UNIVERSIDAD DE GRANADA

DECSAI

Reinforcement Learning: Neural Networks

September 25th – 29th, 2023

E.T.S. de Ingenierías Informática y de Telecomunicación

Universidad de Granada

Manuel Pegalajar Cuellar / Juan Gómez Romero

manupc@ugr.es / jgomez@ugr.es

Departamento de Ciencias de la Computación e Inteligencia Artificial

http://decsai.ugr.es

UNIVERSIDAD
DE GRANADA

Reinforcement Learning:
Neural Networks

Contents

DECSAI

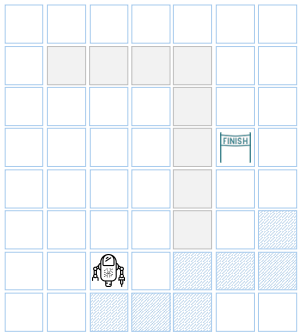
What we know so far

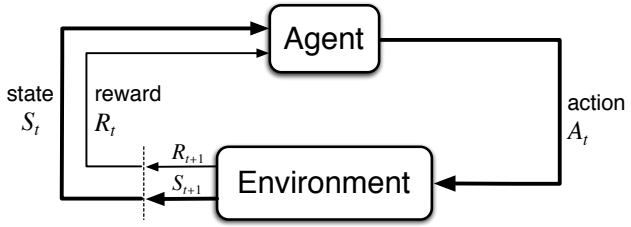
- Fundamental concepts of RL: state, action, reward, policy
- Formulation of a Markov Decision Process
- The value function *V(s)* and state-value function *Q(s, a)*
- Value iteration algorithm
- Q-Learning algorithm

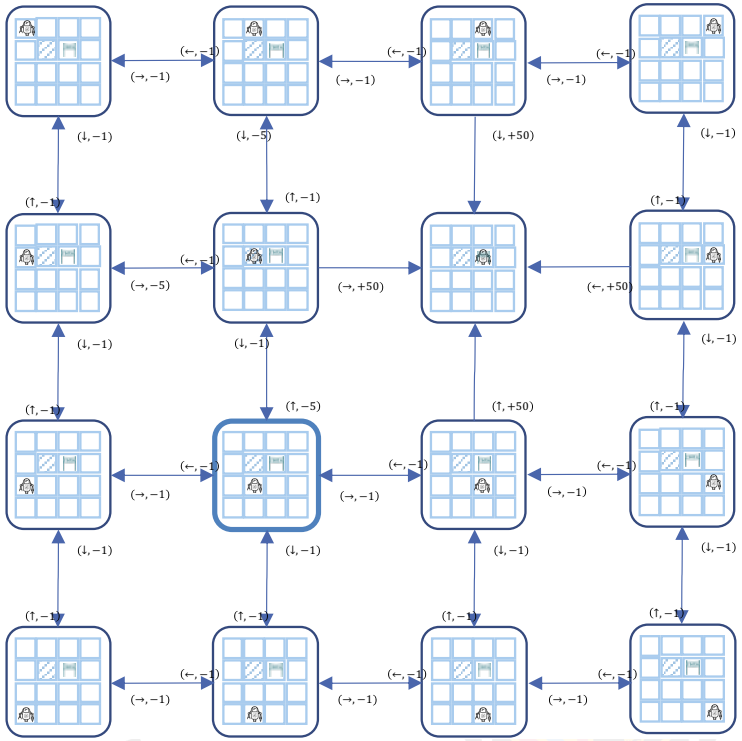
Reinforcement Learning

Find paths in the state/transition graph

Optimize the future accumulated reward (*return*)







Q-Learning

Estimate $Q(s, a)$

Q		Action		
		A_1	...	A^m
State	S^1			
	...			
	S^n			

Limitations with continuous action or state functions

Experience is generated sequentially

From interactions with the environment

$S_0\ A_0\ R_1\ S_1 \mid A_1\ R_2\ S_2 \mid$

$$Q(S_0, A_0) \leftarrow Q(S_0, A_0) + \alpha\left(R_1 + \gamma \max_{a \in \mathcal{A}} Q(S_1, a) - Q(S_0, A_0)\right)$$

$$Q(S_1, A_1) \leftarrow Q(S_1, A_1) + \alpha\left(R_2 + \gamma \max_{a \in \mathcal{A}} Q(S_2, a) - Q(S_1, A_1)\right)$$

The action can be selected based on a policy (on-policy) or by maximizing the Q (off-policy)

A random action may be selected with probability ϵ (epsilon-greedy)

