Training deep resistive networks with equilibrium propagation

Aspen Winter Conference
Computing with Physical Systems

11 January 2024

Benjamin Scellier

Research scientist at Rain AI

Acknowledgements


Jack Kendall


Maxence Ernoult


Suhas Kumar


Ross Pantone

RAIN


Yoshua Bengio


Axel Laborieux


Julie Grollier


Damien Querlioz


Yann Ollivier


Siddhartha Mishra

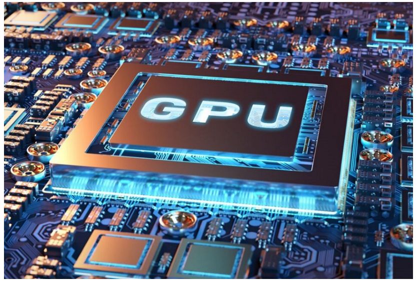

Kalpana Manickavasagam

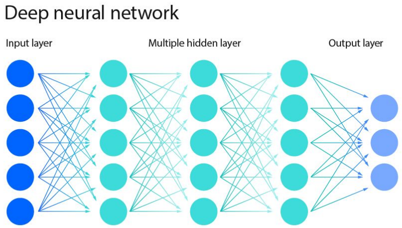

Vidyesh Anisetti

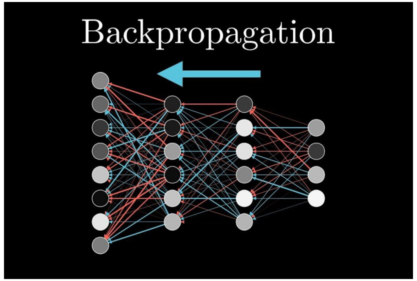

Ananth Kandala

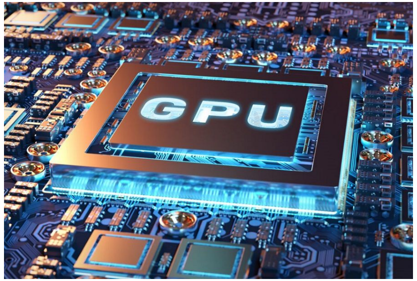

Jennifer Schwarz

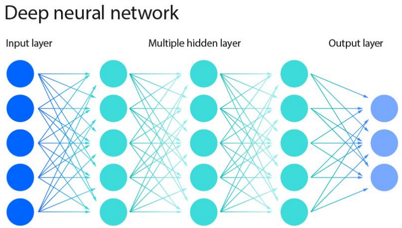
Deep Learning Computing Paradigm

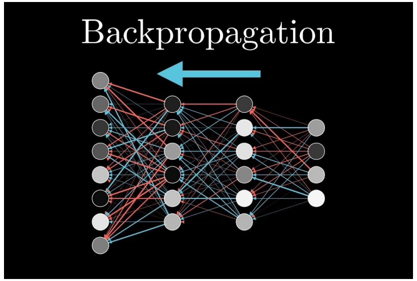


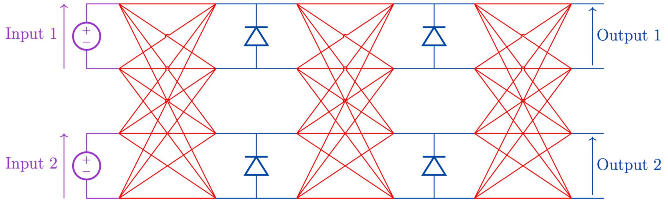




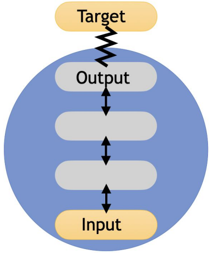








Deep resistive network



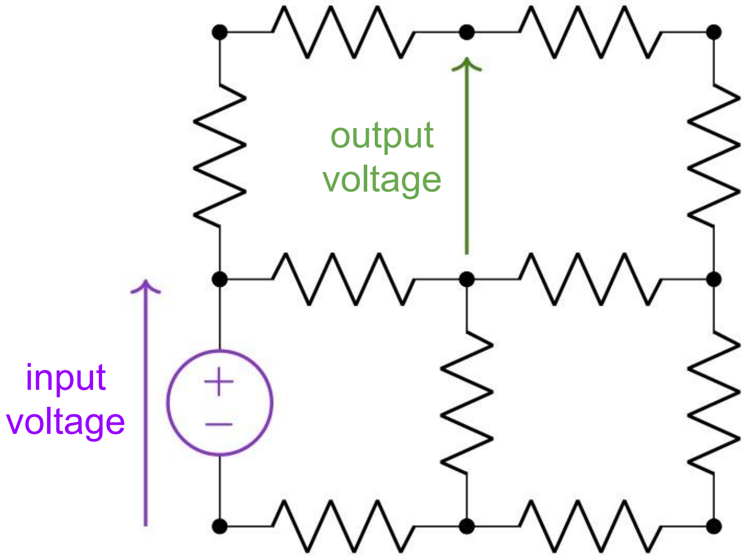
Equilibrium propagation

What is a deep resistive network?

Computing with a Resistor Network

Circuit elements:

- voltage sources (inputs)



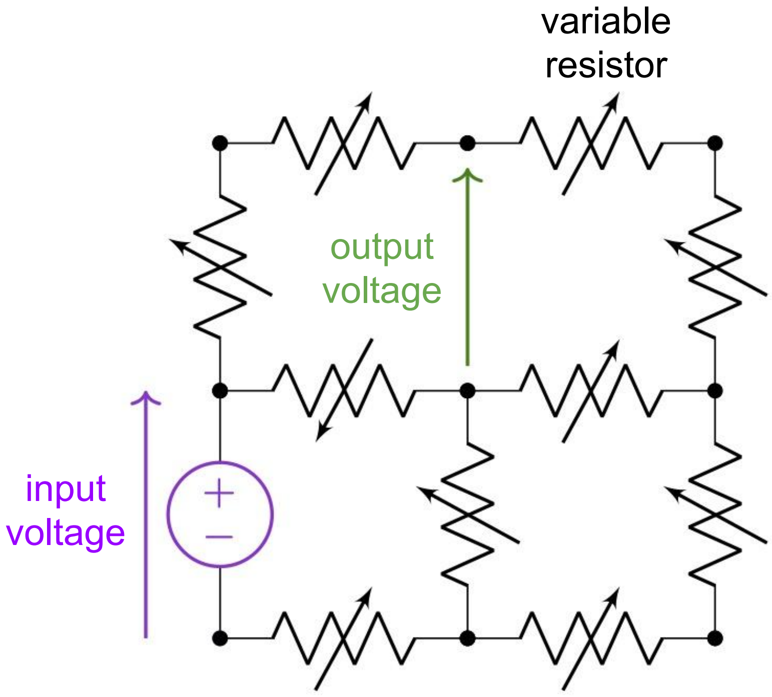
$$\mathrm{output} = F(\mathrm{input})$$

Reference: Scellier and Mishra. “A universal approximation theorem for nonlinear resistive networks” (2023)

Computing with a Resistor Network

Circuit elements:

- voltage sources (inputs)
- variable resistors (trainable weights)



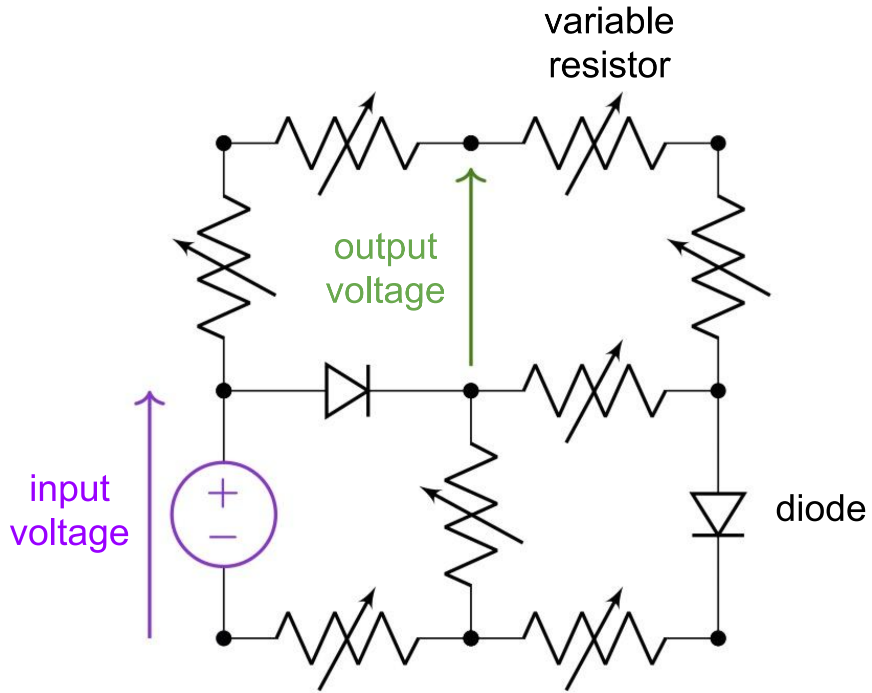
$$\text{output} = F_{\text{conductances}}(\text{input})$$

Reference: Scellier and Mishra. “A universal approximation theorem for nonlinear resistive networks” (2023)

Computing with a Resistor Network

Circuit elements:

- voltage sources (inputs)
- variable resistors (trainable weights)
- diodes (nonlinearities)



$$\text{output} = F_{\text{conductances}}(\text{input})$$

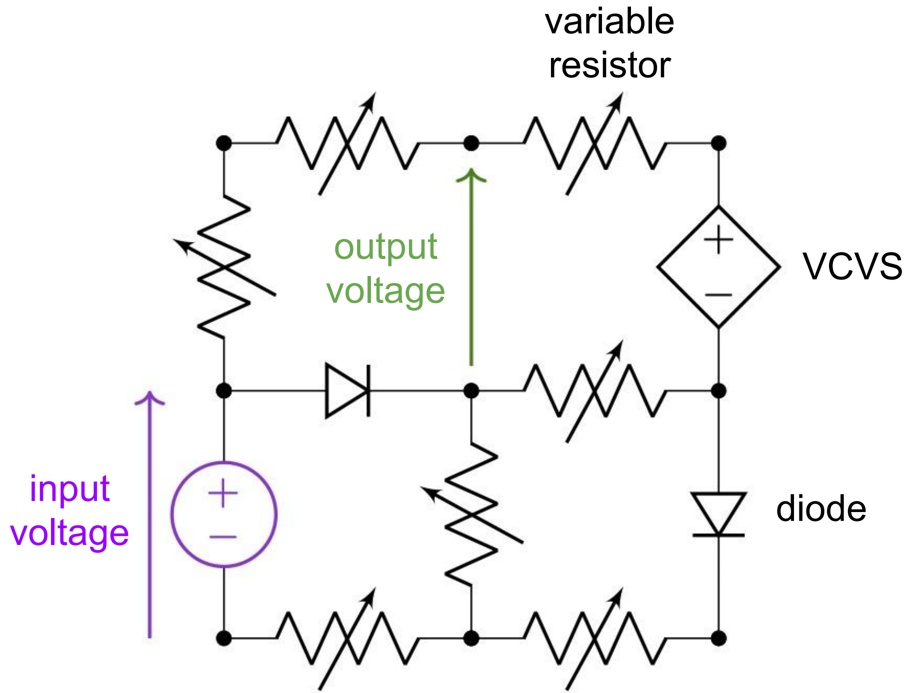
Reference: Scellier and Mishra. “A universal approximation theorem for nonlinear resistive networks” (2023)

Computing with a Resistor Network

Circuit elements:

- voltage sources (inputs)
- variable resistors (trainable weights)
- diodes (nonlinearities)
- VCVS (amplification)

