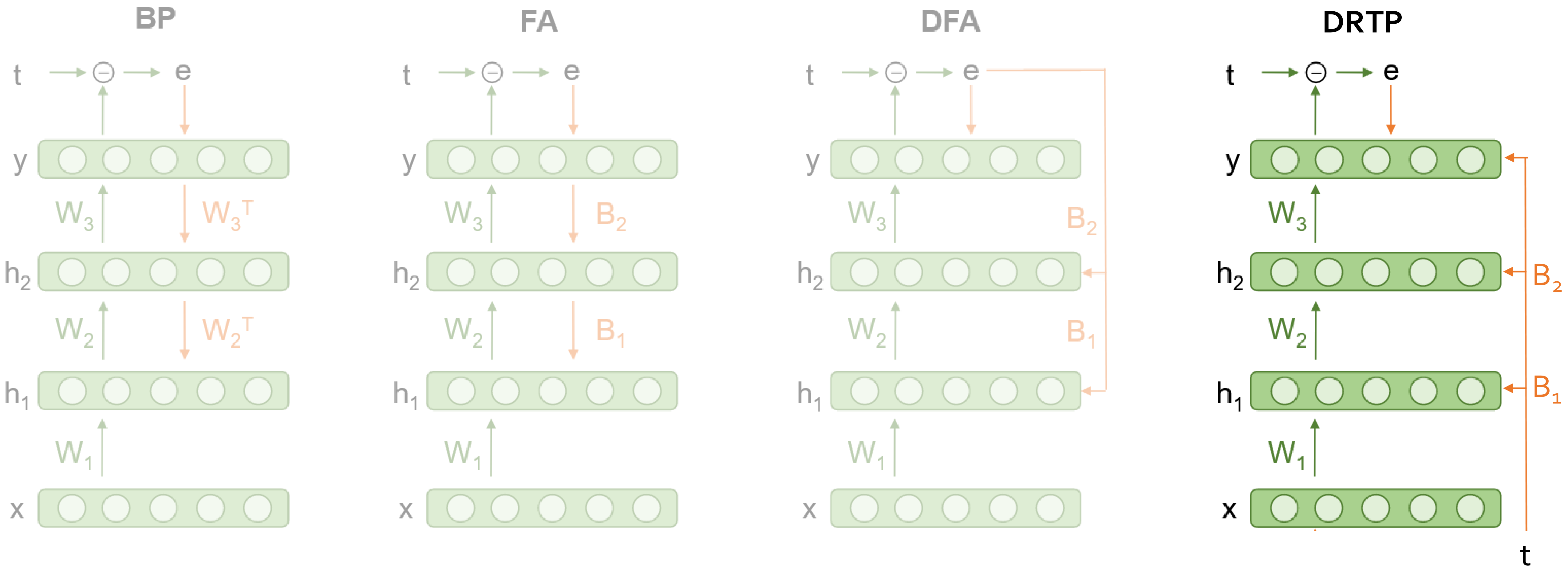


Rumelhart et al., 1995

Lillicrap et al., 2016

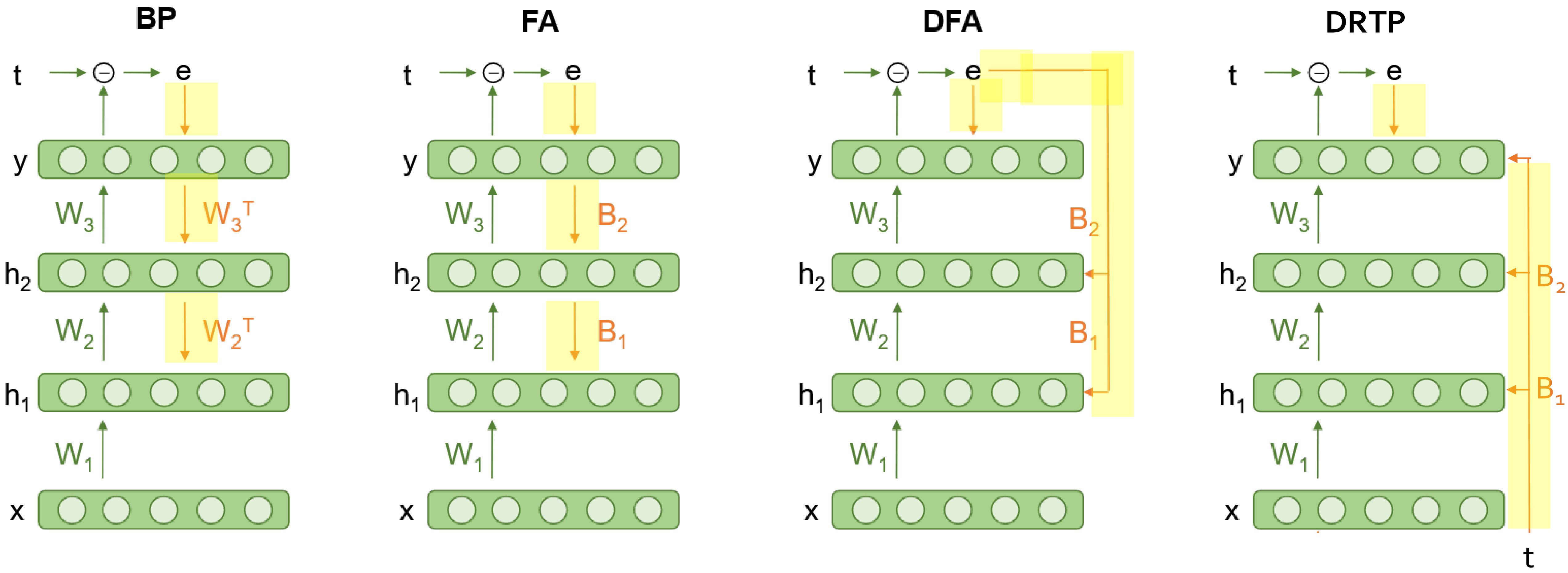
A. Nokland, 2016

Alternatives to BP: relaxing symmetry requirements



Rumelhart et al., 1995 | Lillicrap et al., 2016 | A. Nokland, 2016 | Frenkel et al., 2019

