JUMPSTART GUIDE TO DEEP LEARNING IN AUDIO FOR ABSOLUTE BEGINNERS:

FROM NO EXPERIENCE AND NO DATASETS TO A DEPLOYED MODEL

JAN WILCZEK

Outline

1. Motivation for the talk
2. 4 myths about deep learning
3. Your learning path
4. Introduction to deep learning terminology
5. Live demo

Slides & code:
github.com/JanWilczek/adc22

Who am I?

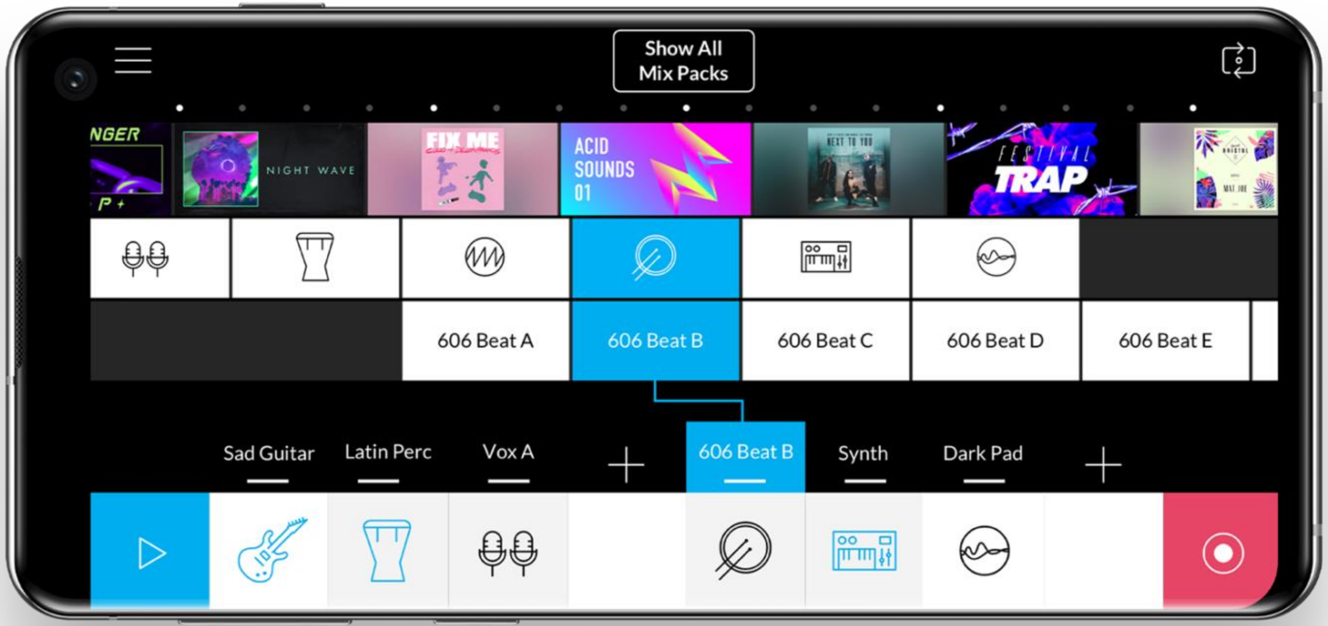
- Founder of an educational and consulting company in the field of audio programming
- Blog: thewolfsound.com
- YouTube channel: youtube.com/c/WolfSoundAudio

WolfSound

Become an Audio Programmer

Loudly

Music Maker JAM



Loudly

SOUND TRACKS

Proceedings of the 25th International Conference on Digital Audio Effects (DAFx20in22), Vienna, Austria, September 6-10, 2022

