INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

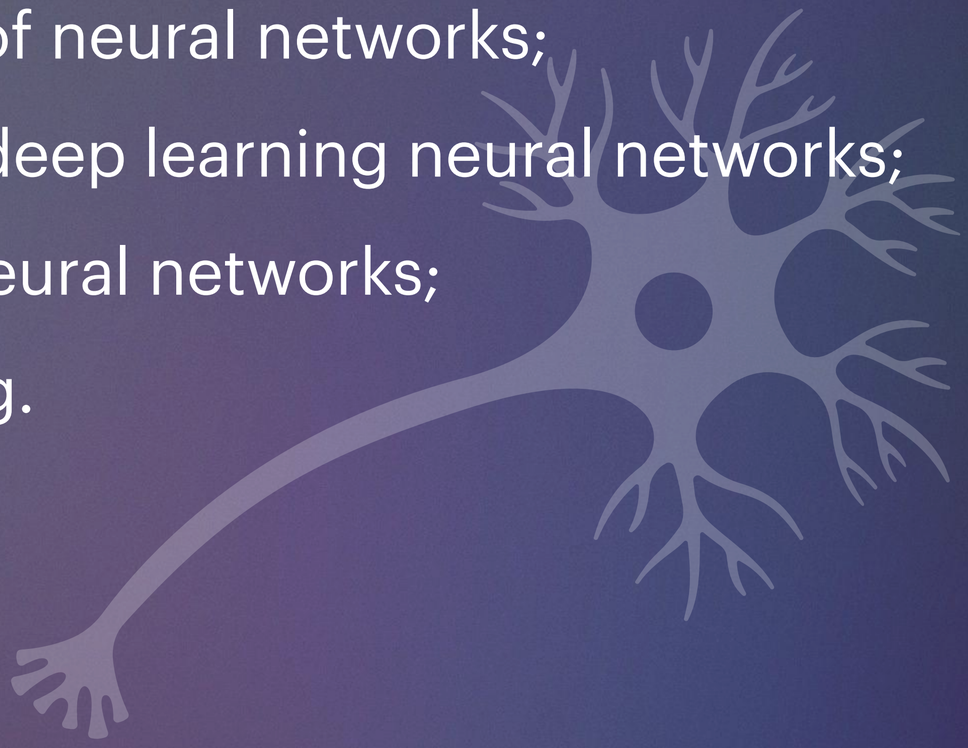
Activation functions and Regularizers

Prof. Me. Saulo A. F. Oliveira
saulo.oliveira@ifce.edu.br

Objectives

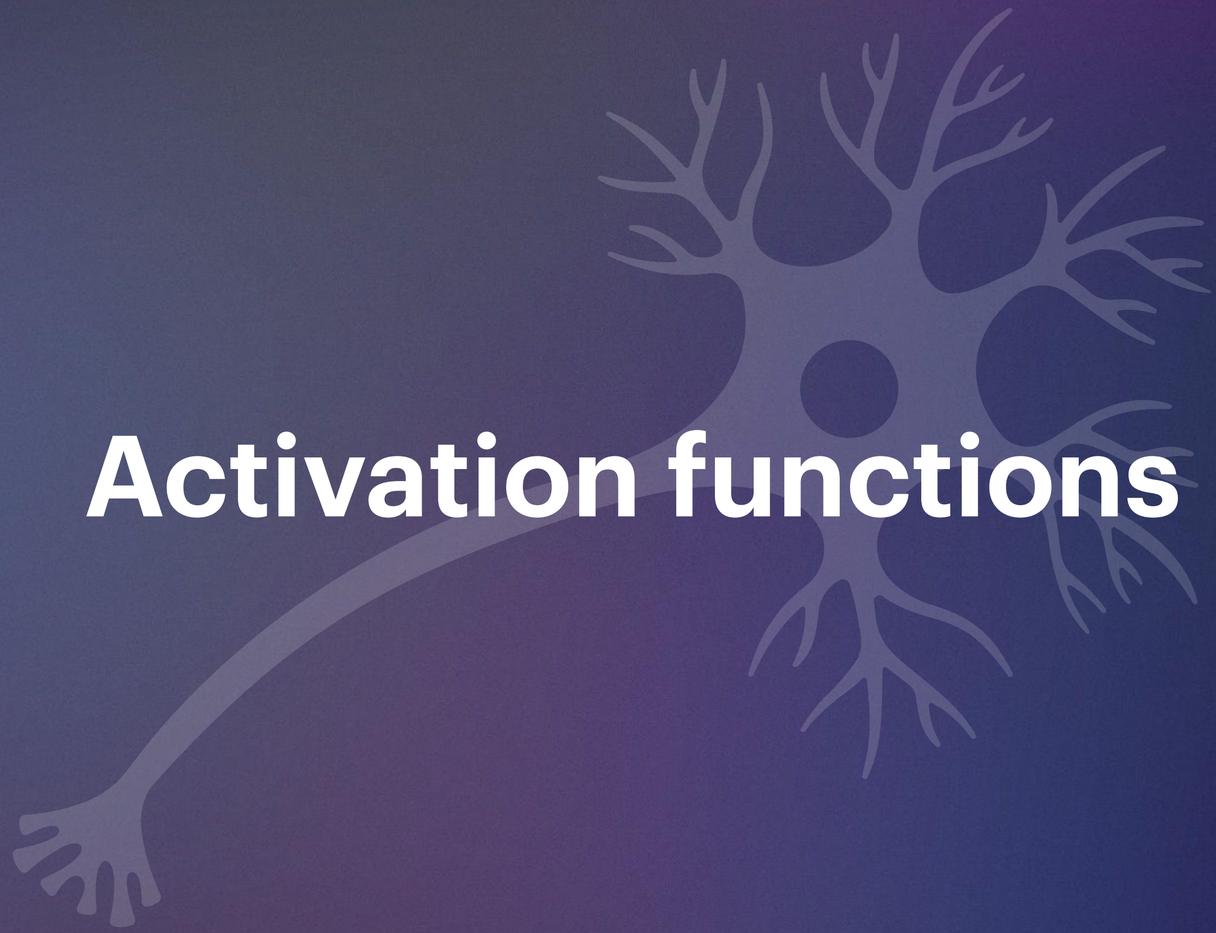
On completion of this course, you will be able to:

- Describe the definition and development of neural networks;
- Learn about the essential components of deep learning neural networks;
- Understand training and optimization of neural networks;
- Describe typical problems in deep learning.