Deep Q-Learning

Model $Q(s,a)$ as a (non-linear) function

$Q(s,a) \rightarrow \mathbb{R}$

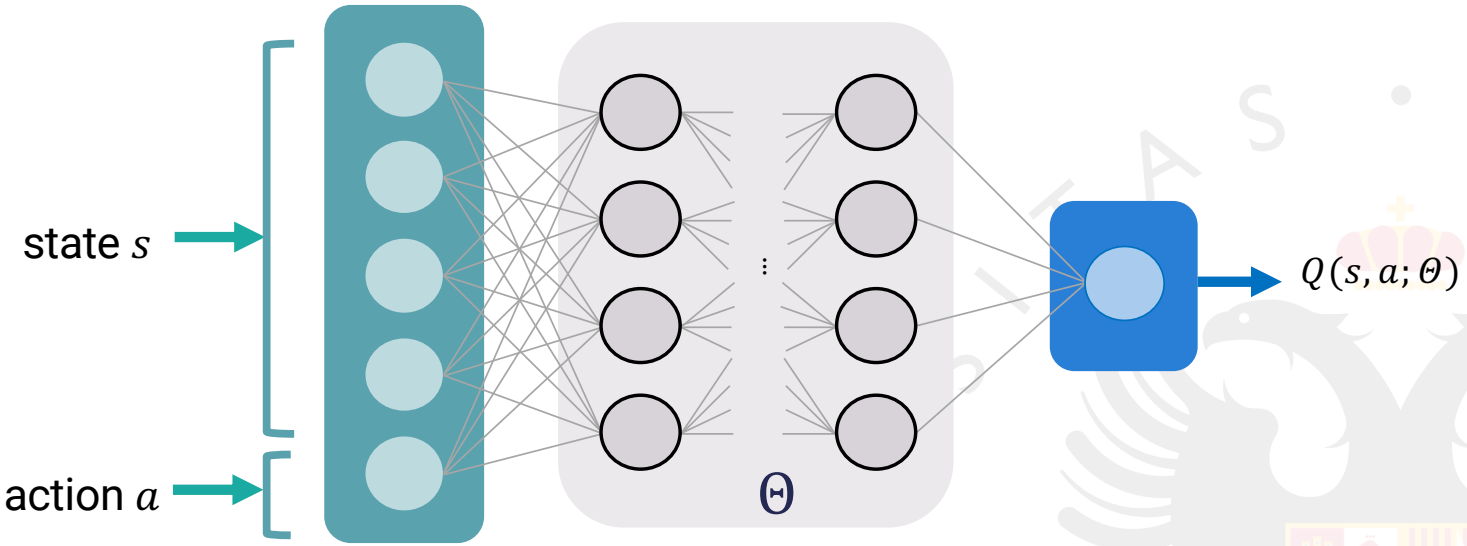
Q can be approximated with a parametric model $Q(s,a;\Theta)$

Parameters of $Q(s,a,\Theta)$ can be adjusted to samples

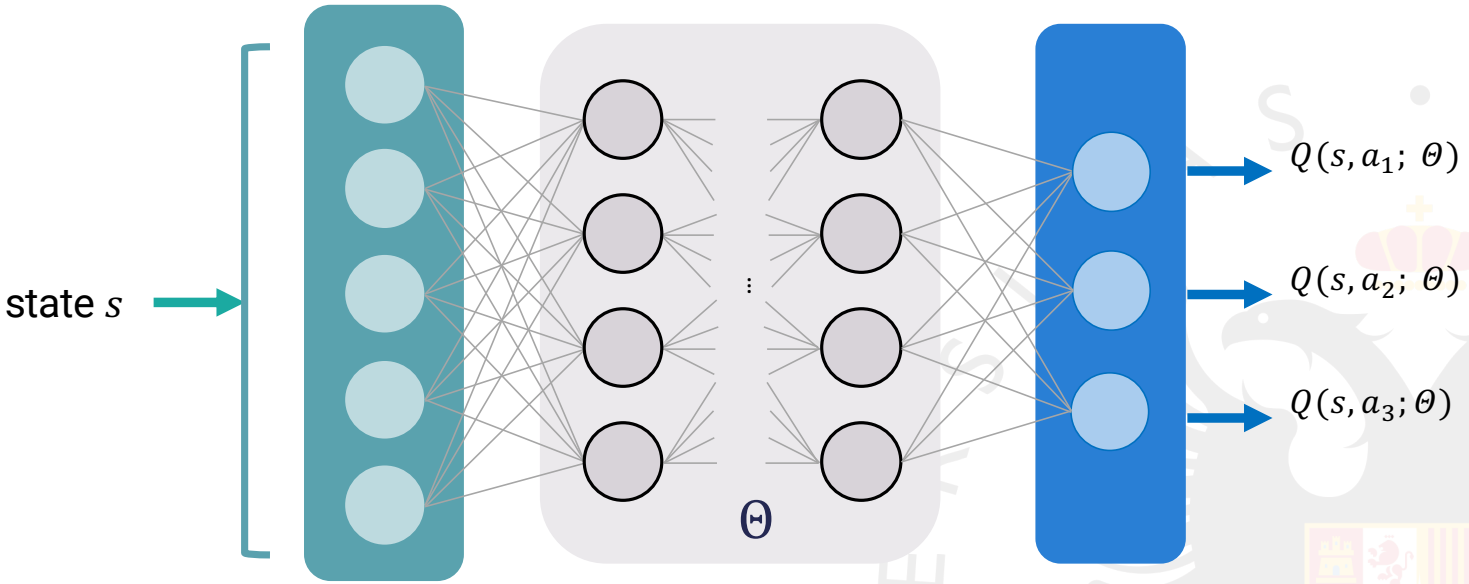
$S_0\ A_0\ R_1\ S_1 \rightarrow \hat{Q}\ (S_0, A_0)$
$S_1\ A_1\ R_2\ S_2 \rightarrow \hat{Q}(S_1, A_1)$
…

