Physics-Informed Neural Networks

Ljubljana, March 2023

Author: Simon Perovnik
Mentor: doc. dr. Simon Čopar

What is this seminar about?

- What is a neural network?
    - Biological analogy
    - Artificial neural networks
- Idea behind physics-informed neural networks
- Applications (Burger’s equation & Allen-Cahn equation)
- Advantages and limitations

What is this seminar about?

- What is a neural network?
  - Biological analogy
  - Artificial neural networks
- Idea behind physics-informed neural networks
- Applications (Burger’s equation & Allen-Cahn equation)
- Advantages and limitations

Physics informed neural networks are machine learning algorithms that provide a powerful and promising numerical method for solving (partial) differential equations.

What is a neural network (NN) ?

What is a neural network (NN) ?

1. What are neurons?
2. How do neurons link together to form a network?

1. What are neurons?

- Excitable and can send pulses
- Synapses

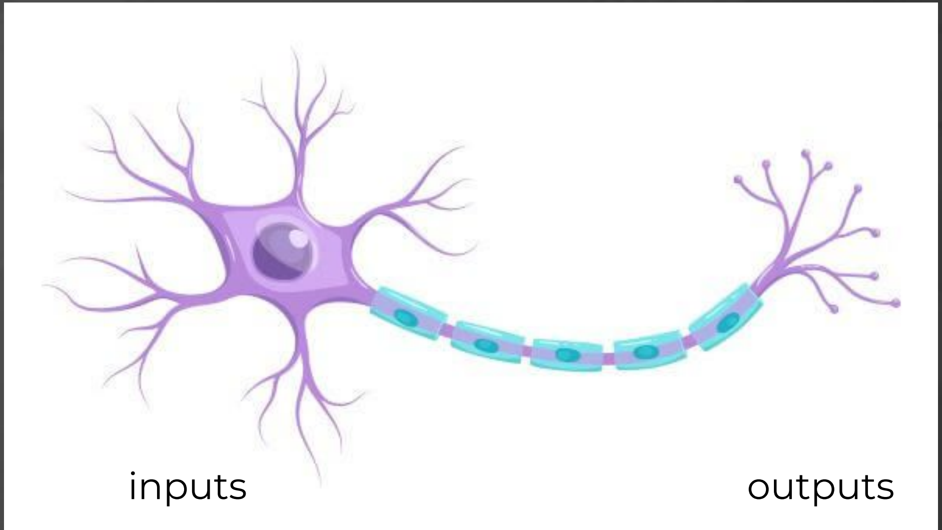


Image source : [10]

Artificial neuron

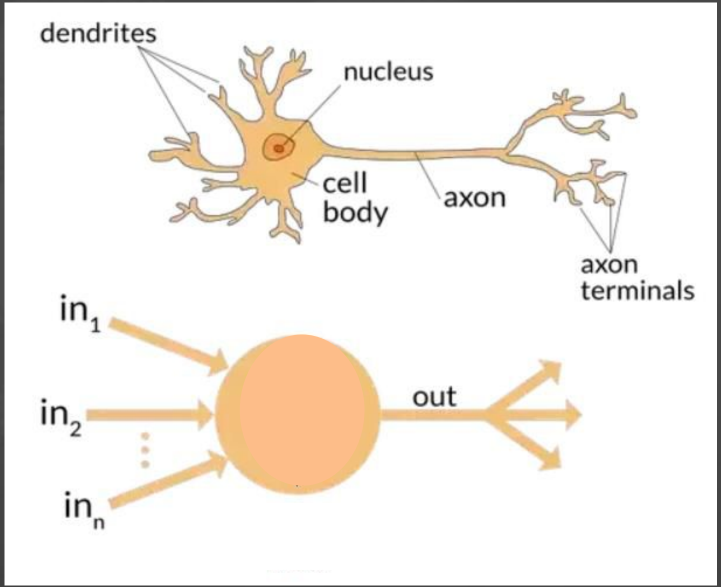


Image source : [9]

2. How do neurons link together to form a network?

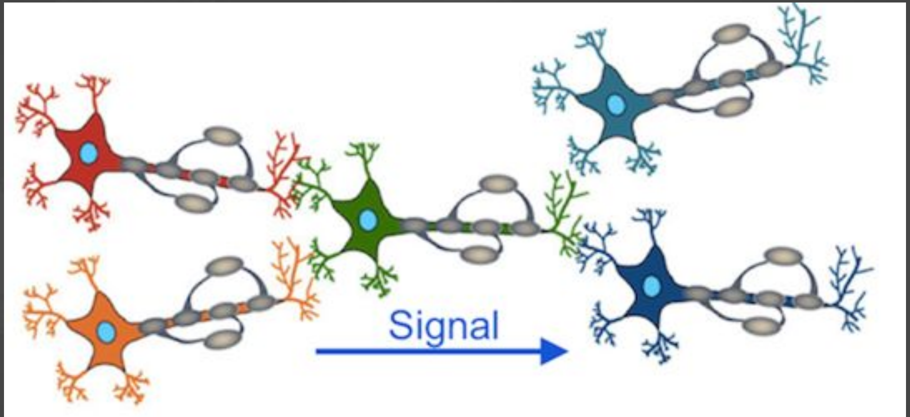


Image source : [10]

Artificial neural network

- Layered
- Feedforward
- Dense

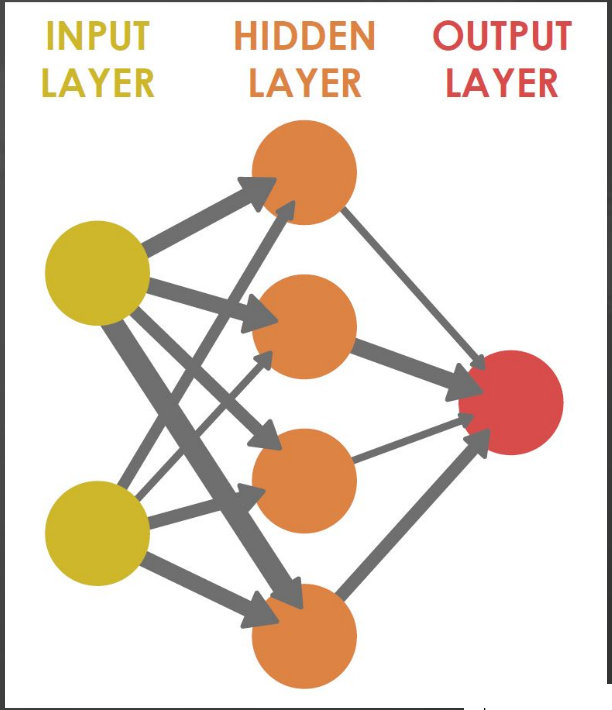


Image source : [11]

Artificial neural network

- Layered
- Feedforward
- Dense

Not yet a construction for machine learning algorithms!

