Scaling EqProp to Deep ConvNets by reducing its gradient estimator bias

Axel Laborieux*, Maxence Ernoult*, Benjamin Scellier, Yoshua Bengio, Julie Grollier, Damien Querlioz

* Corresponding authors:
{axel.laborieux, maxence.ernoult}@universite-paris-saclay.fr

Summary

- Equilibrium Propagation (EP) is a biologically-inspired counterpart of Backpropagation Through Time (BPTT). However :

1. Only shown to work on MNIST
2. Optimizes the squared error loss

- **This work** : identify and cancel a bias in the gradient estimate

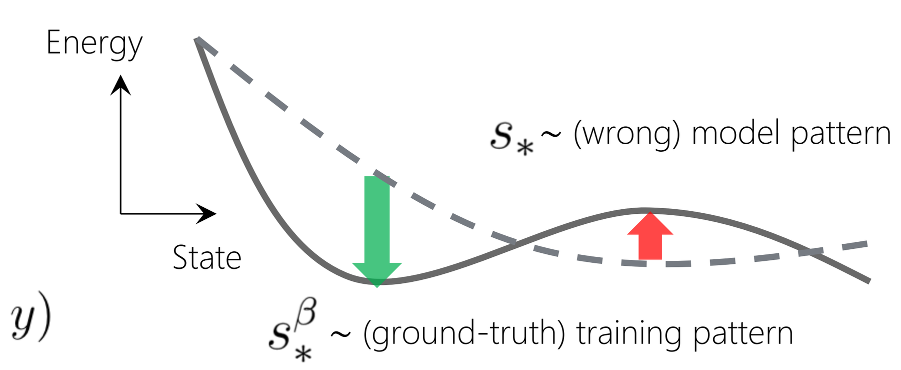
1. Scale to deeper architectures and closely match BPTT on CIFAR10
2. Adapt the model to optimize the cross-entropy loss
3. Still works with untied weights provided an alignment mechanism

Equilibrium Propagation [1]

$$s_{t+1} = \frac{\partial \Phi}{\partial s}(x, s_t, \theta)$$

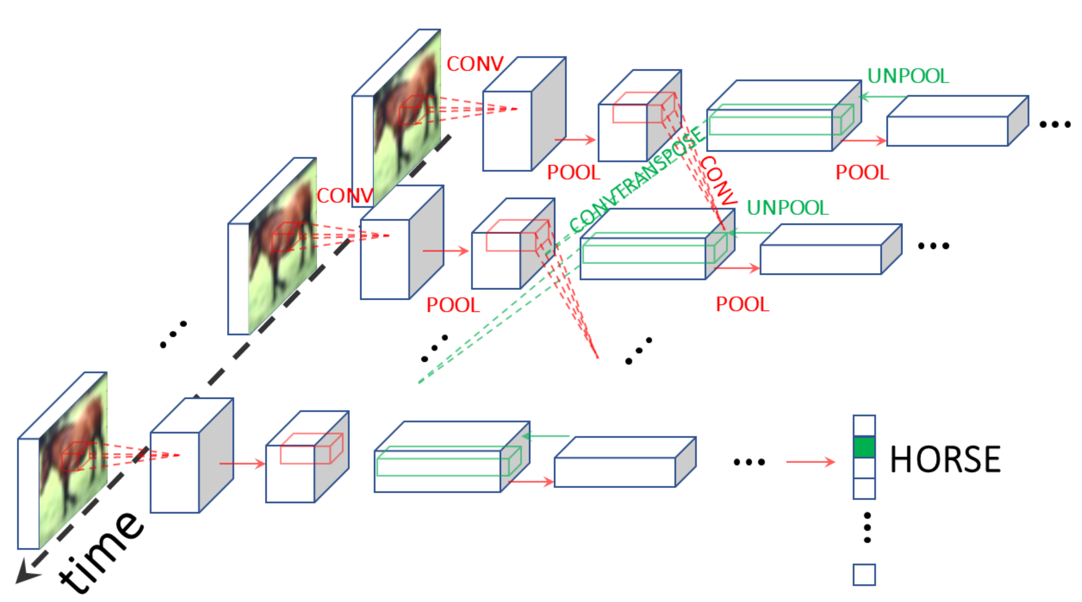
until $\quad s_* = \frac{\partial \Phi}{\partial s}(x, s_*, \theta)$

$$s_{t+1}^\beta = \frac{\partial \Phi}{\partial s}(x, s_t^\beta, \theta) - \beta \frac{\partial \ell}{\partial s}(s_t^\beta, y)$$



$$\widehat{\nabla}^{\mathrm{EP}}(\beta) \triangleq \frac{1}{\beta}\left(\frac{\partial \Phi}{\partial \theta}(x, s_*^\beta, \theta) - \frac{\partial \Phi}{\partial \theta}(x, s_*, \theta)\right)$$

