Selected topics in Artificial Intelligence

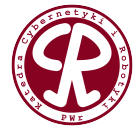
Deep leaning problems

Wojciech Domski

Chair of Cybernetics and Robotics,
Wrocław University of Science and Technology

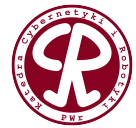
Presentation compiled for taking notes during lecture

Wrocław University
of Science and Technology



Outline

Wrocław University
of Science and Technology



Learning process

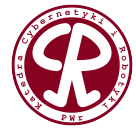
Wrocław University
of Science and Technology

Dataset division (1/3)

When learning of a model is being considered not a small part is played by the dataset. It allows to train the model and asses how well it performs. To achieve it the dataset has to be divided into three separate parts:

- training,
- validation,
- test.

Wrocław University of Science and Technology



Dataset division (2/3)

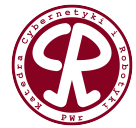
Training dataset

This part of data set is the largest one. Usually it holds around 70% of all samples. It is directly involved in the learning process.

Validation dataset

Similarly to training dataset it is involved in learning process. However, not directly. It is used to internally validate the model against loss function. The size of the dataset is usually around 20% of the entire dataset.

Wrocław University of Science and Technology

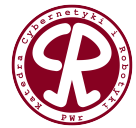


Dataset division (3/3)

Test dataset

The third part of initial dataset. The model is neither trained on samples coming from the this dataset nor validated. The purpose of this part is to test a trained model. It allows to estimate how well model was trained and evaluate it on data which has not been previously seen. The size of this part is the smallest one and is around 10%.

Wrocław University of Science and Technology



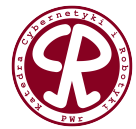
Training

During training process constantly two factors are evaluated: **loss** function and a **metric** e.g. accuracy.
In order to our model perform well it has to achieve both:

- small training error,
- minimize gap between training and test error.

The above are directly related to two challenges of machine learning, thus deep learning – overfitting and underfitting.

Wrocław University of Science and Technology



Over- and underfitting

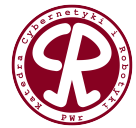
Wrocław University
of Science and Technology

Overfitting (1/4)

Overfitting is a problem which can be described as a tendency of a model to learn the dataset instead of learning the relationships between features and labels. This can happen due to following reasons:

- the model is too complex in respect to size of dataset,
- the data set is noisy,
- the data set is too small.

Wrocław University of Science and Technology



Overfitting (2/4)

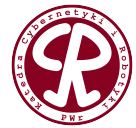
Overfitting can be solved using following approaches:

complexity reduction instead of using deep neural network a shallower can be used or maybe even a polynomial or a linear function can be used instead, also constraining space of solutions is possibility,

increasing the dataset usually it is related to collecting more representative samples,

reducing noise in the dataset the dataset has to be filtered, gaps removed, and exceeding signals limited.

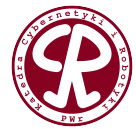
Wrocław University of Science and Technology



Overfitting (3/4)

Regularization is a technique which allows to constrain our model without interfering with the dataset. An example of regularization technique is dropout layer which randomly with a given probability drops signals. Thanks to this the model tries to disperse information over whole network instead of traversing through known paths.
It was shown that using a dropout layer can reduce overfitting of a model. With relatively small values of dropout the model is underfitting. However, when the dropout is kept between $p \in (0.4, 0.8)$ it has positive effect on learning process. Only when the p reaches to 1 the testing error is increasing.

Wrocław University
of Science and Technology



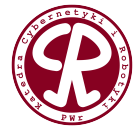
Overfitting (4/4)

Another technique can help overcome overfitting is called batch normalization. It allows to adaptively reparametrize input to network's layer.

$$H' = \frac{H - \mu}{\sigma} \tag{1}$$

where H is initial mini-batch.

Wrocław University of Science and Technology



Underfitting (1/2)

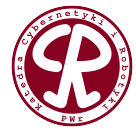
Underfitting is opposite to overfitting. The trained model is not able to grasp the complexity of observed process. This can be caused by:

model simplicity the given model is to simple to represent the given problem,

unrepresentative data collected data does not correlate with the given process,

model is overconstrained too many regularization techniques were used.

Wrocław University of Science and Technology

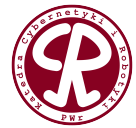


Underfitting (2/2)

The underfitting problem can be solved with following approaches:

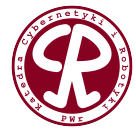
- increasing complexity of a model by introducing more parameters e.g. a new layer or more complex layers with more parameters,
- dataset optimization – feature engineering which tries to find more representative features that describe the problem or relation better,
- relaxation of constraints put on the model – decreasing regularization parameters.

Wrocław University of Science and Technology



Vanishing and exploding gradients

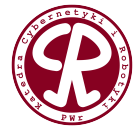
Wrocław University
of Science and Technology



Vanishing gradients

It is a phenomenon when a neural network is being trained and due to the gradients getting smaller and smaller the network does not learn (adjust weights).
All methods which have gradient calculation at their basis are affected. During the process of learning through backpropagation weights in layers are updated starting from output layer up to the input layer. At each step a gradient is being calculated in order to adjust the weights. However, the gradients can get smaller and smaller with every layer. This on the other hand prevents adjustment of weights, thus convergence and learning.