We can keep an experience memory with the Q values calculated at each step

Deep Q-Learning
Feed forward neural network



Deep Q-Learning
Feed forward neural network



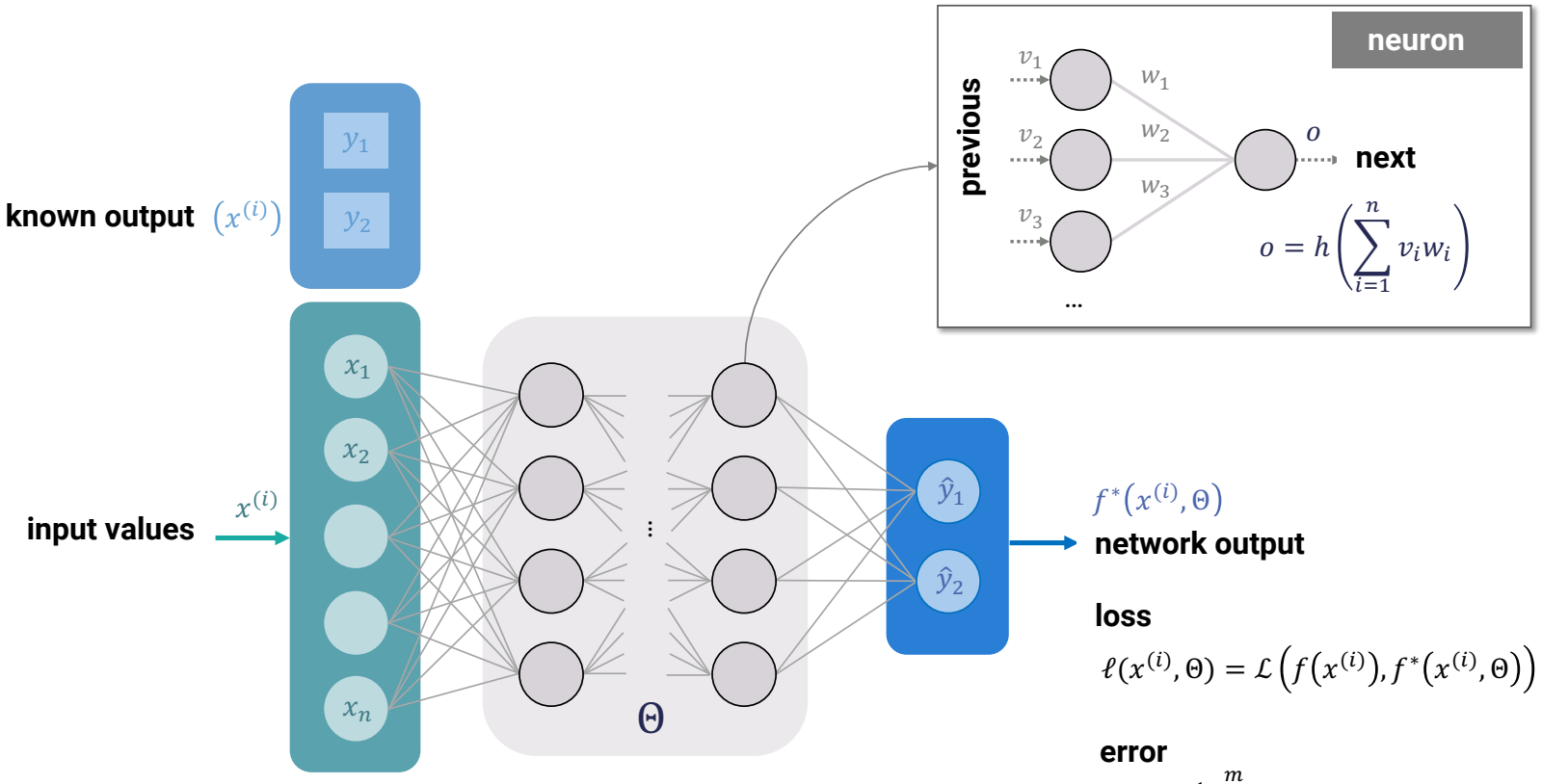
UNIVERSIDAD DE GRANADA

Reinforcement Learning:
Neural Networks

Contents

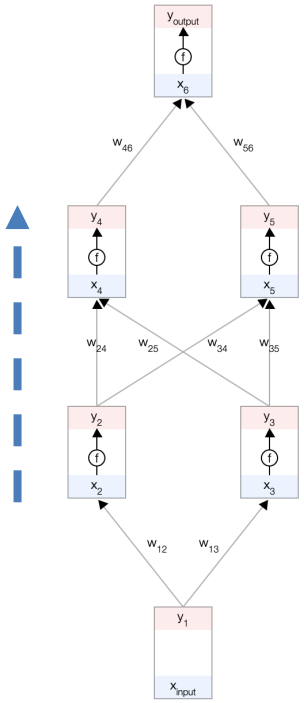
DECSAI

Architecture of feed-forward neural networks

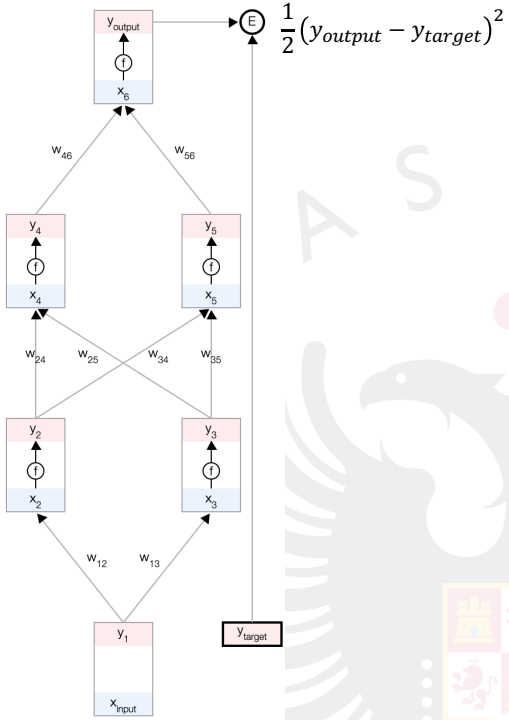


Output & error calculation

https://developers.google.com/machine-learning?hl=en



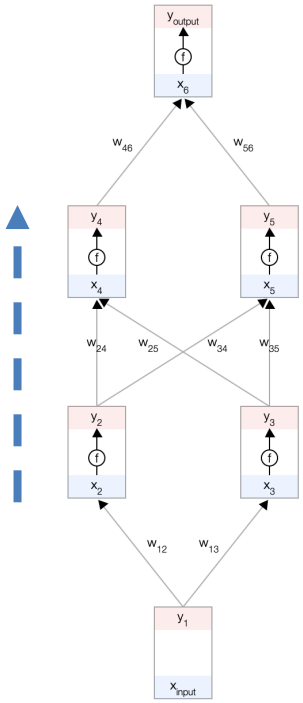
forward pass



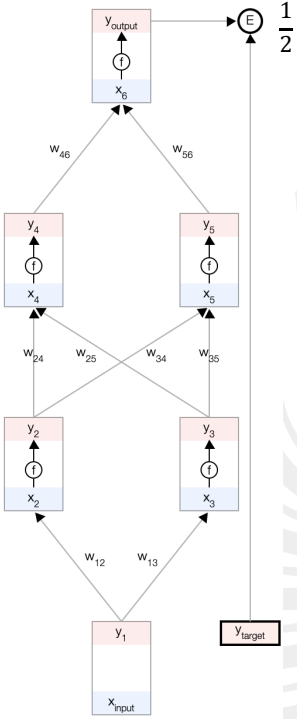
error calculation

Weight adjustment

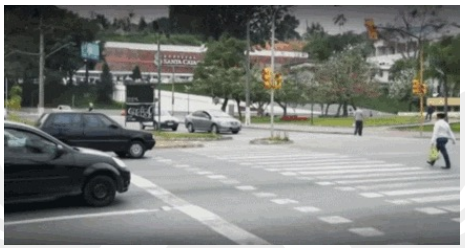
https://developers.google.com/machine-learning?hl=en