VIRTUAL ANALOG MODELING OF DISTORTION CIRCUITS USING NEURAL ORDINARY DIFFERENTIAL EQUATIONS

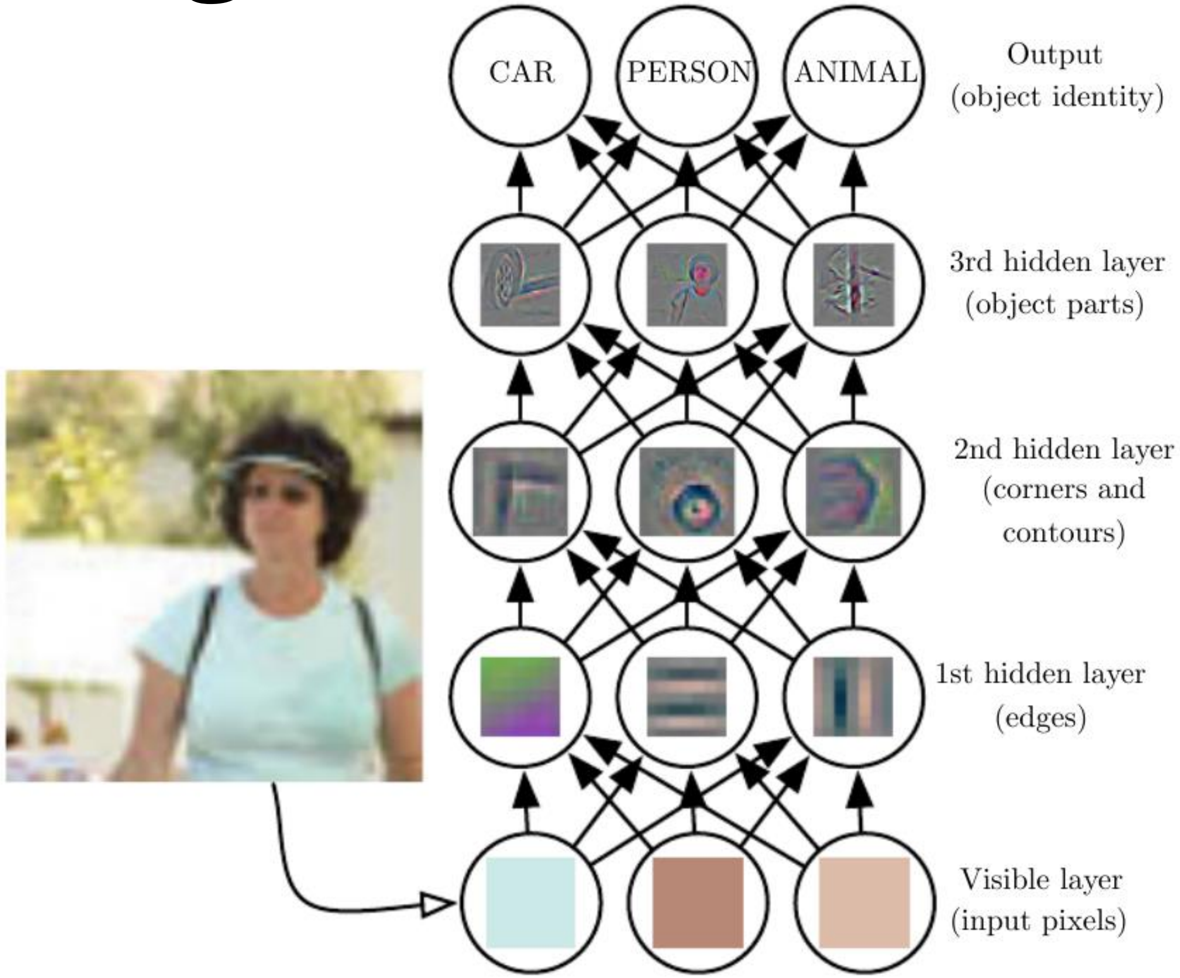
Jan Wilczek *
WolfSound
Katowice, Poland
jan.wilczek@thewolfsound.com

Alec Wright and Vesa Välimäki †
Acoustics Lab
Dept. Signal Processing and Acoustics
Aalto University, Espoo, Finland
alec.wright@aalto.fi

Emanuël A. P. Habets
International Audio Laboratories Erlangen ‡
Erlangen, Germany
emanuel.habets
@audiolabs-erlangen.de

What is deep learning?

- A subfield of machine learning.
- Each concept consists of simpler concepts.
- A „deep” hierarchy of concepts.



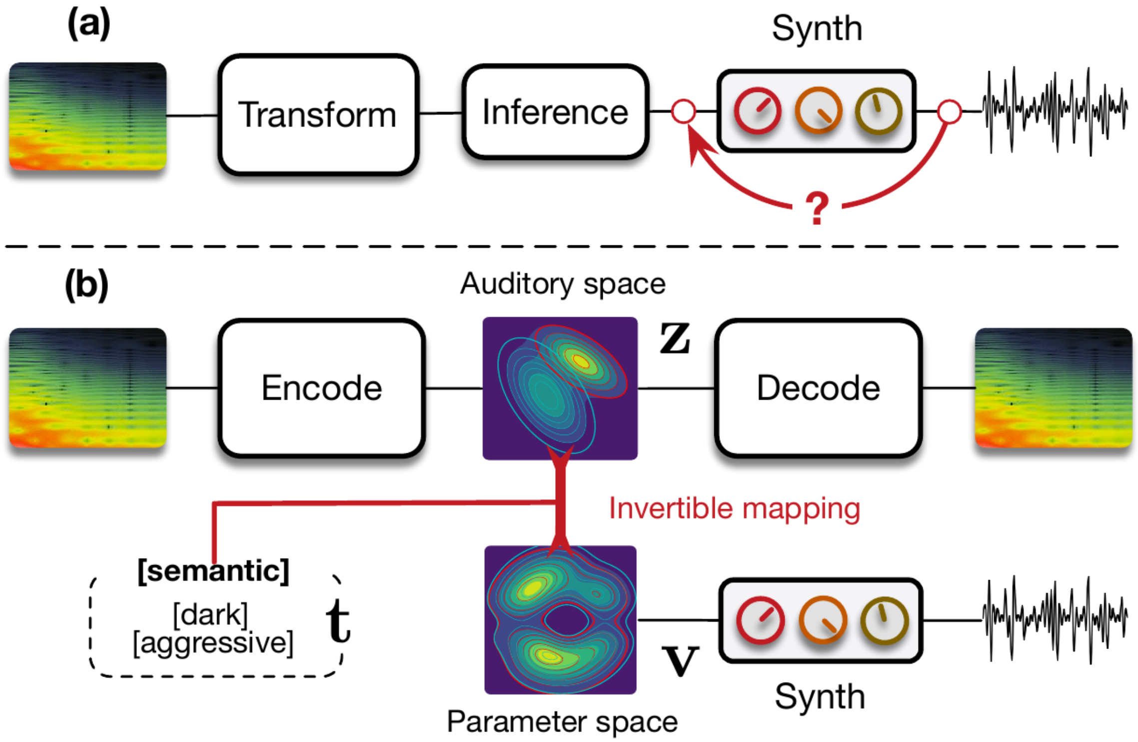
I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*, deeplearning.org, access: 6.11.2022

Why would you want to use deep learning?

- „Everyone does it”
- Sounds cool
- Something to brag about
- …
- Allows achieving much better results than non-deep learning methods…
- …while not being so complicated ☺

Where is deep learning used in audio?

