Master Thesis in Physics of Data

Biological Networks as Defense against Adversarial Attacks

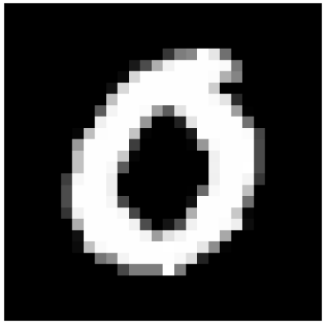
Supervisor
Prof. Marco Baiesi

Master Candidate
Andrea Zanola

Dipartimento di Fisica e Astronomia Galileo Galilei

Università degli Studi di Padova

Introduction

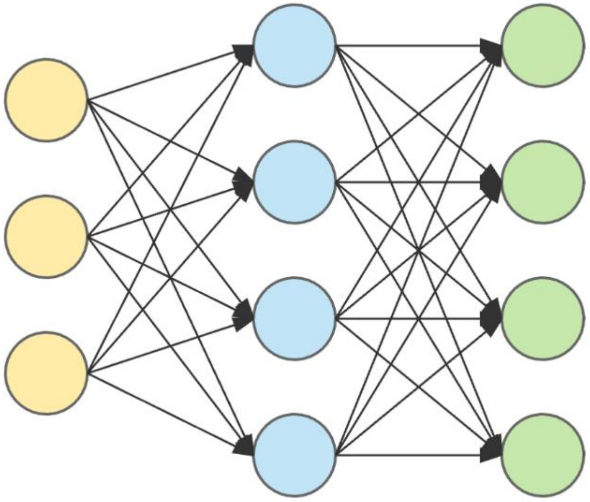


Does artificial neural networks really work?

The first exercise that a student does studying ML, is MNIST digit classification.

Dataset:
MNIST, 70000 b/w images of handwritten digits, 28x28 pixels.

Base solution:
- ANN with min 3 layers, 784 neurons in input and 10 in output;
- Activation Functions: Linear-ReLU-Softmax.
- Loss Function: Cross-Entropy.
- Training Procedure: Back-propagation in PyTorch.



Results:
- Test Accuracy: 96%!
- CPU time: 1-2min

Is there something else, or this is all what we need?

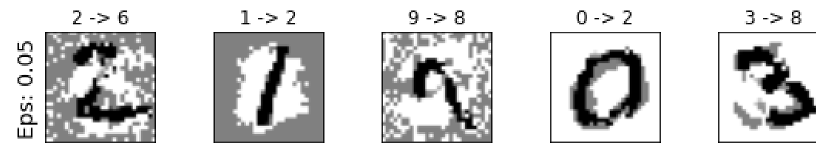
Adversarial Attacks

Suppose to give in input to the network the following images:

The predicted labels are:

6 2 8 2 8

These are not MNIST images!



Several ways to create the mask attack:
the most famous is **FGSM** (Fast Gradient Sign Method) and it was used to generate the examples above.

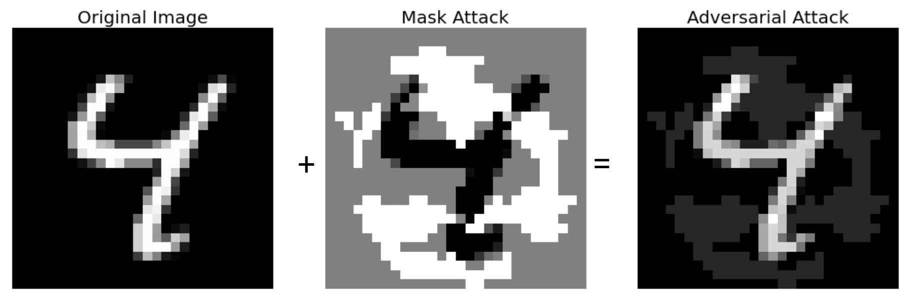


Fig. Example of an adversarial attack.

Szegedy et al. described such weakness of ANN in: “Intriguing properties of neural networks”, 2014.

FGSM Algorithm

Università degli Studi di Padova

The FGSM algorithm calculates the direction that appears to be the fastest route to a misclassification.

The formula defining the FGSM attack is:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boxed{\epsilon \times sign\left[\boldsymbol{\nabla}_{\boldsymbol{x}} J\left(f\left(\boldsymbol{x}, \Theta; l\right)\right)\right]}$$

Mask Attack

where:

- $\boldsymbol{x}$: is the original image.
- $\tilde{\boldsymbol{x}}$: is the adversarial attack.
- ϵ: the strength of the attack.
- J: the loss function of an ANN f.

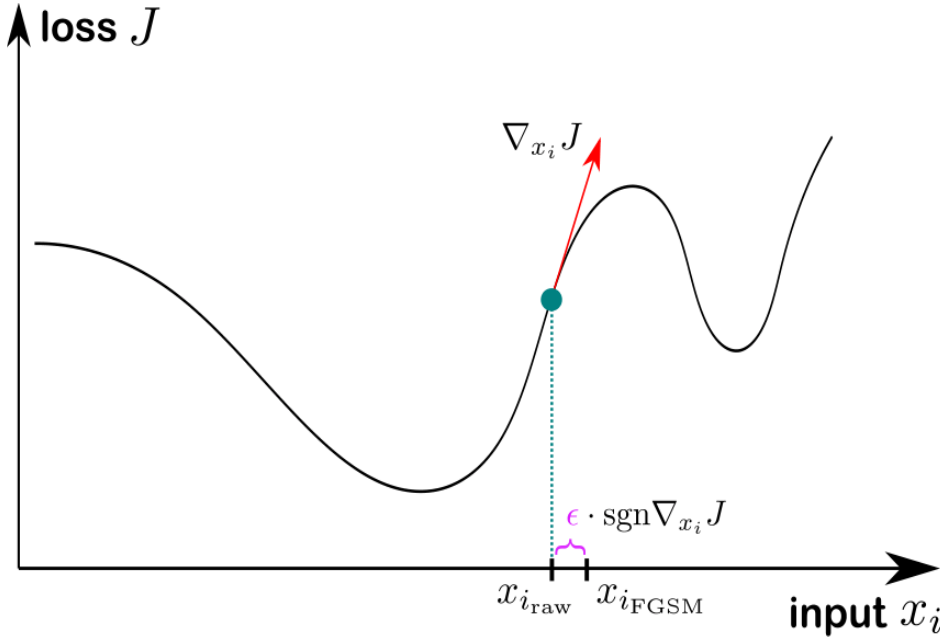
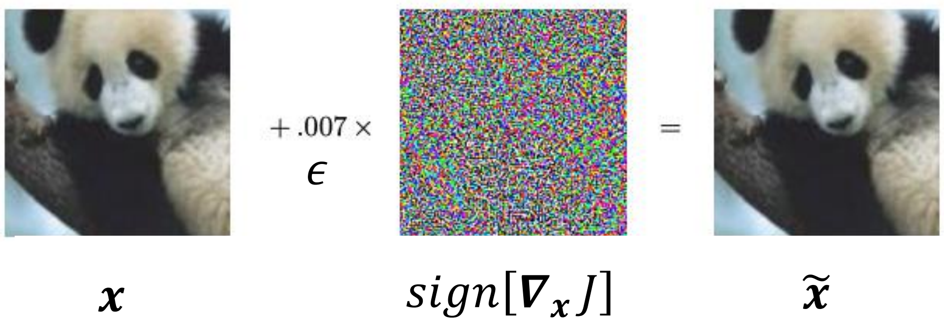


Fig. FGSM algorithm. Image taken from «*Improving robustness of jet taggin algorithms with adversarial training*», by A. Stein et al. (2022).

