Neuromorphic Computing with Spiking Neural Networks

David L Escudero

Introduction - The Problem of Demand

- Machine solutions for abstract problems have increased in popularity
- Power demand for algorithms grows with complexity of network
  - GPT-3 has approximately 175 billion parameters
- New focus on less demanding networks
  - Neuromorphic computing
  - Spiking neural networks

Outline

- Introduction
- ***Background***
- Convolutional neural networks
- Spiking neural networks
- Training of spiking neural networks
- Approach by Ledinauskas et al.
- Results and conclusions

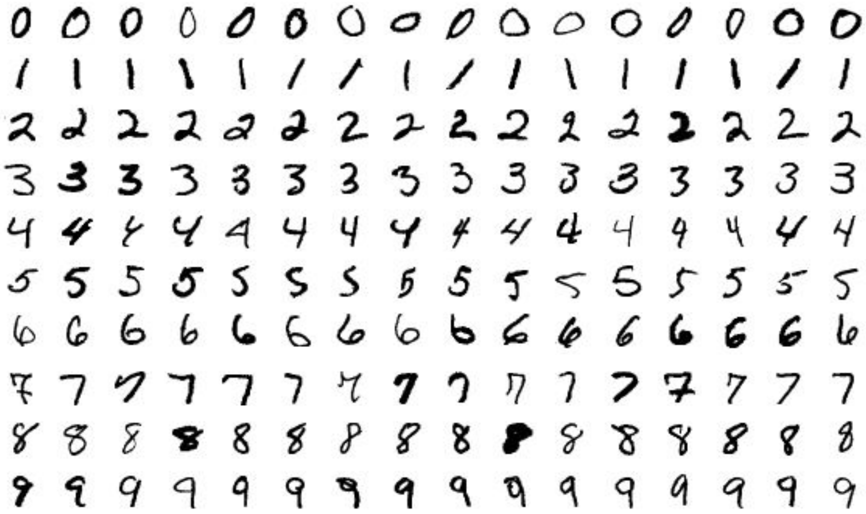
Background - Neuromorphic Computing

Neuromorphic computing

- biologically inspired technology
    - Architectures of the biological neurons
- Biology functional and proven
- Aims
    - Match/exceed human ability
    - Reduce power consumption

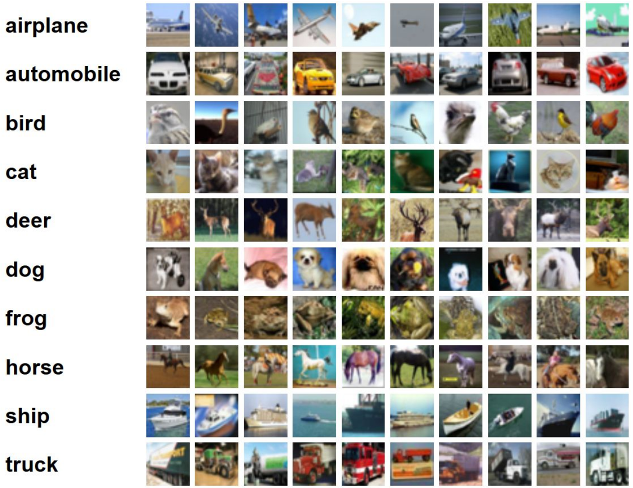
Background - MNIST Dataset

- MNIST - Modified National Institute of Standards and Technology
- Developed by Y. LeCun, C. Cortes, C. Burges in 1998
- Specifications
  - 60,000 hand drawn digits ranging from 0-9
  - Values are black and white, with grey values from interpolation
  - Digits are 28x28 centered boxes
- Used extensively for image classification



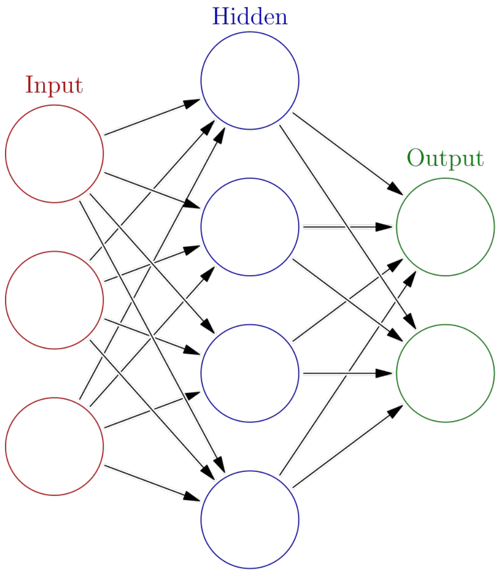
Background - CIFAR datasets

- CIFAR - Canadian Institute for Advanced Research
- Developed by A. Krizhevsky, V. Nair, G. Hinton in 2009
- Specifications, CIFAR-10
  - 60,000 32x32 images from 10 classes
  - Each class has 6,000 images
- Specifications, CIFAR-100
  - 60,000 32x32 images from 100 classes
  - Each class has 600 images