Alternatives to BP: relaxing symmetry requirements

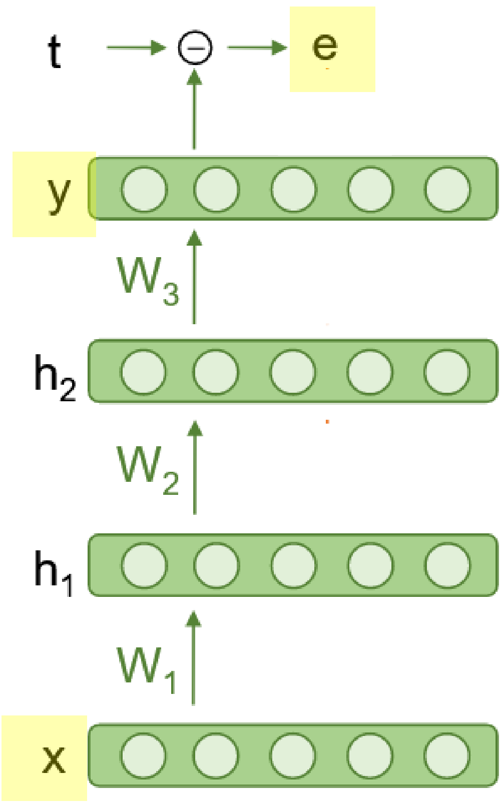


Rumelhart et al., 1995   Lillicrap et al., 2016   A. Nokland, 2016   Frenkel et al., 2019

The backpropagation algorithm

» **Forward pass**
- Network's response to input
- Error function $e = y - t$
- Weight updates proportional to its negative gradient

» **Backward pass**
- Error signal flows backward through the network
- Computed recursively via the chain rule
- Update phase



Lillicrap et al., 2020

The Backward Pass

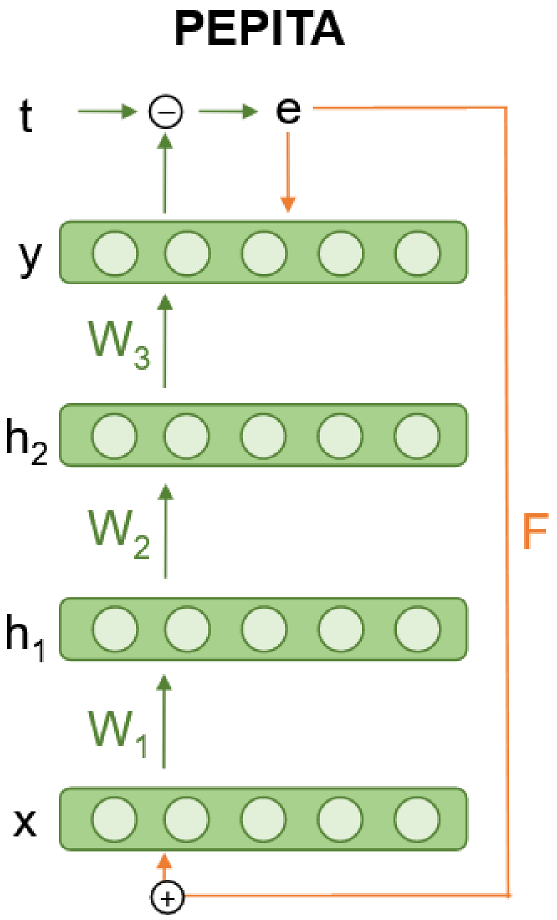
The backward pass implies:

» **Non-locality**

» **Frozen activity**

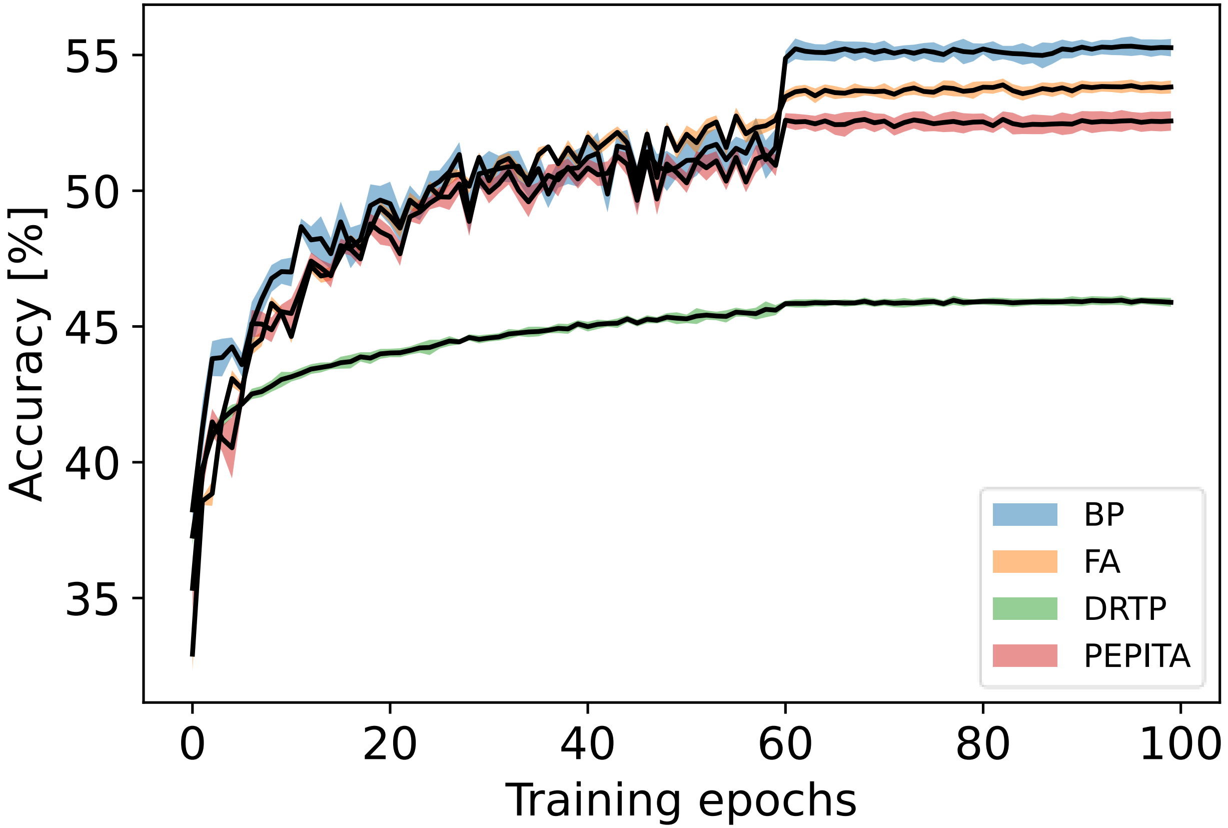
» At least **partial update locking**

Remove the backward pass

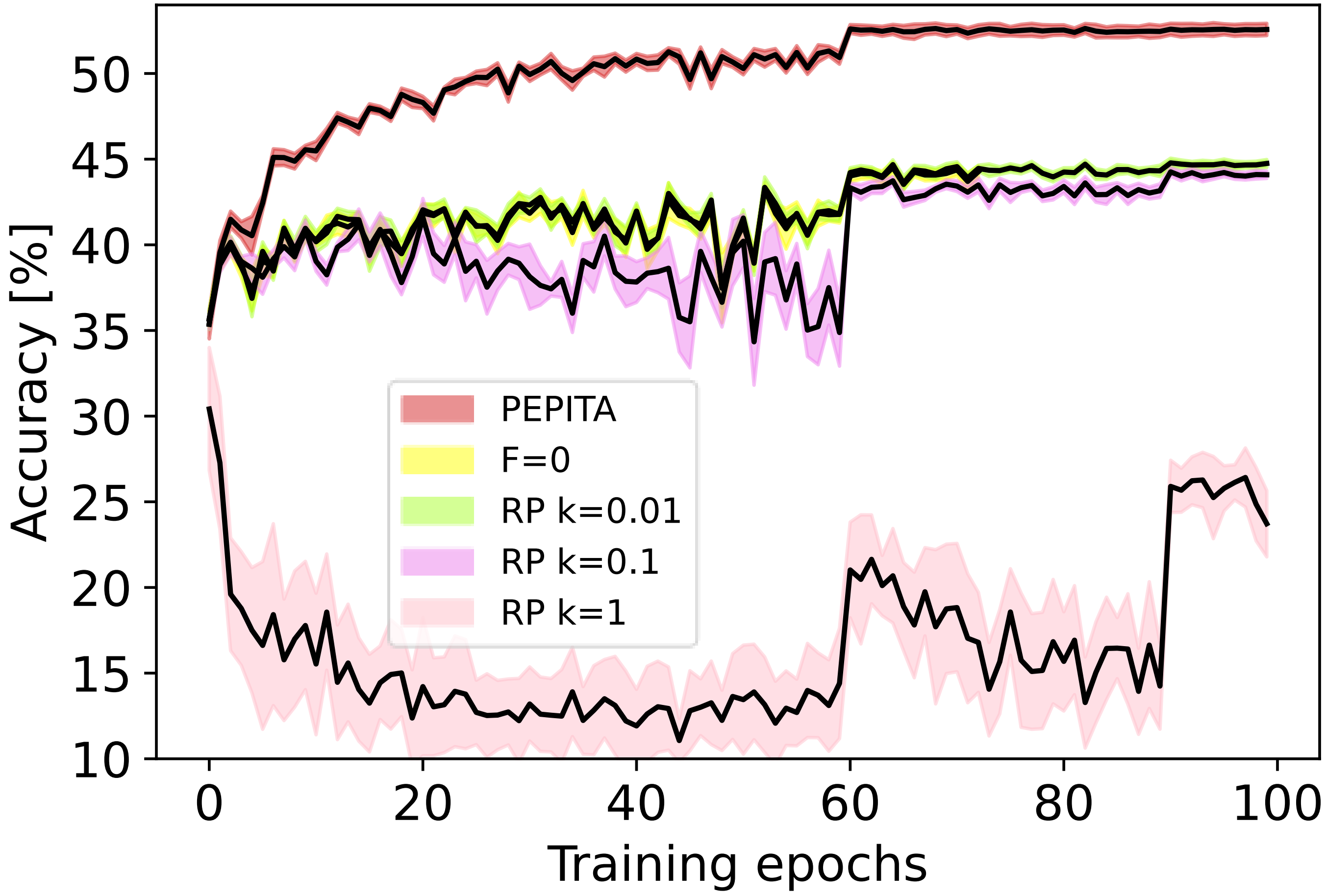


Dellaferrera & Kreiman, ICML 2022

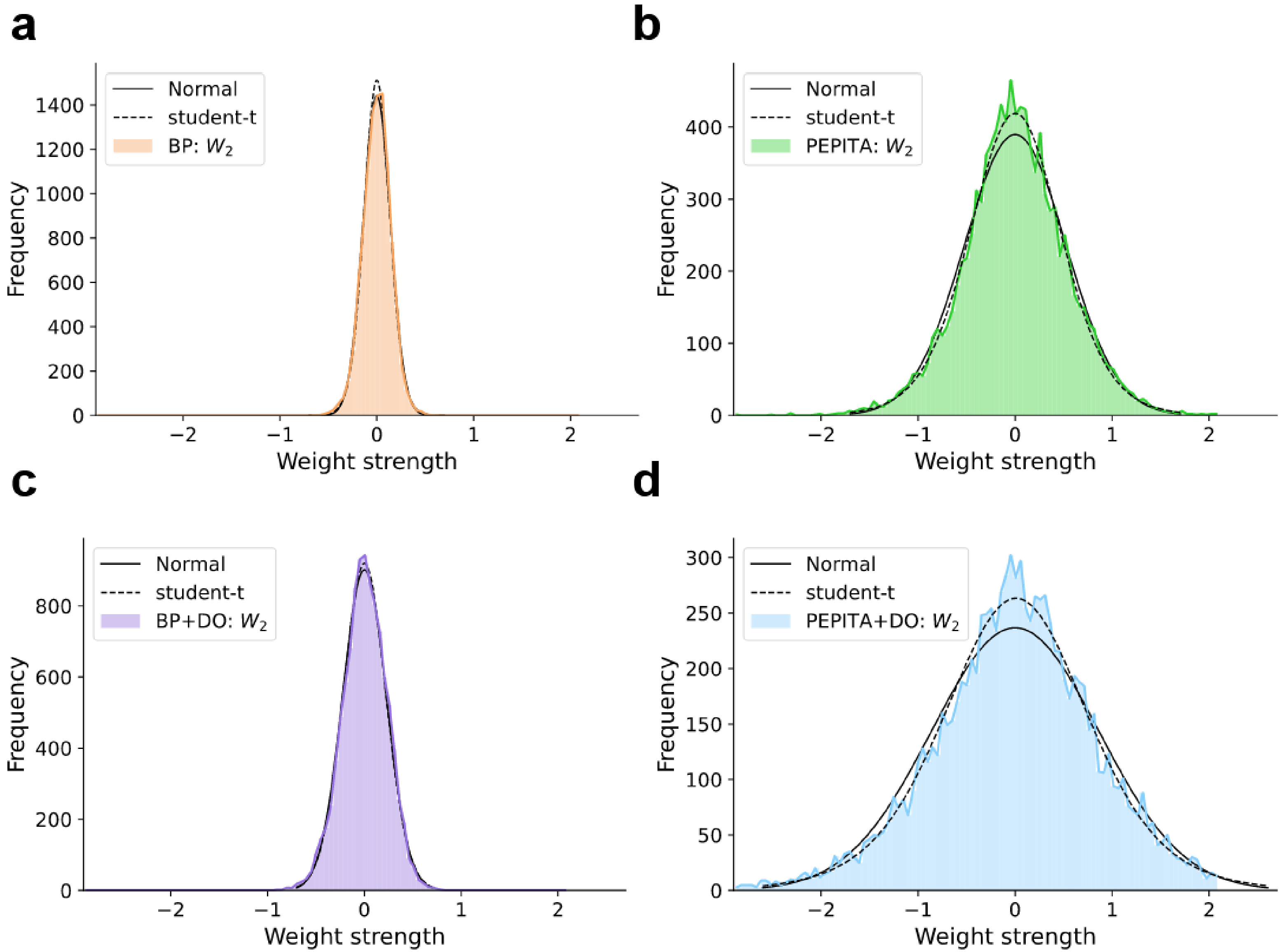
Test curves on CIFAR-10