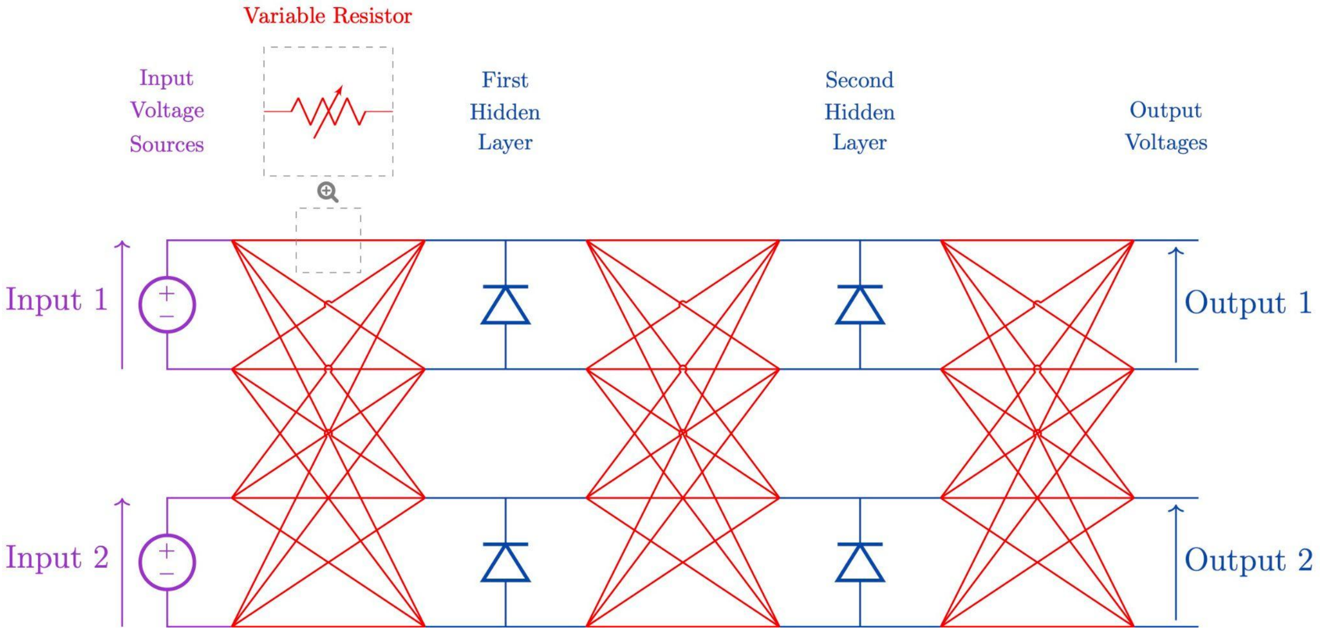
Such electrical circuits are
universal function approximators



$$\text{output} = F_{\text{conductances}}(\text{input})$$

Reference: Scellier and Mishra. "A universal approximation theorem for nonlinear resistive networks" (2023)

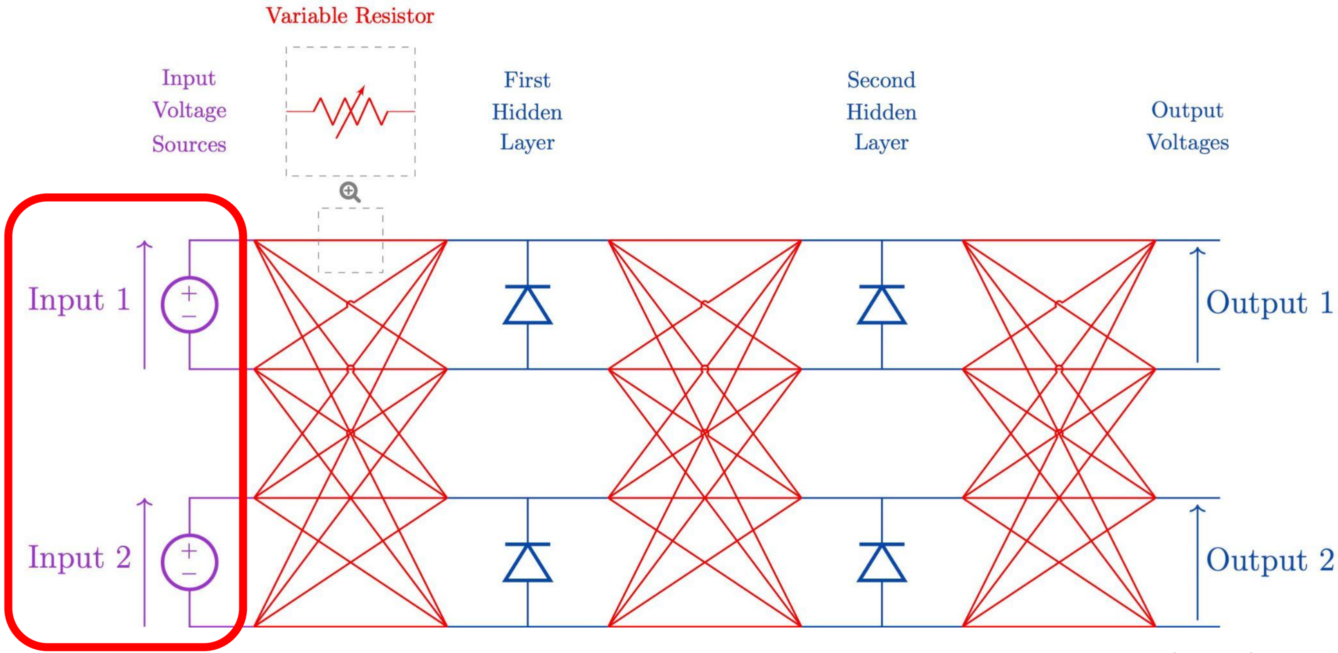
Deep Resistive Network



Reference: Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Deep Resistive Network

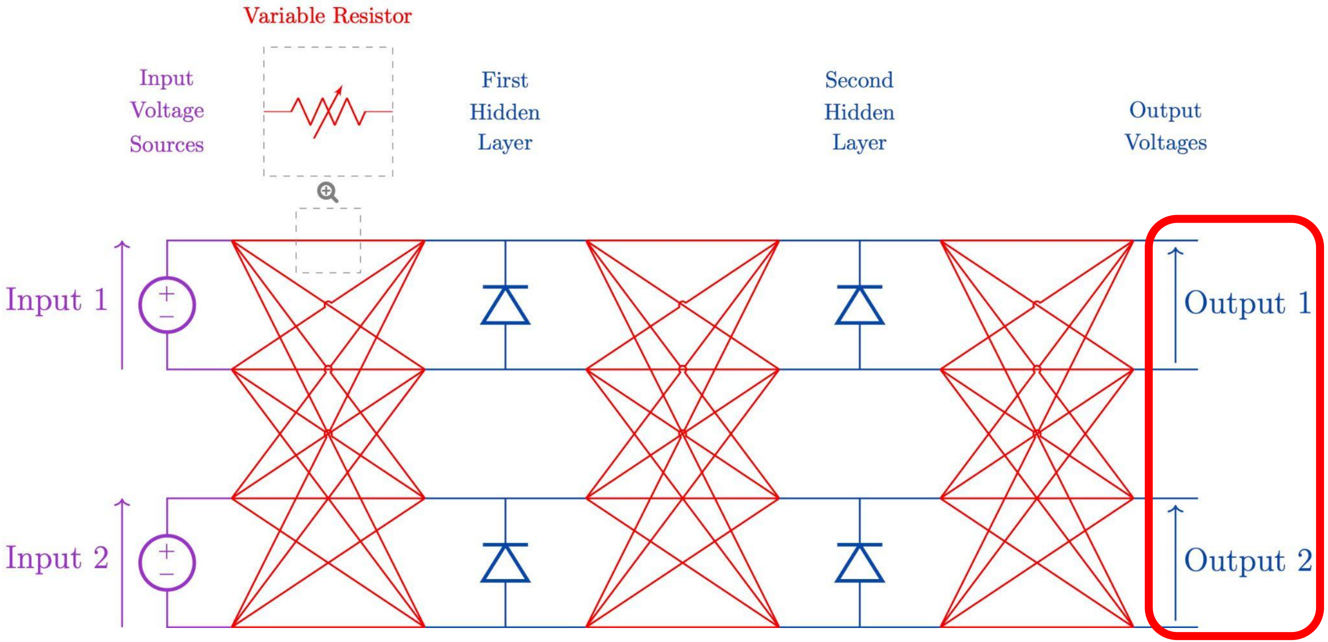
- input voltage sources



Reference: Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Deep Resistive Network

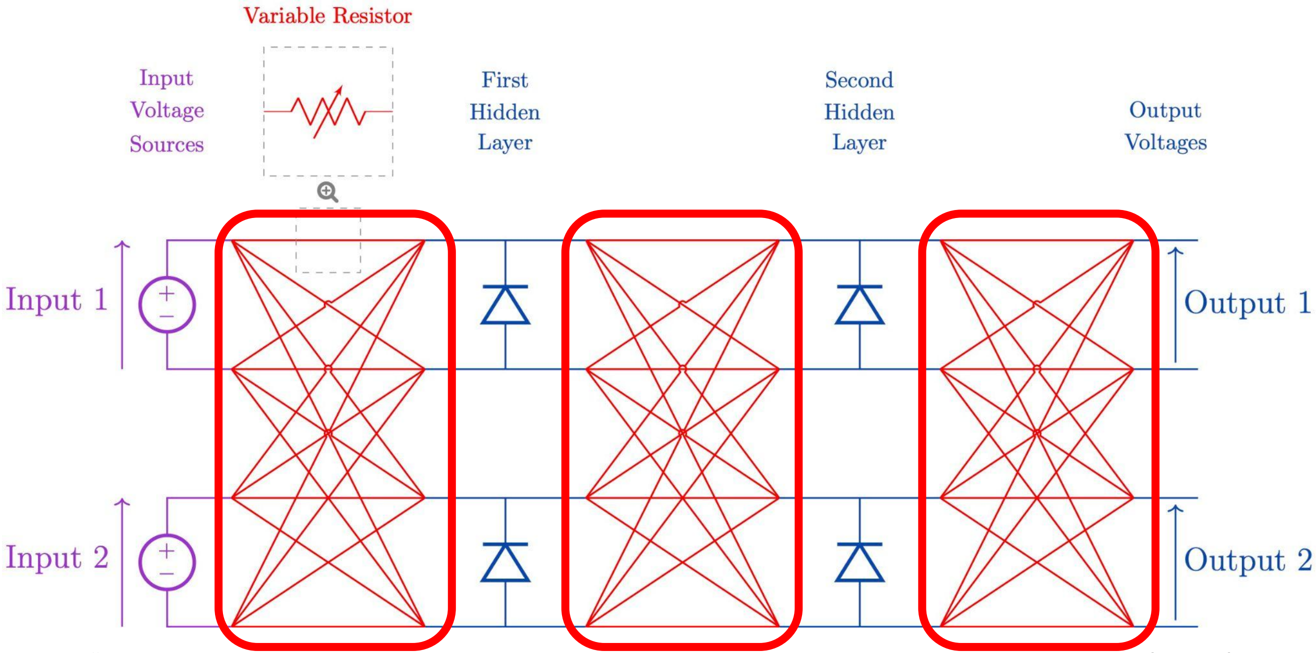
- input voltage sources
- output voltages



Reference: Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Deep Resistive Network