Accuracy Under Attack

One of the most famous examples is that of a panda classified as a gibbon!



PANDA 57% → GIBBON 99.7%

Fig. Goodfellow's famous example. Image taken from «Explaining and harnessing adversarial examples», by I. Goodfellow et al. (2015).

A simple ANN, trained in PyTorch is not adversarial robust.

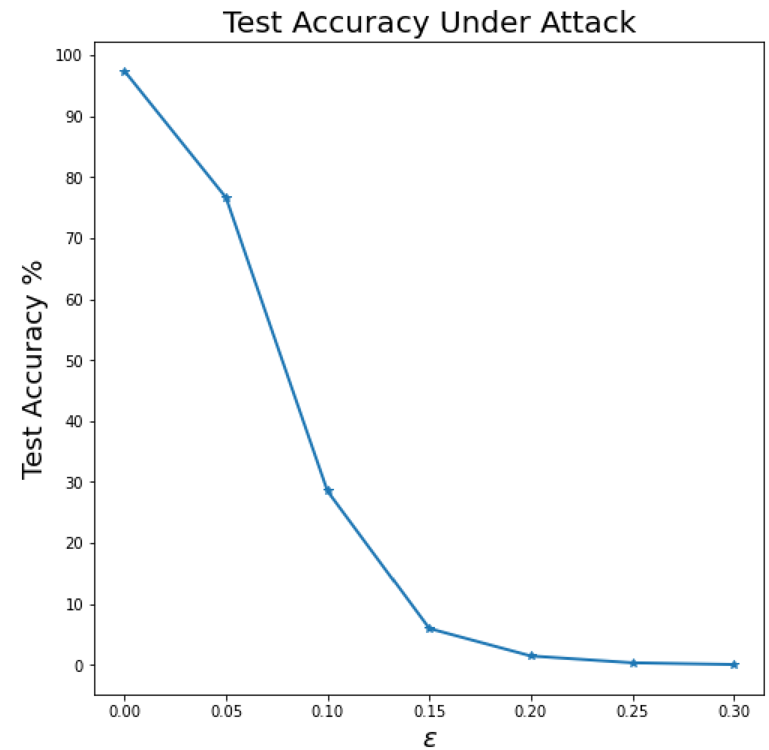


Fig. Test accuracy under attack for an ANN trained only with BP.

Let's call **test-accuracy under attack** the accuracy that an ANN has giving in input as test set images, samples that are adversarial attacks.

EU Guidelines

UNIVERSITÀ DEGLI STUDI DI PADOVA

The adversarial vulnerability is a disaster, since the current ML research is every day more and more focused in finding architectures that not only have a great test-accuracy, but that also have **other qualities, like technical robustness and transparency**.

Robustness: it is the ability to resist against adversarial attacks.

Transparency: a model might be called transparent if a person can contemplate the entire model at once.

Many key-requirements help to build broader trust on AI:

Trustiness: the ability to anticipate the AI behavior.

European Guidelines for Trustworthy AI Models

Key Requirements	**Explanatory Methods/Analyses**
Human agency and oversight	· See "Diversity, non-discrimination, fairness" · User-centered explanations · Explanations in recommender systems
Technical robustness and safety	· Adversarial attacks and defenses · N/A · N/A · Contrast sets , behavioral testing · "Show your work"
Privacy and data governance	· Removal of protected attributes · Detecting data artifacts · N/A
Transparency	· N/A · Saliency maps , self-attention patterns , influence functions , probing · Counterfactual , contrastive , free-text , by-example , concept-level explanations · N/A
Diversity, non-discrimination, fairness	· Debiasing using data manipulation · N/A · N/A
Societal and environmental well-being	· Analayzing individual neurons · Bias exposure · Explanations designed for applications such as fact checking or fake news detection
Accountability	· N/A · N/A · Reporting the robustness-accuracy trade-off or the simplicity-equity trade-off · N/A

Biological Networks

A neuron essentially is composed by:

- Ramifications, called **dendrites** that receive signals in input.
- A **soma**, which contains a nucleus.
- An **axon** that transmits the signal towards other neurons.

The real brain is plastic, a human can ”rewire” it with habits and new experiences.
The simplest form of plasticity is that **neural pathways that are used a lot strengthen.**
This inspired the Hebb’s principle.

$$\tau_w \frac{d\boldsymbol{w}}{dt} = h\boldsymbol{v}$$

where:

- $\boldsymbol{v}$: input vector received from the dendrites.
- h: output value of the neuron.

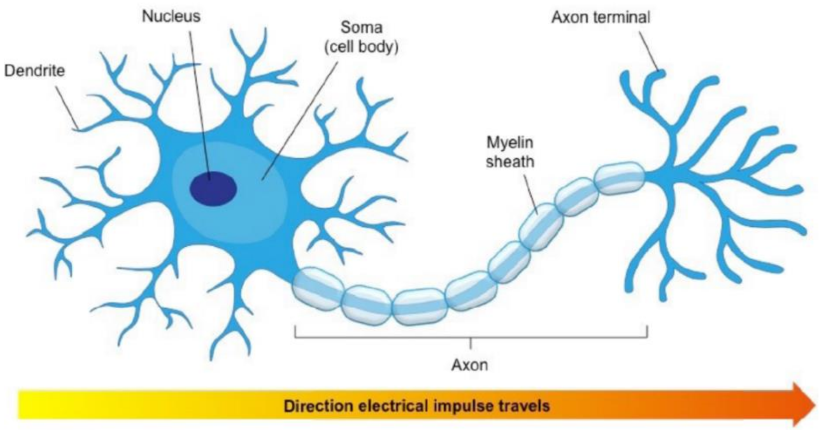


Fig. Biological neuron.