Error-based modulation is key for good performance



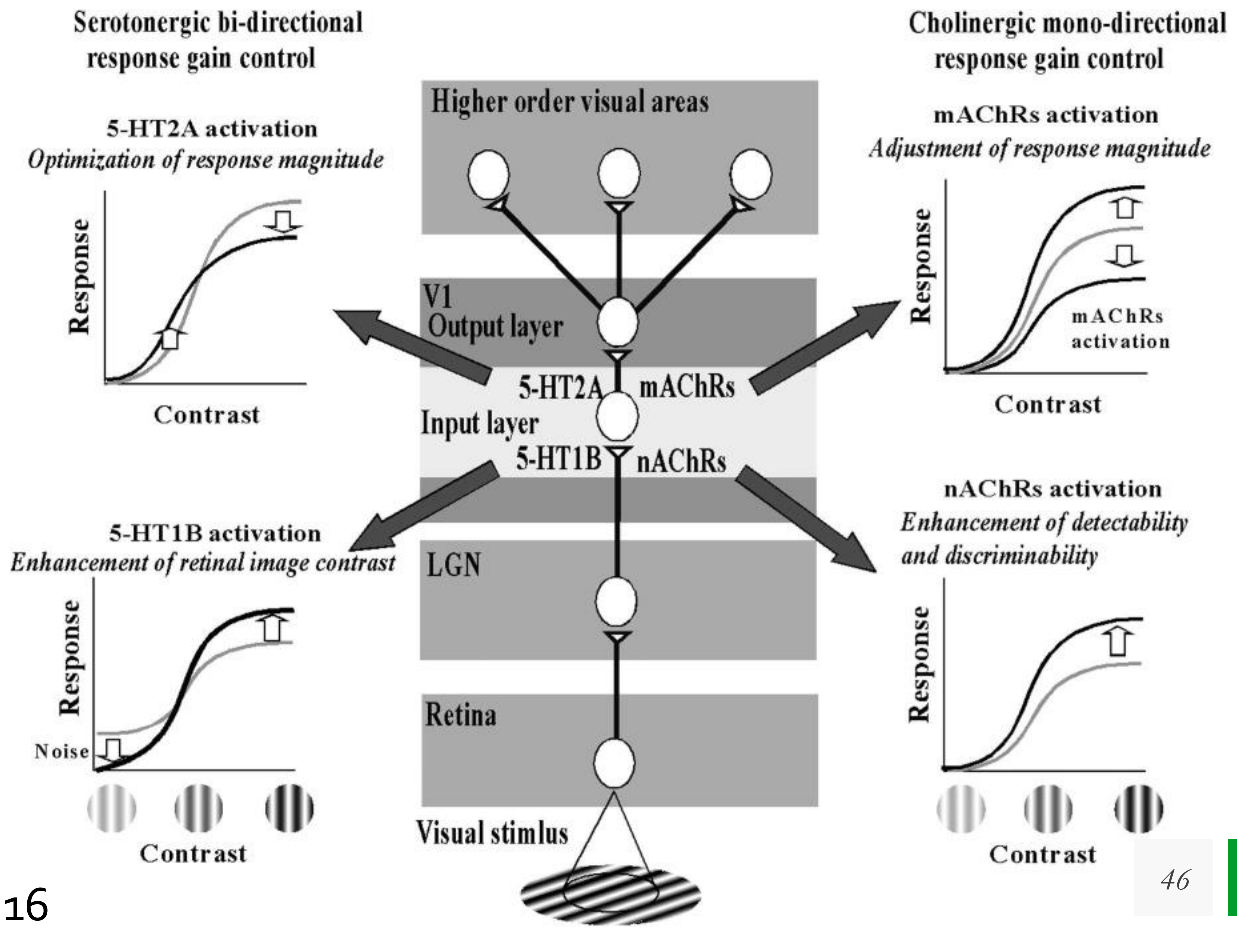
Weight distribution – heavy tailed



Analysis of PEPITA from a biological standpoint

PEPITA solves the BP's issues of biological plausibility, but it introduces additional elements:

» Projection of the error onto the input through a fixed random matrix
  - Global neuromodulatory signals modulate activity in V1

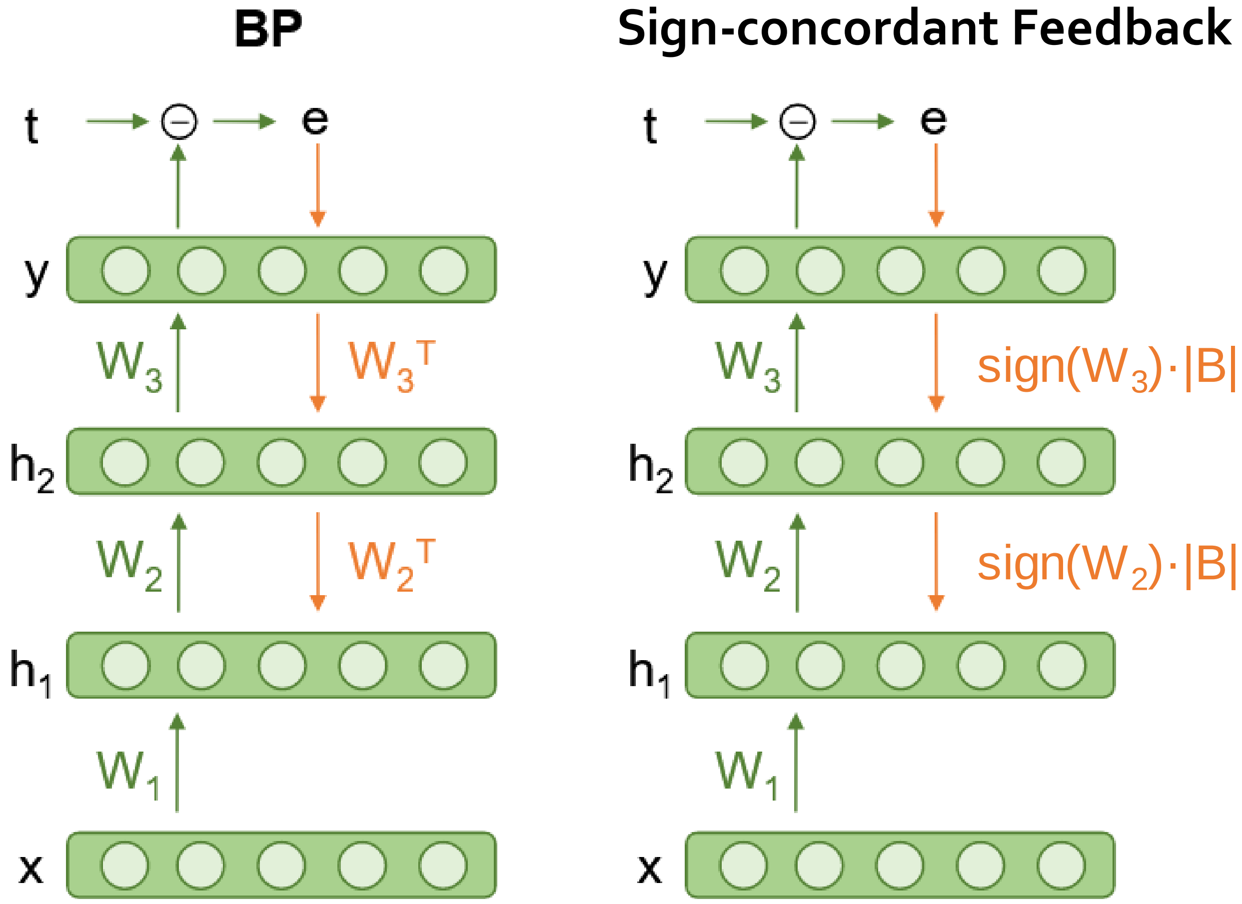


Shimegi et al., 2016

Alternatives to BP: Sign symmetry

» **Asymmetric backpropagation**
- Sign-concordant Feedback

» **Relax weight symmetry requirement**
- the magnitudes of feedback weights do not matter to performance
- the signs of feedback weights do matter — the more concordant signs between feedforward and their corresponding feedback connection



Liao et al., 2016
Xiao et al., 2019

Alternatives to BP: Feedback Alignment

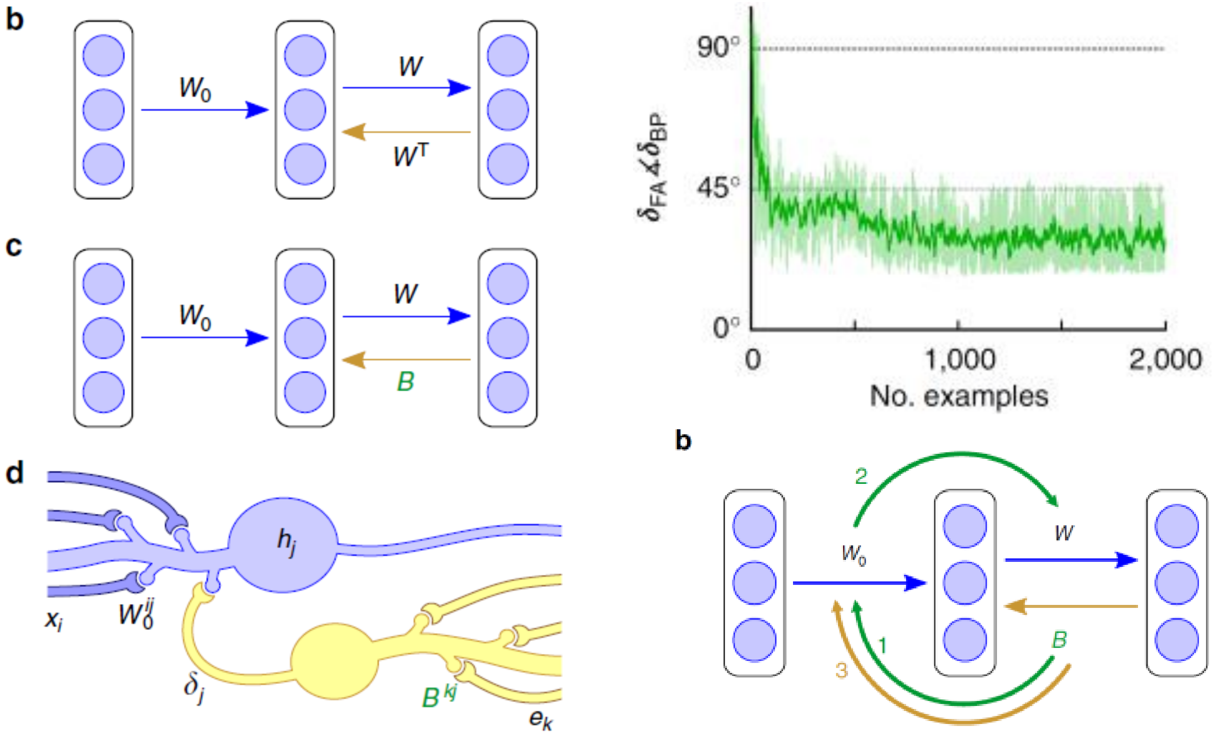
» Precise symmetric connectivity between connected layers is not required to obtain quick learning