- Virtual Analog modeling
- Automatic speech recognition (ASR)
- Speech synthesis
- Timbre/style transfer
- „Intelligent” plugins
- Physical modeling
- Pitch tracking
- Audio tagging
- Sound synthesis
- And much, much more…



P. Esling et al., *Universal Audio Synthesizer Control With Normalizing Flows*, Proc. of the 22nd Int. Conf. on Digital Audio Effects (DAFx-19), Birmingham, UK, September 2–6, 2019

Why people struggle learning deep learning?

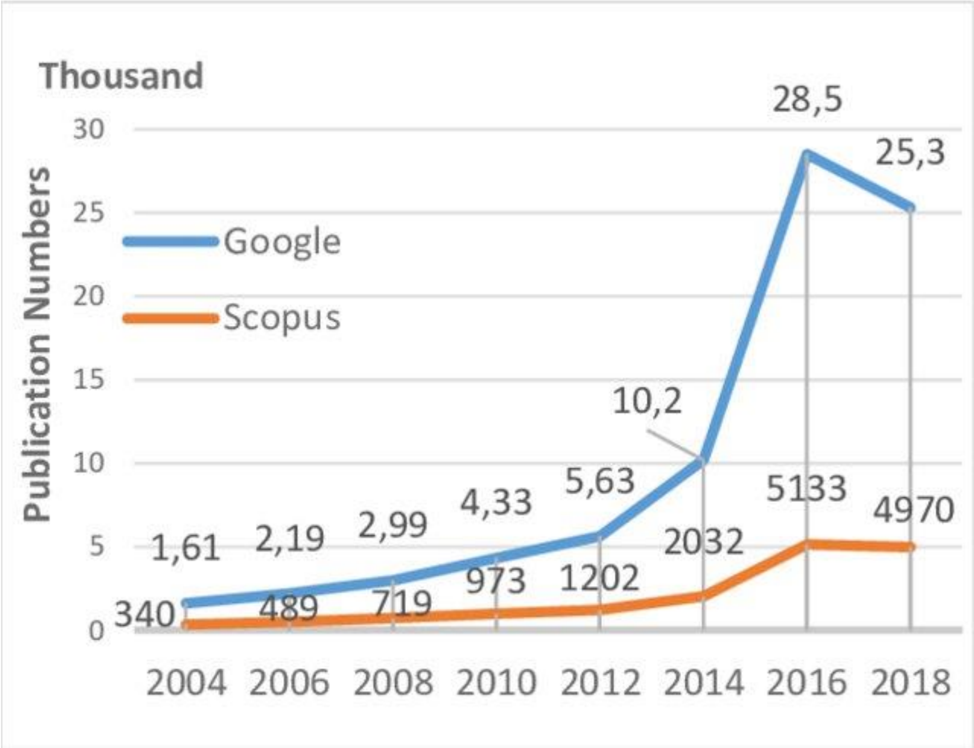
- Myths stopping them
- There seems to be something „magical" about it

Photo by Siora Photography on Unsplash

Myths Concerning Deep Learning

Myth #1: Learning deep learning is hard

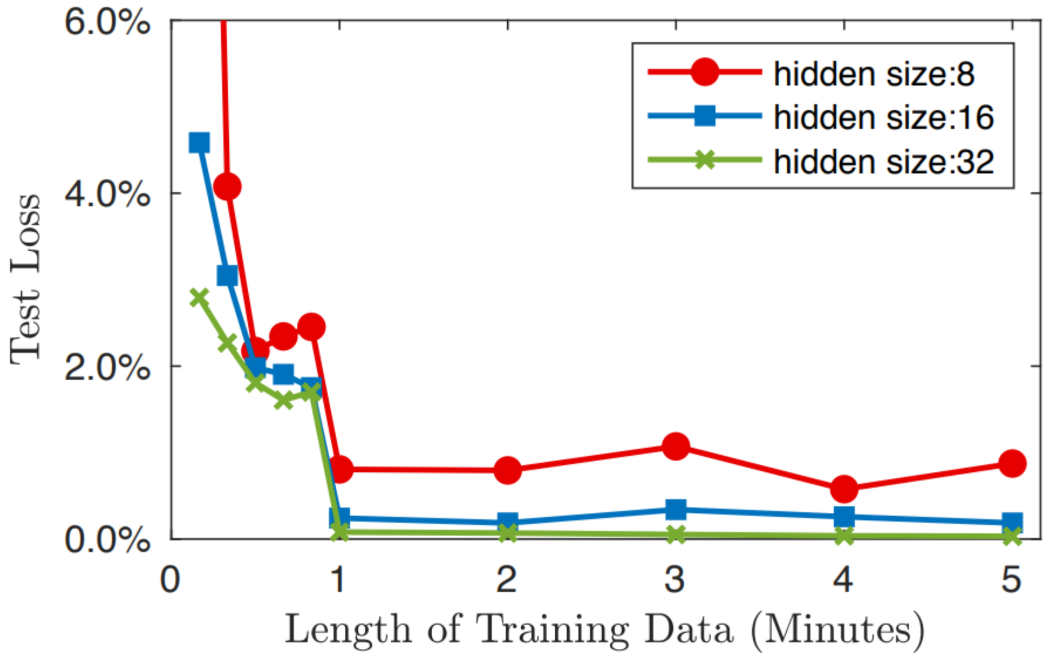
- Huge number of knowledge sources
- Great software support
- Abundance of research



M. M. Yapici, A. Tekerek, N. Topaloglu, *Literature Review of Deep Learning Research Areas*, Gazi Journal of Engineering Sciences 2019