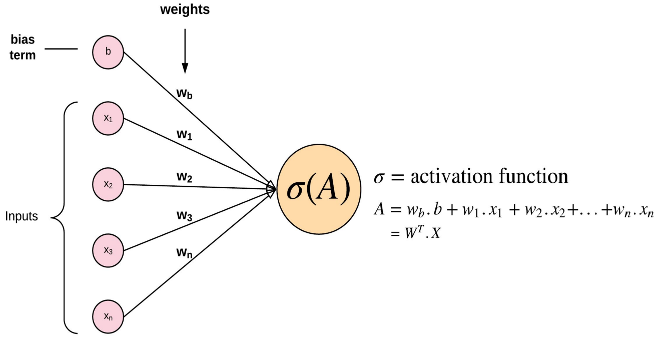
Background - Neural Nets

- Artificial neurons and weights
- *Neural nets approximate functions*
- Simplest neural networks
  - Divided into layers
    - Input
    - Hidden layer(s)
    - Output



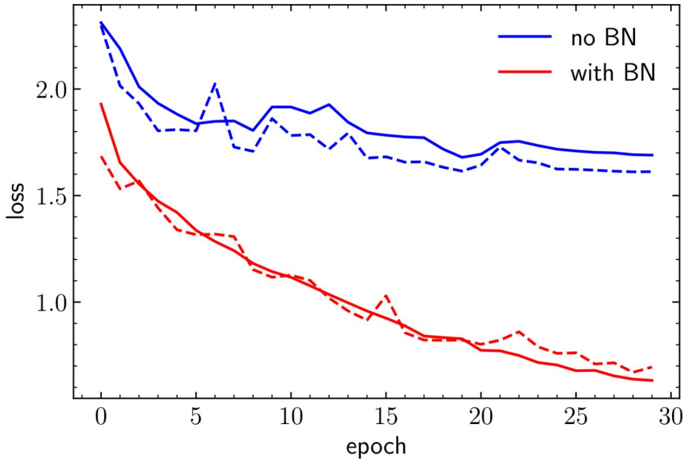
Background - Neural Nets Ct’d

- Activation functions
  - Determine output by given neuron
  - Different functions exist for different applications
- The role of weights
  - Weights summed together
  - *Training* - process of making a neural network nonfunctional to functional