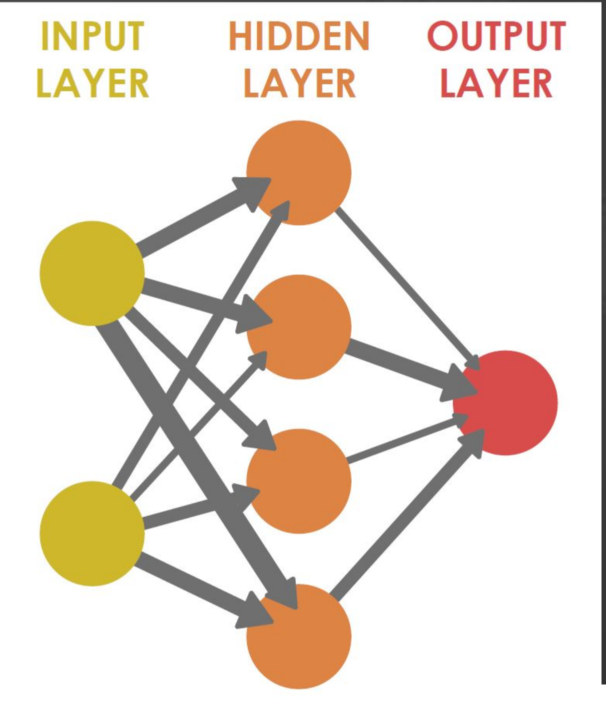


Image source : [11]

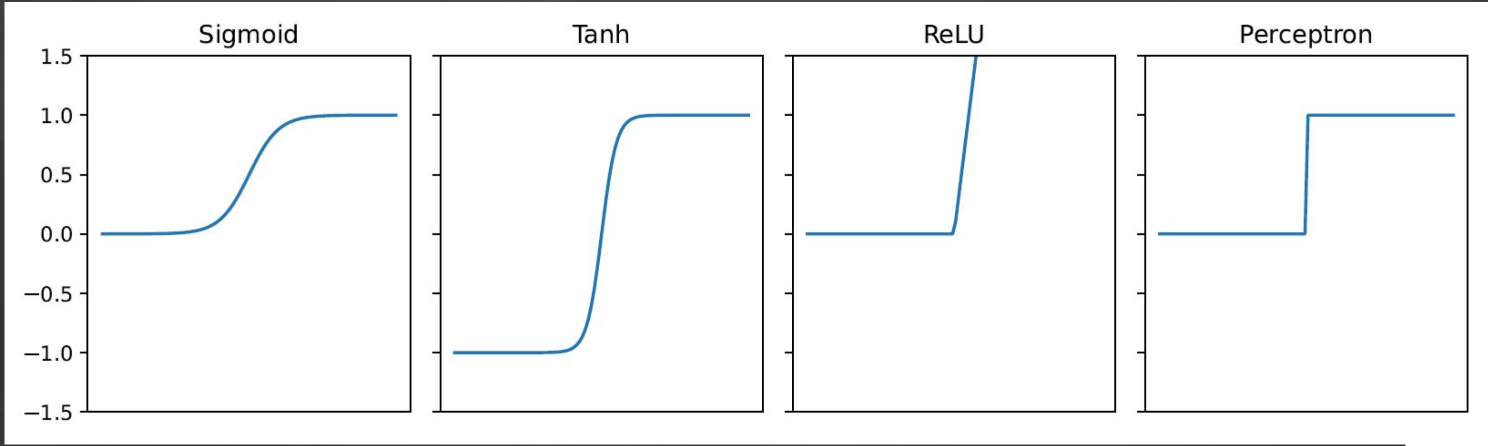
1. Activation function - σ

- Biological neurons are binary
- Universal approximation theorem ➡ We can model any function with a large enough composition of non-linear functions.
    - Non-linear
- Continuous
- Differentiable

1. Activation function - σ

- Biological neurons are binary
- Universal approximation theorem
  - Non-linear
- Continuous
- Differentiable

We can model any function with a large enough composition of non-linear functions.



source : [11]

2. Weights and biases

$$y = \quad x_1 w_{1,1} + x_2 w_{1,2} + b_1$$

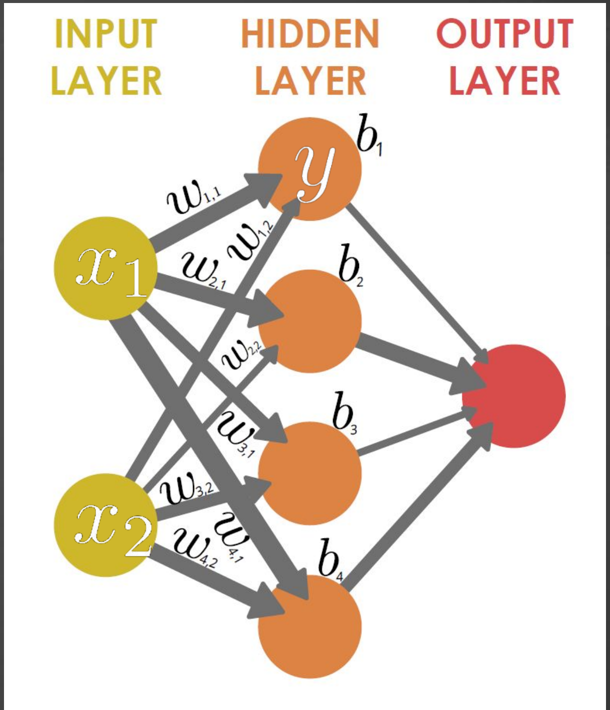


Image source : [11]

2. Weights and biases

$$y = \sigma\left(x_1 w_{1,1} + x_2 w_{1,2} + b_1\right)$$

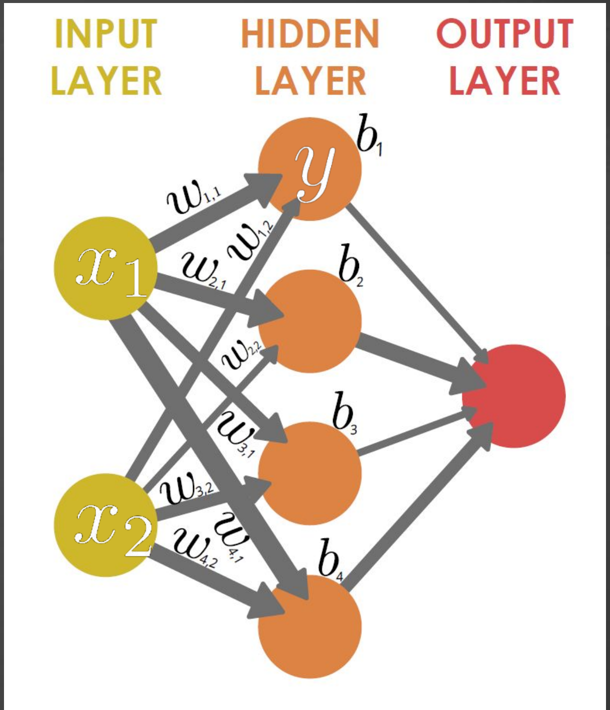


Image source : [11]