» **Replaces W^T with a matrix of fixed random weights B**

- Each neuron in the hidden layer receives a random projection of the error vector
- Avoids all transport of synaptic weight information

» **The circuit learns by encouraging a soft alignment of W with B^T**

- The angle between modulator vectors prescribed by feedback alignment and backprop decreases
- As W aligns with B^T, B begins to act like W^T, sending useful teaching signals to the hidden units



Lillicrap et al., 2016

Alternatives to BP: Direct and Indirect Feedback Alignment

» The FA principle is used for training hidden layers more independently from the rest of the network

» **Feedback path disconnected from the forward path**

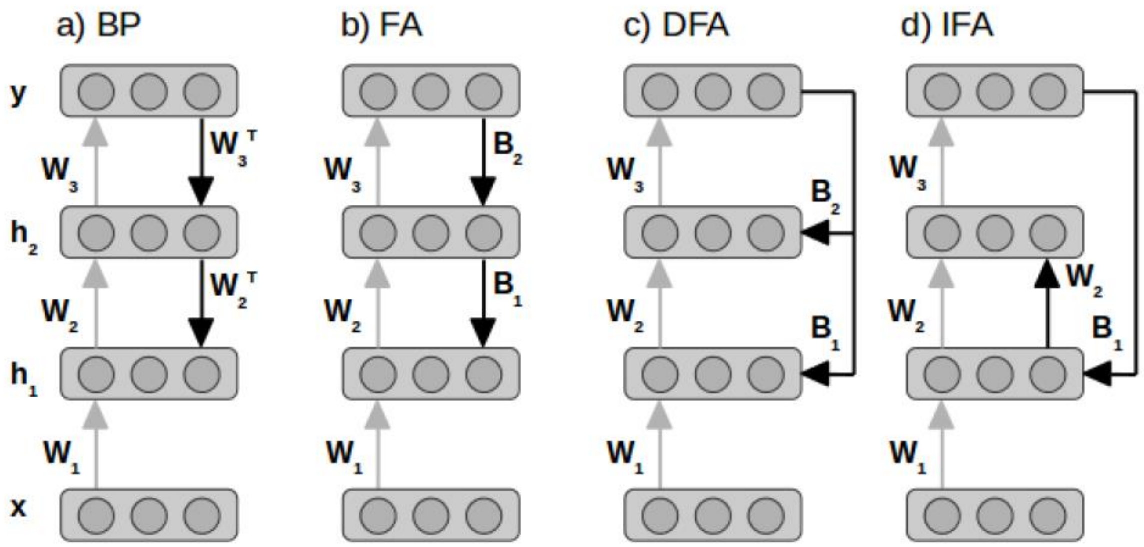
- Possibility that the error in the feedback layer is represented by neurons not participating in the forward pass
- layer is no longer reciprocally connected to the layer above

» **DFA**

- direct feedback path to each hidden layer

» **IFA**

- direct feedback path connecting to the first hidden layer
- then visiting every layer on its way forward



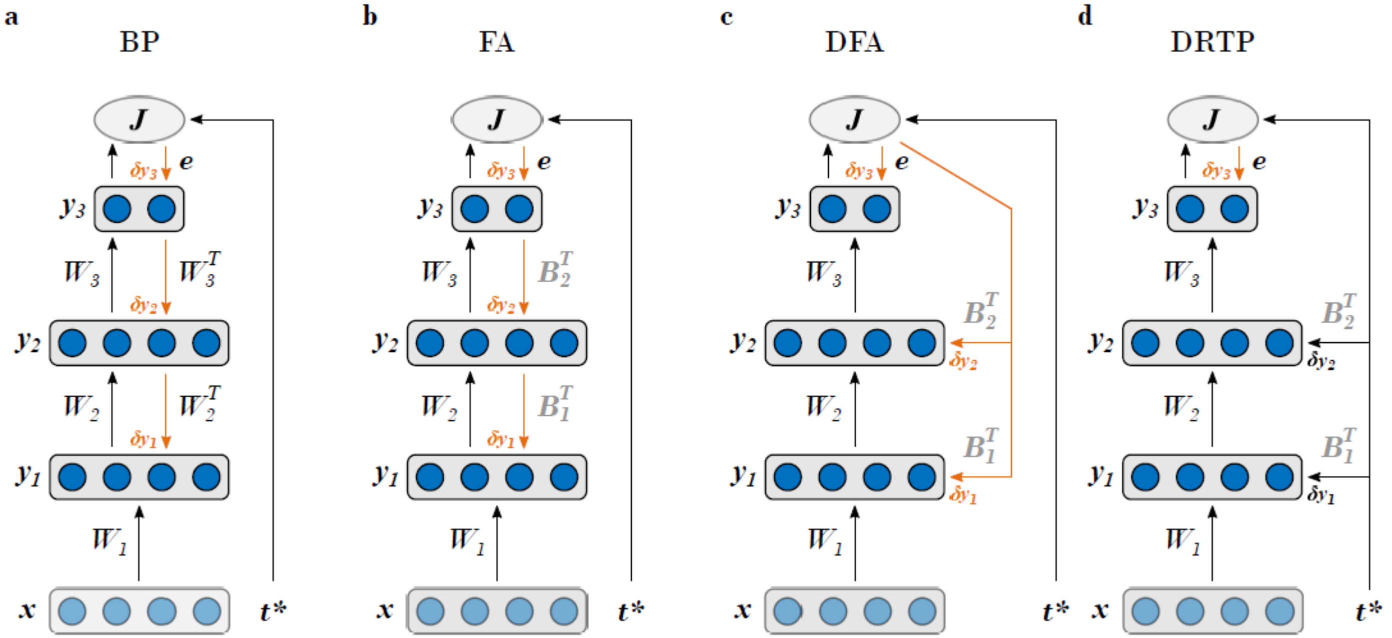
MODEL	BP	FA	DFA
7x240 Tanh	2.16 ± 0.13%	2.20 ± 0.13% (0.02%)	2.32 ± 0.15% (0.03%)
100x240 Tanh			3.92 ± 0.09% (0.12%)
1x800 Tanh	1.59 ± 0.04%	1.68 ± 0.05%	1.68 ± 0.05%
2x800 Tanh	1.60 ± 0.06%	1.64 ± 0.03%	1.74 ± 0.08%
3x800 Tanh	1.75 ± 0.05%	1.66 ± 0.09%	1.70 ± 0.04%
4x800 Tanh	1.92 ± 0.11%	1.70 ± 0.04%	1.83 ± 0.07% (0.02%)
2x800 Logistic	1.67 ± 0.03%	1.82 ± 0.10%	1.75 ± 0.04%
2x800 ReLU	1.48 ± 0.06%	1.74 ± 0.10%	1.70 ± 0.06%
2x800 Tanh + DO	1.26 ± 0.03% (0.18%)	1.53 ± 0.03% (0.18%)	1.45 ± 0.07% (0.24%)
2x800 Tanh + ADV	1.01 ± 0.08%	1.14 ± 0.03%	1.02 ± 0.05% (0.12%)