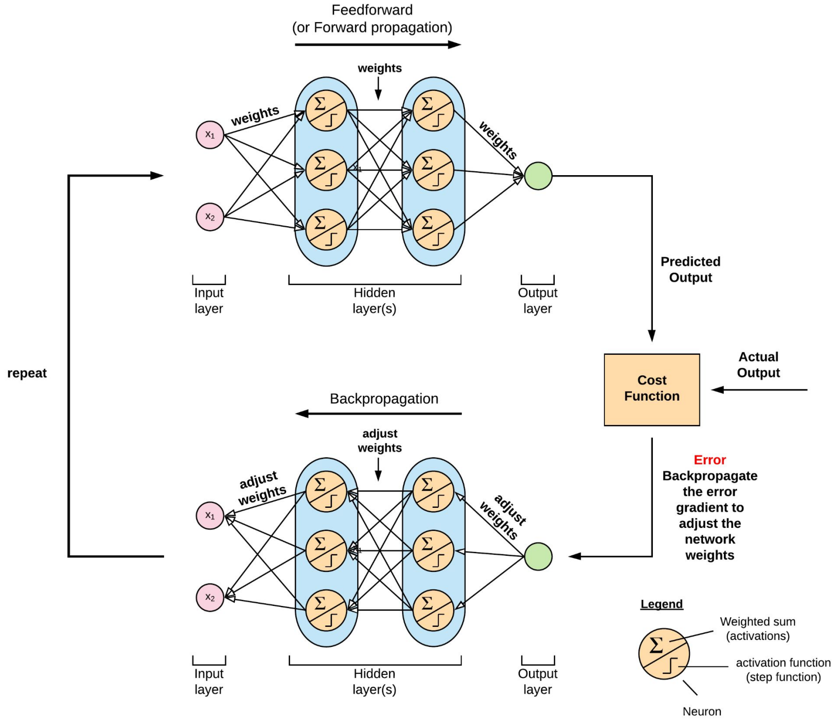
Background - Neural Nets Ct’d

- Weight balancing through backpropagation
  - Weights randomly assigned
  - Inputs ‘fed’ into the network, output recorded
  - Outputs compared to ‘ground truth’
  - Loss/cost function is generated
  - Calculus is used to guide loss downward
- Process of training and backprop is called *gradient descent*
- Each loop of the gradient descent cycle is an *epoch*



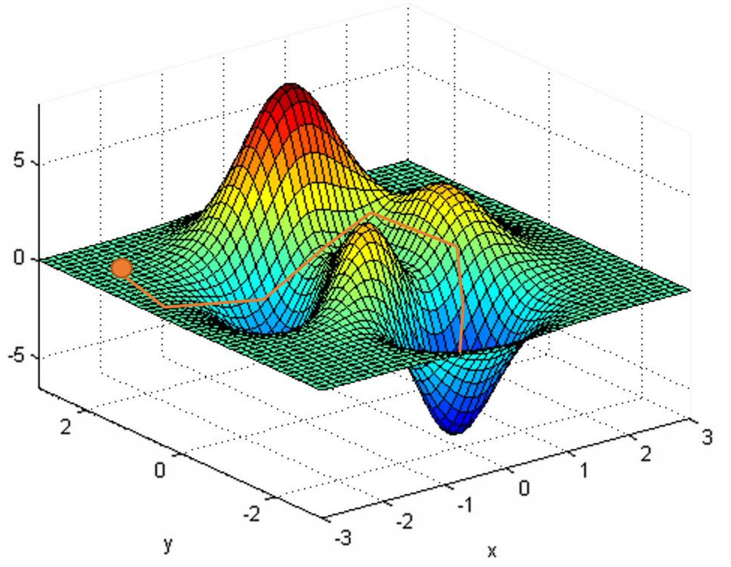
Basic Gradient Descent Cycle

1. A network is made and weights are assigned at random
2. Known Input is fed into network using input layer and output is recorded
3. Predicted output is compared to ground truth output, generating a cost function
4. Cost function is used to ‘nudge’ weights to minimize cost function
5. Repeat from 2 until predicted output matches actual output.



Pitfalls of gradient descent training

- Gradient descent is not perfect
  - Model trapped in local minima
  - Adjustment of *hyperparameters* may resolve this
- Vanishing/exploding gradient problem
  - Vanishing - slope is too low, no 'distance' is travelled
  - Exploding - slope is too high, model 'overshoots'



Background - Comparing Neural Networks

- Neural networks organized by purpose
  - Examples
    - Natural language processing (NLP)
    - Image classification
    - Speech synthesis
- Common, well-defined datasets
  - MNIST
  - CIFAR-10
  - CIFAR-100

Outline

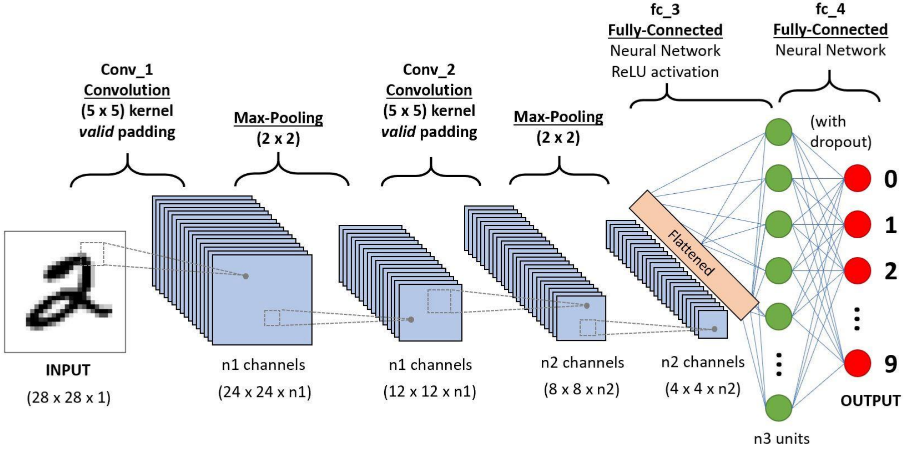
- Introduction
- Background
- ***Convolutional neural networks***
- Spiking neural networks
- Training of spiking neural networks
- Approach by Ledinauskas et al.
- Results and conclusions