2. Weights and biases

$$y = \sigma\left(x_1 w_{1,1} + x_2 w_{1,2} + b_1\right)$$

$$y_i = \sigma\left(\sum_{j=1}^{N} x_j w_j + b_i\right)$$

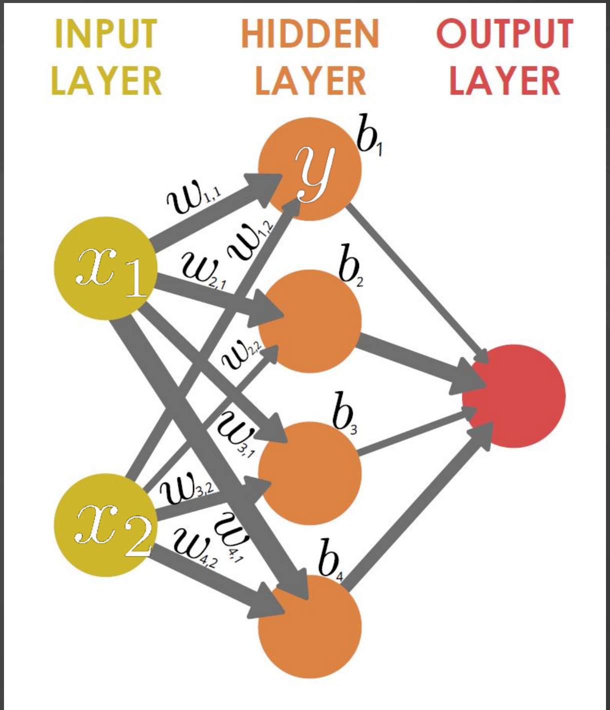


Image source : [11]

2. Weights and biases

$$y_i = \sigma\left(\sum_{j=1}^{N} x_j w_j + b_i\right)$$

$$\mathbf{y}^{(i)} = \sigma\left(\mathbf{W}\mathbf{y}^{(i-1)} + \mathbf{b}^{(i)}\right)$$

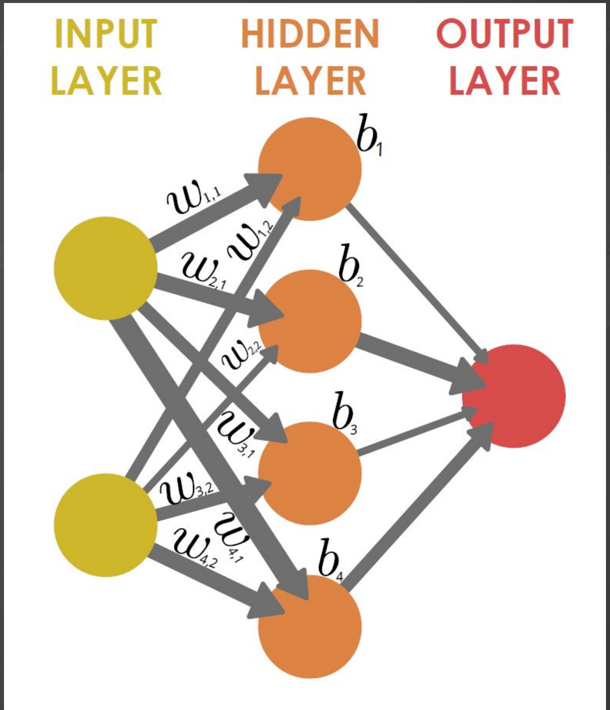


Image source : [11]

NN is a mathematical model! More than an "electrical circuit".

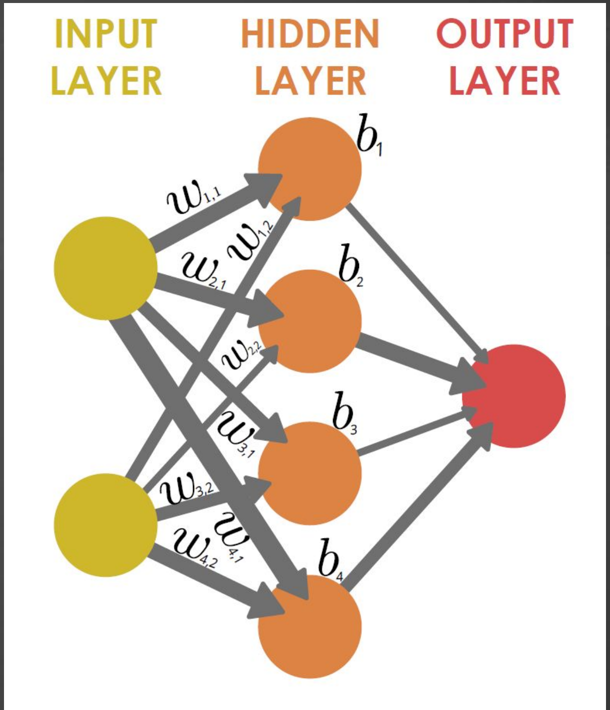


Image source : [11]

NN is a mathematical model! More than an "electrical circuit".

NN is a universal function!