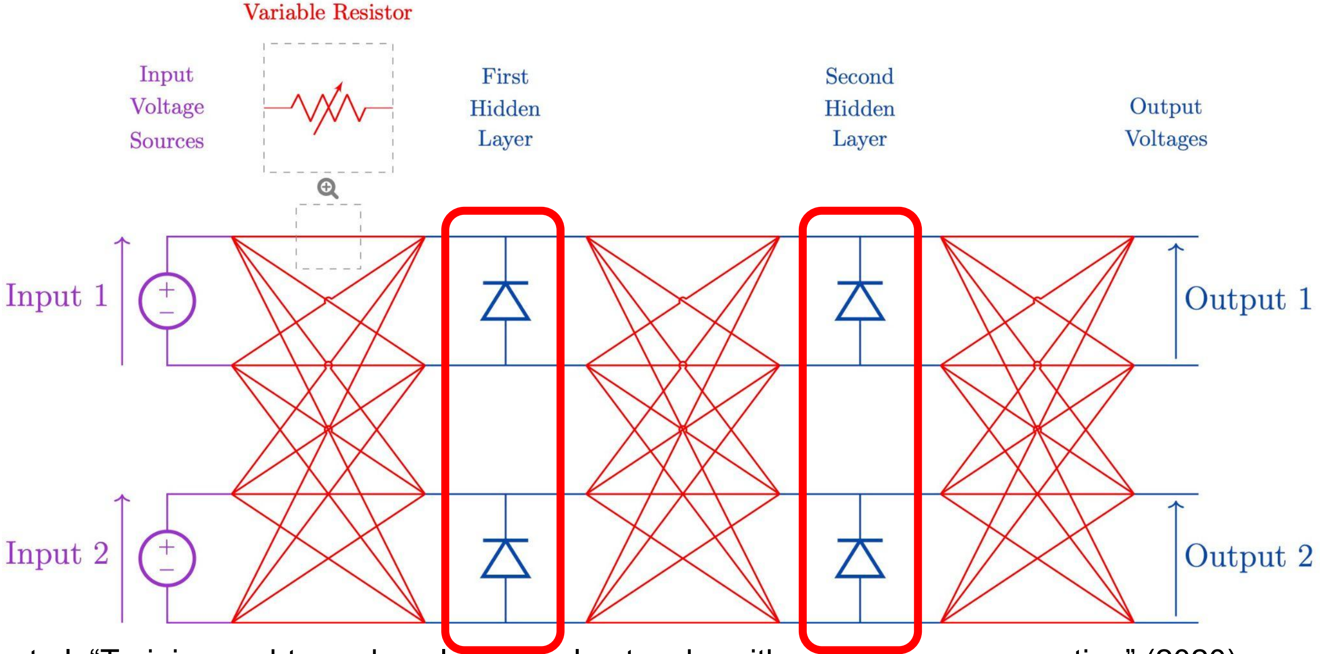
- input voltage sources
- output voltages
- crossbar arrays of variable resistors (linear trainable weights)



Reference: Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Deep Resistive Network

- input voltage sources
- output voltages
- crossbar arrays of variable resistors (linear trainable weights)
- diodes (nonlinearities)

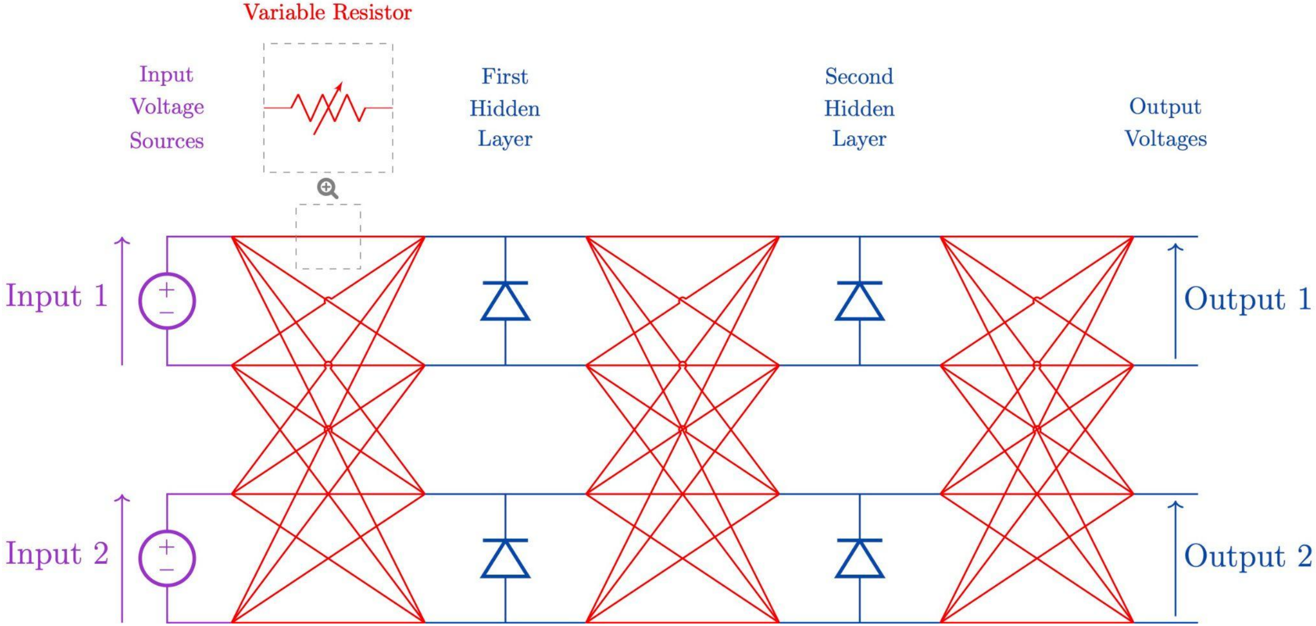


<u>Reference</u>: Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Deep Resistive Network

Properties:

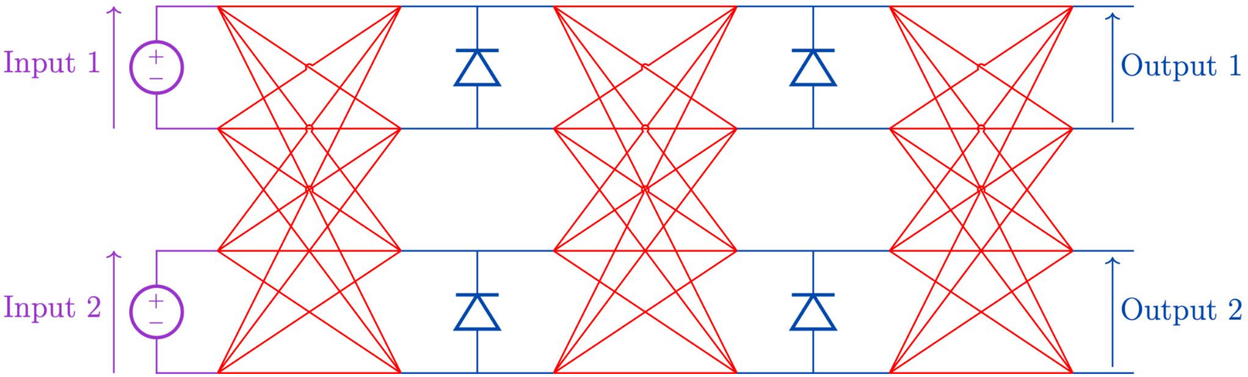
- universal function approximator
- can be trained via equilibrium propagation



Reference: Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

What is equilibrium propagation?

Equilibrium Propagation in a Deep Resistive Network

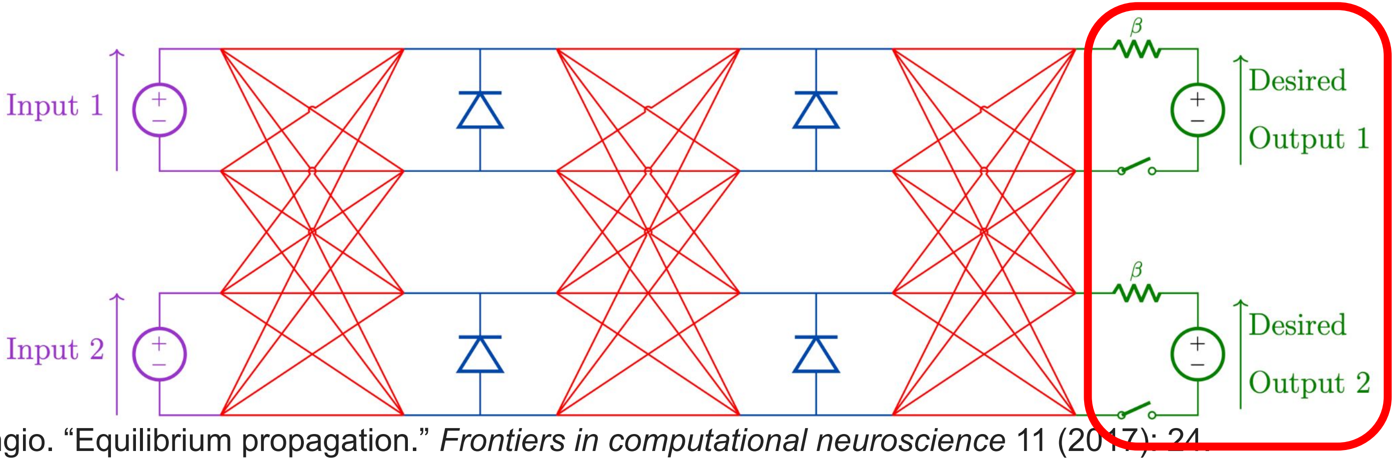


References:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Equilibrium Propagation in a Deep Resistive Network

EP learning requires augmenting the network:

- for each pair of output nodes, add switch + voltage source + resistor



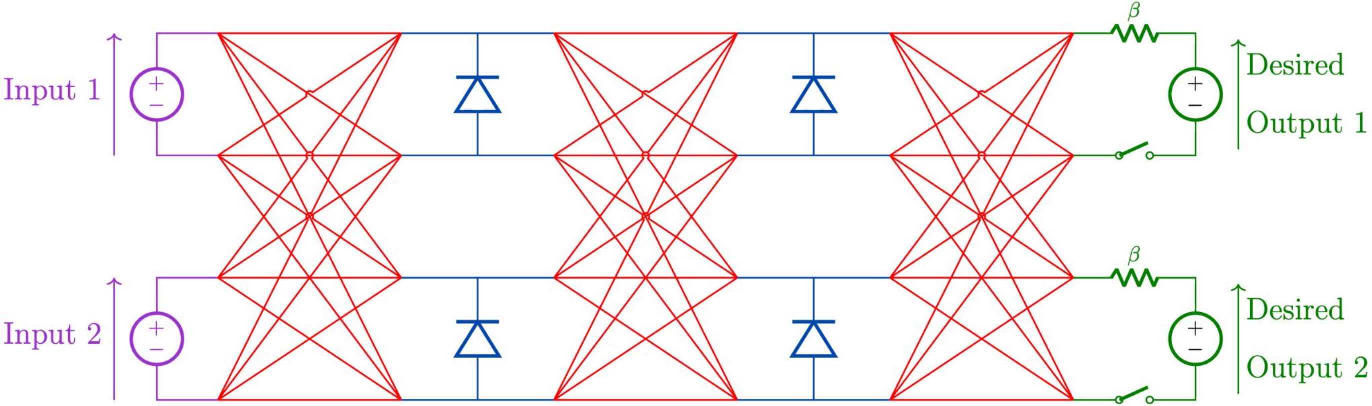
References:

Scellier and Bengio. "Equilibrium propagation." *Frontiers in computational neuroscience* 11 (2017): 24.