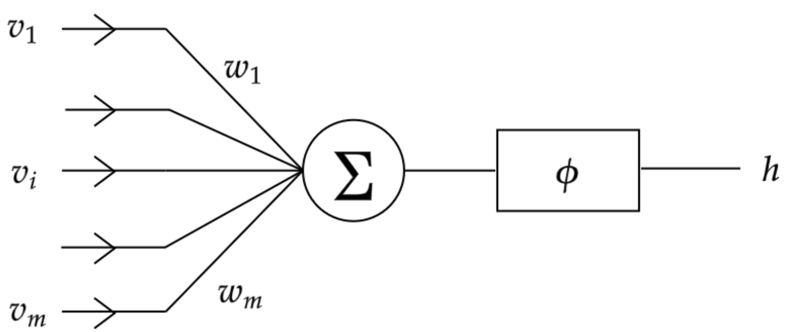


Fig. McCulloch-Pitts’ model of a neuron.

However, two missing ingredients: **competition** and **synapses weakening.**

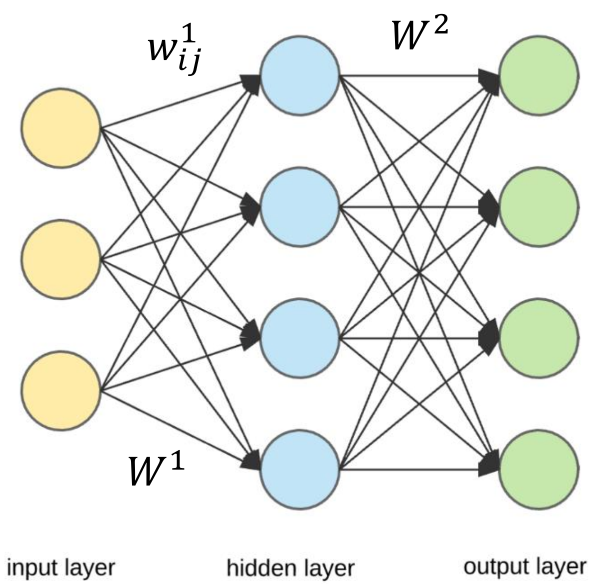
Artificial Networks

"It is widely believed that end-to-end training with the back-propagation algorithm is essential [...]. At the same time, the traditional form of back-propagation is biologically implausible"

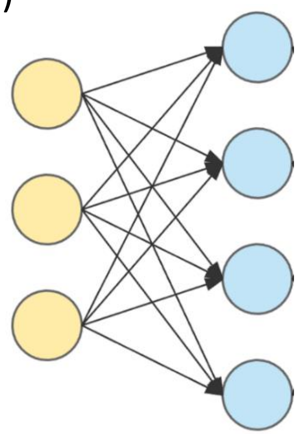
D.Krotov - J.J. Hopfield

Back-propagation is not biologically plausible, since the change of a specific weight w_{ij} requires the knowledge of all the previous weights up to the output layer of the network.
Plasticity is only between two consequent neurons.

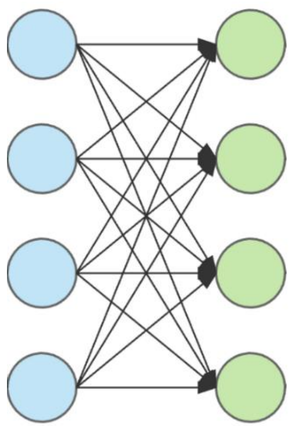
This means that the neuronal dynamic is local while BP is not.
Fundamental idea: avoid back-propagation!



a)



b)



First layer is trained in the first phase, while the second in the second phase.

First Layer

The first layer is trained in a biological way and the training is constituted by the iteration of two steps, for each training example:

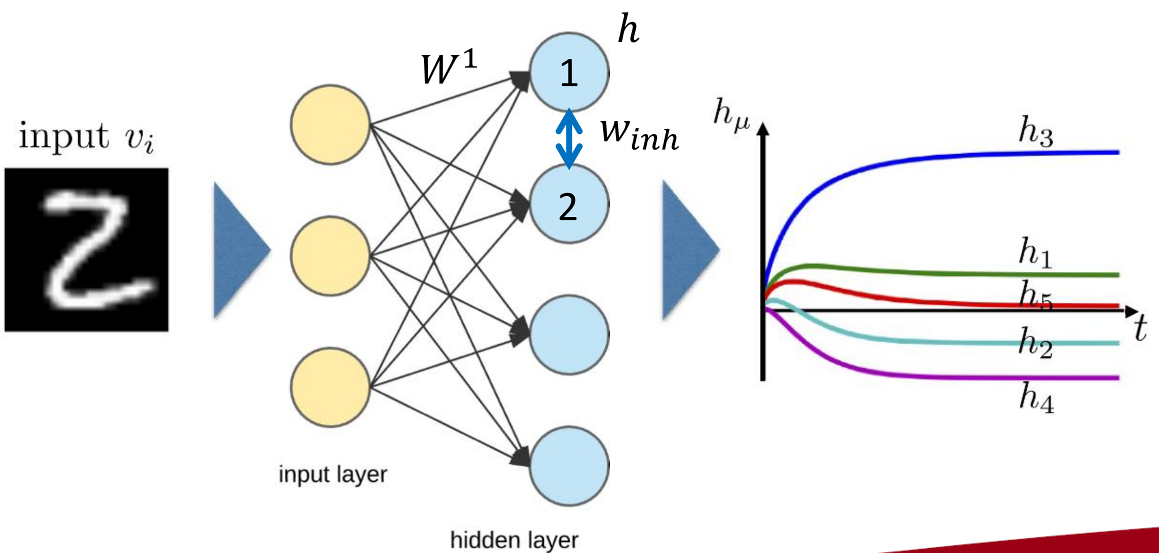
1. Give an image in input. Wait until the hidden neurons reach a steady state solution $\boldsymbol{h}_\infty$. The dynamic is described by an ODE that is like the v-equation of the **firing rate model**.

$$\tau_r \frac{dh_\mu}{dt} = I_\mu - w_{inh} \sum_{\nu \neq \mu} r(h_\nu) - h_\mu$$

2. Once the steady state is achieved, the matrix of weights W^1 is modified.

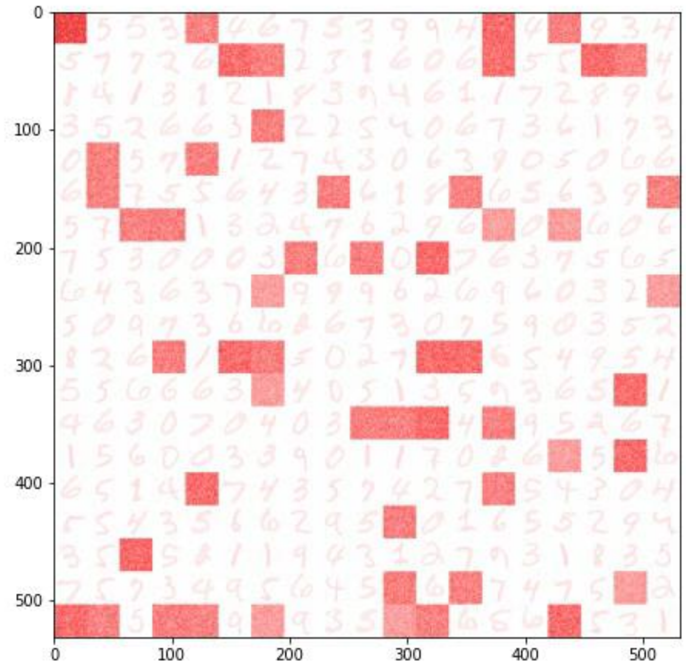
$$\tau_w \frac{dw_{\mu i}}{dt} = \mathrm{g}(h_\mu) \left[v_i - \left(\sum_{k=1}^{784} w_{\mu k}\, v_k \right) w_{\mu i} \right]$$