Myth #2: You need a lot of data

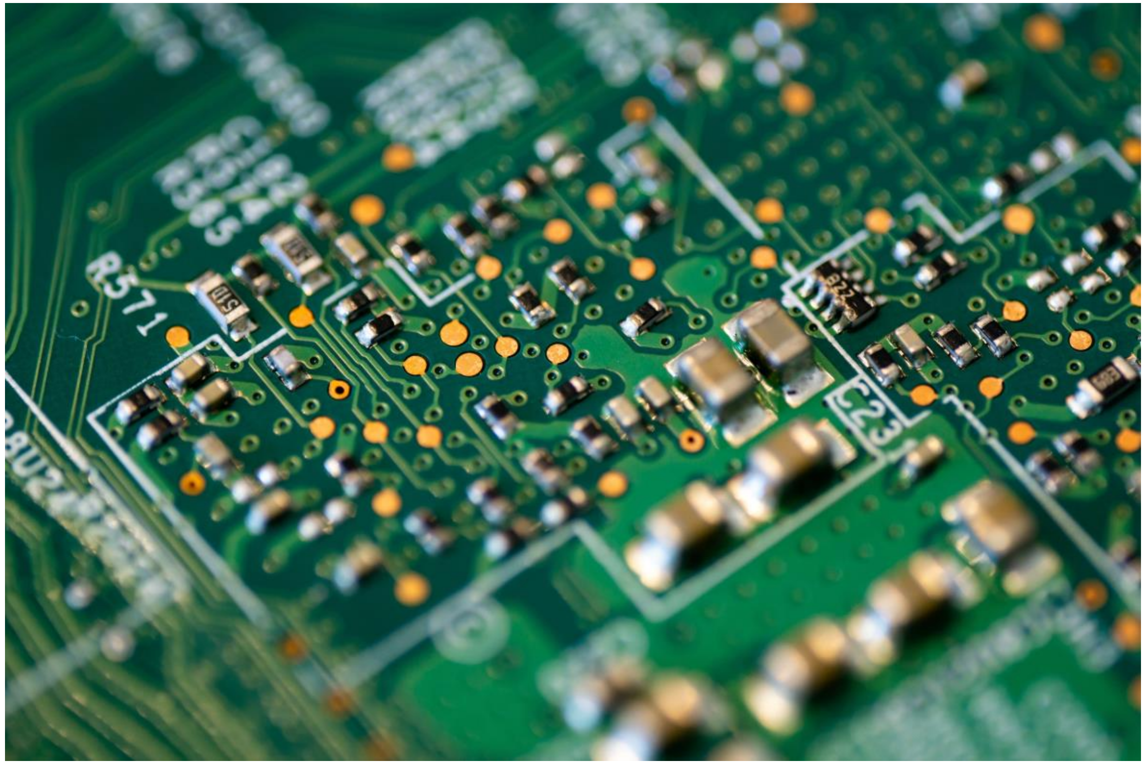
- 1 minute of audio may suffice
- Online samples or whole datasets available



A. Wright and V. Välimäki, *Neural Modelling Of Periodically Modulated Time-varying Effects*, Proc. of the 23rd Int. Conf. on Digital Audio Effects (DAFx-20), Vienna, Austria, September 2020-21

Myth #3: You need a lot of computing power

- Local GPU may suffice for simple models
- Some models may be faster to train on a CPU
- Computing clusters available:
  - Google Cloud Platform
  - Amazon AWS



Myth #4: Deep learning models cannot run in real time

- Commercial plugins available.
- Techniques for reducing the network size, e.g., prunning.
- Matrix computation libraries, e.g., Eigen.

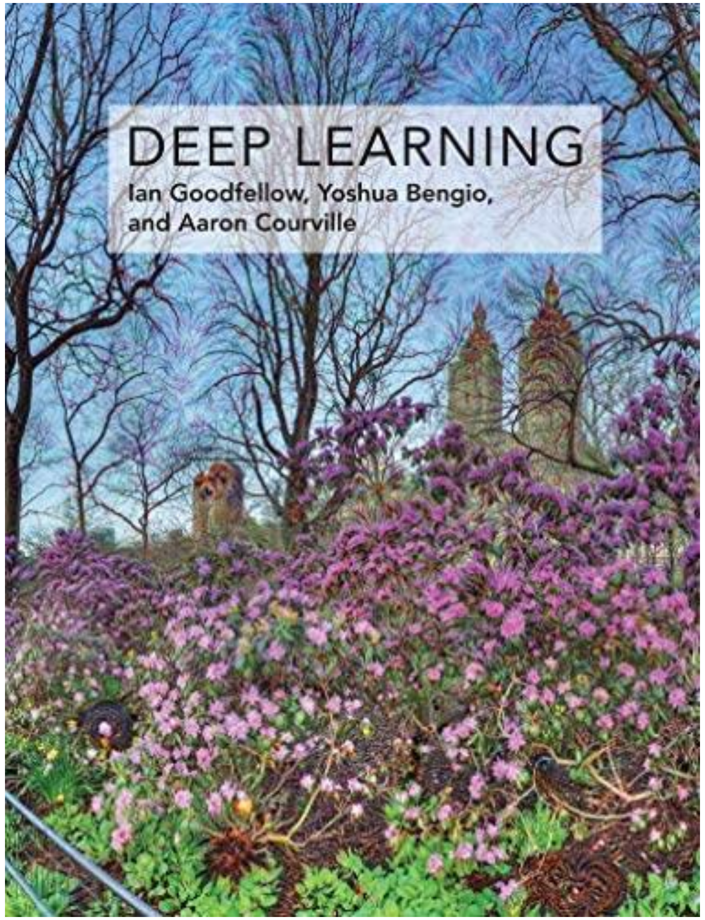
How to learn deep learning in 4 steps

Step 1: Read the free "Deep Learning Book"