Kendall et al. "Training end-to-end analog neural networks with equilibrium propagation" (2020)

Equilibrium Propagation in a Deep Resistive Network

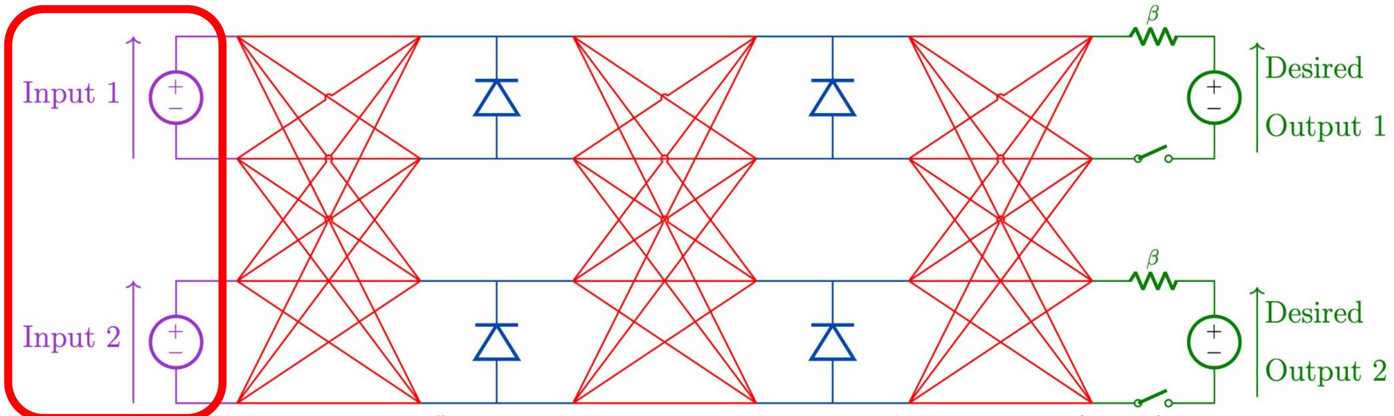
Training procedure:



References:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Equilibrium Propagation in a Deep Resistive Network

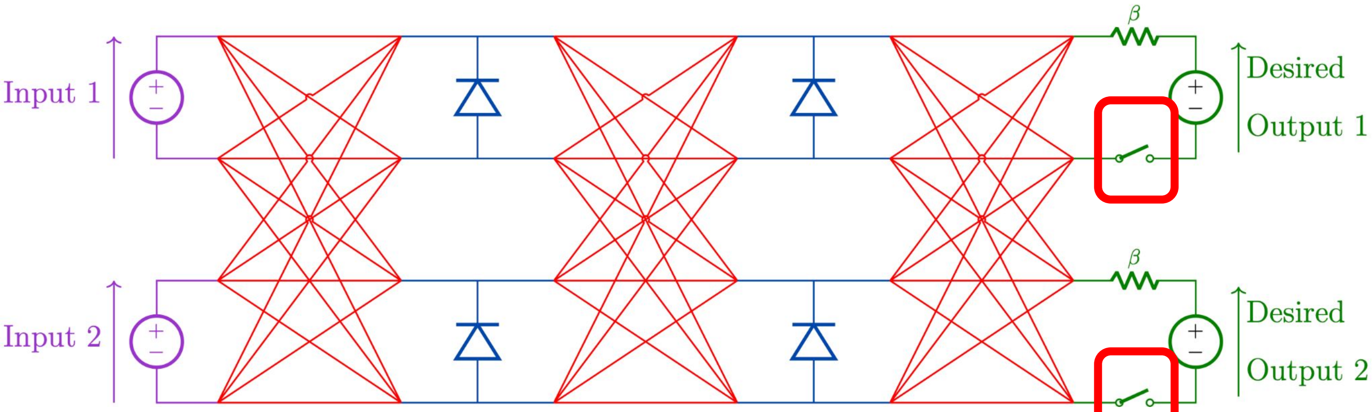
Training procedure:
1. Set input voltages.



References:
Scellier and Bengio. "Equilibrium propagation." *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. "Training end-to-end analog neural networks with equilibrium propagation" (2020)

Equilibrium Propagation in a Deep Resistive Network

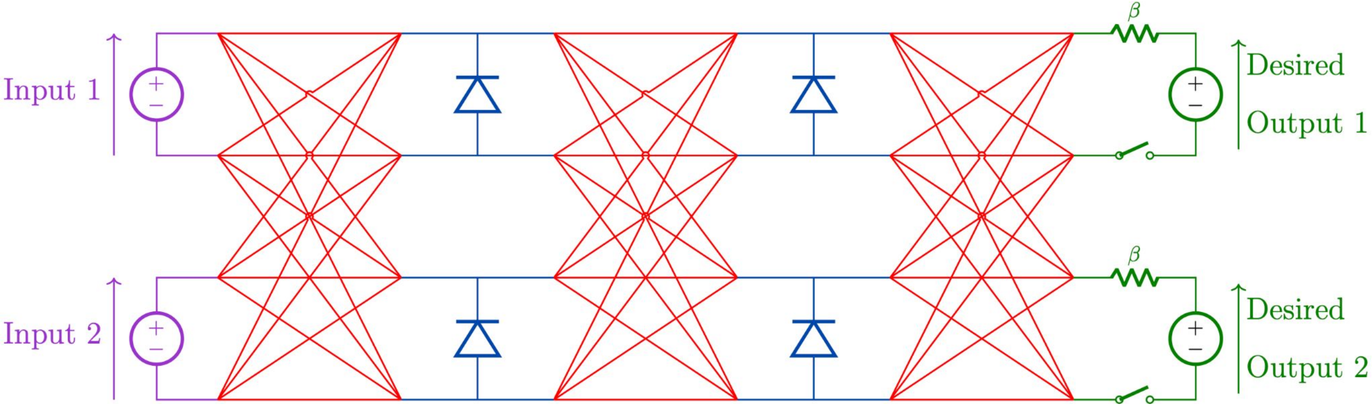
<u>Training procedure:</u>
1. Open output switches.



<u>References</u>:
Scellier and Bengio. "Equilibrium propagation." *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. "Training end-to-end analog neural networks with equilibrium propagation" (2020)

Equilibrium Propagation in a Deep Resistive Network

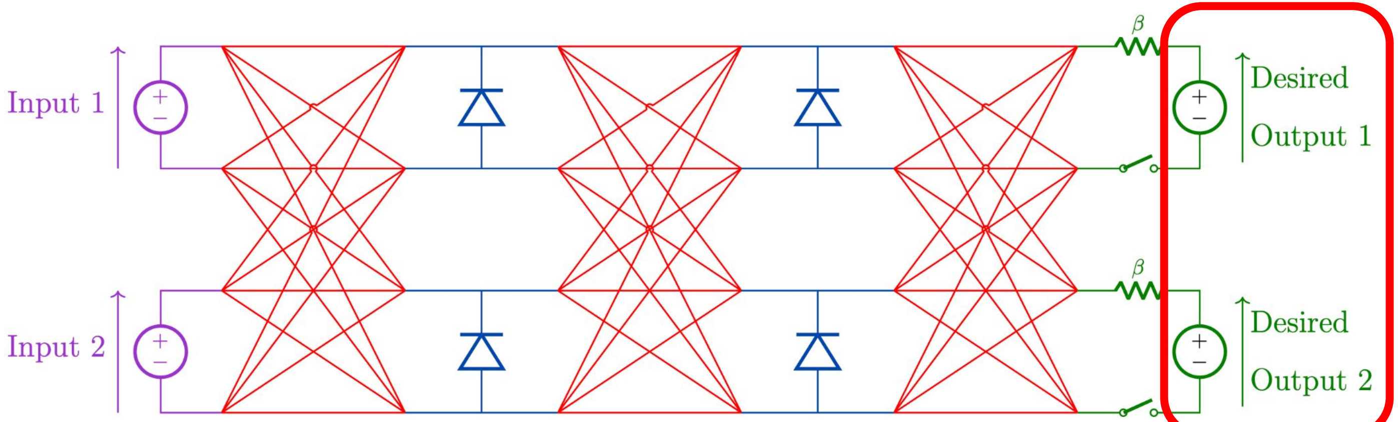
<u>Training procedure:</u>
1. Open output switches. Observe the steady state (“free state”).



<u>References</u>:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Equilibrium Propagation in a Deep Resistive Network

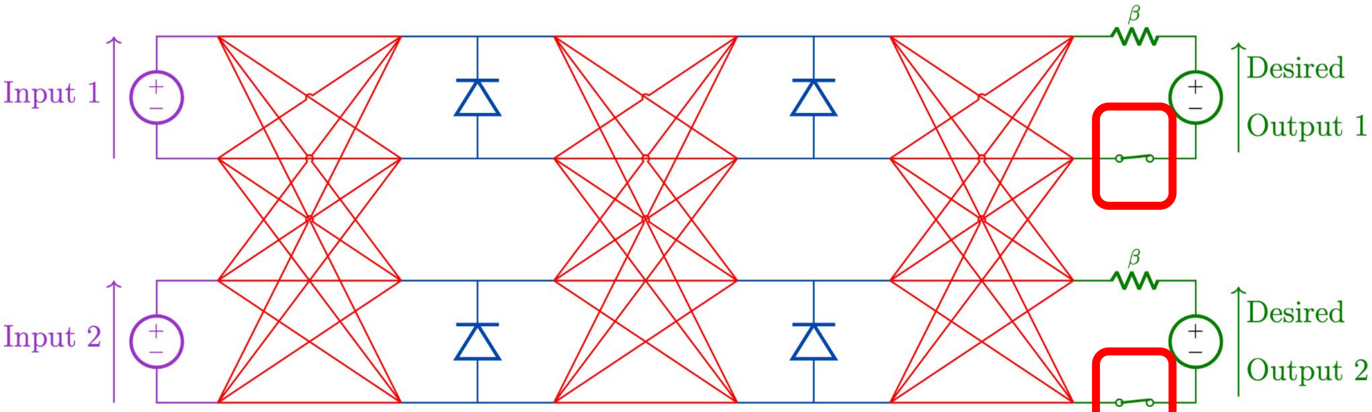
<u>Training procedure:</u>
1. Open output switches: “free state”
2. Set desired output voltages.



<u>References</u>:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Equilibrium Propagation in a Deep Resistive Network

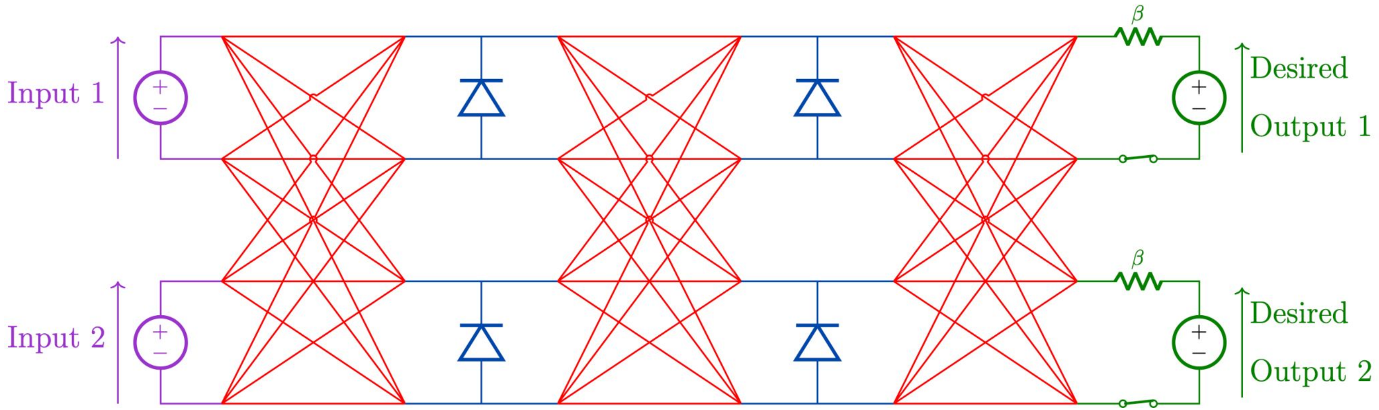
<u>Training procedure:</u>
1. Open output switches: “free state”
2. Close output switches.



<u>References</u>:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

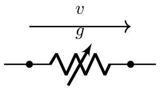
Equilibrium Propagation in a Deep Resistive Network

<u>Training procedure:</u>
1. Open output switches: “free state”
2. Close output switches. Observe the new steady state (“nudged state”).