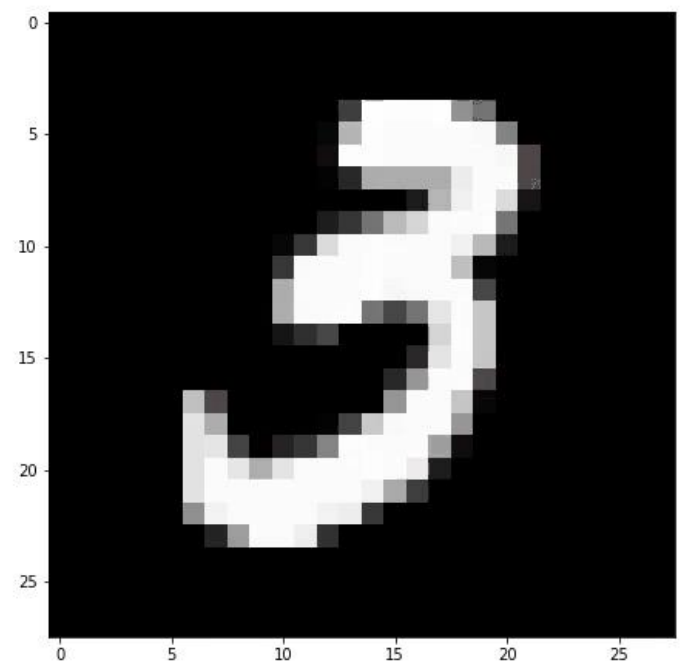
The learning dynamic is described by a plasticity rule developed by Krotov and Hopfield.



$$\tau_w \frac{d\boldsymbol{W}}{dt} = h\boldsymbol{v}$$

Visualization 1° Phase

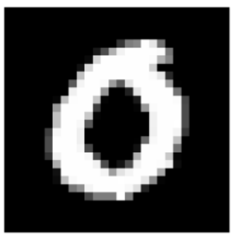


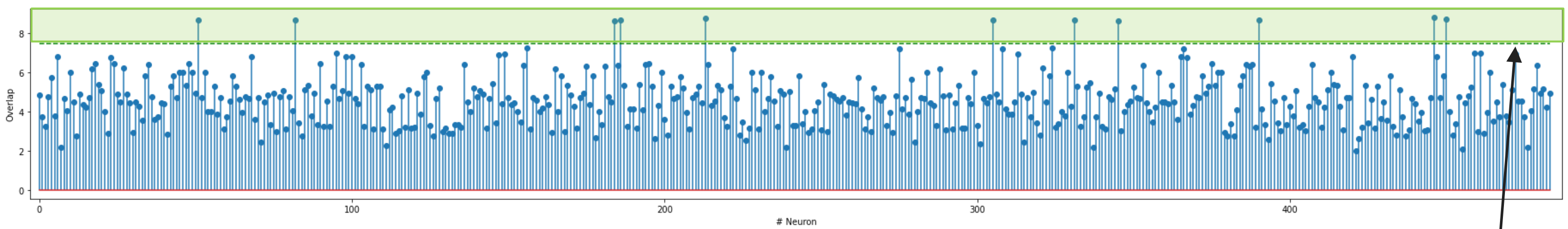


Super-active Neurons

A network trained with the algorithm proposed by Krotov and Hopfield **is not adversarial robust**!
The second layer is trained with a new algorithm proposed by me.

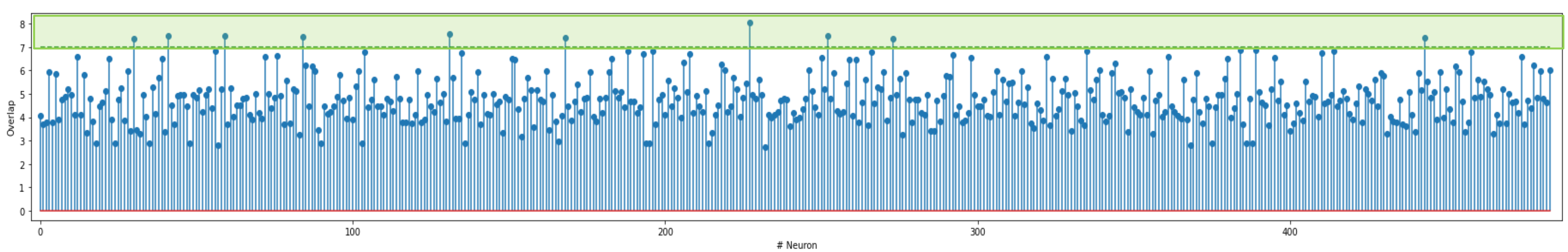
Turns out that exists **«super-active» hidden neurons** to some type of digits. They are defined as the top 5% neurons with the higher activation value.





Top active hidden neurons

Fig. Two examples of activation profile in the hidden layer.



Second Layer

The **matrix of weights W^1 is fixed**, W^2 is initialized to 0.

The train is performed in this way:

1. Give in input an image.
2. The top hidden neurons are set to 1, the others to 0.
3. If a hidden neuron is active (e.g. A), then the connection between neuron A and the output neuron corresponding to the label 0 (D) should be increased.
4. Do steps 1-2-3 for every test-image
5. The weights are properly normalized.