- Deep Learning
  by Ian Goodfellow, Yoshua Bengio, and Aaron Courville.

  deeplearningbook.org

- free
- approachable



Step 2: Watch a free YouTube lecture on deep learning

- 2x the speed if you need to
- Make notes
- Learn the vocabulary

Alexander Maier
University of Erlangen-Nürnberg (FAU)

Alexander Amini
Massachusetts Institute of Technology (MIT)

Step 3: Read at least 3 free research papers on deep learning in audio

1. J. Engel, L. Hantrakul, C. Gu, A. Roberts, *DDSP: Differentiable Digital Signal Processing*, published as a conference paper at ICLR 2020.
2. J. D. Parker, F. Esqueda, A. Bergner, *Modelling Of Nonlinear State-space Systems Using A Deep Neural Network*, Proc. of the 22nd Int. Conf. on Digital Audio Effects (DAFx-19), Birmingham, UK, 2019.
3. A. Wright, E.-P. Damskägg, V. Välimäki, *Real-Time Black-Box Modelling with Recurrent Neural Networks*, Proc. of the 22nd Int. Conf. on Digital Audio Effects (DAFx-19), Birmingham, UK, 2019.

Step 4: Train your first model using free tools

- Python
- PyTorch
- Visual Studio Code
- Local CPU or GPU
- Architecture and dataset from GitHub

What do all 4 steps have in common?

(they are free)

Basic Concepts in Deep Learning

Machine Learning

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P, improves with experience E."

~ T. M. Mitchell, *Machine Learning* (1997)

Deep Learning

- A subfield of machine learning, which learns high-level concepts in their relations to low-level (simple) concepts → „deep” hierarchy of concepts.
- „Deep” also refers to the application of deep neural networks (networks with hidden layers) which can learn these concept hierarchies.

Dataset

- Ranges from a few minutes to hours of audio.
- Either raw audio or **features** extracted from it.
- Consists of **examples**: (input, target) pairs.
- Typically split into **training** set, **validation** set, and **test** set.

Sample dataset for Virtual Analog modeling

- Input signal: 10 minutes of guitar and bass recordings.
- Target signal: recorded output of the distortion circuit with the above input.
- Test set: 2 minutes of input and output.
- Training set: 6.4 minutes of the input and output.
- Validation set: 1.6 minutes of the input and output.
- Features: raw samples.
- Example: a pair of input and corresponding output sequences of length 0.5 second.
- Minibatch: 64 input + target sequences, 0.5 second each.

Neural Network

- „Universal function approximator”
- „Filter with trainable coefficients”
- A function
- An input-to-output mapping

input example

output

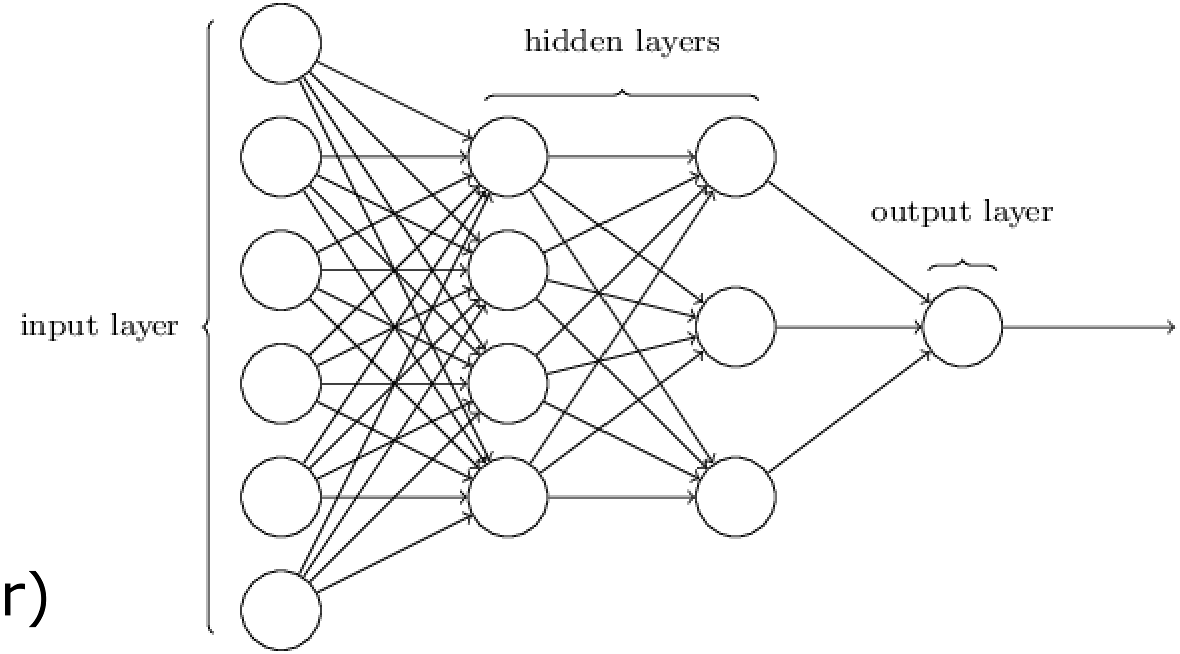
$$\hat{y} = f(x)$$

neural
network

Multilayer Perceptron (MLP)