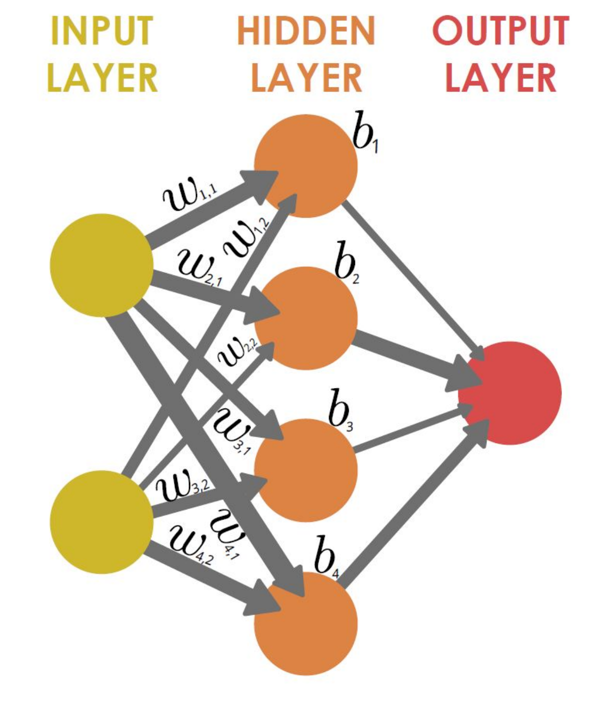
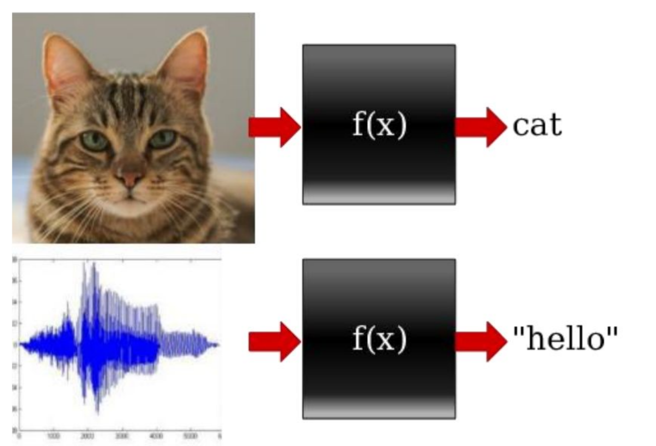


Image source : [11]

NN is a mathematical model! More than an “electrical circuit”.

NN is a universal function!



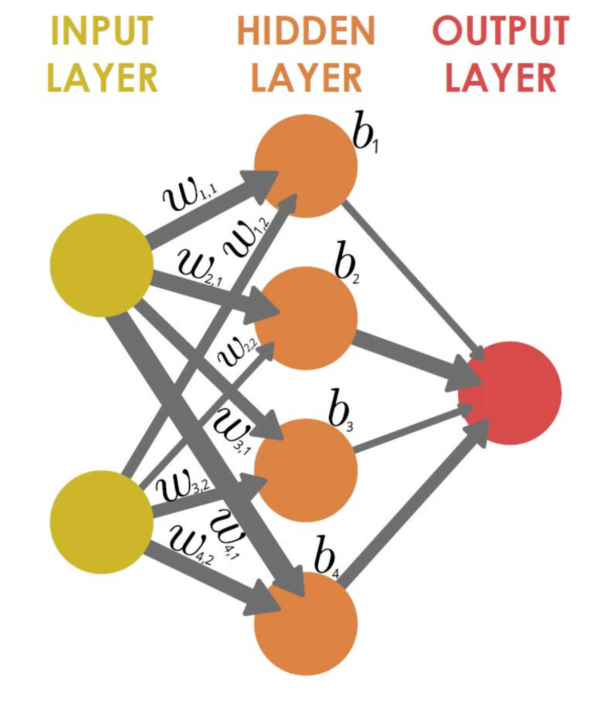


Image source : [9], [11]

How to model with NN?

(How to set the weights and biases?)

Training a neural network

- Training - In the same way that a person's abilities can be improved through repetition and feedback, the weights of a neural network are improved through exposure to many examples and adjusting based on the error in the network's predictions.

Training a neural network

- Training - In the same way that a person's abilities can be improved through repetition and feedback, the weights of a neural network are improved through exposure to many examples and adjusting based on the error in the network's predictions.

Training a neural network

- Training - In the same way that a person's abilities can be improved through repetition and feedback, the weights of a neural network are improved through exposure to many examples and adjusting based on the error in the network's predictions.
- Labeled data

Image source : [11]

Training a neural network

- Training - In the same way that a person's abilities can be improved through repetition and feedback, the weights of a neural network are improved through exposure to many examples and adjusting based on the error in the network's predictions.
- Labeled data

Image source : [11]

Training a neural network

- Training - In the same way that a person's abilities can be improved through repetition and feedback, the weights of a neural network are improved through exposure to many examples and adjusting based on the error in the network's predictions.
- Labeled data
- Loss function

$$\mathcal{L}(\mathbf{W}) = \sum_{i=0}^{N_L} (y_i - \hat{y}_i)^2$$

prediction label

label

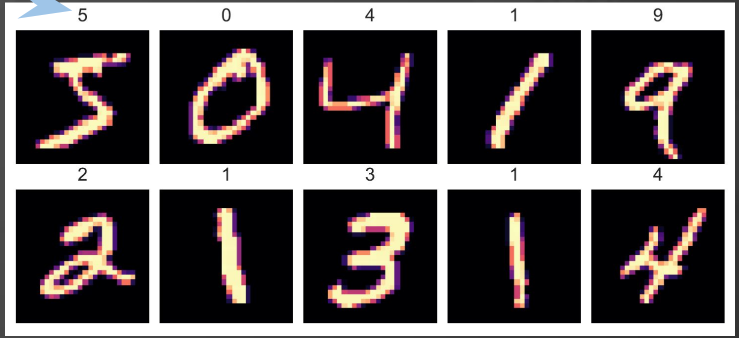


Image source : [11]

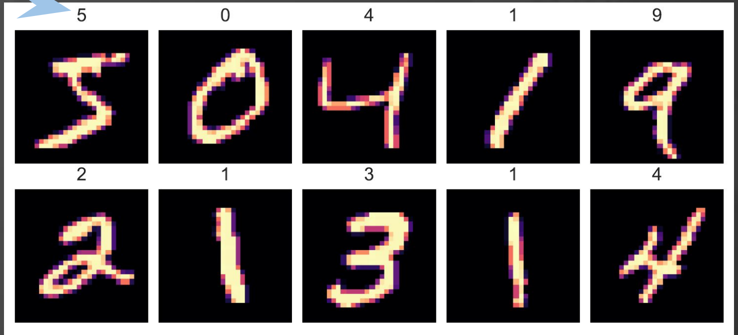
Training a neural network

- Training - In the same way that a person's abilities can be improved through repetition and feedback, the weights of a neural network are improved through exposure to many examples and adjusting based on the error in the network's predictions.
- Labeled data
- Loss function