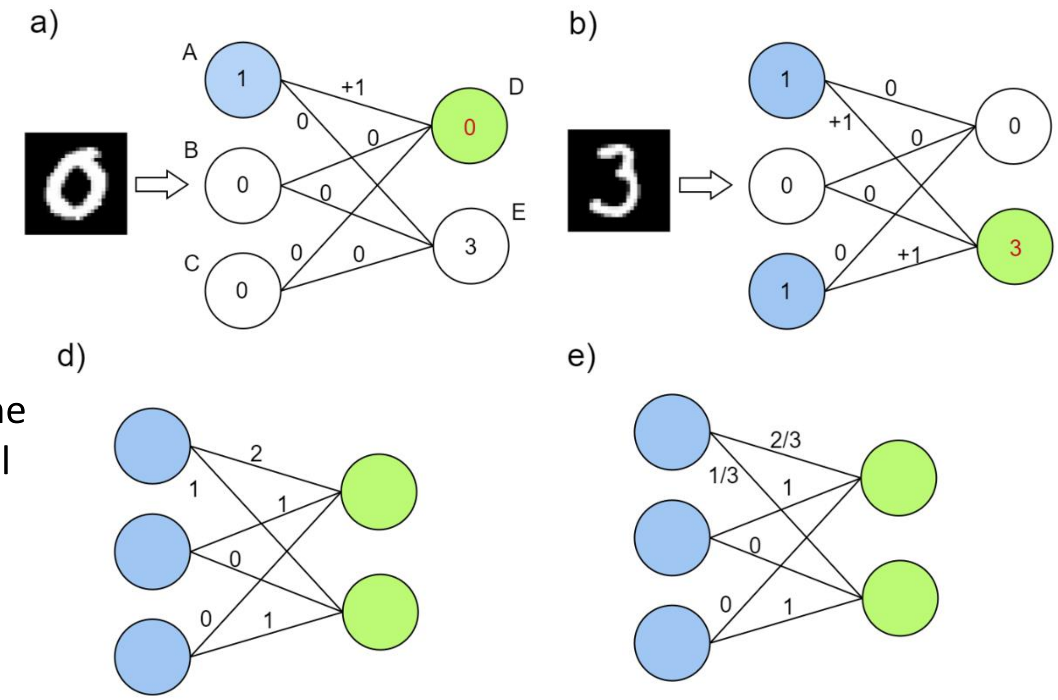


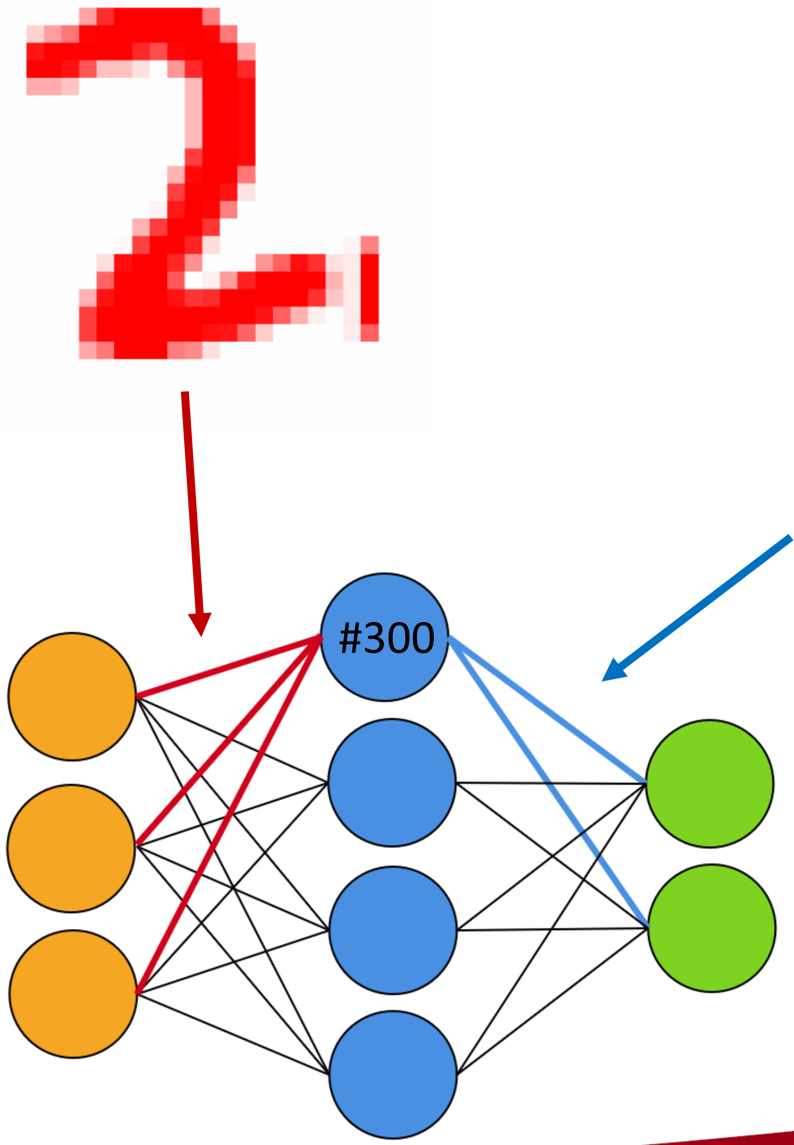
Fig. Illustrated training procedure.

$$w_{AD} = \frac{\#\ times\ A\ and\ D\ are\ both\ active}{\#\ times\ A\ is\ active} = P(D|A)$$