Convolutional Neural Networks

- Useful in image processing and classification
- Loosely mimics the human visual system
    - Neural circuits ‘look’ for features in vision
- Images goes through processes of *convolution* and *pooling* before being fed into a connected neural network
    - Convolution and pooling extract key information from images to minimize processing time

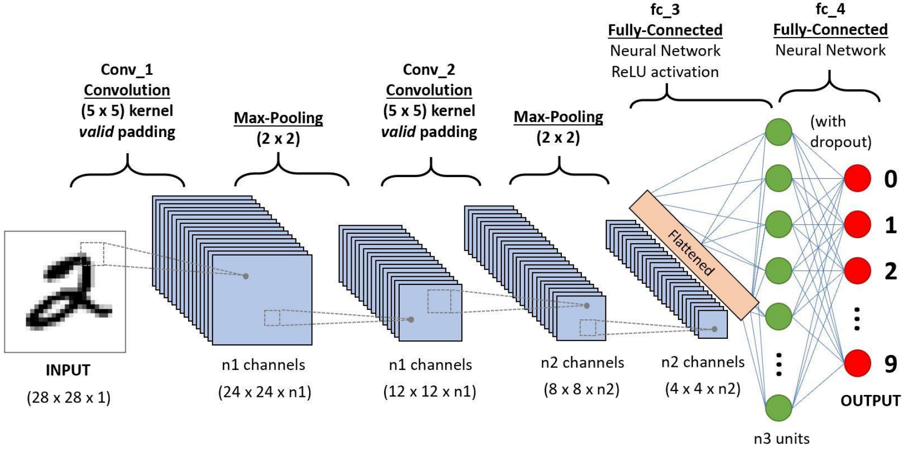
Convolution & Pooling

- Convolution is a process of comparing an image to a ‘mask’
- Convolution process:
  - An input image is transformed into a numerical representation
  - A square mask or *kernel*, moves through the image left to right, up to down
  - Each step the kernel convolutes itself with the image to build a *feature map*
- Multiple kernels define multiple features



Convolution & Pooling

- Often there are multiple convolution and pooling steps
- MNIST convolution
  - Input is given
  - Multiple 5x5 kernels pass through input generating multiple 24x24 feature maps
  - A 2x2 max pool kernel passes through the feature map, generating a 12x12 feature map
  - Convolution/pooling step happens again, generating final 4x4 feature maps
  - Feature maps are passed into neural network



Convolution & Pooling - Advantages

- For a 1920x1080 image
  - 2,073,600 pixels = 2,073,600 inputs
  - Layers are often *fully connected*
  - 1 layer in → 4.3 trillion connections
- Rule of thumb for neural nets: more weights require more training data
  - 4.3 trillion connections needs a LOT of training data.

Outline

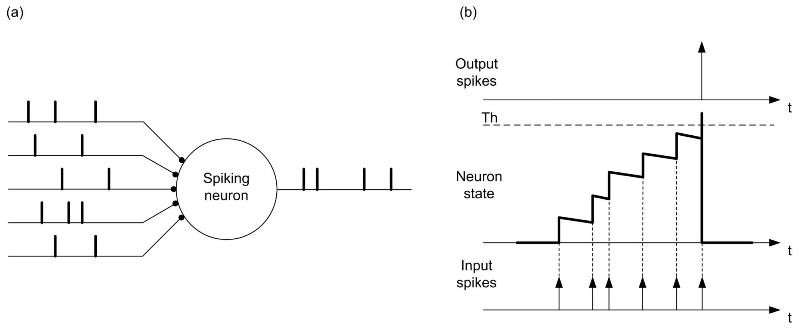
- Introduction
- Background
- Convolutional neural networks
- ***Spiking neural networks***
- Training of spiking neural networks
- Approach by Ledinauskas et al.
- Results and conclusions

Spiking Neural Networks

- Spiking neural networks (SNNs) are a newer type of NN
- SNNs are more closely related to biological neurons
- Utilize a series of pulses, known as *spikes*
  - Spikes are binary
  - *Pulses happen over time*
- Pulses contribute to *synaptic potential*
  - Once a *synaptic threshold* is reached neuron fires a spike and potential returns to baseline
- Efficiency is derived from fewer training and inference steps