$$\mathcal{L}(\mathbf{W}) = \sum_{i=0}^{N_L} (y_i - \hat{y}_i)^2$$

prediction label

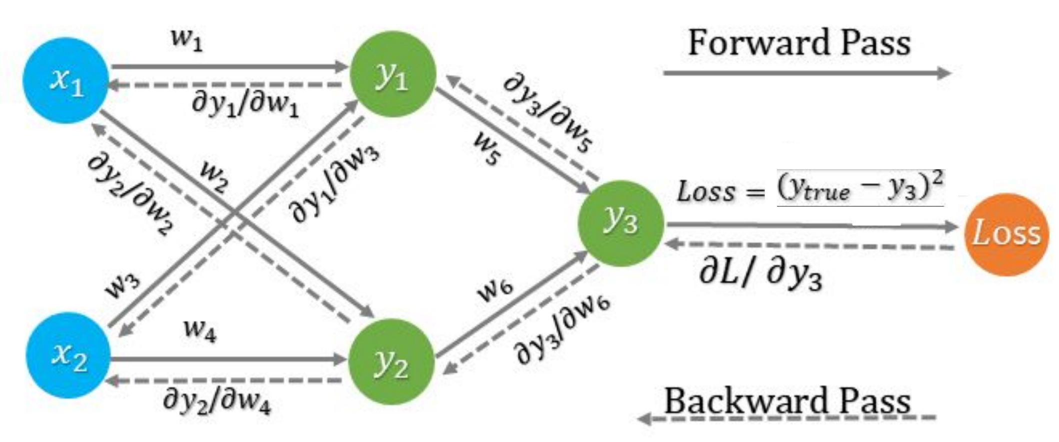
label



The goal of training is to find the set of weights and biases that result in the lowest error, or best fit, between the network's predictions and the true (labeled) values.

Training a neural network - Backpropagation

- Backpropagation - propagating error from output layer to the input layer to calculate how to change internal parameters so that we will reduce loss function the most.
    - Forward pass + backward pass



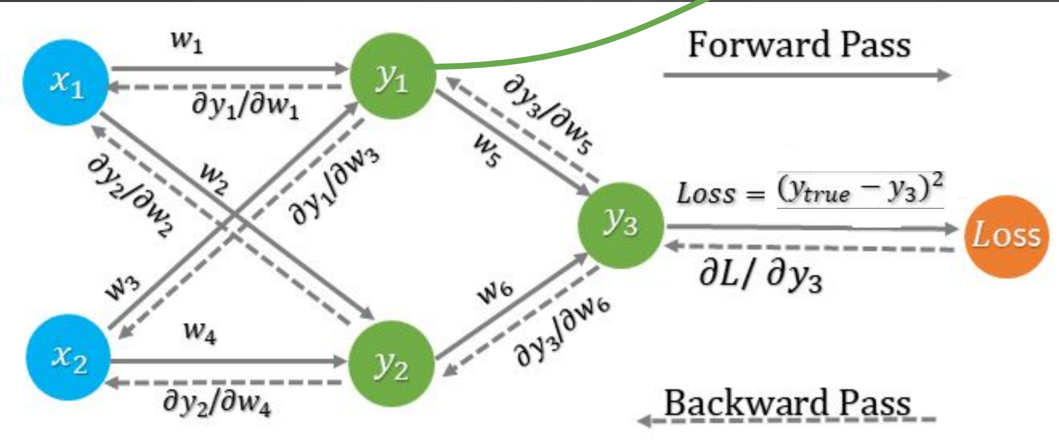
$$\Delta \mathbf{W} \propto -\nabla_{\mathbf{W}} \mathcal{L}(\mathbf{W})$$

Image source : [10]

Training a neural network - Backpropagation

- Backpropagation - propagating error from output layer to the input layer to calculate how to change internal parameters so that we will reduce loss function the most.
    - Forward pass + backward pass

$$y_1 = \sigma\left(w_1 x_1 + w_3 x_2 + b_1\right)$$



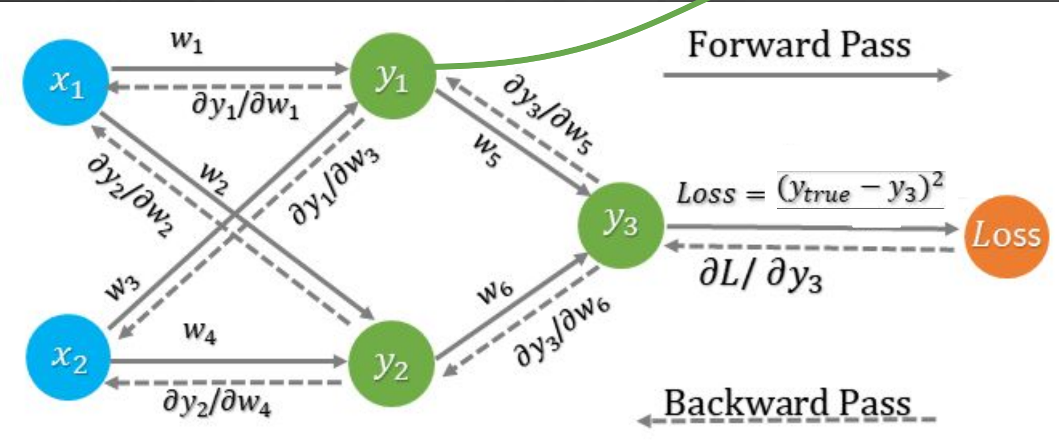
$$\Delta \mathbf{W} \propto -\nabla_{\mathbf{W}} \mathcal{L}(\mathbf{W})$$

Image source : [10]

Training a neural network - Backpropagation

- Backpropagation - propagating error from output layer to the input layer to calculate how to change internal parameters so that we will reduce loss function the most.
  - Forward pass + backward pass

$$y_1 = \sigma\left(w_1 x_1 + w_3 x_2 + b_1\right)$$

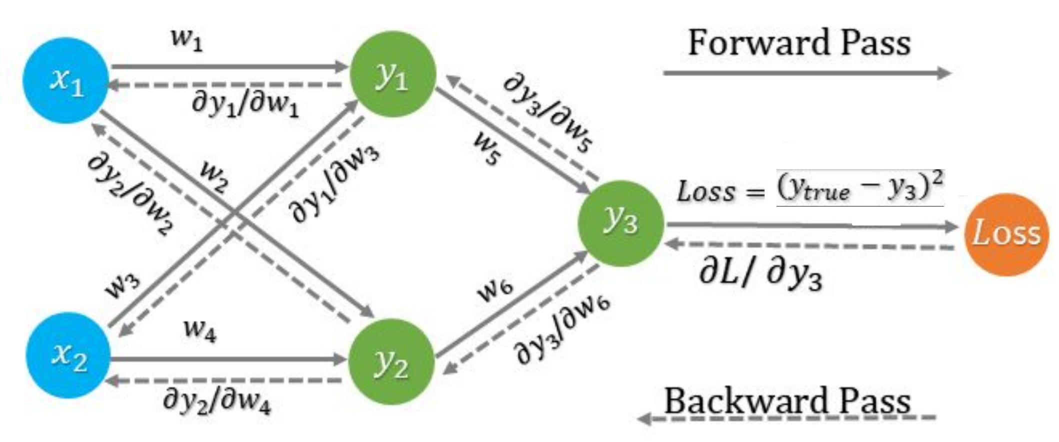


$$\Delta \mathbf{W} \propto -\nabla_{\mathbf{W}} \mathcal{L}(\mathbf{W})$$

Image source : [10]