$$P(A) = \frac{\#\ times\ A\ is\ active}{\#\ images\ presented\ in\ input}$$

$$P(D) = \sum_i P(D|i)P(i) \quad i \in \{A, B, \dots\}$$

Visualize W^2



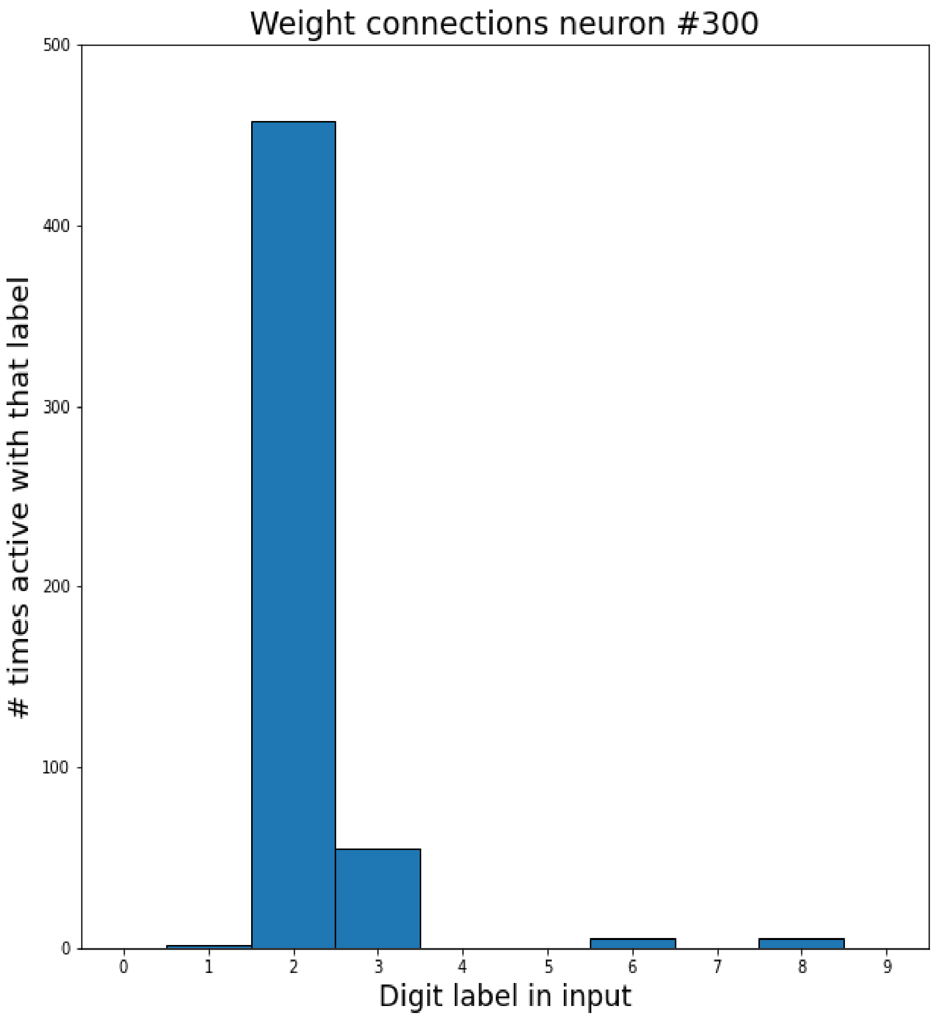


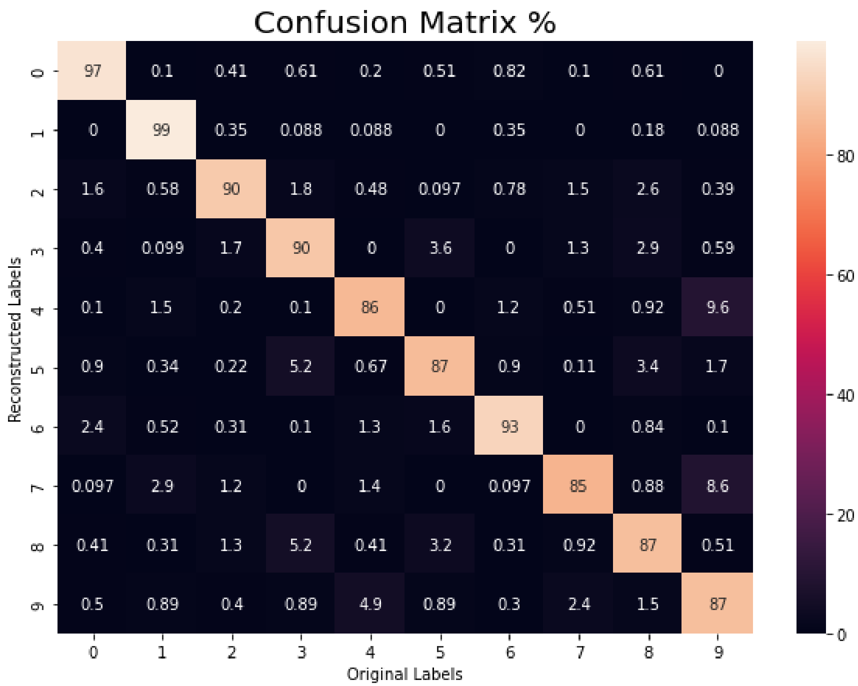
Fig. Visualization of counts for neuron n°300.

Results

UNIVERSITÀ DEGLI STUDI DI PADOVA

Main drawbacks:

- Not an excellent test accuracy, 90.16% (97% with BP).
- Slow training procedure 10min (2min with BP).



Main advantages:

- Network is transparent.
- Adversarial robustness is greatly improved.
- Network robust respect many types of attacks like BIM, PGD, FFGSM and others.

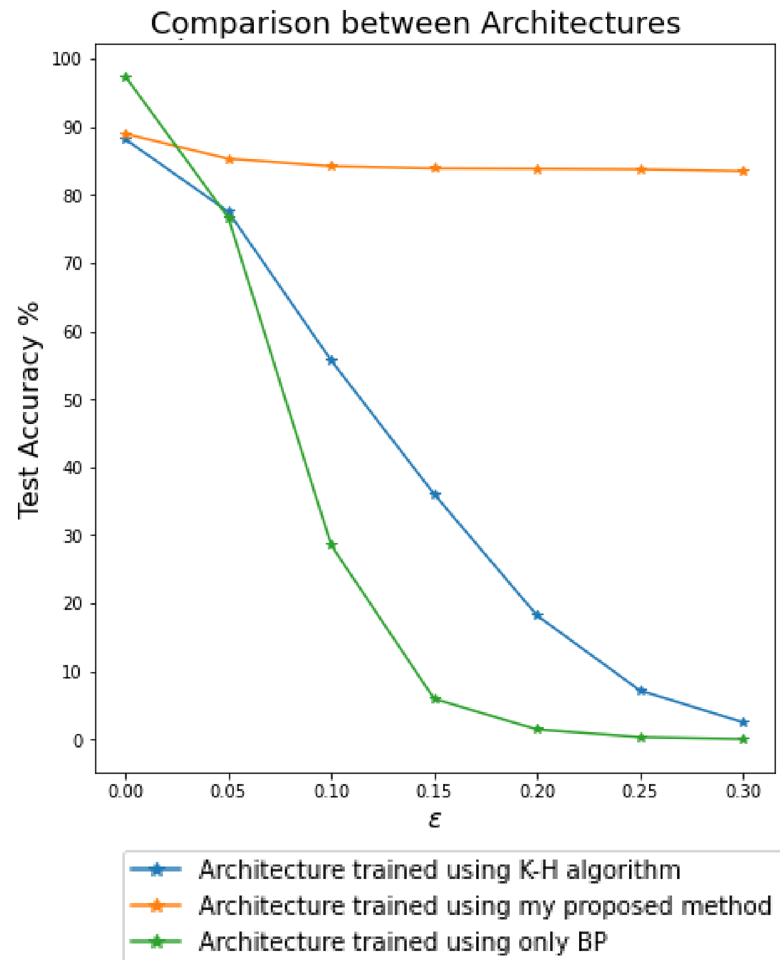


Fig. Comparison between training procedures.

Conclusions

Nowadays, machine learning would be applied in every sector of the market: finance, health-care, cooking instruments, smartphone, cyber-security and many others.