Convolutional Architecture [2]



$$\begin{cases} s_{t+1}^n &= \sigma\left(\mathcal{P}\left(w_{n-1} \star s_t^{n-1}\right) + \tilde{w}_{n+1} \star \mathcal{P}^{-1}\left(s_t^{n+1}\right)\right), & \text{for convolutional layers,} \\ s_{t+1}^n &= \sigma\left(w_{n-1} \cdot s_t^{n-1} + w_{n+1}^\top \cdot s_t^{n+1}\right), & \text{for fully connected layers.} \end{cases}$$

Paper

https://arxiv.org/pdf/2006.03824.pdf

Code

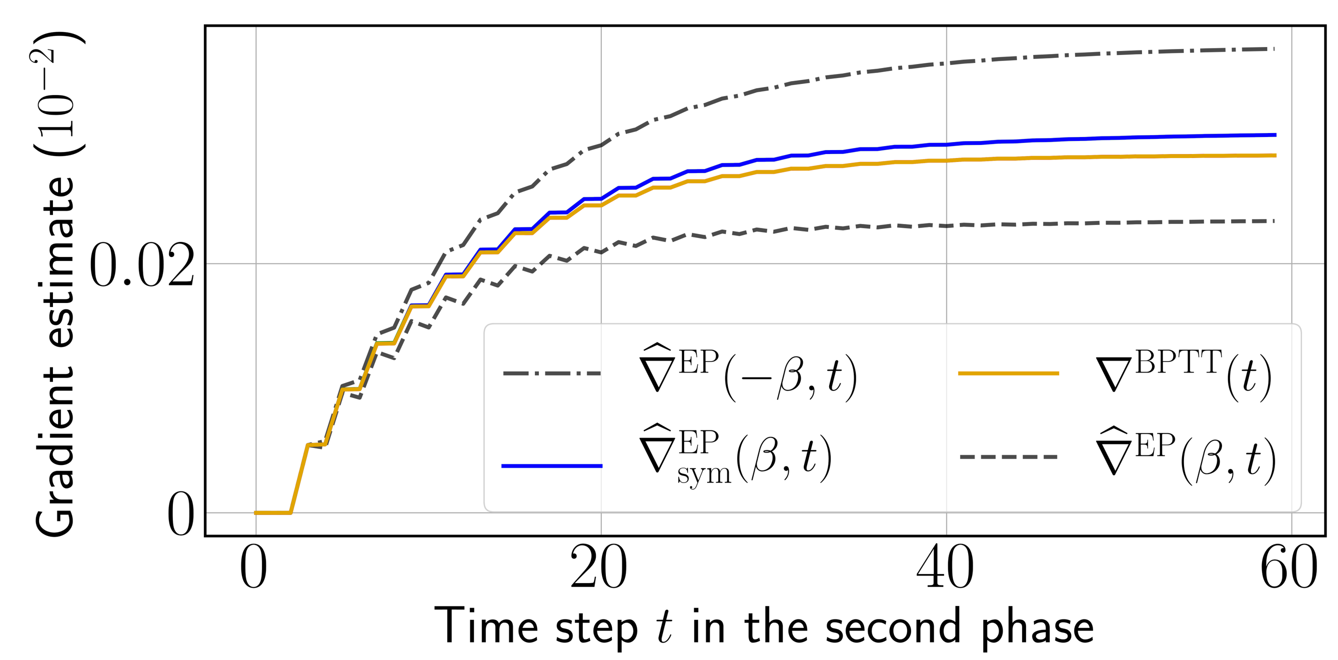
https://github.com/Laborieux-Axel/Equilibrium-Propagation

Symmetric estimate of the loss gradient

$$\widehat{\nabla}^{\mathrm{EP}}(\beta) = -\frac{\partial \mathcal{L}^*}{\partial \theta} + O(\beta) \quad \text{: first order bias in } \beta$$

$$\widehat{\nabla}^{\mathrm{EP}}_{\mathrm{sym}}(\beta) \triangleq \frac{\widehat{\nabla}^{\mathrm{EP}}(\beta) + \widehat{\nabla}^{\mathrm{EP}}(-\beta)}{2} = -\frac{\partial \mathcal{L}^*}{\partial \theta} + O(\beta^2)$$