Backward pass

- Automatic differentiation - method of calculating derivatives

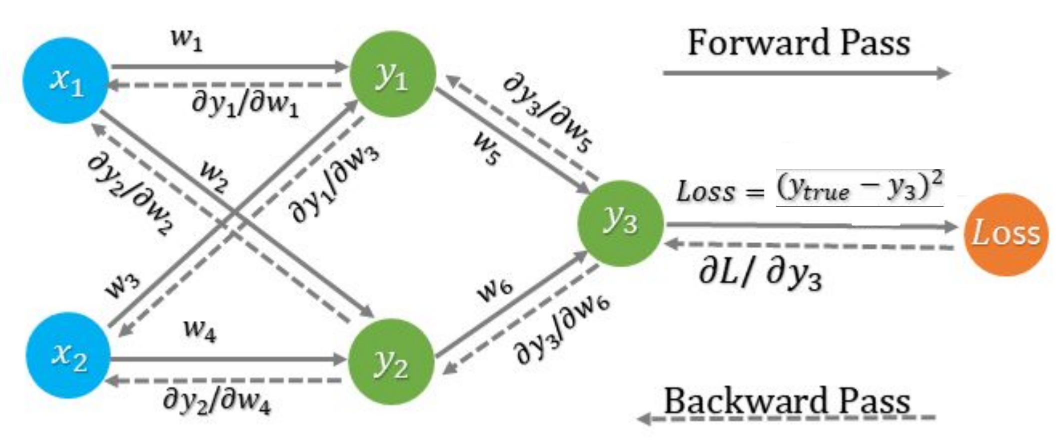


$$\frac{\partial L}{\partial w_5} = \frac{\partial L}{\partial y_3}\frac{\partial y_3}{\partial w_5}$$

Image source : [10]

Backward pass

- Automatic differentiation - method of calculating derivatives



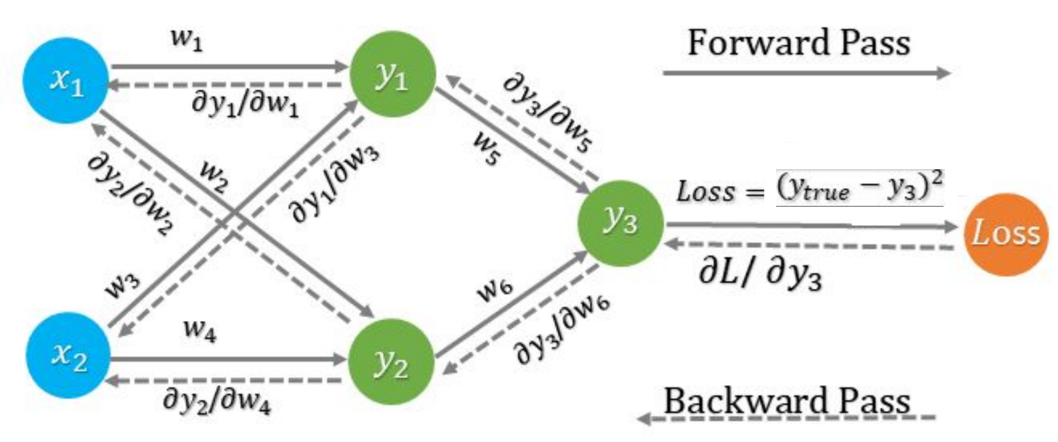
$$\frac{\partial L}{\partial w_5} = \frac{\partial L}{\partial y_3}\frac{\partial y_3}{\partial w_5}$$

$$\frac{\partial L}{\partial y_3} = -2(y_{true} - y_3)$$

Image source : [10]

Backward pass

- Automatic differentiation - method of calculating derivatives



$$\frac{\partial L}{\partial w_5} = \frac{\partial L}{\partial y_3}\frac{\partial y_3}{\partial w_5}$$

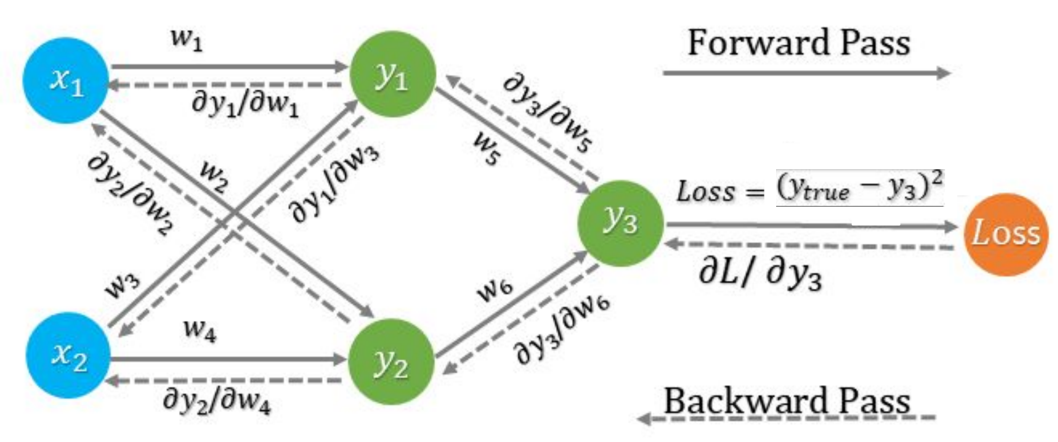
$$\frac{\partial L}{\partial y_3} = -2(y_{true} - y_3)$$

$$\frac{\partial y_3}{\partial w_5} = \sigma'(y_1 w_5 + y_2 w_6 + b_3) \cdot y_1$$

Image source : [10]

Backward pass

- Automatic differentiation - method of calculating derivatives



$$\frac{\partial L}{\partial w_5} = \frac{\partial L}{\partial y_3}\frac{\partial y_3}{\partial w_5}$$

$$\frac{\partial L}{\partial y_3} = -2(y_{true} - y_3)$$

$$\boxed{\frac{\partial y_3}{\partial x_1} = \frac{\partial y_3}{\partial y_1}\frac{\partial y_1}{\partial x_1}}$$

$$\frac{\partial y_3}{\partial w_5} = \sigma'(y_1 w_5 + y_2 w_6 + b_3) \cdot y_1$$