01

Activation functions

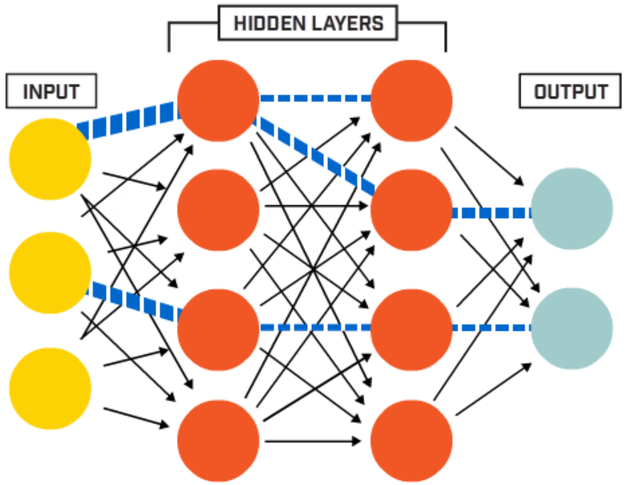


INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

Activation Function

- Activation functions are crucial for the neural network model to learn and understand complex non-linear functions. They allow the introduction of non-linear features to the network.
- In a neural network, the activation function is responsible for transforming the summed weighted input from the node into the node's activation or output for that input.