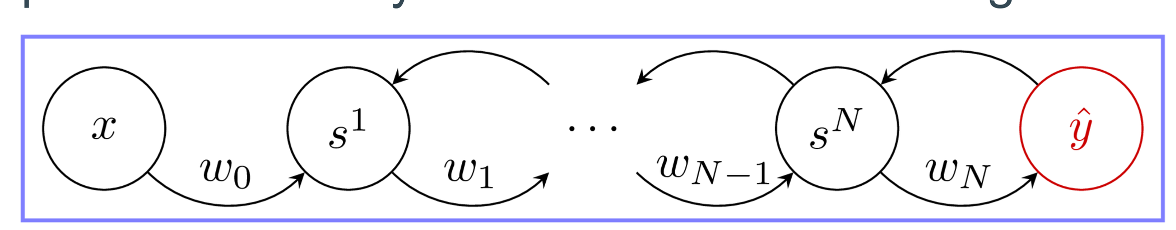
Third phase: $s_*^{-\beta}$



Loss Function	EP Gradient Estimate	EP Error (%) Test	EP Error (%) Train	BPTT Error (%) Test	BPTT Error (%) Train
Squared Error	2-Phase / $\hat{\nabla}^{\mathrm{EP}}$	86.7 (5.8)	84.9	11.1 (0.1)	3.7
	Random Sign	21.6 (20.0)	20.0		
	3-Phase / $\hat{\nabla}^{\mathrm{EP}}_{\mathrm{sym}}$	12.5 (0.2)	7.8		

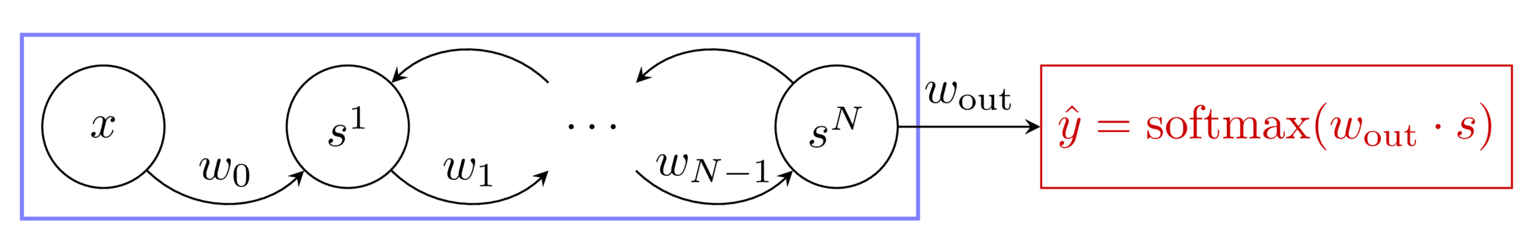
Optimize the cross-entropy loss with EqProp

Output units in the system : difficult to define logits



h_t^β: hidden layer

$$\hat{y}_{t+1}^\beta = \frac{\partial \Phi}{\partial \hat{y}}(x, h_t^\beta, \hat{y}_t^\beta, \theta) + \beta\left(y - \hat{y}_t^\beta\right)$$



$$s_{t+1}^\beta = \frac{\partial \Phi}{\partial s}(x, s_t^\beta, \theta) + \beta\, w_{\mathrm{out}}^\top \cdot \left(y - \hat{y}_t^\beta\right)$$

Loss Function	EP Gradient Estimate	EP Error (%) Test	EP Error (%) Train	BPTT Error (%) Test	BPTT Error (%) Train
Cross-Ent.	3-Phase / $\hat{\nabla}^{\mathrm{EP}}_{\mathrm{sym}}$	**11.7 (0.2)**	**5.0**	11.1 (0.2)	2.2
Cross-Ent. (Dropout)	3-Phase / $\hat{\nabla}^{\mathrm{EP}}_{\mathrm{sym}}$	11.9 (0.3)	6.5	10.7 (0.1)	2.9

Distinct forward and backward weights

Generalized framework [3] : $\quad s_{t+1} = F(x, s_t, \theta)$

$$\widehat{\nabla}^{\mathrm{VF}}_{\mathrm{sym}}(\beta) \triangleq \frac{1}{2\beta}\frac{\partial F}{\partial \theta}(x, s_*, \theta)^\top \cdot \left(s_*^\beta - s_*^{-\beta}\right)$$