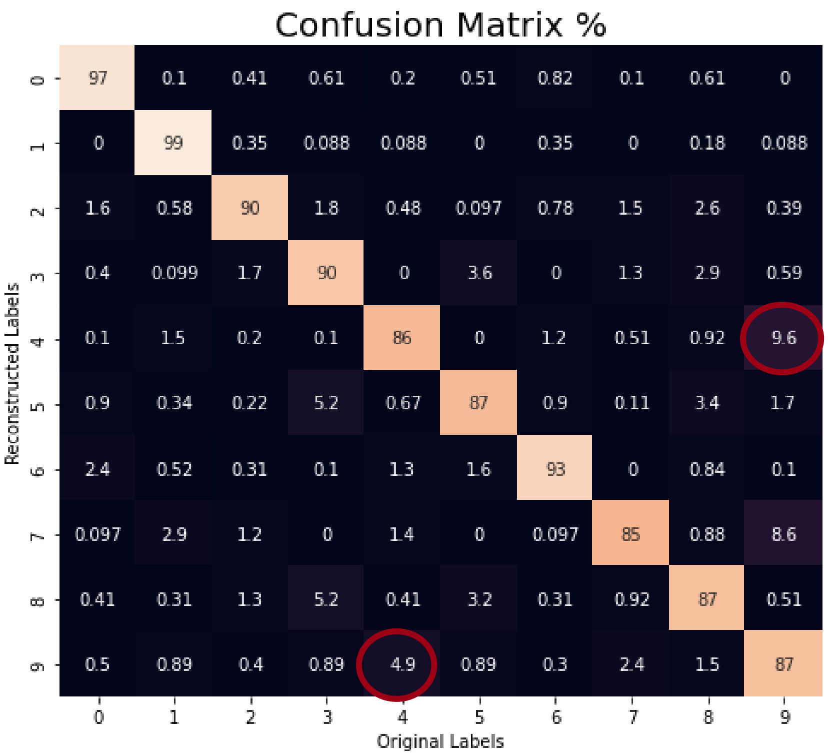
Lot of companies and investors are investing money in developing ML algorithm. The problem is that ML gives right answers only to right questions.

Assume a ML algorithm capable to give almost always the right answers to your problem: if you don't have trust in it, all the ML framework is pointless.

The final philosophical take-home message is: frame your problem from the human point of view. Indeed, the decisions suggested by a decisional model are made **for humans**!

Thank you for your attention!

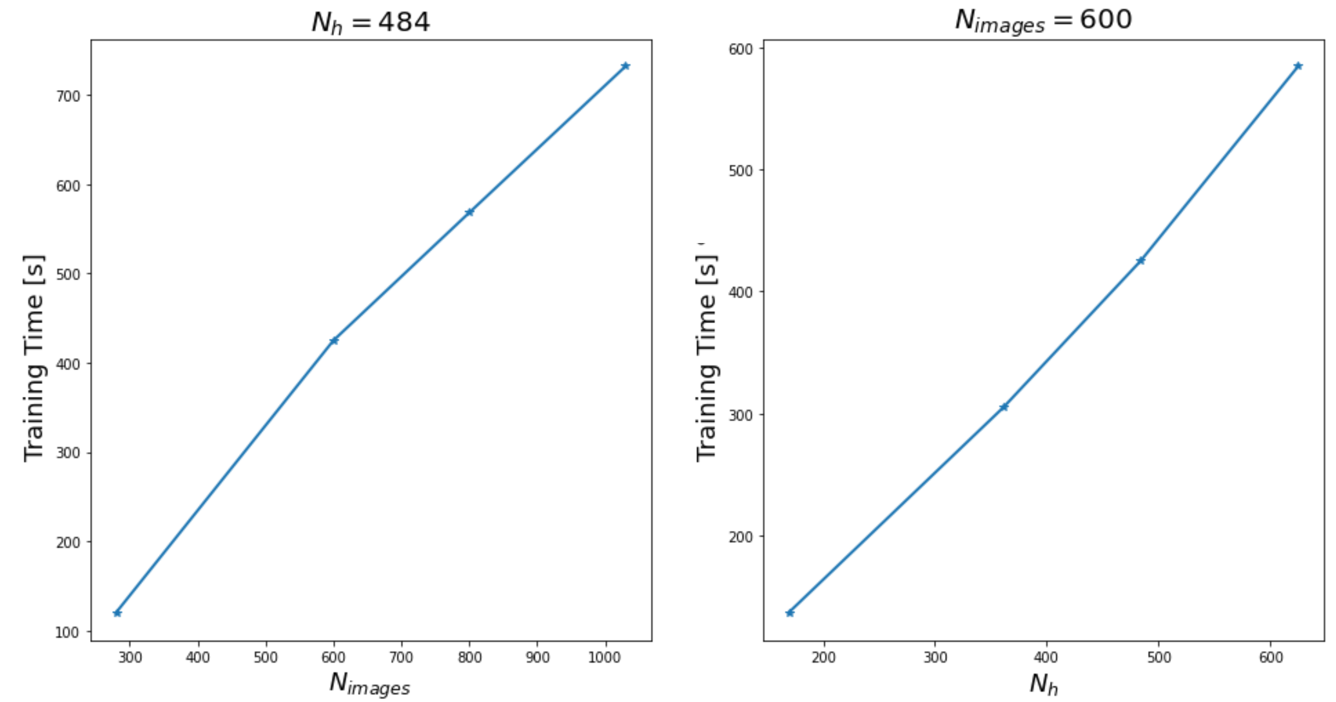
Low Test-Accuracy Explanation



Training Time

UNIVERSITÀ DEGLI STUDI DI PADOVA

N_{images} | $N_{hidden\ n.}$ = 484	Training time [s]	$N_{hidden\ n.}$ | N_{images} = 600	Training time [s]
280	199.8	169	136.9
600	424.9	361	304.8
800	568.2	484	424.9
1030	732.2	625	584.3



Feature Maps Comparison

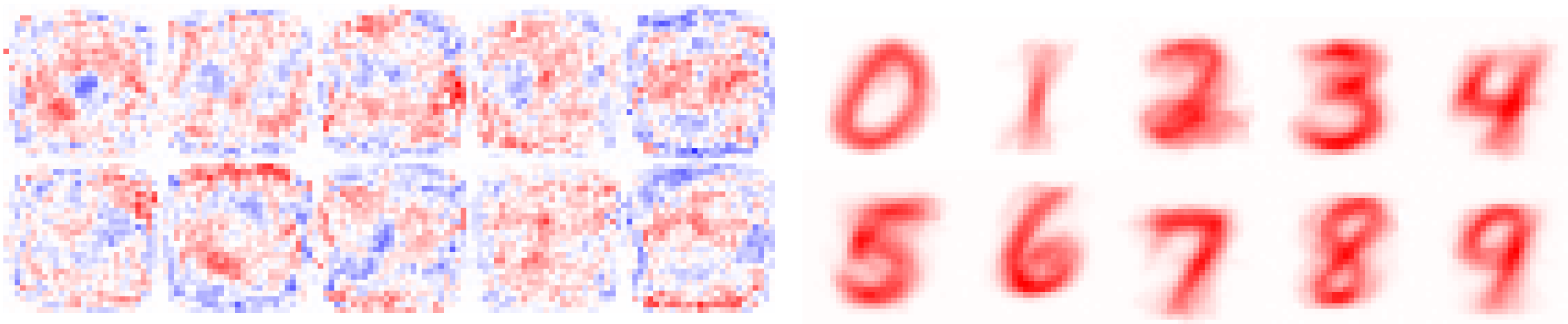


Fig. Comparison between feature maps of the second layer.