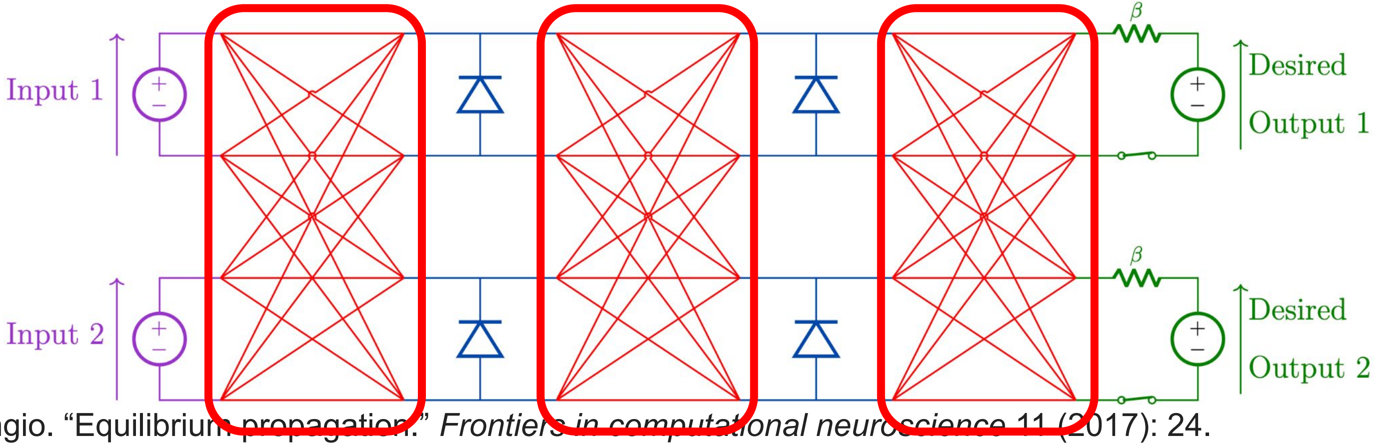


<u>References</u>:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Equilibrium Propagation in a Deep Resistive Network

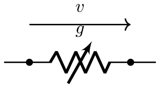
<u>Training procedure:</u>
1. Open output switches: “free state”
2. Close output switches: “nudged state”
3. For each resistor 

update the conductance g $\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$



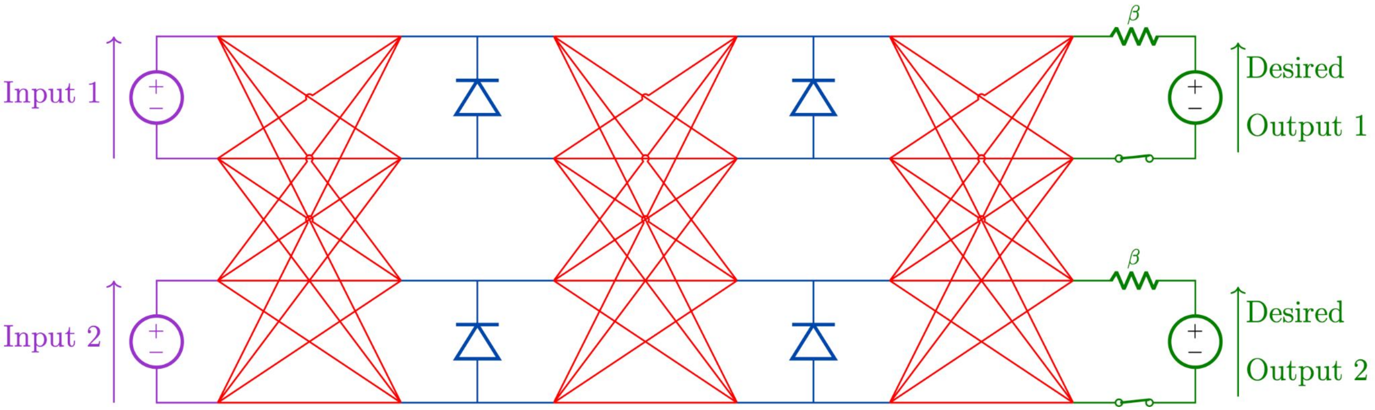
<u>References</u>:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Equilibrium Propagation in a Deep Resistive Network

Training procedure:
1. Open output switches: “free state”
2. Close output switches: “nudged state”
3. For each resistor  update the conductance g

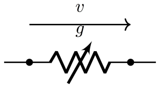
$$\boxed{\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)}$$

local in space



References:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

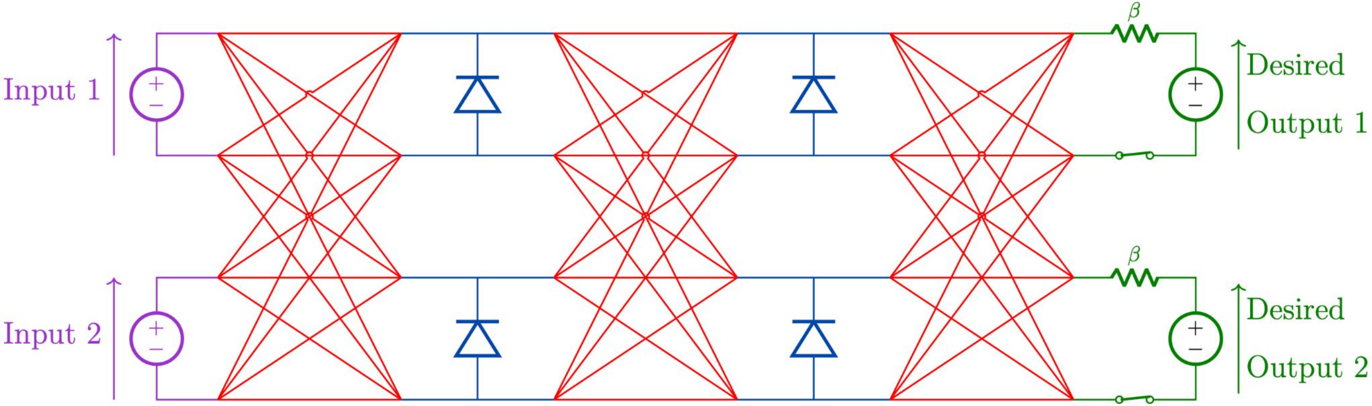
Equilibrium Propagation in a Deep Resistive Network

<u>Training procedure:</u>
1. Open output switches: “free state”
2. Close output switches: “nudged state”
3. For each resistor 

update the conductance g $\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$

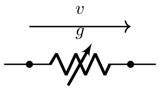
<u>Theorem</u>: weight updates approximate gradient descent on the MSE

$$\Delta g = -\eta \nabla_g \text{MSE} + O(\beta)$$



<u>References</u>:
Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Equilibrium Propagation in a Deep Resistive Network

<u>Training procedure:</u>
1. Open output switches: “free state”
2. Close output switches: “nudged state”
3. For each resistor 

update the conductance g $$\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$$

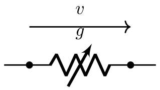
<u>Theorem</u>: weight updates approximate gradient descent on the MSE

$$\Delta g = -\eta \nabla_g \mathrm{MSE} + O(\beta)$$

<u>Bottom line</u>: **local learning rule** performs **stochastic gradient descent (SGD)**

Equilibrium Propagation in a Deep Resistive Network

Training procedure:
1. Open output switches: “free state”
2. Close output switches: “nudged state”

3. For each resistor 

update the conductance g $$\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$$

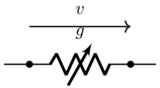
Theorem: weight updates approximate gradient descent on the cost function

$$\Delta g = -\eta \nabla_g C + O(\beta)$$

Bottom line: **local learning rule** performs **stochastic gradient descent (SGD)**

Remarks:
1. applicable to any cost function C (not just the MSE)

Equilibrium Propagation in a Deep Resistive Network

Training procedure:
1. Open output switches: “free state”
2. Close output switches: “nudged state”
3. For each resistor 

update the conductance g $\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$

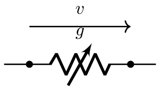
Theorem: weight updates approximate gradient descent on the cost function

$$\Delta g = -\eta \nabla_g C + O(\beta)$$

Bottom line: **local learning rule** performs **stochastic gradient descent (SGD)**

Remarks:
1. applicable to any cost function C (not just the MSE)
2. applicable to any network topology (not just a DRN)

Equilibrium Propagation in a Deep Resistive Network

Training procedure:
1. Open output switches: “free state”
2. Close output switches: “nudged state”
3. For each resistor 

update the conductance g $$\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$$

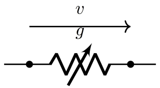
Theorem: weight updates approximate gradient descent on the cost function

$$\Delta g = -\eta \nabla_g C + O(\beta)$$

Bottom line: **local learning rule** performs **stochastic gradient descent (SGD)**

Remarks:
1. applicable to any cost function C (not just the MSE)
2. applicable to any network topology (not just a DRN)

Equilibrium Propagation in a Deep Resistive Network

Training procedure:
1. Open output switches: “free state”
2. Close output switches: “nudged state”
3. For each resistor 

update the conductance g $\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$

Theorem: weight updates perform gradient descent on a surrogate function

$$\Delta g = -\eta \nabla_g \mathcal{L}_\beta$$ where $$\mathcal{L}_\beta = C + O(\beta)$$

Bottom line: **local learning rule** performs **stochastic gradient descent (SGD)**

Remarks:
1. applicable to any cost function C (not just the MSE)
2. applicable to any network topology (not just a DRN)

Equilibrium Propagation in a Deep Resistive Network

<u>Sketch of the proof:</u>

- Equilibrium propagation generally applies to systems whose steady state minimizes a functional (variational principle) [1,2]

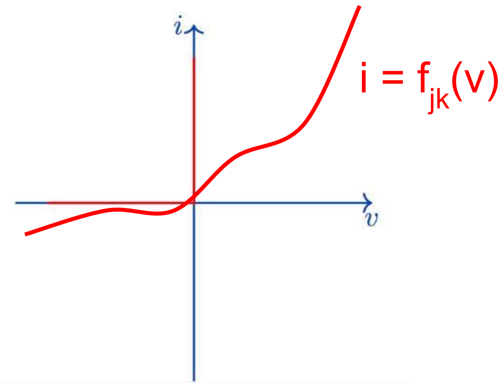
- Nonlinear resistive networks minimize a functional called *co-content* [3]

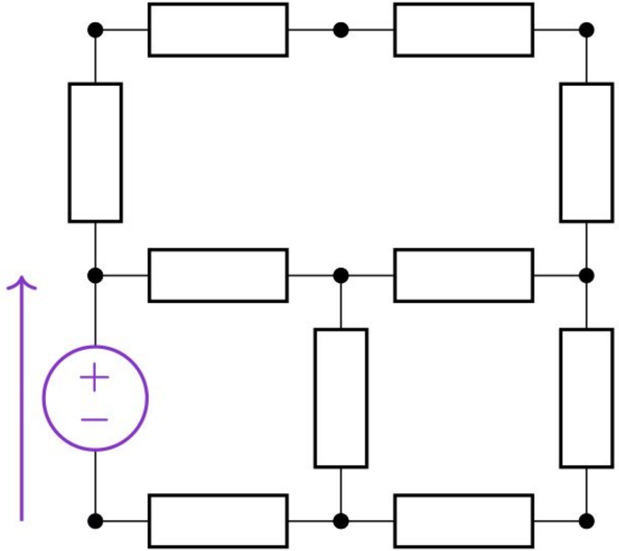
<u>References</u>

[1] Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
[2] Scellier. “A deep learning theory for neural networks grounded in physics.” *PhD thesis, Université de Montréal* (2021).
[3] Millar. "CXVI. Some general theorems for non-linear systems possessing resistance." The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 42.333 (1951): 1150-1160.