- Feedforward network = FIR filter

$$\boldsymbol{y}^{(l)} = g(\boldsymbol{W}^{(l)}\boldsymbol{y}^{(l-1)} + \boldsymbol{b}^{(l)}),$$



$\boldsymbol{y}^{(l)}$ output of the l-th layer

g nonlinear function (hyperparameter)

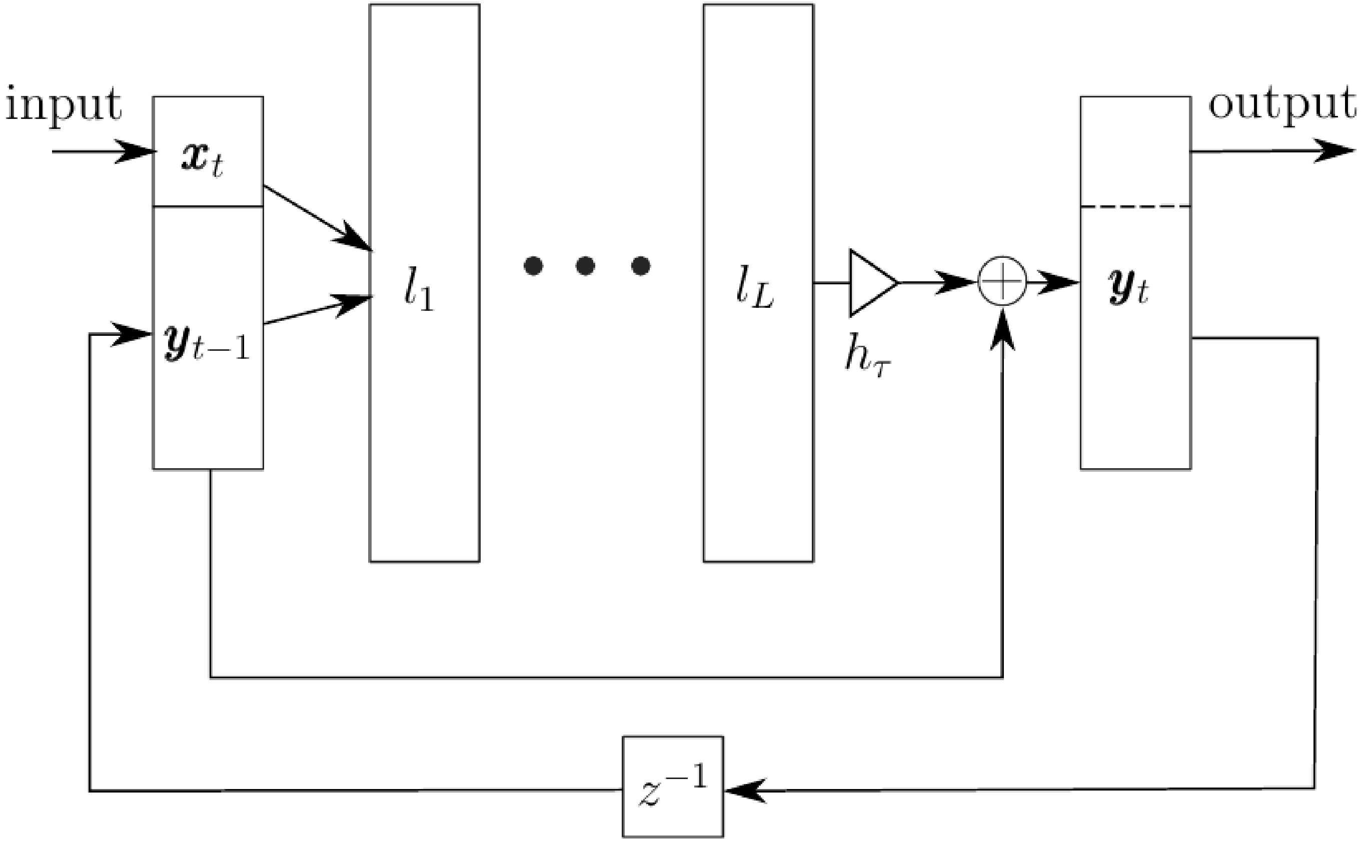
$\boldsymbol{W}^{(l)}$ matrix of weights (training parameters)

$\boldsymbol{b}^{(l)}$ vector of biases (training parameters)

M. Nielsen, *Neural Networks and Deep Learning,* neuralnetworksanddeeplearning.com

Recurrent Neural Network (RNN)