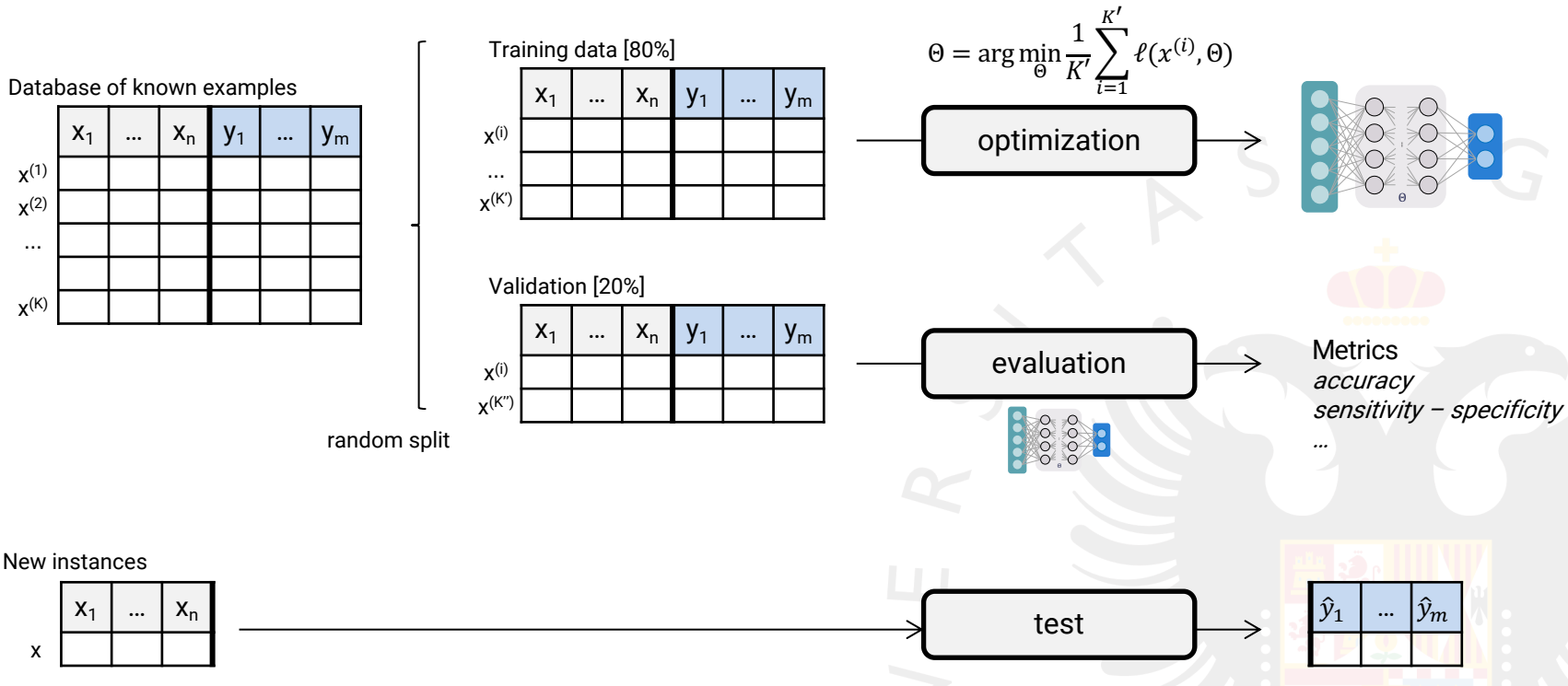
forward pass

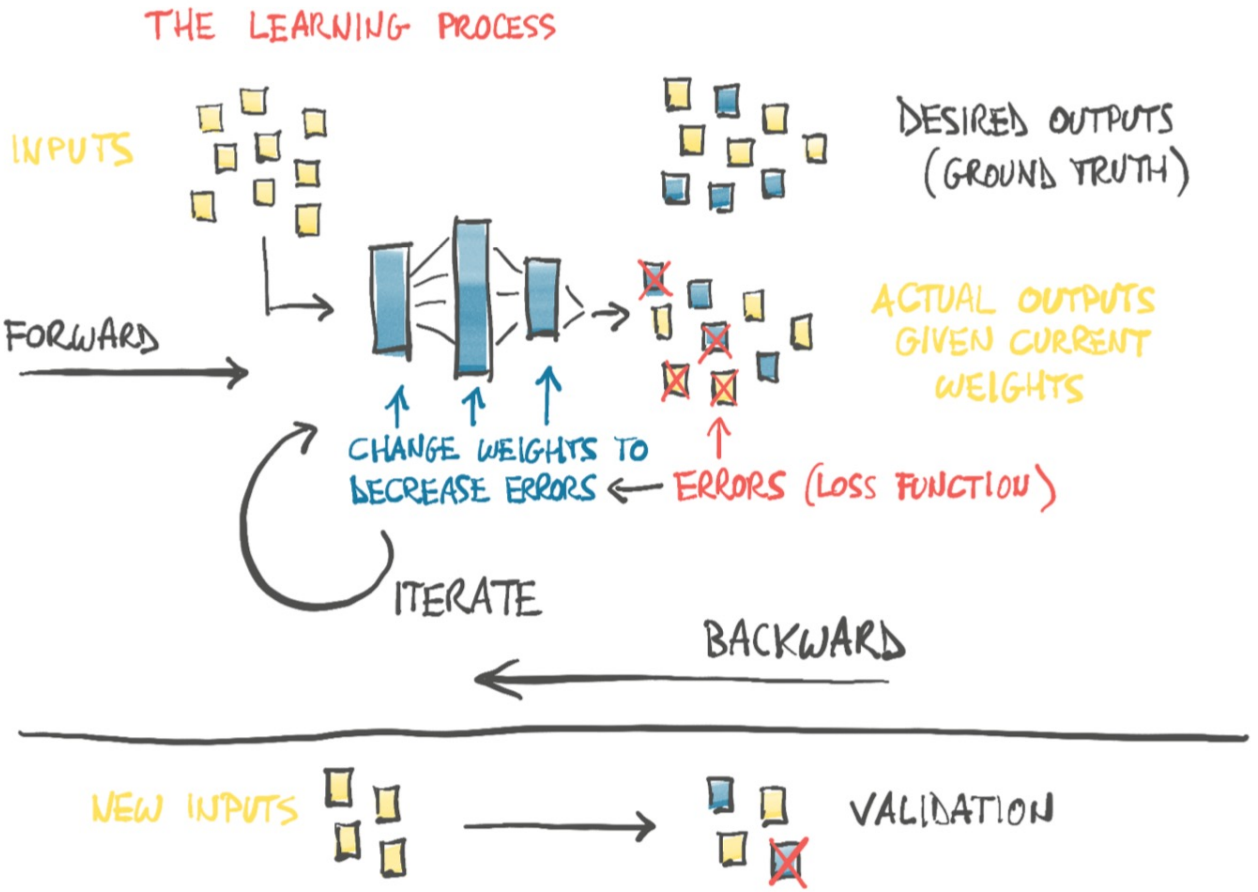


error calculation



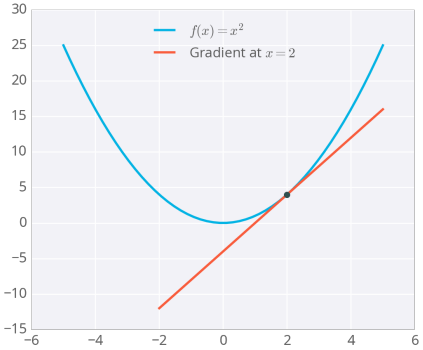
Training methodology





E. Stevens, L. Antiga (2019) **Deep Learning with PyTorch**. Manning.

Gradient descent

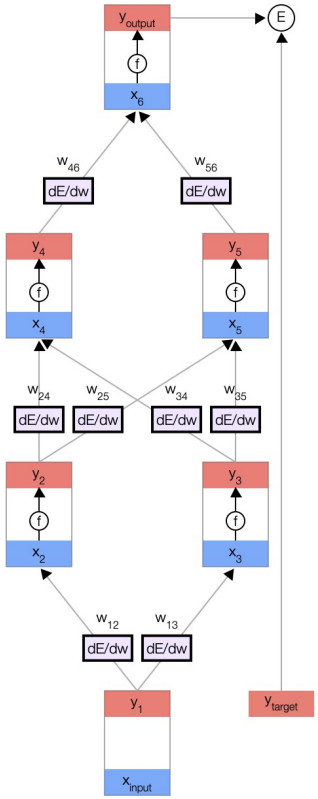


$$\nabla E = \frac{\partial f(w)}{\partial w}$$

$$\Delta w_{ij} \propto -\frac{\partial E}{\partial w_{ij}}$$

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$$

Backpropagation