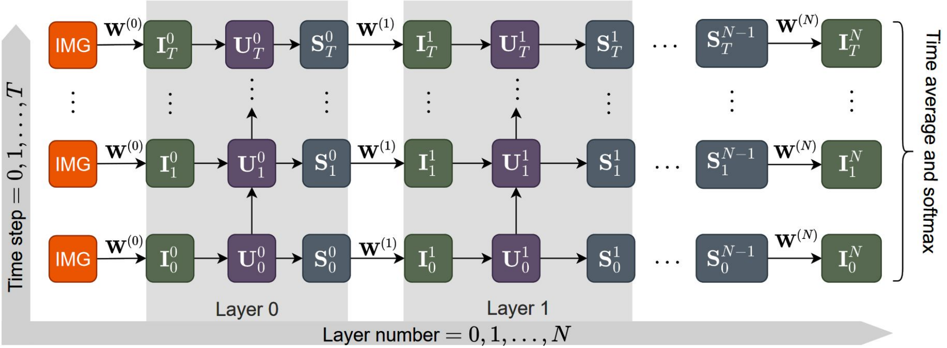
Neurons and Spiking Neurons

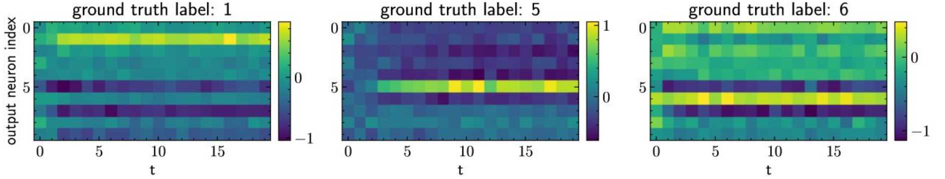
- Traditional neurons
  - Don’t really rely on time
  - Images are mapped to intensities
  - Strength of inputs determines given output
- Spiking neurons
  - Learning happens over time
  - Neurons with no input lose synaptic potential
  - Training becomes a problem since time affects output



Spiking Neural Networks - Visualization

- Superscripts denote layer, subscripts denote time
- **W** is a matrix of weights leading into each layer
- Each layer has 3 parts
  - I - Currents (intensity)
  - U - Synaptic potentials
  - S - Spikes
- For outputs time is averaged and max value is taken at each timestep





Outline

- ❖ Introduction
- ❖ Background
- ❖ Convolutional neural networks
- ❖ Spiking neural networks
- ❖ ***Training of spiking neural networks***
- ❖ Approach by Ledinauskas et al.
- ❖ Results and conclusions

Training SNNs

- In 2015, P.U. Diehl et al. found that convolutional neural networks (CNN) were able to be relatively easily translated to SNNs
  - Creation and training of CNN with uniform activation function
  - Mapping weights of trained CNN to equivalent SNN
  - Method works, but benefits of SNN diminish from higher training time as well as high *inference time*
- Other alternative is direct training using a *surrogate gradient*, developed by Lee et al. in 2016

Training SNNs - Surrogate Gradients

- Backpropagation relies upon differentiable functions
  - *Total accuracy is not required for use of backprop*
  - This allows for the use of a surrogate gradient which allows for direct training of SNNs
- Surrogate gradient acts as an approximation of a gradient, allowing for SNN simulated gradient descent
- A problem with this method is that for deeper SNNs, gradients tend to explode or vanish.

Outline

- Introduction
- Background
- Convolutional neural networks
- Spiking neural networks
- Training of spiking neural networks
- ***Approach by Ledinauskas et al.***
- Results and conclusions

Approach by Ledinauskas et al.