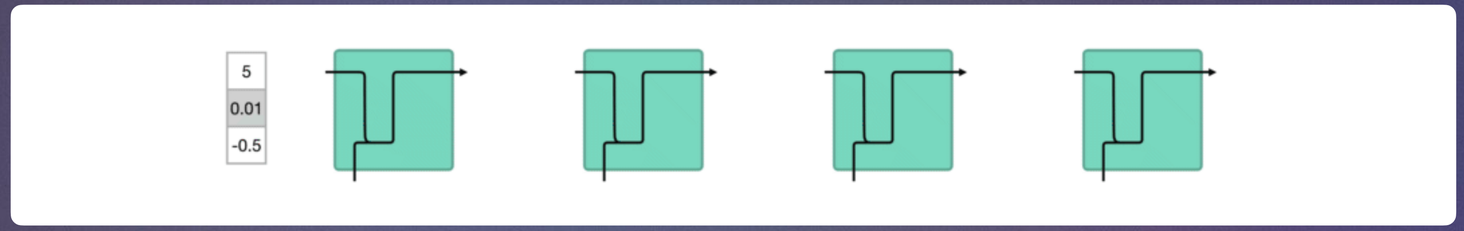
https://laptrinhx.com/introduction-to-neural-networks-and-their-key-elements-part-c-activation-functions-layers-1195957448/

- Without activation functions, output signals are only simple linear functions. The complexity of linear functions is limited, and the capability of learning complex function mappings from data is low.

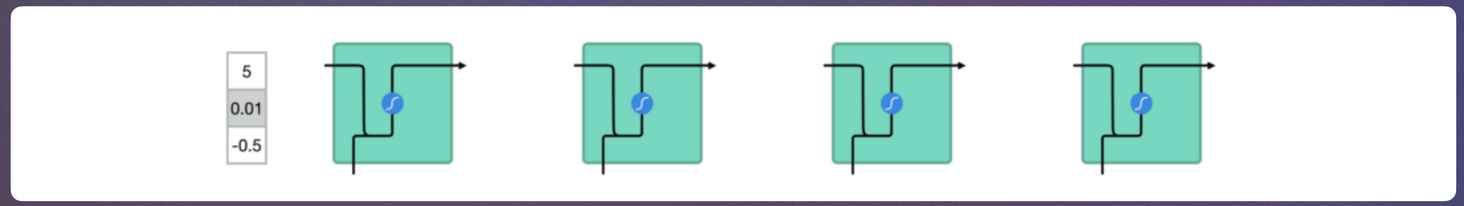
$$\text{output} = f\left(w_1 x_1 + w_2 x_2 + \ldots, w_n x_n + \text{bias}\right) = f\left(\mathbf{w}^\intercal \mathbf{x} + \text{bias}\right)$$

Activation Function

- When vectors are flowing through a neural network, it undergoes many transformations due to various math operations. So imagine a value that continues to be multiplied by let's say 3. You can see how some values can explode and become astronomical, causing other values to seem insignificant.



- For example, a $\tanh$ function ensures that the values stay between -1 and $+1$, thus regulating the neural network's output. One can see how the same values from above remain between the boundaries allowed by the tanh function.



https://laptrinhx.com/introduction-to-neural-networks-and-their-key-elements-part-c-activation-functions-layers-1195957448/