$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial \left(\frac{1}{2}(y - \hat{y}(w))^2\right)}{\partial w_{ij}}$$

$$= -(y - \hat{y})\, h'(x_j)\, y_i$$

UNIVERSIDAD DE GRANADA

Reinforcement Learning:
Neural Networks

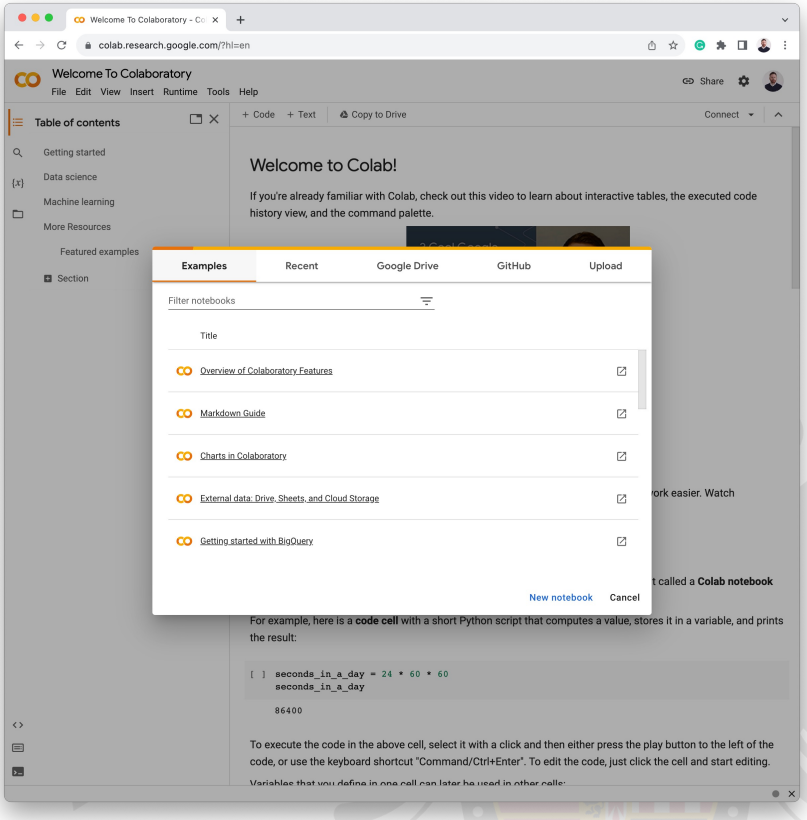
Contents

DECSAI

Google Colaboratory

https://colab.research.google.com

- Cloud-based environment for notebooks *a la* Jupyter (code + text)
- Write Python code, storage in Google account
- GPU-enabled execution
- PyTorch is pre-installed