Image source : [10]

Quick recap

- NNs are machine learning algorithms that receive an input x and generate y according to currently set weights.
- Modeling by setting appropriate weights
- Loss function -> error propagation -> weights correction
- Automatic differentiation allows for quick and simple evaluation of derivatives

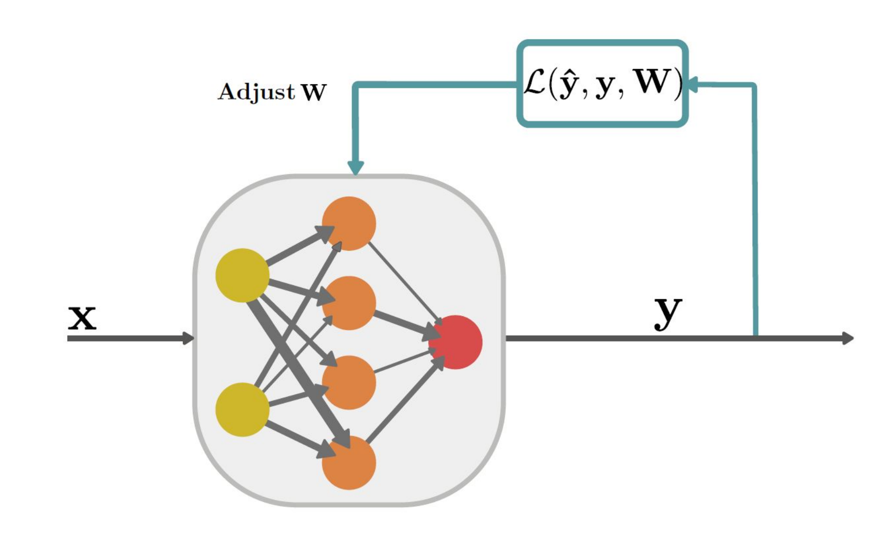


Image source : [11]

Physics-informed NN

Motivation

- Physical laws often in form of (P)DE
- In principle solvable with NN, but can we impose physics?

$$A\frac{\mathrm{d}^2y}{\mathrm{d}x^2} + B\frac{\mathrm{d}y}{\mathrm{d}x} + Cy = 0$$

$$y(0) = y_0, \qquad y(1) = y_1$$

Modifying the loss function

$$\mathcal{L} = \mathcal{L}_d$$

$$\mathcal{L}_d(\mathbf{W}) = (y - \hat{y})^2$$

$$A\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} + B\frac{\mathrm{d}y}{\mathrm{d}x} + Cy = 0$$

$$y(0) = y_0, \qquad y(1) = y_1$$

Modifying the loss function

$$\mathcal{L} = \mathcal{L}_d + \mathcal{L}_r$$

$$A\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} + B\frac{\mathrm{d}y}{\mathrm{d}x} + Cy = 0$$

$$y(0) = y_0, \qquad y(1) = y_1$$

$$\mathcal{L}_d(\mathbf{W}) = (y - \hat{y})^2$$

$$\boxed{\frac{\partial y_3}{\partial x_1} = \frac{\partial y_3}{\partial y_1}\frac{\partial y_1}{\partial x_1}}$$

$$\mathcal{L}_r(\mathbf{W}) = \left|A\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} + B\frac{\mathrm{d}y}{\mathrm{d}x} + Cy\right|^2$$

Modifying the loss function

$$\mathcal{L} = \mathcal{L}_d + \mathcal{L}_r$$

$$A\frac{\mathrm{d}^2y}{\mathrm{d}x^2} + B\frac{\mathrm{d}y}{\mathrm{d}x} + Cy = 0$$

$$y(0) = y_0, \qquad y(1) = y_1$$

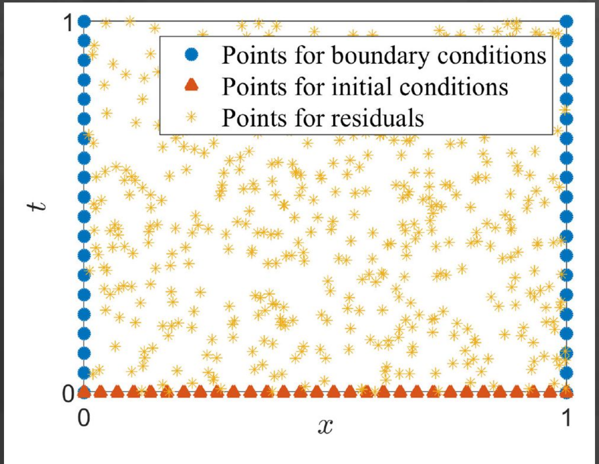
$$\mathcal{L}_d(\mathbf{W}) = (y - \hat{y})^2$$

$$\mathcal{L}_r(\mathbf{W}) = \left| A\frac{\mathrm{d}^2y}{\mathrm{d}x^2} + B\frac{\mathrm{d}y}{\mathrm{d}x} + Cy \right|^2$$

Lagrangian would be even better!

Modifying the loss function

$$\mathcal{L} = \mathcal{L}_d + \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$$



$$\mathcal{L}_b(\mathbf{W}) = (y^b - \hat{y}^b)^2$$

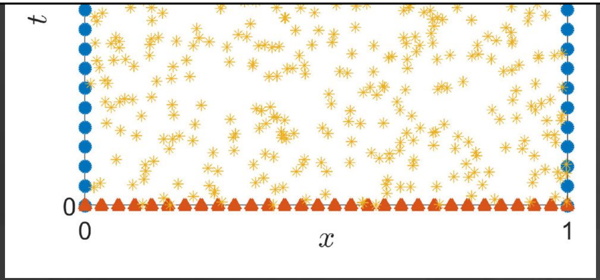
$$\mathcal{L}_0(\mathbf{W}) = (y^0 - \hat{y}^0)^2$$

Image source : [1]

Modifying the loss function

$$\mathcal{L} = \cancel{\mathcal{L}_d} + \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$$

We can train without data points!