Test error on MNIST

A. Nokland, 2016

Alternatives to BP: Direct Random Target Propagation

- **The error sign provides useful modulatory signals to multi-layer networks**
  - Targets (i.e. one-hot-encoded labels) used in place of the output error
  - Targets are projected onto the hidden layers
- Fully solves both the weight transport and the update locking problems
- BUT: lower performance than BP, FA, DFA

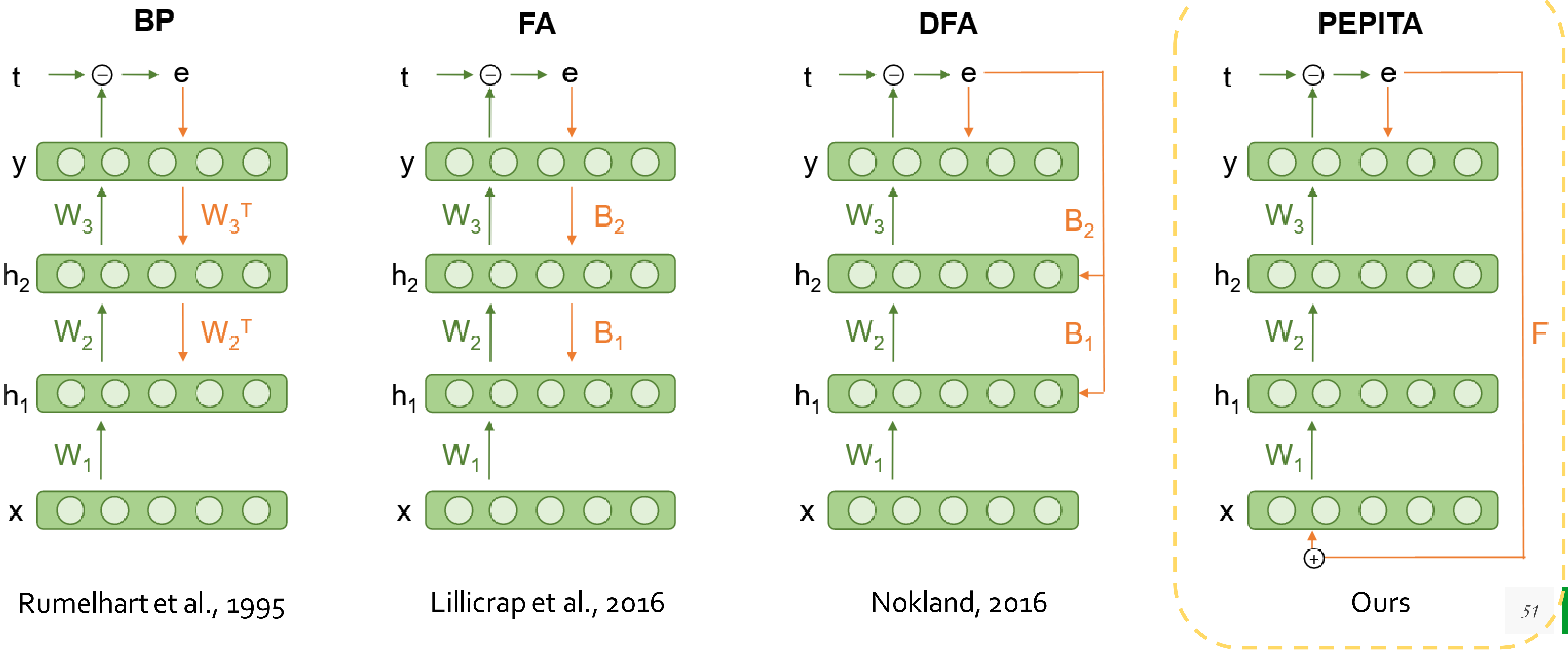


Network		BP	FA	DFA	DRTP
FC1-500	DO 0.0	1.72±0.08%	1.92±0.08%	2.59±0.11%	4.58±0.12%
	DO 0.1	1.55±0.03%	1.66±0.06%	2.17±0.10%	4.65±0.13%
	DO 0.25	1.64±0.06%	1.73±0.05%	2.32±0.08%	5.36±0.11%