What is the co-content?

i-v curve of branch (j,k)





Co-content of branch (j,k)

$$E_{jk} = \int_0^{v_{jk}} f_{jk}(u)du$$

Total co-content of the circuit

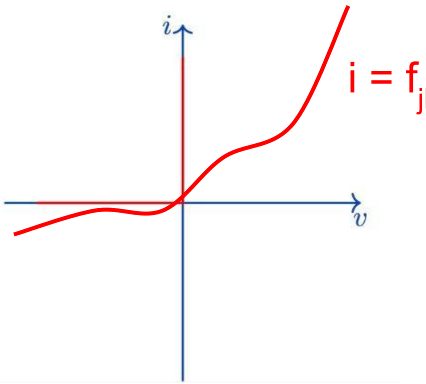
$$E_{\text{total}} = \sum_{j,k} E_{jk}$$

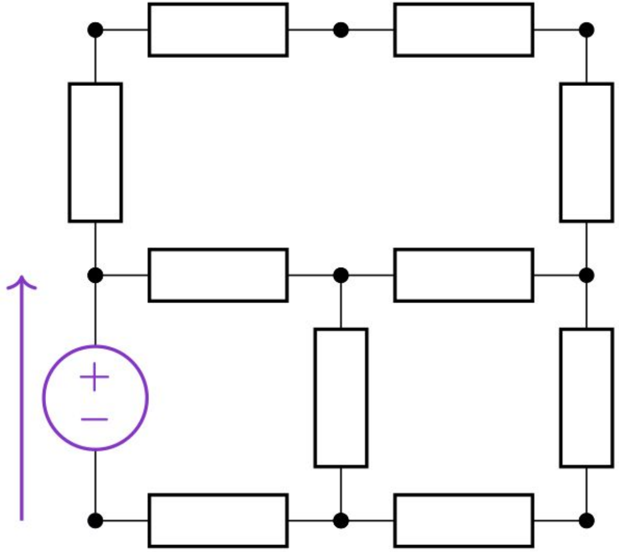
References

Millar. "CXVI. Some general theorems for non-linear systems possessing resistance." The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 42.333 (1951): 1150-1160.

Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

i-v curve of branch (j,k)





Co-content of branch (j,k)

$$E_{jk} = \int_0^{v_{jk}} f_{jk}(u)du$$

Total co-content of the circuit

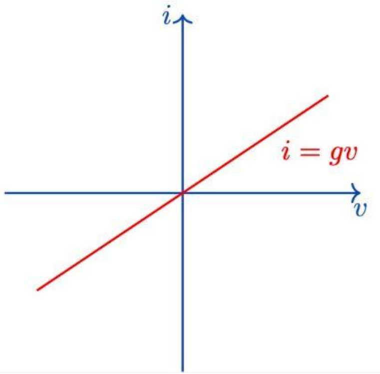
$$E_{\text{total}} = \sum_{j,k} E_{jk}$$

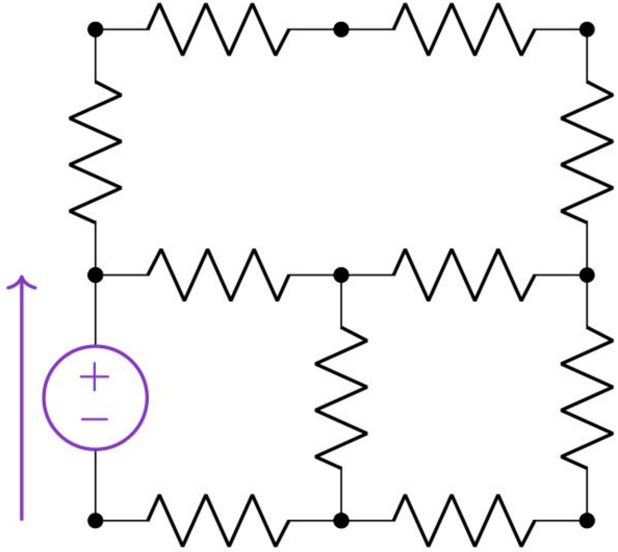
branch equation + KVL

KCL

$$\delta E_{\text{total}} = \sum_{j,k} \delta E_{jk} = \sum_{j,k} f_{jk}\, \delta v_{jk} \overset{\downarrow}{=} \sum_{j,k} i_{jk}(\delta v_j - \delta v_k) = \sum_j \delta v_j \left(\sum_k i_{jk} \right) \overset{\downarrow}{=} 0$$

Linear resistor





Co-content of linear resistor

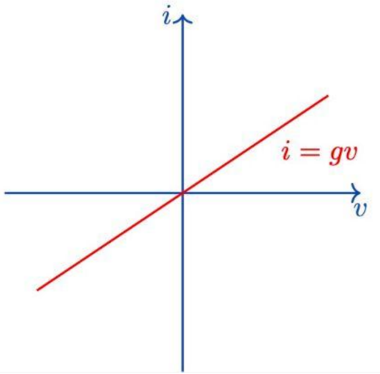
$$E = \frac{1}{2} g v^2$$

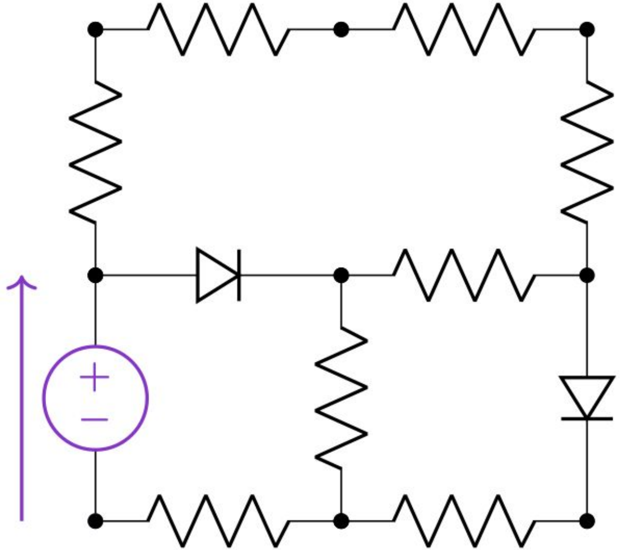
(half the power dissipation)

Total co-content of the circuit

$$E_{\text{total}} = \frac{1}{2} \sum_{j,k} g_{jk} v_{jk}^2$$

Linear resistor





Co-content of linear resistor

$$E = \frac{1}{2} g v^2$$

(half the power dissipation)

Total co-content of the circuit

$$E_{\text{total}} = \frac{1}{2} \sum_{j,k} g_{jk} v_{jk}^2 + E_{\text{diodes}}$$

Equilibrium Propagation Learning Rule

Learning rule

$$\Delta g = \frac{\eta}{2\beta}\left(\underbrace{v_{\text{free}}^2}_{\partial_g E(\text{free})} - \underbrace{v_{\text{nudged}}^2}_{\partial_g E(\text{nudged})}\right)$$

Co-content

$$E = \frac{1}{2}\sum_{\text{resistor}} g v^2 + E_{\text{diodes}}$$

Reference: Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.

Equilibrium Propagation Learning Rule

Learning rule

$$\Delta g = \frac{\eta}{\beta}\left(\partial_g E(\text{free}) - \partial_g E(\text{nudged})\right)$$

Co-content

$$E = \frac{1}{2}\sum_{\text{resistor}} g v^2 + E_{\text{diodes}}$$

Reference: Scellier and Bengio. "Equilibrium propagation." *Frontiers in computational neuroscience* 11 (2017): 24.

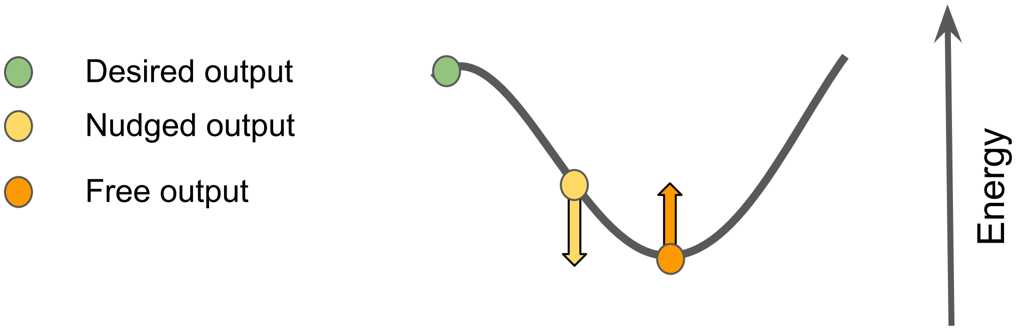
Equilibrium Propagation Learning Rule

Learning rule

$$\Delta w = \frac{\eta}{\beta} \left(\partial_w E(\text{free}) - \partial_w E(\text{nudged}) \right)$$

Reference: Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.

Equilibrium Propagation Learning Rule



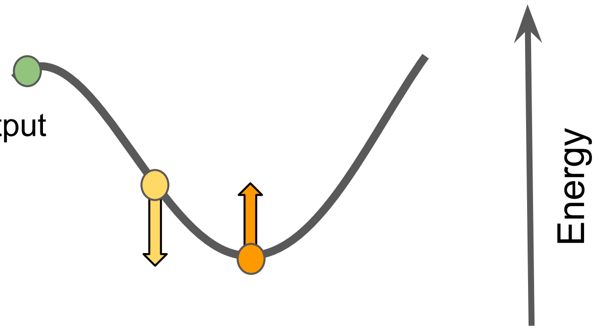
Learning rule

$$\Delta w = \frac{\eta}{\beta}\left(\partial_w E(\mathrm{free}) - \partial_w E(\mathrm{nudged})\right)$$

Reference: Scellier and Bengio. "Equilibrium propagation." *Frontiers in computational neuroscience* 11 (2017): 24.

Variants of Equilibrium Propagation (EP)

- Desired output
- Positively-perturbed output
- Free output



Positively-perturbed EP
pull outputs **towards** desired outputs

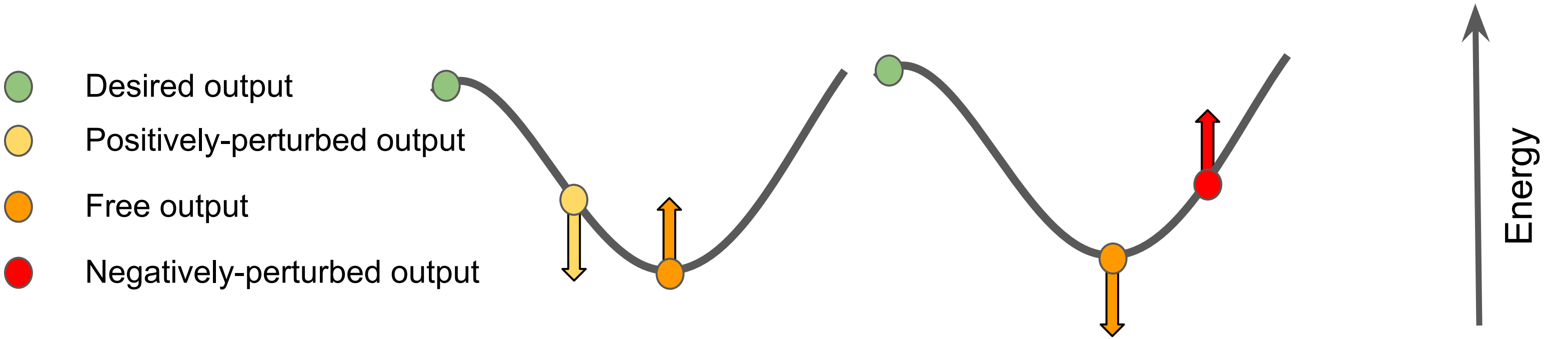
References:

Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.