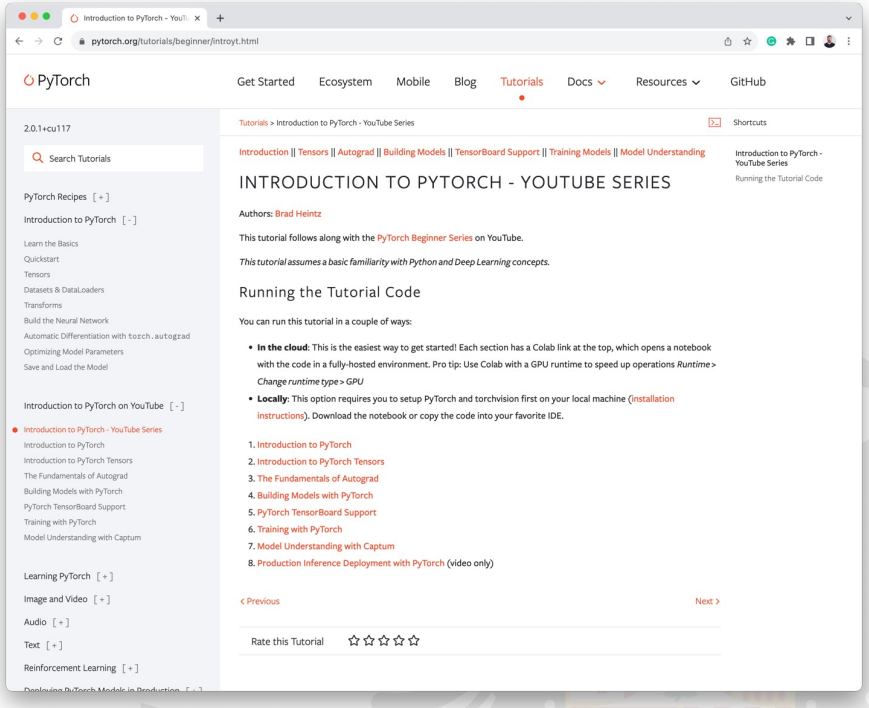


Open in Colab

https://github.com/jgromero/rl_seminar_2023/tree/main/code

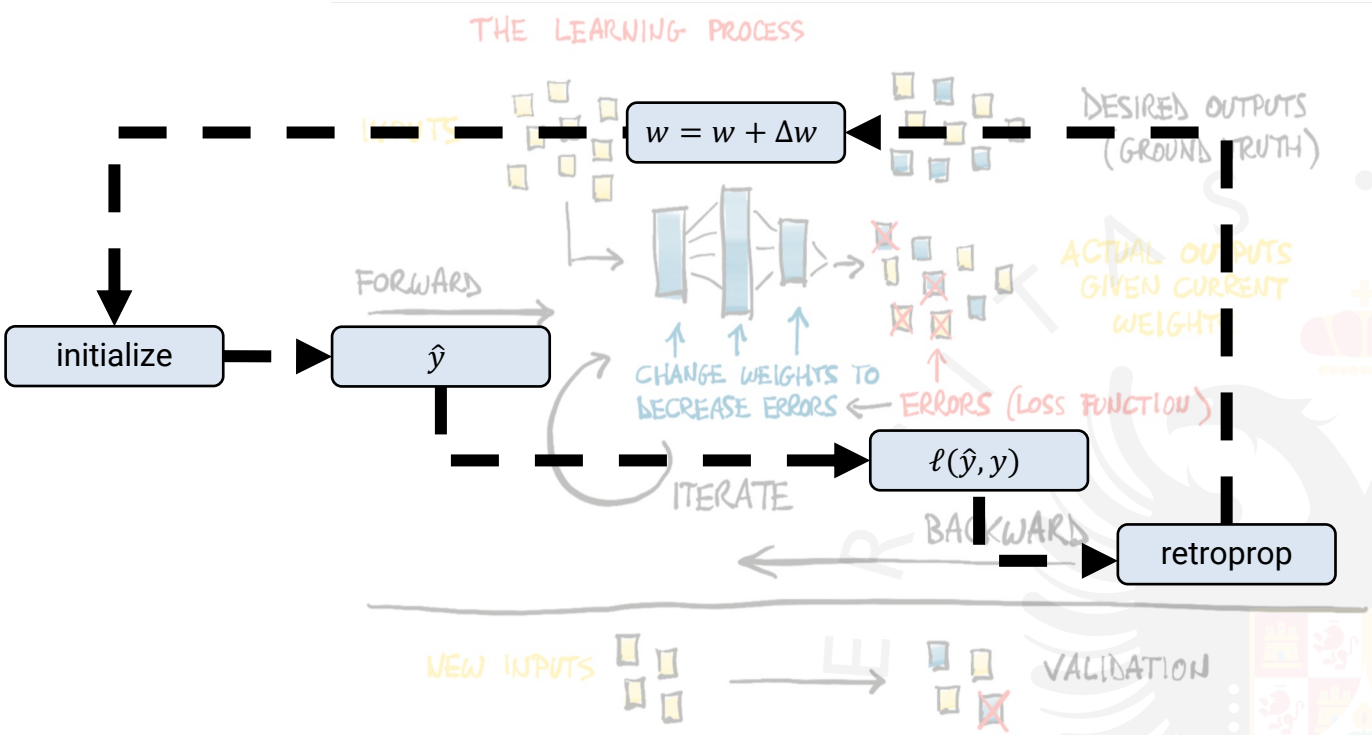
PyTorch

- Python library for machine learning with neural networks
- Based on *tensors* and *computation graphs*
- Automatic differentiation & optimization
- GPU-enabled
- Repository of pre-trained models