Notorious Activation Functions

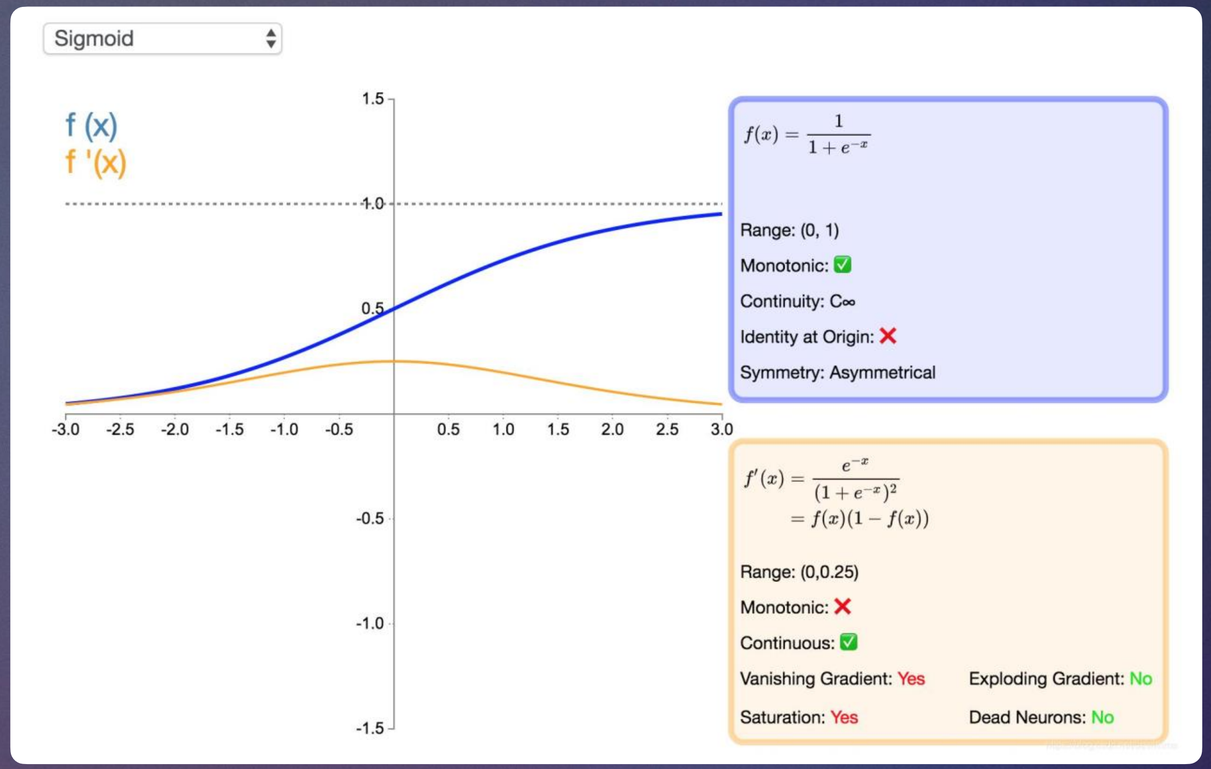
Sigmoid

$$\text{sigmoid}\,(x) = \frac{1}{1 + \exp(-x)}$$

A sigmoid activation is similar to the $\tanh$ activation. Instead of squishing values between -1 and $+1$, it squishes values between 0 and 1. That helps update data because any number getting multiplied by 0 is 0, causing values to disappear or be “forgotten.”

Any number multiplied by 1 is the same value; therefore, that value stays the same or is “kept.” The network can learn which data is not important; therefore, it can be forgotten or kept.

The neurons that have values near zero will have less impact, and the neurons that produce values near 1 will have more impact.



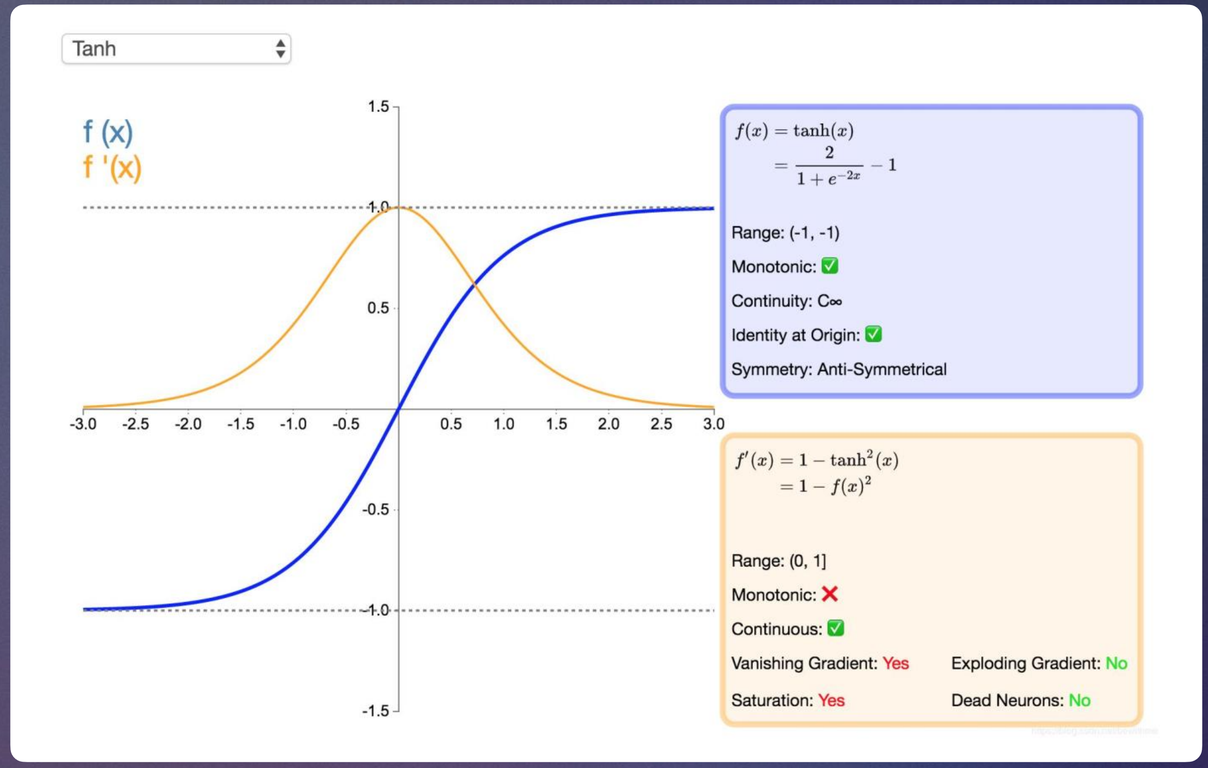
https://laptrinhx.com/introduction-to-neural-networks-and-their-key-elements-part-c-activation-functions-layers-1195957448/