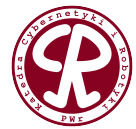
Wrocław University
of Science and Technology



Exploding gradients

Exploding gradient is a adversarial phenomenon to vanishing gradient. Through the course of backpropagation gradients are increasing exponentially.
This leads to large numerical values which can easily accumulate. In turn, large values which are outside of numerical value representation lead to NaN values.

Wrocław University
of Science and Technology

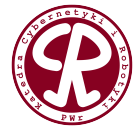


Root cause (1/3)

The main cause for gradient issues can be related to:

- saturation functions such as sigmoid,
- initial values of weights.

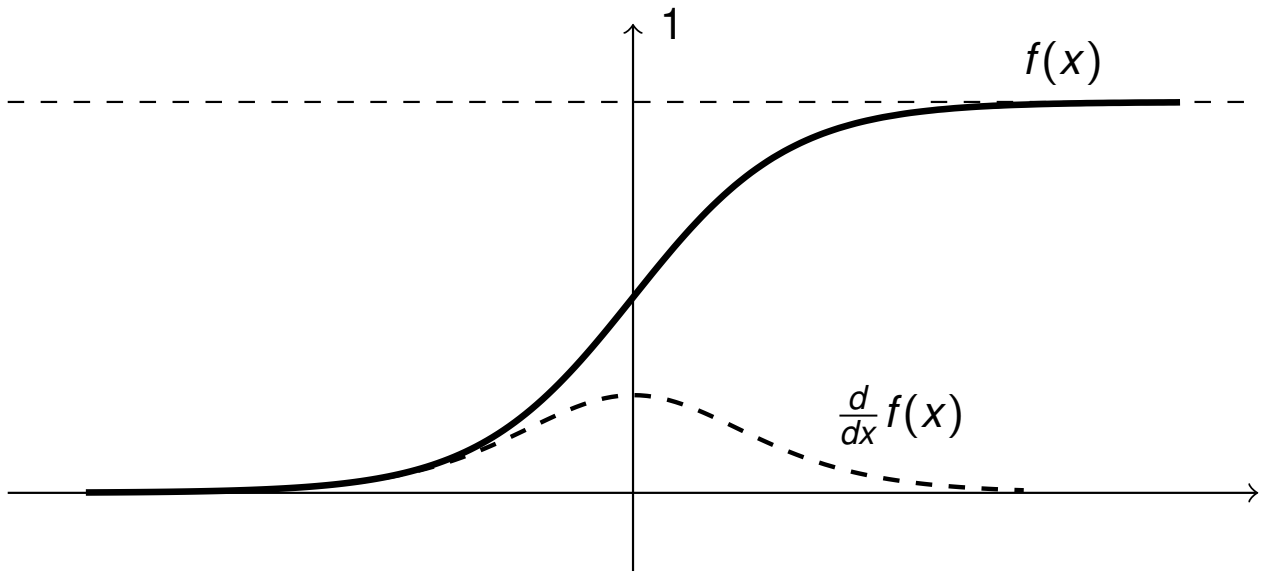
Wrocław University
of Science and Technology



Root cause (2/3)

Sigmoid function (logistic function)

$$f(x) = \frac{1}{1 + e^{-x}}, \qquad \frac{d}{dx} f(x) = \frac{e^{-x}}{(1 + e^{-x})^2}$$



Root cause (3/3)

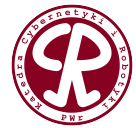
Gradient for sigmoid function converges to 0 for very big positive or negative values.

$$\lim_{x \to +\text{ inf}} \frac{d}{dx} f(x) = 0$$

$$\lim_{x \to -\text{ inf}} \frac{d}{dx} f(x) = 0$$

Therefore for substantial values of errors the gradient is close to zero.

Wrocław University of Science and Technology

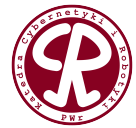


Solutions

There are several methods which can mitigate the gradient related problems:

- proper weight initialization,
- non-saturation activation layers,
- batch normalization,
- gradient saturation.

Wrocław University of Science and Technology



Weight initialization (1/2)

Currently, there is little information on how the initialization influences learning process. However, it is know that the identical initialization to adjacent units with the same inputs is not advised since the correction will affect the unit in the same degree.

Wrocław University of Science and Technology

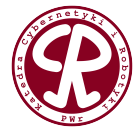
Weight initialization (2/2)

There are multiple techniques for parameter initialization. It was proposed by Glorot and Bengio that initial weights of DNN should be normalized through initialization from uniform distribution

$$U\left(-\sqrt{\frac{6}{m+n}}, +\sqrt{\frac{6}{m+n}}\right) \tag{2}$$

where m designates number of inputs and n number of outputs. This approach allows to have the same gradient variance across network. Although, this technique is used for networks containing only matrix multiplication and no nonlinearities.