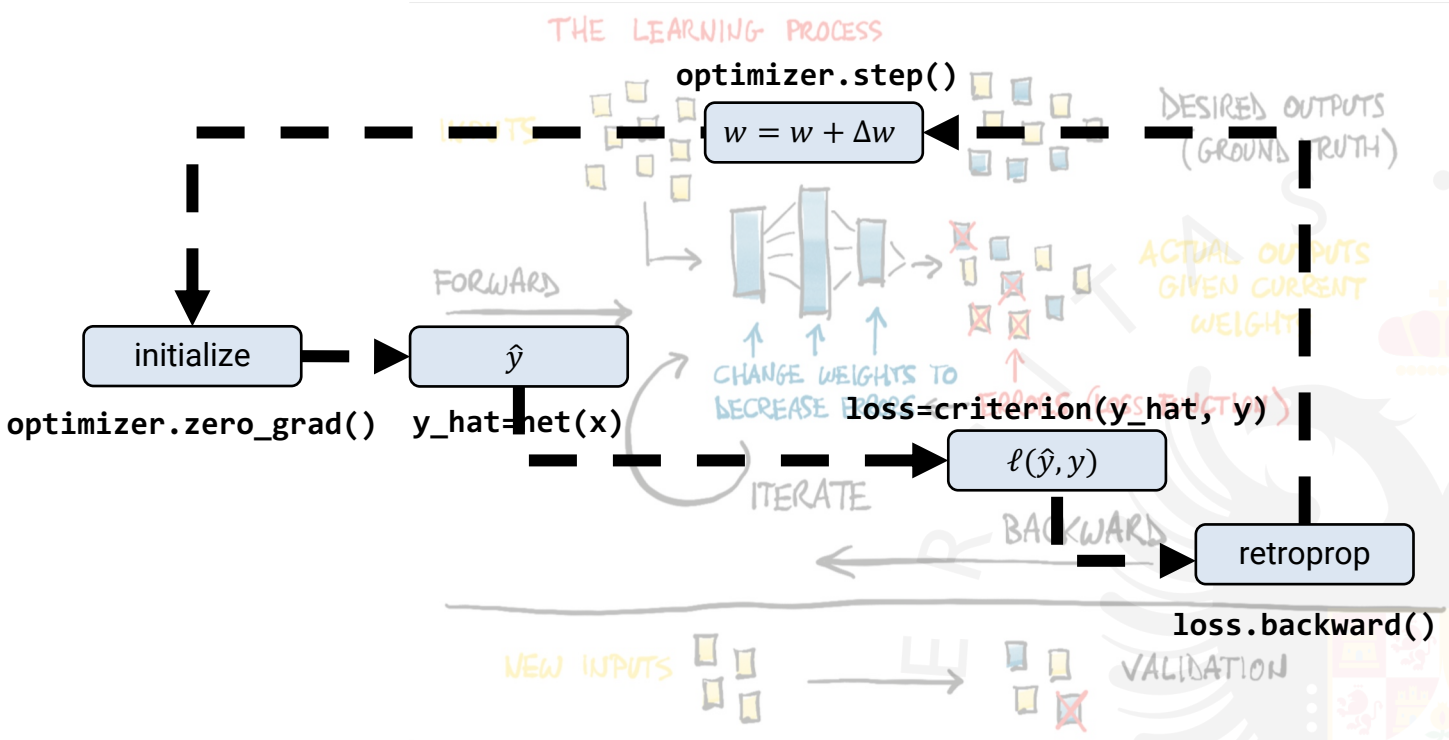
https://pytorch.org/tutorials/beginner/introyt.html
[images]

PyTorch



E. Stevens, L. Antiga (2019) **Deep Learning with PyTorch**. Manning.

PyTorch



E. Stevens, L. Antiga (2019) **Deep Learning with PyTorch**. Manning.

PyTorch

Example: classification with the *iris* dataset
https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html

Predict flower class from shape measurements

R. Fisher (1936) *The use of multiple measurements in taxonomic* problems

Data
3 classes: iris setosa, iris virginica, iris versicolor
50 samples of each class
4 features: length and width of petals and sepals

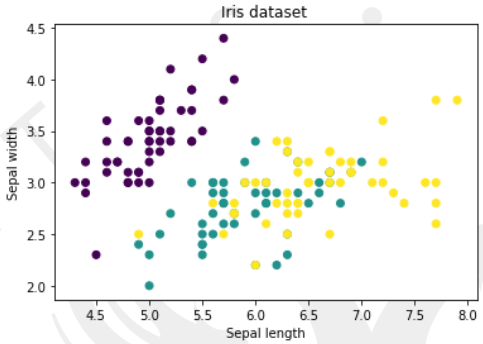
Target: find the class of an unknown sample given the values of the 4 features

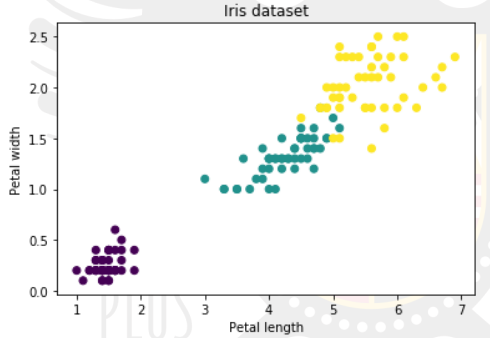
PyTorch

Example: classification with the *iris* dataset

https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html

sepal length	sepal width	petal length	petal width	class
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
5.0	3.6	1.4	0.2	Iris-setosa
7.0	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4.0	1.3	Iris-versicolor
6.5	2.8	4.6	1.5	Iris-versicolor
6.7	2.5	5.8	1.8	Iris-virginica
7.2	3.6	6.1	2.5	Iris-virginica
6.5	3.2	5.1	2.0	Iris-virginica
6.4	2.7	5.3	1.9	Iris-virginica
6.8	3.0	5.5	2.1	Iris-virginica
...				