Notorious Activation Functions

Tanh

The tanh activation is used to help regulate the values flowing through the network. The tanh function squishes values to always be between -1 and $+1$.

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$



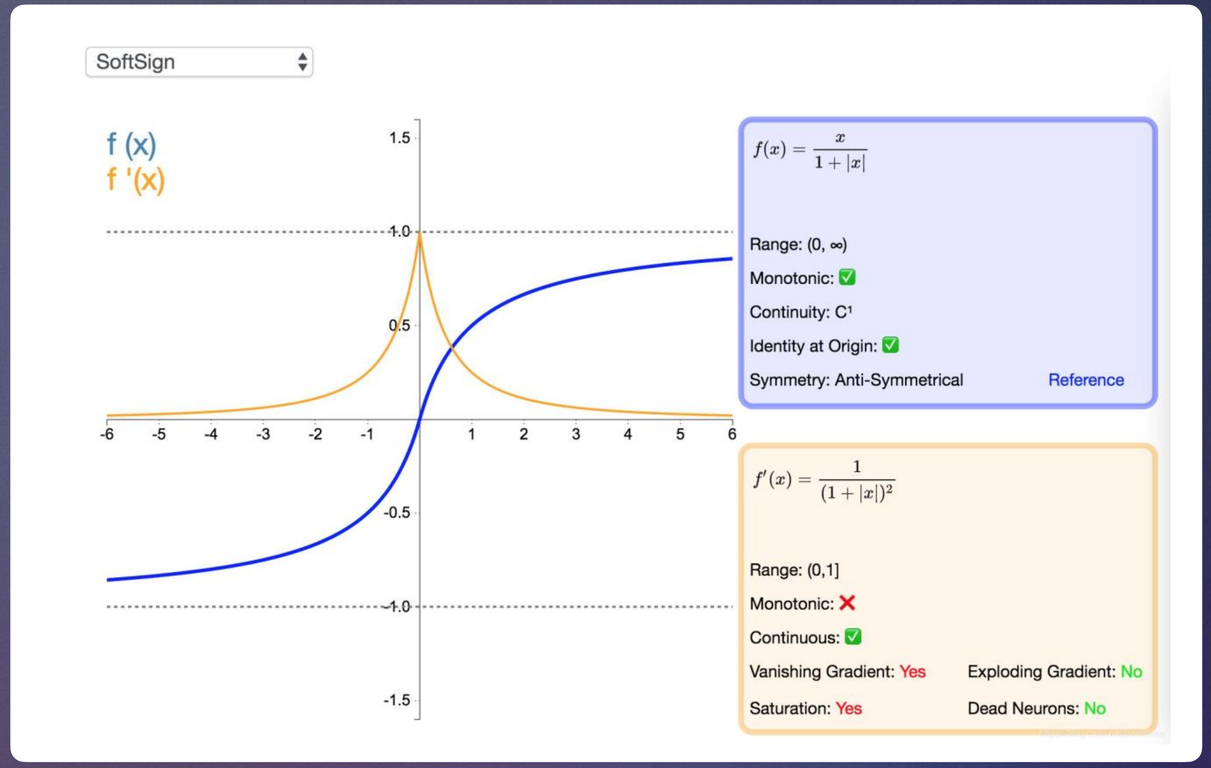
https://laptrinhx.com/introduction-to-neural-networks-and-their-key-elements-part-c-activation-functions-layers-1195957448/