$$\mathcal{L}_b(\mathbf{W}) = (y^b - \hat{y}^b)^2$$

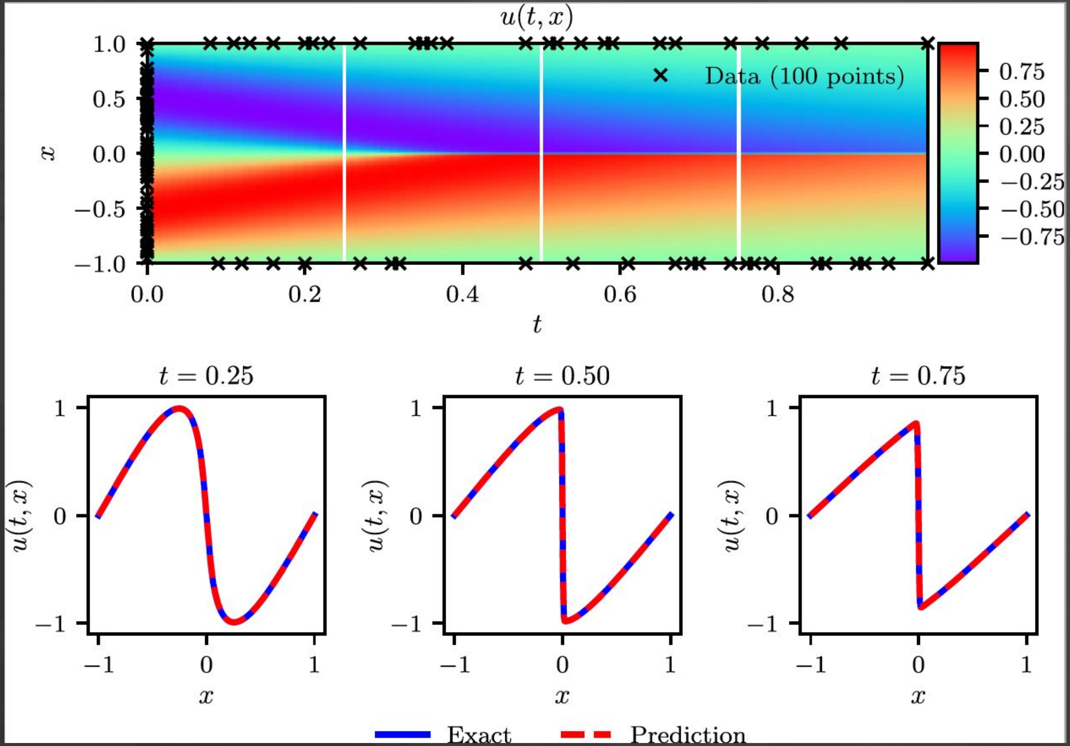
$$\mathcal{L}_0(\mathbf{W}) = (y^0 - \hat{y}^0)^2$$

Image source : [1]

Applications

$$u_t + uu_x = (0.01/\pi)u_{xx},\ x \in [-1,1],\ \ t \in [0,1],$$
$$u(0,x) = -\sin(\pi x),$$
$$u(t,-1) = u(t,1) = 0$$

1. Burger's equation

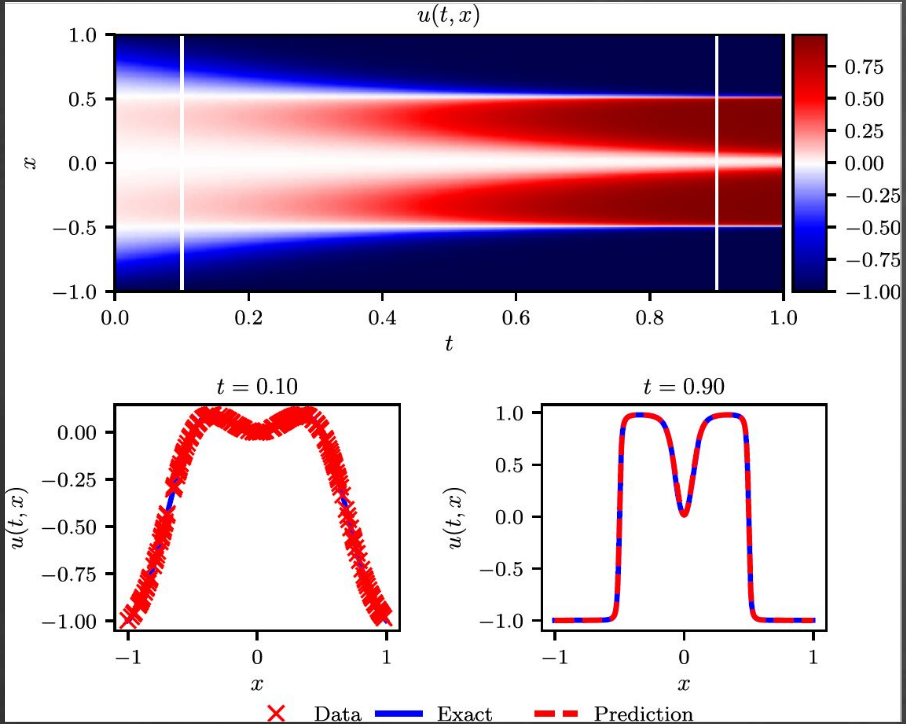


- NN with 7 hidden layers, 20 neurons per layer, tanh as activation function
- No experimental data, no $\mathcal{L}_d$

Image source : [2]

$$u_t - 0.0001u_{xx} + 5u^3 - 5u = 0, \;\; x \in [-1,1], \;\; t \in [0,1],$$
$$u(0,x) = x^2 \cos(\pi x),$$
$$u(t,-1) = u(t,1), \quad u_x(t,-1) = u_x(t,1),$$

1. Allen-Cahn equation



- NN with 4 hidden layers, 200 neurons per layer
- Input - x(t), output - function evaluations according to Runge-Kutta integration

Image source : [2]

Limitations and advantages

Limitations

- Interpretability problems, too much like a "black-box"
- Quantifying the uncertainty of results
- Searching for optimal architecture
- Slower and less accurate then finite differences methods

Advantages

- Combining experimental data with modeling
- Once trained, very fast to obtain solution in arbitrary point in space
- Complex geometries where discretisation is not trivial
- Solving inverse problems

Conclusion

Physics informed neural networks are machine learning algorithms that provide a powerful and promising numerical method for solving (partial) differential equations.

References

[1] Shengze Cai, Zhicheng Wang, Sifan Wang, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 143(6), 2021.

[2] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[3] Kevin Gurney. *An introduction to neural networks*. CRC press, 2018.

[4] Wikipedia: Neural network (https://en.wikipedia.org/wiki/neural_network), last accessed on 5.2. 2023.

[5] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018.

[6] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

[7] Tamara G Grossmann, Urszula Julia Komorowska, Jonas Latz, and Carola-Bibiane Schönlieb. Can physics-informed neural networks beat the finite element method? *arXiv preprint arXiv:2302.04107*, 2023.

[8] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM review*, 63(1):208–228, 2021.

[9] Simon Čopar. Nevronske mreže - psuf, 2023.

[10] Yubiao Sun, Qiankun Sun, and Kan Qin. Physics-based deep learning for flow problems. *Energies*, 14(22):7760, 2021.

[11] Pictures produced by the author