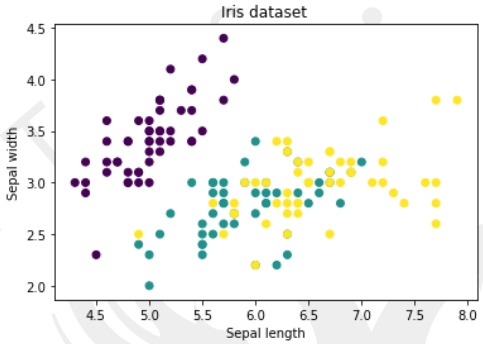


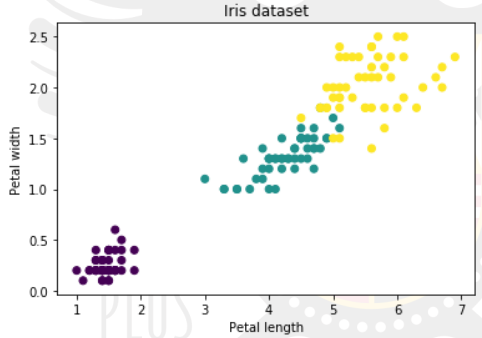
PyTorch

Example: classification with the *iris* dataset

https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html

sepal length	sepal width	petal length	petal width	class
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
5.0	3.6	1.4	0.2	Iris-setosa
7.0	3.2	4.7	1.4	Iris-versicolor
6.4	3.2	4.5	1.5	Iris-versicolor
6.9	3.1	4.9	1.5	Iris-versicolor
5.5	2.3	4.0	1.3	Iris-versicolor
6.5	2.8	4.6	1.5	Iris-versicolor
6.7	2.5	5.8	1.8	Iris-virginica
7.2	3.6	6.1	2.5	Iris-virginica
6.5	3.2	5.1	2.0	Iris-virginica
6.4	2.7	5.3	1.9	Iris-virginica
6.8	3.0	5.5	2.1	Iris-virginica
...				





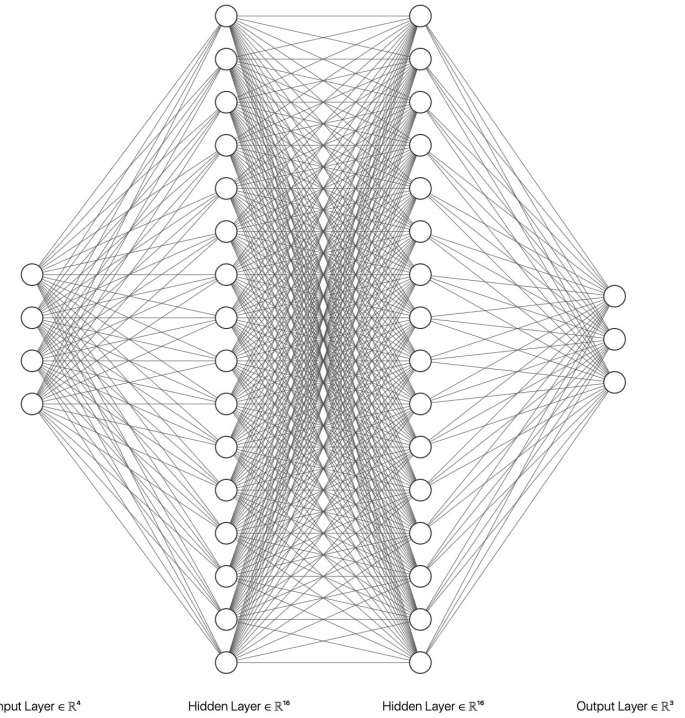
PyTorch

Example: classification with the *iris* dataset
https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html

1. Load dataset
2. Define network architecture
3. Split training and validation datasets
4. Train network with train split
5. Validate with validation Split

Hyperparameters

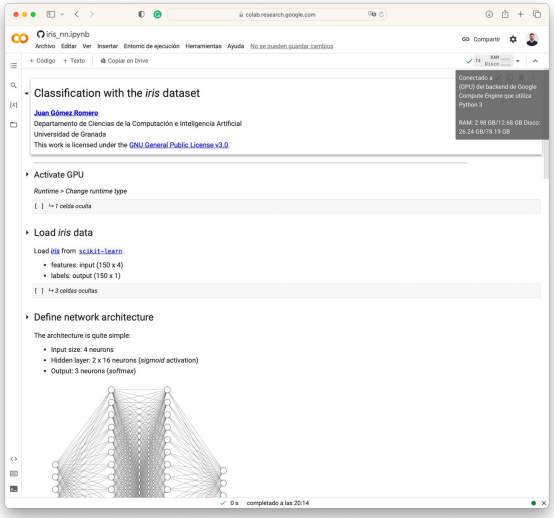
- batch size: 50
- epochs: 1000

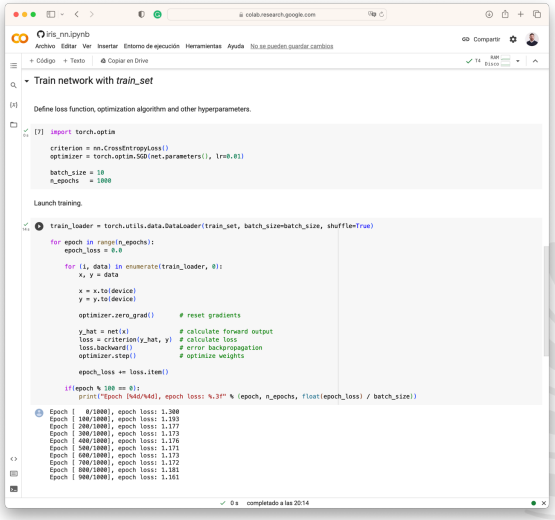


PyTorch

Example: classification with the *iris* dataset

https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html





Open in Colab

https://github.com/jgromero/rl_seminar_2023/tree/main/code

iris-nn.ipynb