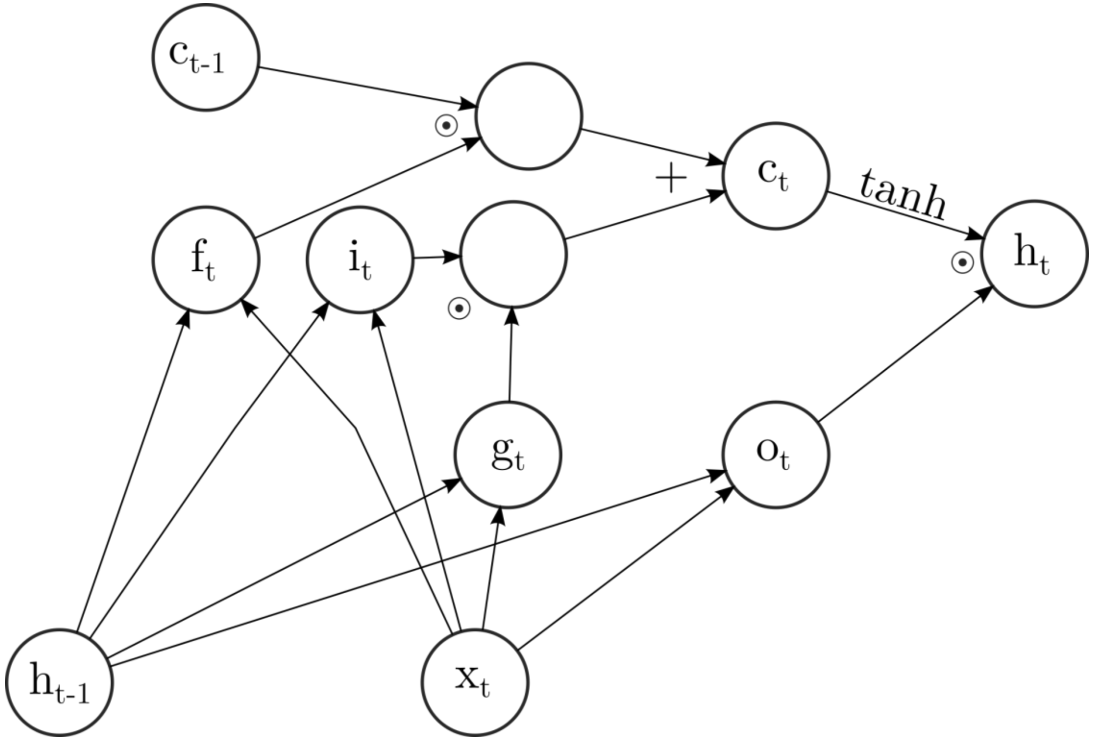
- A network that has a **feedback** connection from the „previous" output to the „current" input
- Recurrent network = IIR filter



After: J. D. Parker, F. Esqueda, and A. Bergner, *Modelling of Nonlinear State-Space Systems Using a Deep Neural Network*, in Proc. of the 22nd Int. Conf. on Digital Audio Effects (DAFx-19), Birmingham, UK, 2019.

Long Short-Term Memory (LSTM)

- S. Hochreiter and J. Schmidhuber (1997)



Loss function

- „How far is the network's output from the desired output (the target)?"
- Mean squared error (MSE)

$$\mathcal{E}_{\mathrm{MSE}}(y, \hat{y}) = \frac{1}{N} \sum_{n=0}^{N-1} (y[n] - \hat{y}[n])^2,$$

- Error-to-signal ratio (ESR)

$$\mathcal{E}_{\mathrm{ESR}}(y, \hat{y}) = \frac{\sum_{n=0}^{N-1} (y[n] - \hat{y}[n])^2}{\sum_{n=0}^{N-1} (y[n])^2},$$

Optimization (learning) algorithm

- A method of calculating the coefficients update to reduce the value of the loss function after each minibatch.
- **Stochastic gradient descent** with adaptive parameters; Adam by Kingma and Ba is one of the most popular.
- Supported out-of-the-box by PyTorch.

Hyperparameters

- Control the behavior of the learning algorithm
- Are not adapted during learning
- Most popular hyperparameters:
  - Learning rate (gradient step size)
  - Number of epochs in training
  - Minibatch size
  - Weight decay
  - Architecture
  - Optimization algorithm
  - …

How to train a network?

1. For each epoch:
    1. For each minibatch in the training set:
        1. Run the minibatch through the network
        2. Calculate the loss function (training loss)
        3. Calculate the gradient of the loss
        4. Update the coefficients according to the optimization algorithm
    2. Every few epochs:
        1. Run the validation set input through the network
        2. Calculate the loss function (validation loss)
        3. If better than current best: store the loss value and current parameters
2. Adjust the hyperparameters **based on the validation loss**
3. Repeat

How to evaluate a model?

- Run the trained, validation set-adjusted model on the test set
- Calculate the value of the test loss function
- Compare against other models

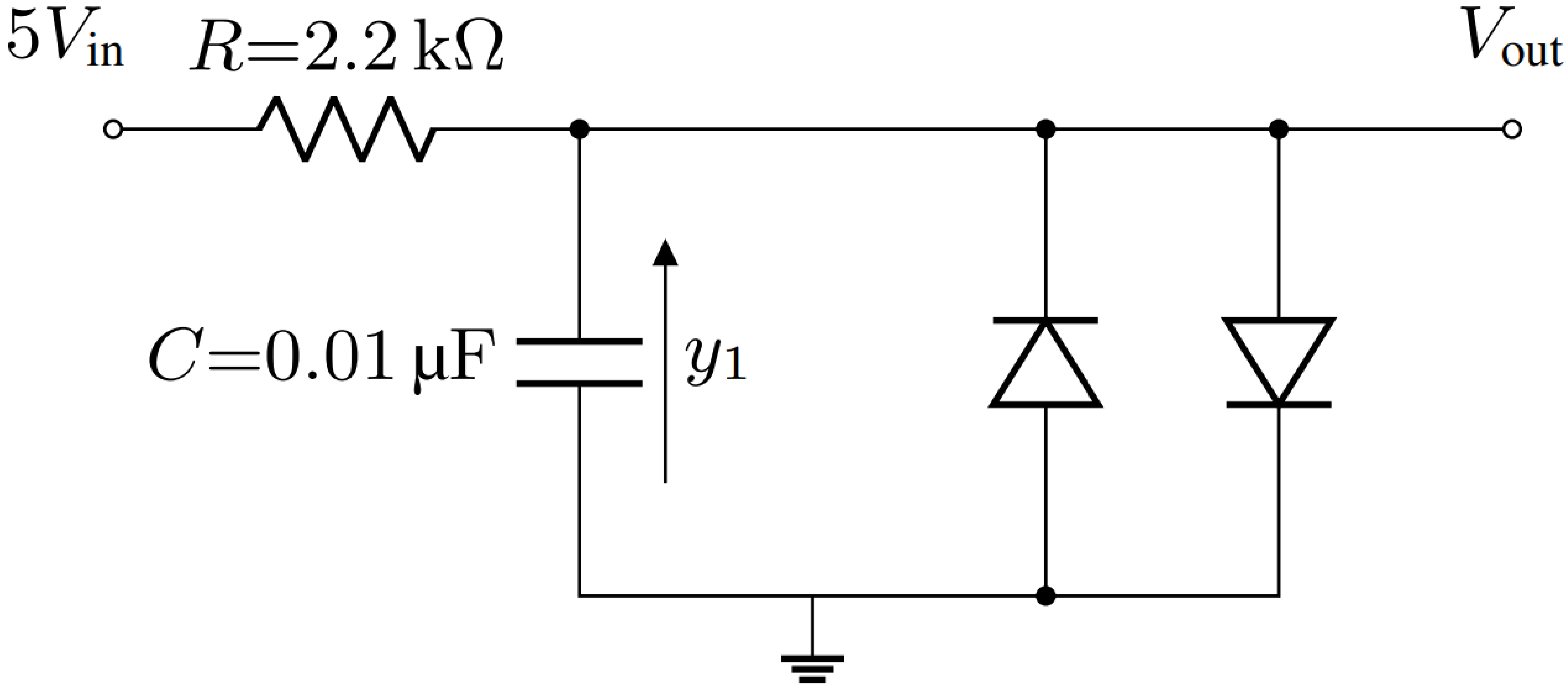
What is needed for network training?