Laborieux et al. “Scaling equilibrium propagation to deep ConvNets by reducing its gradient estimator bias” (2021)

Scellier et al. “Energy-based learning algorithms: a comparative study” NeurIPS (2023)

Variants of Equilibrium Propagation (EP)



Positively-perturbed EP	Negatively-perturbed EP
pull outputs **towards** desired outputs	**push** outputs **away from** desired outputs

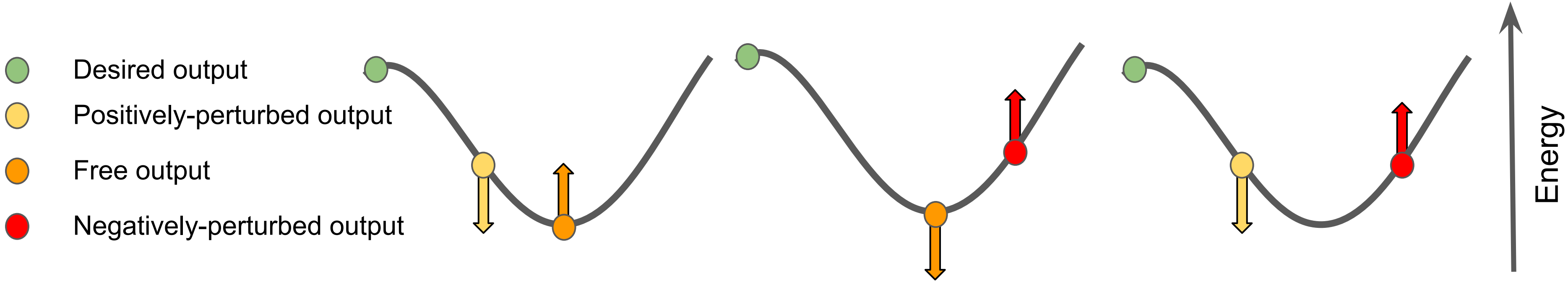
References:

Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.

Laborieux et al. “Scaling equilibrium propagation to deep ConvNets by reducing its gradient estimator bias” (2021)

Scellier et al. “Energy-based learning algorithms: a comparative study” NeurIPS (2023)

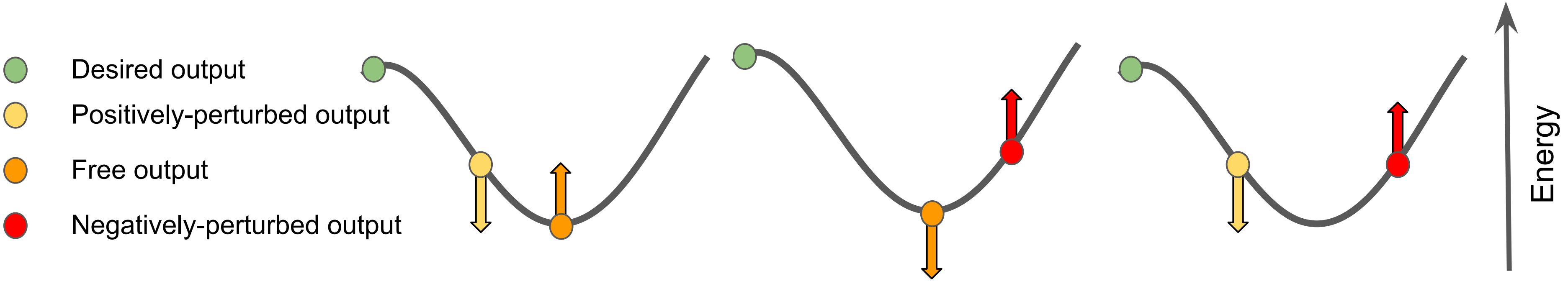
Variants of Equilibrium Propagation (EP)



Positively-perturbed EP	Negatively-perturbed EP	Centered EP
pull outputs **towards** desired outputs	**push** outputs **away from** desired outputs	➔ one **positive** perturbation ➔ one **negative** perturbation

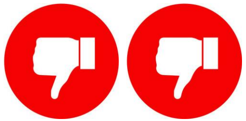
References:
Scellier and Bengio. "Equilibrium propagation." *Frontiers in computational neuroscience* 11 (2017): 24.
Laborieux et al. "Scaling equilibrium propagation to deep ConvNets by reducing its gradient estimator bias" (2021)
Scellier et al. "Energy-based learning algorithms: a comparative study" NeurIPS (2023)

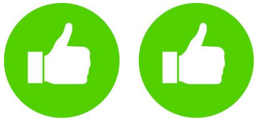
Variants of Equilibrium Propagation (EP)



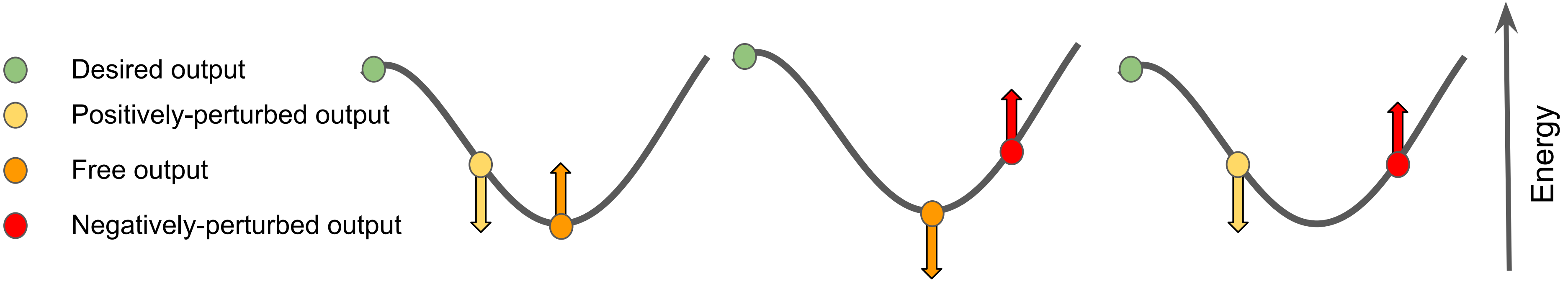
Positively-perturbed EP	Negatively-perturbed EP	Centered EP
pull outputs **towards** desired outputs	**push** outputs **away from** desired outputs	➔ one **positive** perturbation ➔ one **negative** perturbation

Performance in practice?





Variants of Equilibrium Propagation (EP)



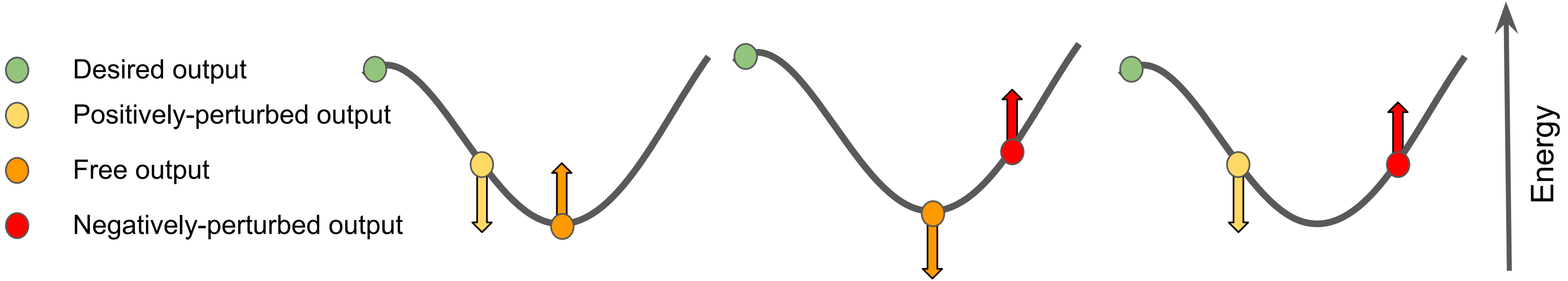
	Positively-perturbed EP	Negatively-perturbed EP	Centered EP

<u>Theorem</u>: EP computes the gradient of a surrogate function that approximates the cost function

$$\Delta w = -\eta \nabla_w \mathcal{L}_\beta \qquad\qquad \mathcal{L}_\beta = C + O(\beta)$$

<u>Reference:</u> Scellier et al. “Energy-based learning algorithms: a comparative study” NeurIPS (2023)

Variants of Equilibrium Propagation (EP)

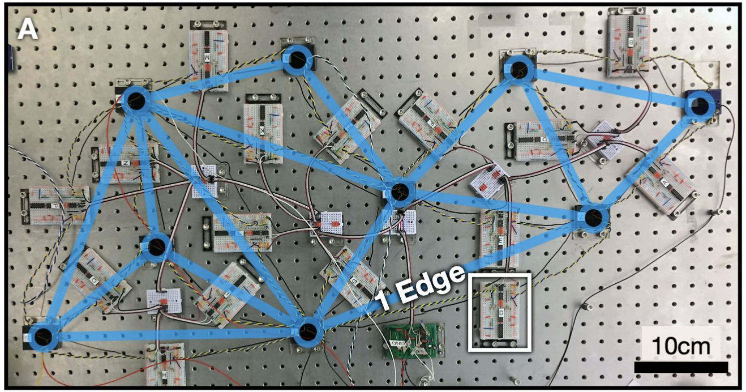


	Positively-perturbed EP	Negatively-perturbed EP	Centered EP
Properties of the surrogate function	$L_{P\text{-}EP} = C +$ **O(β)** $L_{P\text{-}EP} \leq C$ (**lower bound**)	$L_{N\text{-}EP} = C +$ **O(β)** $L_{N\text{-}EP} \geq C$ (**upper bound**)	$L_{C\text{-}EP} = C +$ **O(β²)**

<u>Reference:</u> Scellier et al. "Energy-based learning algorithms: a comparative study" NeurIPS (2023)

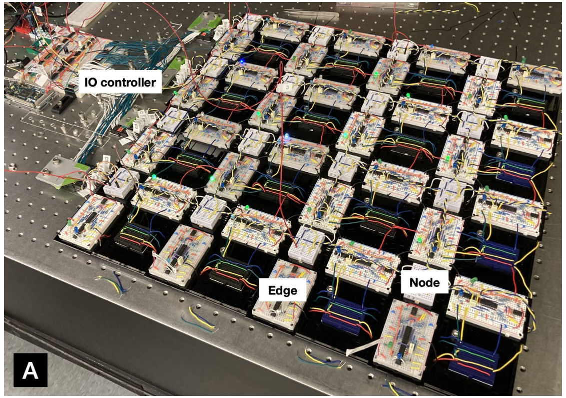
Hardware Experiments

Linear resistor network [1]



- Network of 9 nodes and 16 edges
- Task: classify the Iris dataset

Nonlinear resistive network [2]

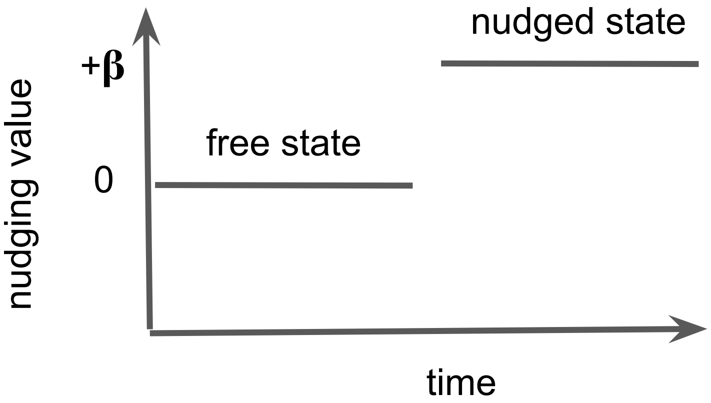


- Network of 32 (transistor-based) twin edges
- Tasks: XOR and nonlinear regression

Caveat: use two copies of the same network
(one for the free state, one for the nudge state)