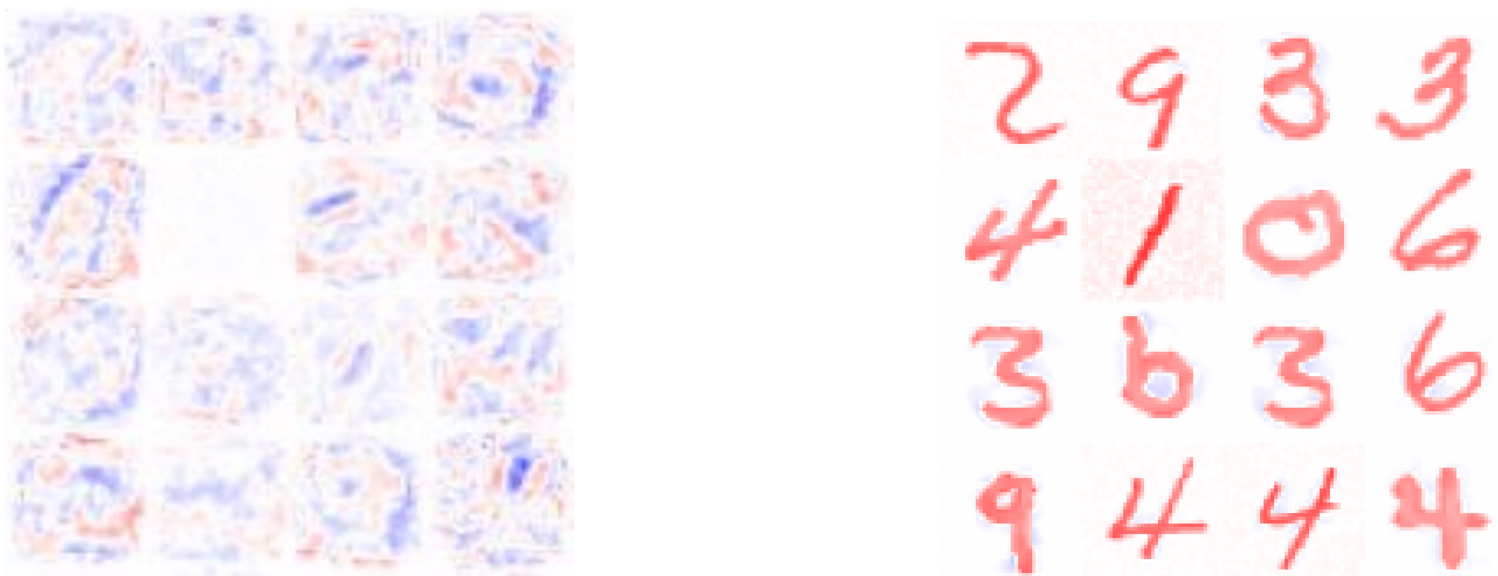
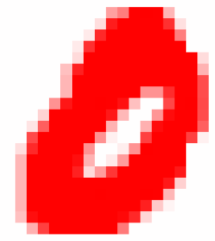
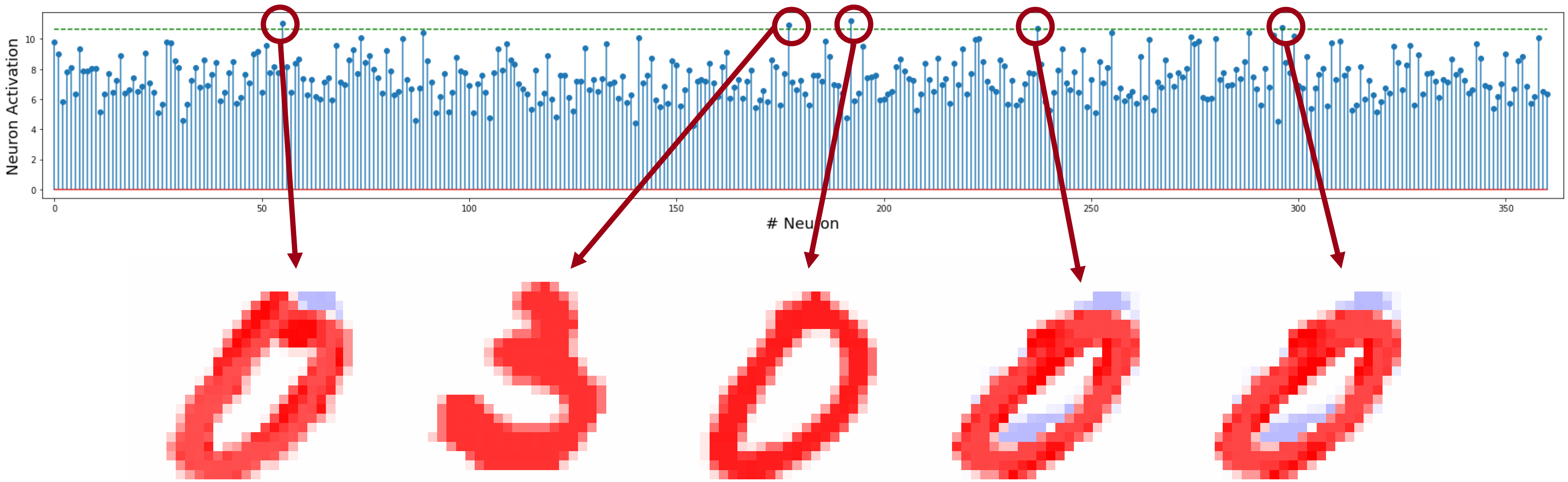


Fig. Comparison between some feature maps of the first layer.

Interpretation Second Layer

Input Image





Feature maps of the top-hidden neurons

Biological ODE

Within-layer competition between neurons

ReLU activation function

$$\boxed{\tau_r}\frac{dh_\mu}{dt} = I_\mu - \boxed{w_{inh}}\sum_{\nu\neq\mu}\boxed{r(h_\nu)} - h_\mu$$

Characteristic time scale of the input dynamic

v-equation of the firing rate model

$$\tau_h\frac{dh_\mu}{dt} = \Psi(I_\mu) - h_\mu$$

v-equation of the firing rate model

Activation function determining the learning paradigm

No-learning

Anti-Hebbian Regime

$$g(h) = \begin{cases} 0 \ if \ h < 0 \\ -\Delta \ if \ 0 < h < h^* \\ 1 \ if \ h^* < h \end{cases}$$

Hebbian Regime

$$\boxed{\tau_w}\frac{dw_{\mu i}}{dt} = \boxed{g(h_\mu)}\left[v_i - \left(\sum_{k=1}^{784} w_{\mu k}\, v_k\right)w_{\mu i}\right]$$

Characteristic time scale of the weights dynamic

Normalization Constraint