- Problem to solve
- Dataset
- Hyperparameters
- Python
- PyTorch
- IDE
- Local CPU or GPU
- Understanding which hyperparameters to change

No data or computational cluster?

[Live Demo]

Problem description: Virtual Analog modeling

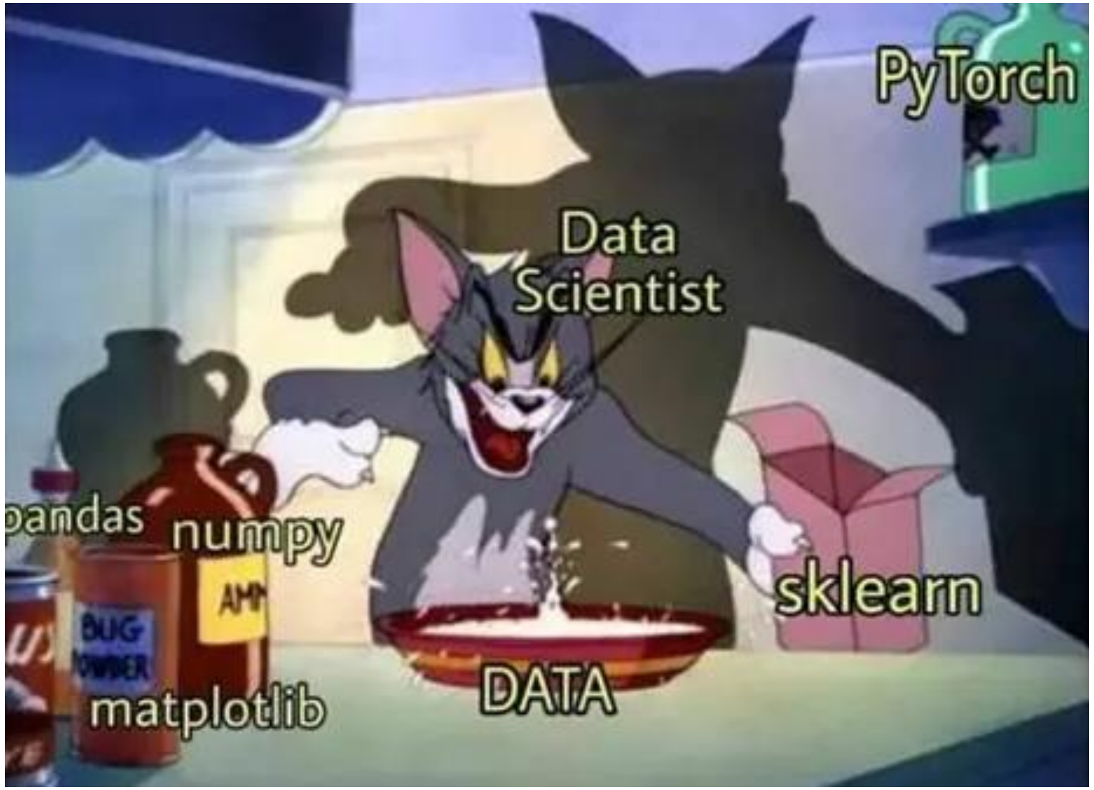


More resources

- Fraunhofer IDMT datasets: https://www.idmt.fraunhofer.de/en/publications/datasets.html
- LTSpice: https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html
- CoreAudioML by Alec Wright: https://github.com/Alec-Wright/CoreAudioML
- Repository for guitar amplifier modeling by Alec Wright: https://github.com/Alec-Wright/Automated-GuitarAmpModelling
- arxiv.org
- researchgate.net
- my master thesis ☺

Conclusion

1. Learning deep learning is not hard…
2. …but successfully training new architectures is challenging.



WolfSound

Become an Audio Programmer

jan.wilczek@thewolfsound.com

Loudly

loudly.com

Thank you!

Slides & code:

github.com/JanWilczek/adc22