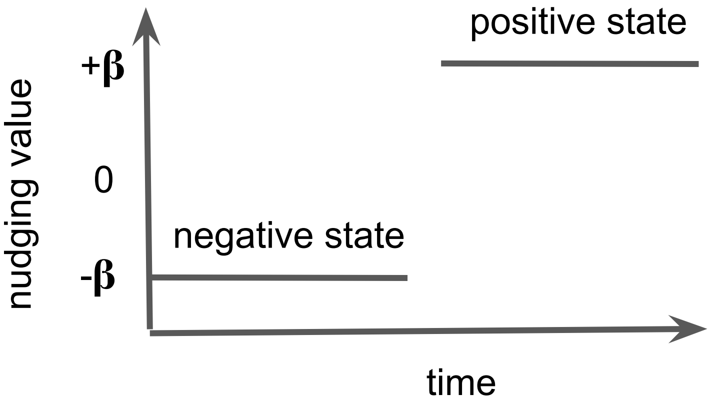
References
[1] Dillavou et al. "Demonstration of decentralized physics-driven learning." Physical Review Applied (2022)
[2] Dillavou et al. "Circuits that train themselves: decentralized, physics-driven learning." (2023)

Equilibrium propagation

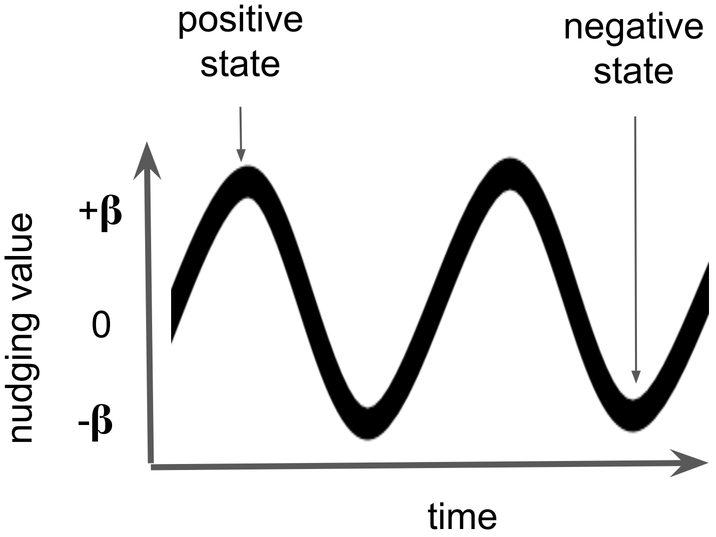


Learning rule $\Delta g = \frac{\eta}{2\beta}\left(v_{\text{free}}^2 - v_{\text{nudged}}^2\right)$

Equilibrium propagation



Frequency propagation [1]



Learning rule $$\Delta g = \frac{\eta}{4\beta}\left(v_{\mathrm{neg}}^2 - v_{\mathrm{pos}}^2\right) = \frac{\eta}{4\beta}\left(v_{\mathrm{neg}} + v_{\mathrm{pos}}\right)\left(v_{\mathrm{neg}} - v_{\mathrm{pos}}\right)$$

state(t) = mean + amplitude * sin(ωt) + O($\boldsymbol{\beta}^2$)

Reference
[1] Anisetti et al. "Frequency propagation: Multi-mechanism learning in nonlinear physical networks." (2022)

Simulations of deep resistive networks

Task: train a DRN to classify MNIST digits

	Network size (number of parameters)	Number of epochs	Total duration	Test error rate
SPICE simulations [1]	0.16M	10	1 week	3.43%

References
[1] Kendall et al. “Training end-to-end analog neural networks with equilibrium propagation” (2020)

Simulations of deep resistive networks

Task: train a DRN to classify MNIST digits

	Network size (number of parameters)	Number of epochs	Total duration	Test error rate
SPICE simulations [1]	0.16M	10	1 week	3.43%
DRN simulator [2,3]	51M	50	6 hours	1.40%
	320x larger	5x more	28x shorter	

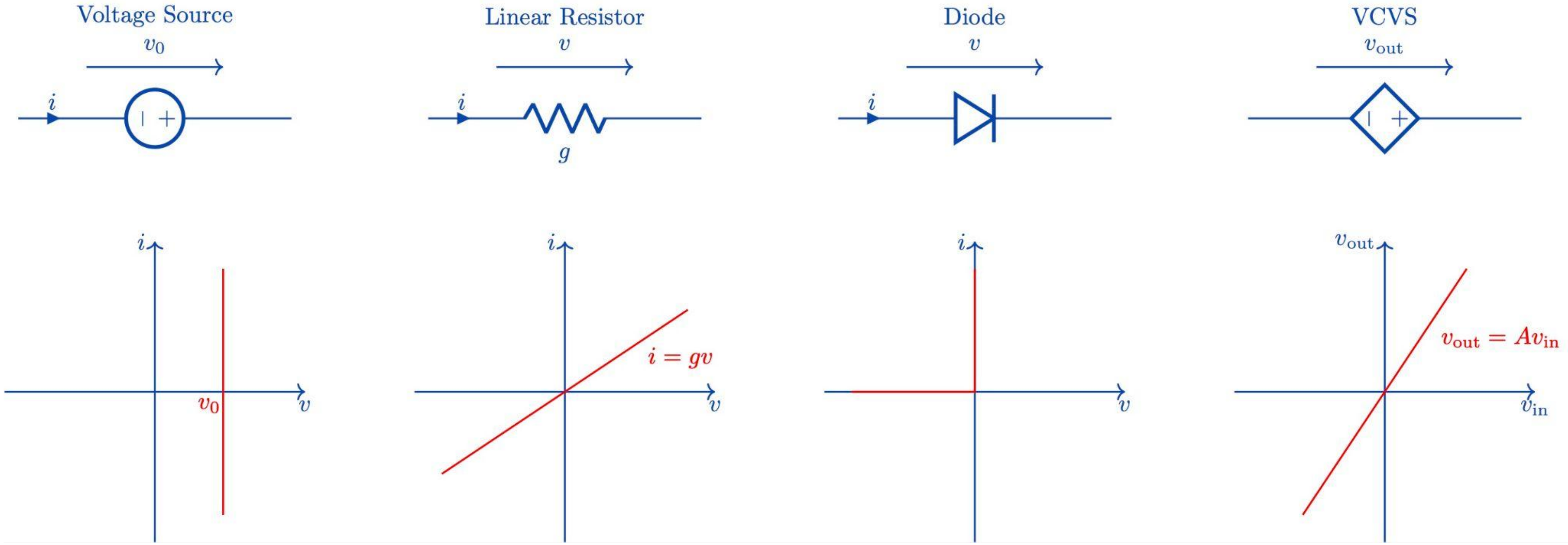
The "network size" to "epoch duration" ratio is 45000 times larger

References
[1] Kendall et al. "Training end-to-end analog neural networks with equilibrium propagation" (2020)
[2] "A fast algorithm to simulate nonlinear resistive networks". To appear.
[3] Code will be available at https://github.com/rain-neuromorphics/energy-based-learning

Simulations of deep resistive networks

Main idea:

- assumption: the circuit elements are **ideal**



Reference: “A fast algorithm to simulate nonlinear resistive networks”. To appear.

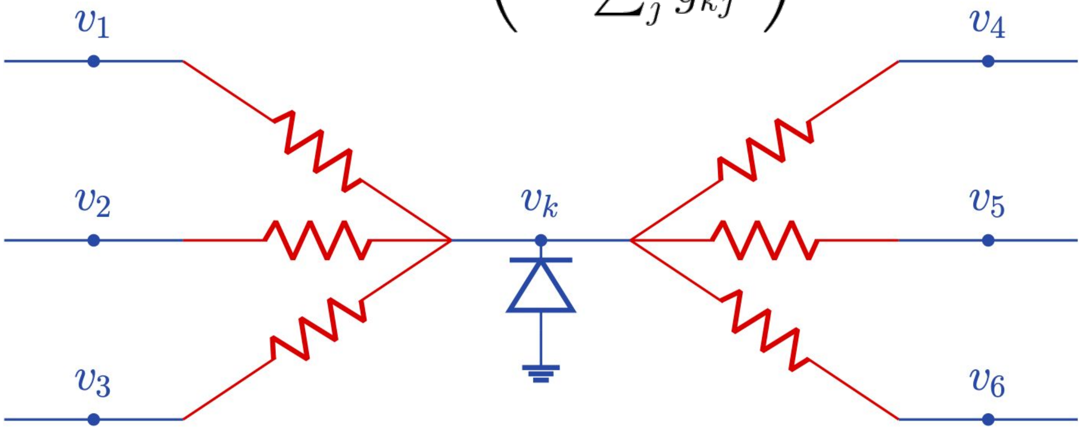
Simulations of deep resistive networks

Main idea:

- assumption: the circuit elements are **ideal**
- the math considerably simplifies

Equation of a node voltage

$$v_k = \max\left(0, \frac{\sum_j g_{kj} v_j}{\sum_j g_{kj}}\right)$$



Reference: “A fast algorithm to simulate nonlinear resistive networks”. To appear.

Simulator for energy-based algorithms

Three key abstractions

Energy function

- deep resistive network (DRN)
- …

Energy minimizer

(in the space of network configurations)

- block coordinate descent
- …

Learning algorithm

(in the weight space)

- equilibrium propagation (positive, negative, centered, …)
- …

Link: Code will be available at https://github.com/rain-neuromorphics/energy-based-learning

Simulator for energy-based algorithms

Three key abstractions

Energy function

- deep resistive network (DRN)
- deep Hopfield network (DHN)
- …

Energy minimizer

(in the space of network configurations)

- block coordinate descent
- gradient descent
- …

Learning algorithm

(in the weight space)

- equilibrium propagation (positive, negative, centered, …)
- contrastive (Hebbian) learning
- coupled learning
- …
- truncated backpropagation (baseline)
- recurrent backpropagation (baseline)

Link: Code will be available at https://github.com/rain-neuromorphics/energy-based-learning

Simulations of Hopfield networks

Scellier and Bengio [1]

MNIST

Ernoult et al. [2]

MNIST
Convolutional HN

Laborieux et al. [3]

CIFAR-10
3 days on a single GPU

Laborieux and Zenke [4]

ImageNet 32*
* downsampled to 32x32 pixels
6 days on 4 GPUs

Scellier et al. [5]

CIFAR-100
3 hours on a single GPU
13.5 faster than [3]

References

[1] Scellier and Bengio. “Equilibrium propagation.” *Frontiers in computational neuroscience* 11 (2017): 24.
[2] Ernoult et al. "Updates of equilibrium prop match gradients of backprop through time." NeurIPS (2019).
[3] Laborieux et al. “Scaling equilibrium propagation to deep ConvNets.” Frontiers in neuroscience (2021).
[4] Laborieux and Zenke. "Holomorphic equilibrium propagation." NeurIPS (2022).
[5] Scellier et al. “Energy-based learning algorithms: a comparative study.” NeurIPS (2023)

Hardware experiments of Hopfield networks

nature electronics

Article https://doi.org/10.1038/s41928-022-00869-w

Activity-difference training of deep neural networks using memristor crossbars

Received: 10 March 2022

Accepted: 13 October 2022

Su-in Yi[1], Jack D. Kendall[2], R. Stanley Williams[1] & Suhas Kumar[3]

- 64 × 64 memristor crossbar array
- Task: classify Braille words
- demonstrate **10,000x improvement** (over GPUs) in energy efficiency for training

Caveat: use additional (SRAM) memory to store the free and nudged states

Reference: Yi et al. "Activity-difference training of deep neural networks using memristor crossbars." Nature Electronics (2023)

Summary

Training deep resistive networks (DRNs) with equilibrium propagation (EP):

1. DRNs are universal function approximators
2. EP performs gradient descent on a cost function
3. DRN simulator is 45,000x faster than SPICE simulations
4. Small-scale experiments demonstrate 10,000x energy efficiency gains over GPUs

Thanks to all my collaborators!

Jack Kendall

Maxence Ernoult

Suhas Kumar

Ross Pantone

RAIN

Yoshua Bengio

Axel Laborieux

Julie Grollier

Damien Querlioz

Yann Ollivier

Siddhartha Mishra

Kalpana Manickavasagam

Vidyesh Anisetti

Ananth Kandala

Jennifer Schwarz

Thank you!