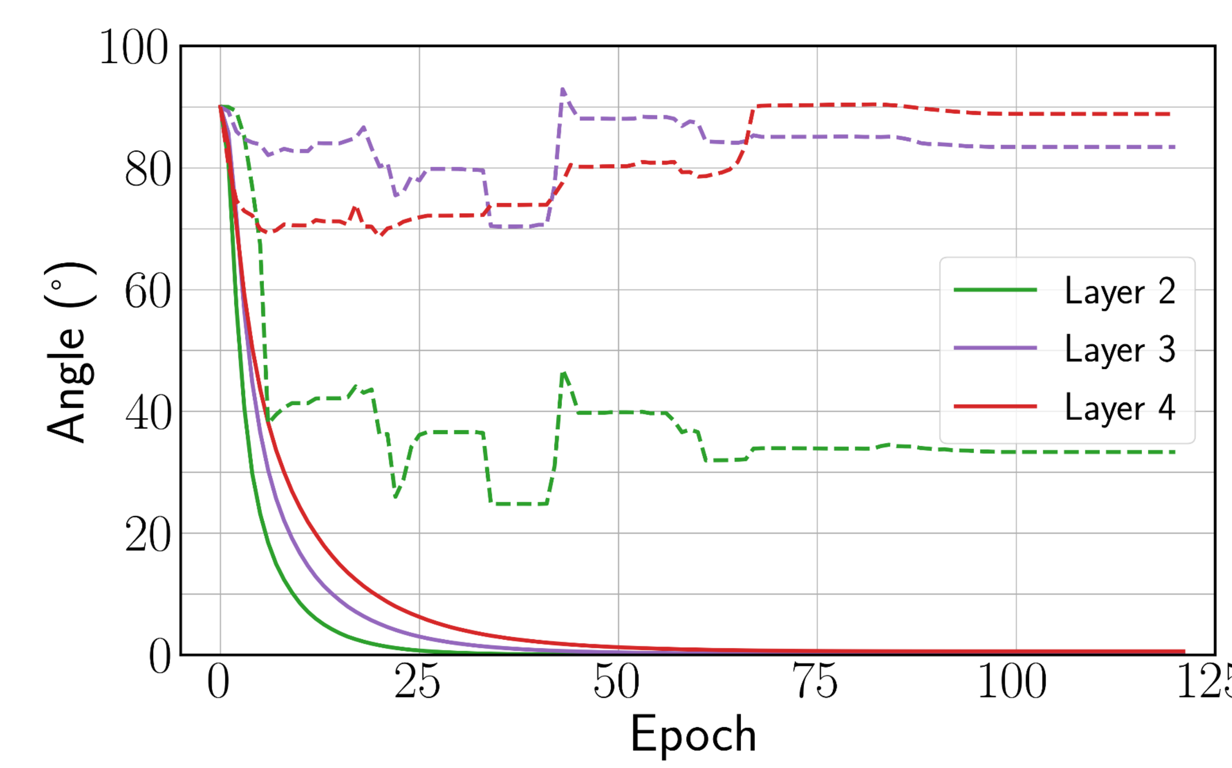
Need for an alignment mechanism, we take inspiration from [4] :

$$\begin{cases} \Delta\theta_{\mathrm{f}} = \eta\left(\widehat{\nabla}^{\mathrm{KP-VF}}_{\mathrm{sym}}(\beta) - \lambda\theta_{\mathrm{f}}\right) \\ \Delta\theta_{\mathrm{b}} = \eta\left(\widehat{\nabla}^{\mathrm{KP-VF}}_{\mathrm{sym}}(\beta) - \lambda\theta_{\mathrm{b}}\right) \end{cases}$$

$$\theta_{\mathrm{f}}(t) - \theta_{\mathrm{b}}(t) = (1 - \eta\lambda)^t\left(\theta_{\mathrm{f}}(0) - \theta_{\mathrm{b}}(0)\right)$$

Loss Function	EP Gradient Estimate	EP Error (%) Test	EP Error (%) Train	BPTT Error (%) Test	BPTT Error (%) Train
Cross-Ent.	3-Phase / $\widehat{\nabla}^{\mathrm{VF}}_{\mathrm{sym}}$	75.5 (4.7)	78.0	9.5 (0.2)	0.8
	3-Phase / $\widehat{\nabla}^{\mathrm{KP-VF}}_{\mathrm{sym}}$	**13.2 (0.5)**	8.9		

We observe that weights do not align (dashed) without the alignment mechanism (solid)



References

Scellier, B. Bengio, Y. (2017). "Equilibrium propagation: Bridging the gap between energy-based models and backpropagation" In: Frontiers in computational neuroscience 11, 24

Ernoult, M. et al. (2019). "Updates of equilibrium prop match backprop through time in an RNN with static input." In: Advances in Neural Information Processing Systems 7081-7091

Scellier, B. et al. (2018). "Generalization of Equilibrium Propagation to vector fields dynamics." arXiv:1808.04873

Kolen, J. F. and Pollack J. B. (1994). "Backpropagation without weight transport". In: IEEE ICCN'94.