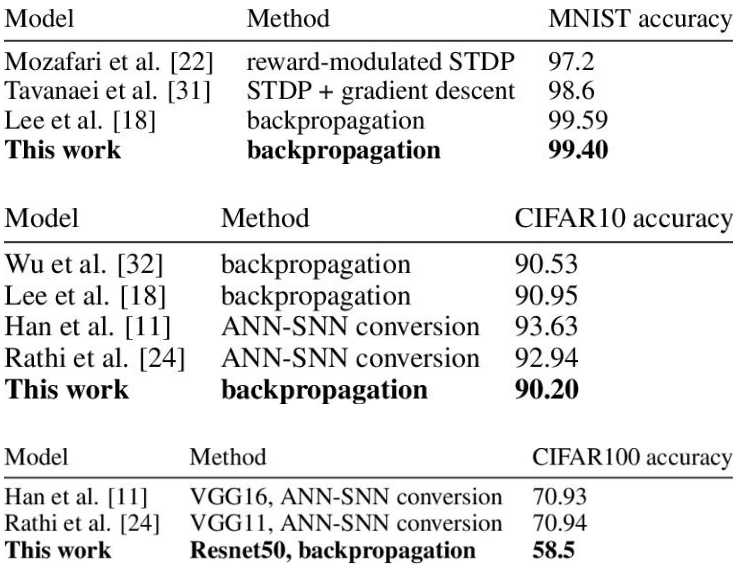
- Ledinauskas et al. described in a 2020 preprint that the problem of vanishing/exploding could be mitigated by modifications to the surrogate gradient
- By tuning hyperparameters, γ and β, where γ is the width of the surrogate gradient and β is the height of the gradient, gradient problems diminish
- They also do this via implementation of *batch normalization* for SNNs

Outline

- Introduction
- Background
- Convolutional neural networks
- Spiking neural networks
- Training of spiking neural networks
- Approach by Ledinauskas et al.
- ***Results and conclusions***

Results of tuned surrogate gradients

- Top table: performance in comparison to other NNs for MNIST
- Middle table: performance of SNNs for CIFAR-10
- Bottom table: performance of SNNs for CIFAR-100
- Result: similar performance while using significantly less inference time



Model	Method	MNIST accuracy
Mozafari et al. [22]	reward-modulated STDP	97.2
Tavanaei et al. [31]	STDP + gradient descent	98.6
Lee et al. [18]	backpropagation	99.59
This work	**backpropagation**	**99.40**

Model	Method	CIFAR10 accuracy
Wu et al. [32]	backpropagation	90.53
Lee et al. [18]	backpropagation	90.95
Han et al. [11]	ANN-SNN conversion	93.63
Rathi et al. [24]	ANN-SNN conversion	92.94
This work	**backpropagation**	**90.20**

Model	Method	CIFAR100 accuracy
Han et al. [11]	VGG16, ANN-SNN conversion	70.93
Rathi et al. [24]	VGG11, ANN-SNN conversion	70.94
This work	**Resnet50, backpropagation**	**58.5**

Conclusions

- Neuromorphic computing is the direction of research relating to developing computing methods more closely resembling biology
- Neural networks work to deliver machine-based solutions to abstract problems
- Spiking neural networks are a novel approach being studied to make low-power neural networks which perform similarly to modern methods

Suggested Topics for Further Study

- Von Neumann architecture/bottleneck
- Beyond CMOS
- Memristors
- Optical computing

Acknowledgements

This project would not have been possible without the help and feedback of Peter Dolan and Elena Machkasova as well as the continual support of my family and friends. Thank you.

Sources

- T. B. Brown, B. Mann, N. Ryder, M. Subbiah et al. *Language Models are Few-Shot Learners*. Advances in Neural Information Processing Systems 33. NeurIPS. Dec 2020
- Y. LeCun, C. Cortes, and C. J. Burges. The MNIST database.
- A. Krizhevsky, V. Nair, and G. Hinton. The CIFAR-10 dataset.
- S. Rajan. *An introduction to artificial neural networks*, 2020. Towards Data Science
- S. Saha. *A comprehensive guide to convolutional neural networks - the eli5 way*, 2018. Towards Data Science
- P. U. Diehl, D. Neil, J. Binas, M. Cook, S. Liu, and M. Pfeiffer. *Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing*. In 2015 International Joint Conference on Neural Networks (IJCNN), pages 1–8, 2015.
- E. Ledinauskas, J. Ruseckas, A. Jurˇ s˙ enas, and G. Buraˇ cas. *Training deep spiking neural networks*, 2020. Arxiv. Preprint.
- Lee, J. H., Delbruck, T., and Pfeiffer, M. (2016). *Training deep spiking neural networks using backpropagation*. Front. Neurosci. 10:508.
- M. Pfeiffer and T. Pfeil. *Deep learning with spiking neurons: Opportunities and challenges*. Frontiers in Neuroscience, 12:774, 2018.

Image Sources

- https://commons.wikimedia.org/wiki/File:Colored_neural_network.svg
- https://ekababisong.org/ieee-ompi-workshop/deep_learning/
- https://towardsdatascience.com/gradient-descent-vs-neuroevolution-f907dace010f [edited]
- https://commons.wikimedia.org/wiki/File:MnistExamples.png
- https://www.mdpi.com/1996-1944/12/17/2745