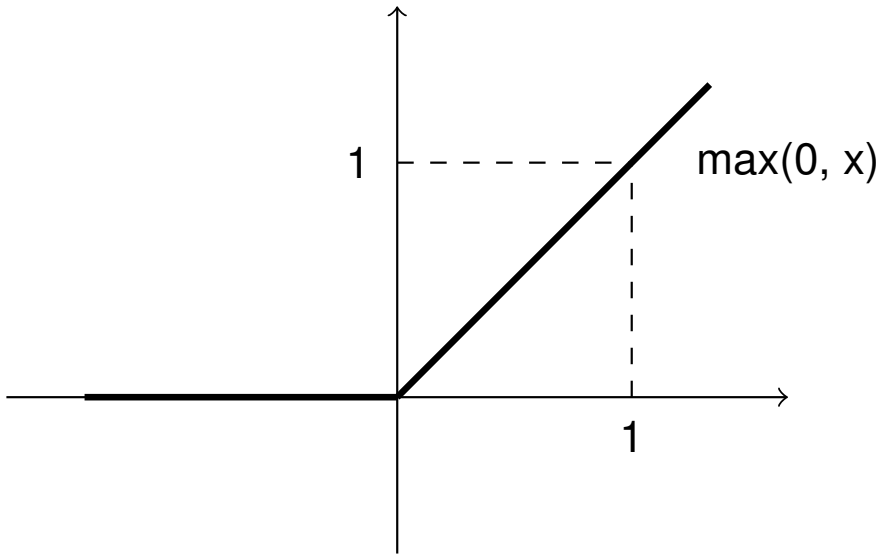
Wrocław University of Science and Technology



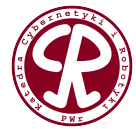
Nonsaturated activation functions (1/2)

One of the most popular non saturated activation function is ReLU (max function with 0).

$$f(x) = \begin{cases} x, & \text{when } x \geq 0, \\ 0, & \text{when } x < 0. \end{cases} \tag{3}$$



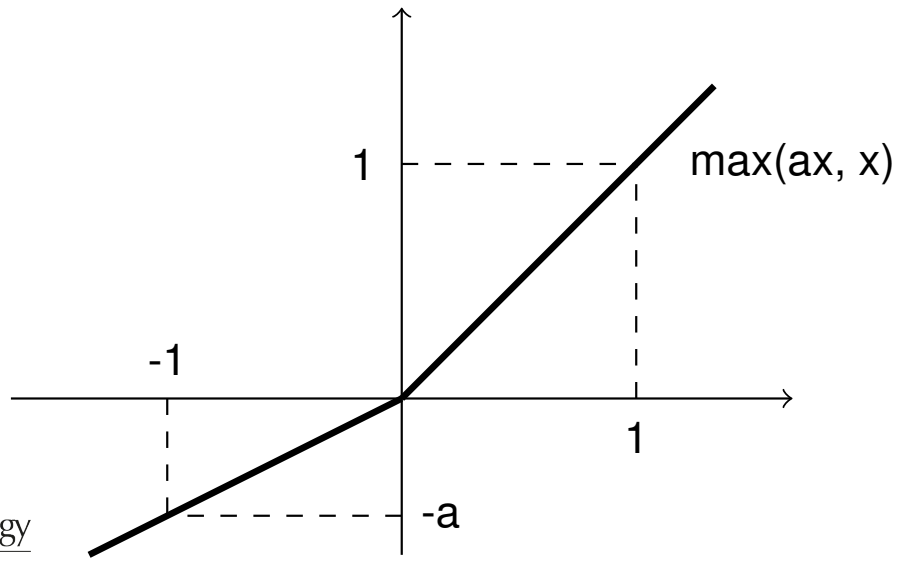
Wrocław University of Science and Technology



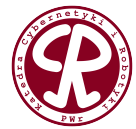
Nonsaturated activation functions (2/2)

However, this function can suffer from so-called dying ReLUs where some neurons produce only 0 values during the training. To avail this behaviour some alternatives have emerged like LReLU (Leaky ReLU)

$$f(x) = \begin{cases} x, & \texttt{when}\, x \geq 0, \\ ax, & \texttt{when}\, x < 0. \end{cases} \tag{4}$$



Wrocław University of Science and Technology

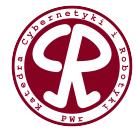


Batch normalization (1/1)

Batch normalization (1) was introduced as a technique to overcome overfitting of a model, however its prime purpose was to overcome issue with vanishing/exploding gradients.
It works on each mini-batch passed to the model during training and it calculates two parameters offset μ and scaling factor σ.
Thanks to this the batch normalization allows for zero-centering and normalization of each mini-batch.

It should be applied before or after activation function for each layer. The additional computational effort exists but it is not substantial.

Wrocław University of Science and Technology



Gradient saturation (1/1)

It allows to ease problems related to exploding gradients. It can be achieved by applying clipping to calculated gradient between given value, e.g. $[-1, 1]$.

Wrocław University
of Science and Technology