Test error on MNIST

Frenkel et al., 2019

Training without a backward path: modulating the input through the error

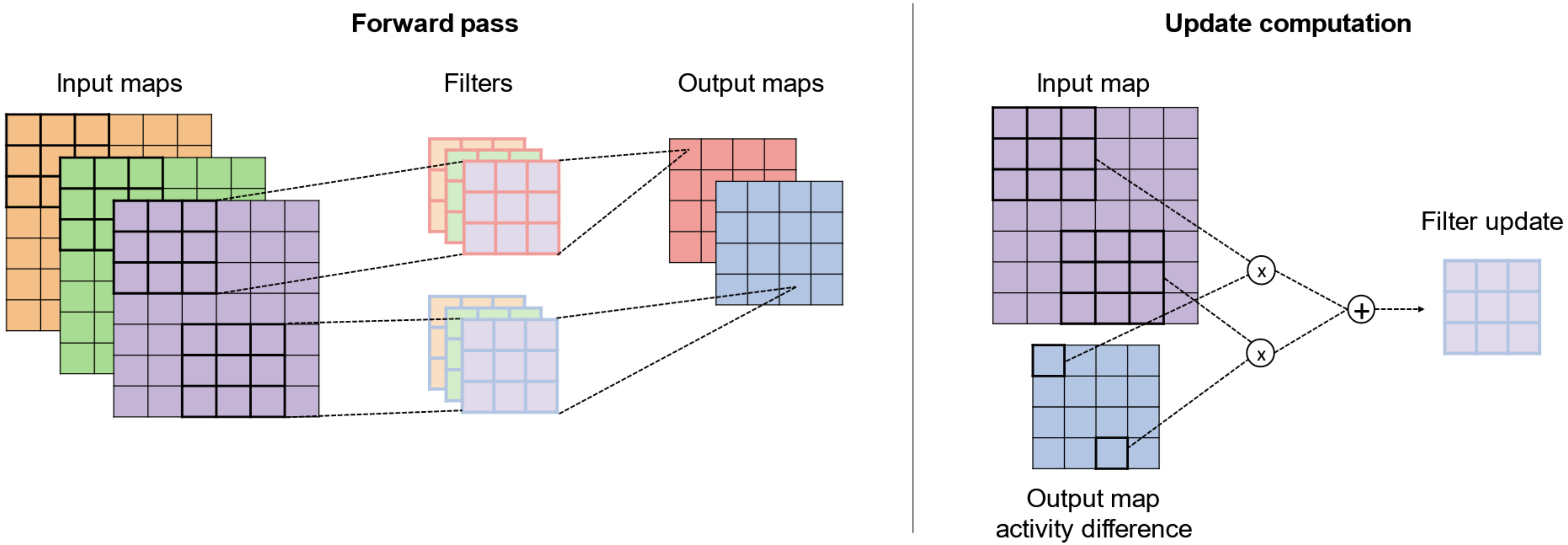


The PEPITA learning rule for Convolutional Neural Networks

» Same approach with *Standard* and *Modulated pass*

» Takes into account *weight sharing* of convolutional layers

» Each filter is updated based on the contributions of each *input-map-region – output-map-element* pair



Testing PEPITA on image classification tasks - experimental results

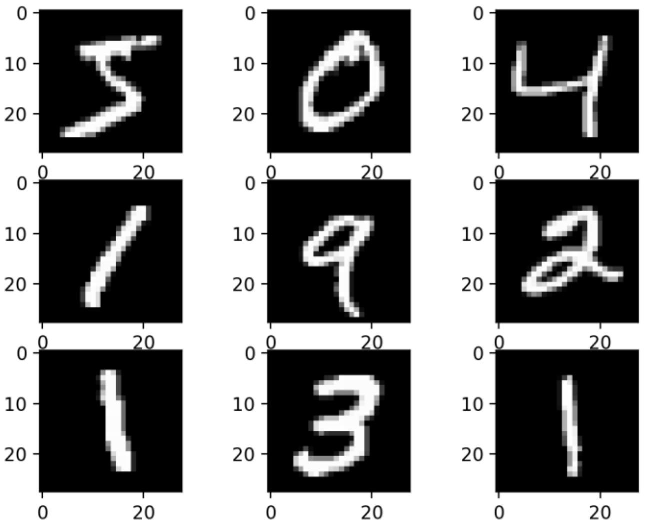
» Res ... nce

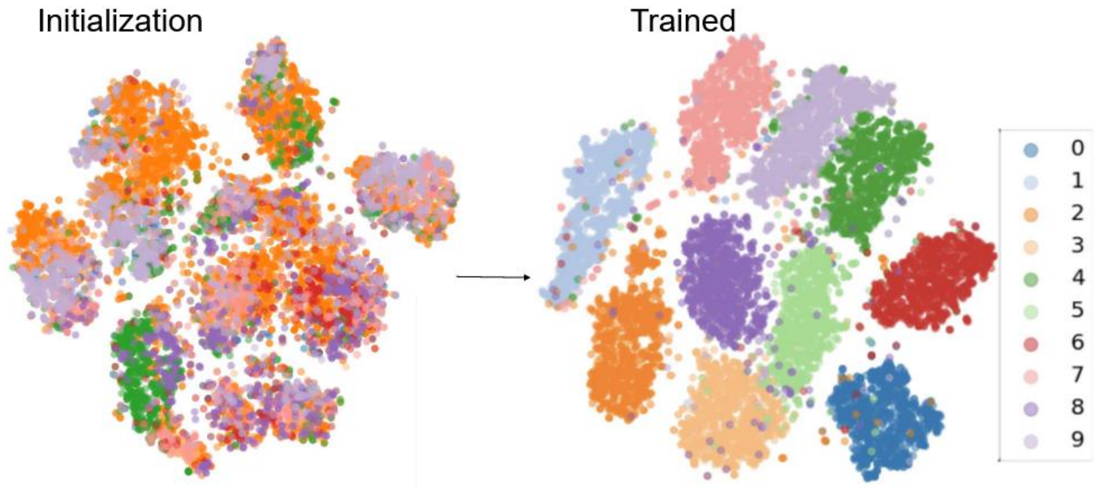
» In s

» PEF

» The

- C
- Is





	Fully connected models			Convolutional models		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

Acknowledgements

Gabriel Kreiman
Harvard, Boston Children's Hospital

Will Xiao
Harvard Medical School