Notorious Activation Functions

SoftSign

The Softsign function is an activation function which rescales the values between -1 and $+1$ by applying a threshold just like a sigmoid function. The advantage, that is, the value is zero-centered which helps the next neuron during propagating.

$$\text{SoftSign}\,(x) = \frac{x}{1 + |x|}$$



https://analyticsindiamag.com/the-number-game-behind-advanced-activation-functions-in-machine-learning/

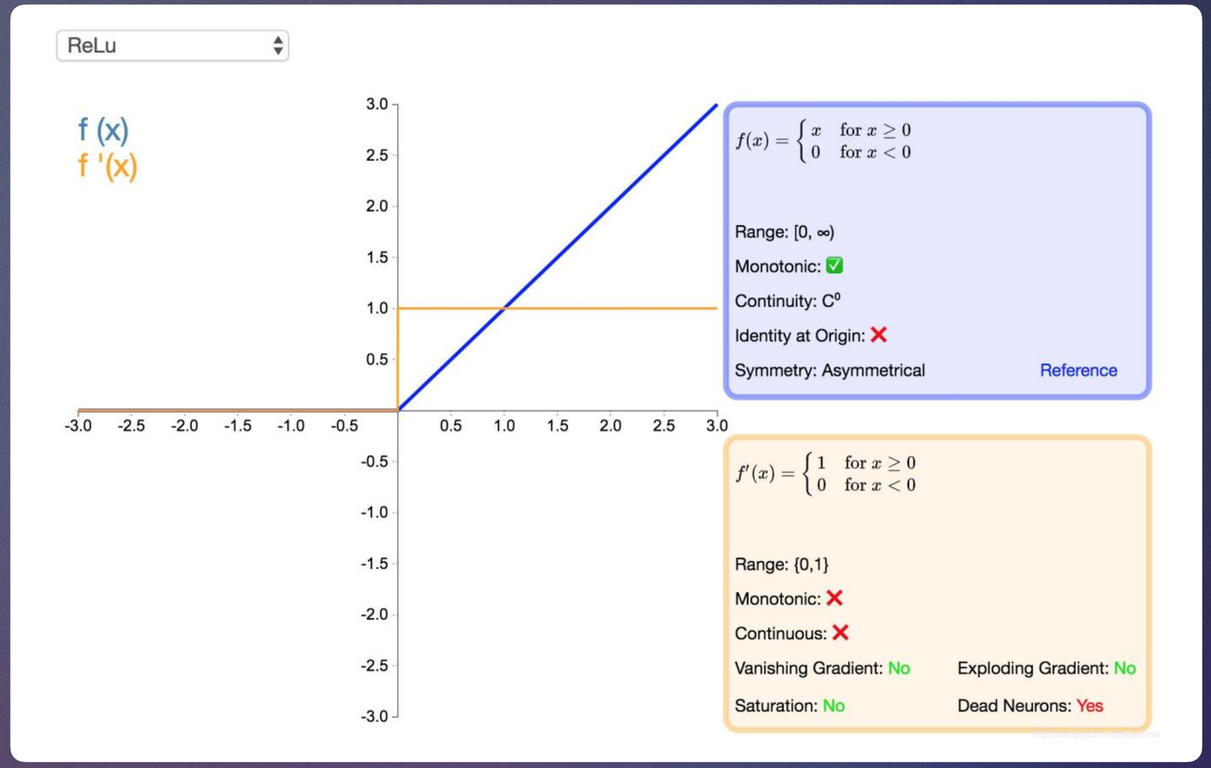
Notorious Activation Functions

Rectified Linear Unit (ReLU)

The ReLU is the most used activation function in the world right now. Since it is used in almost all the convolutional neural networks or deep learning. ReLU stands for a rectified linear unit. If you are unsure what activation function to use in your network, ReLU is usually a good first choice.

$$\text{ReLu}\,(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

A queridinha dos devs!



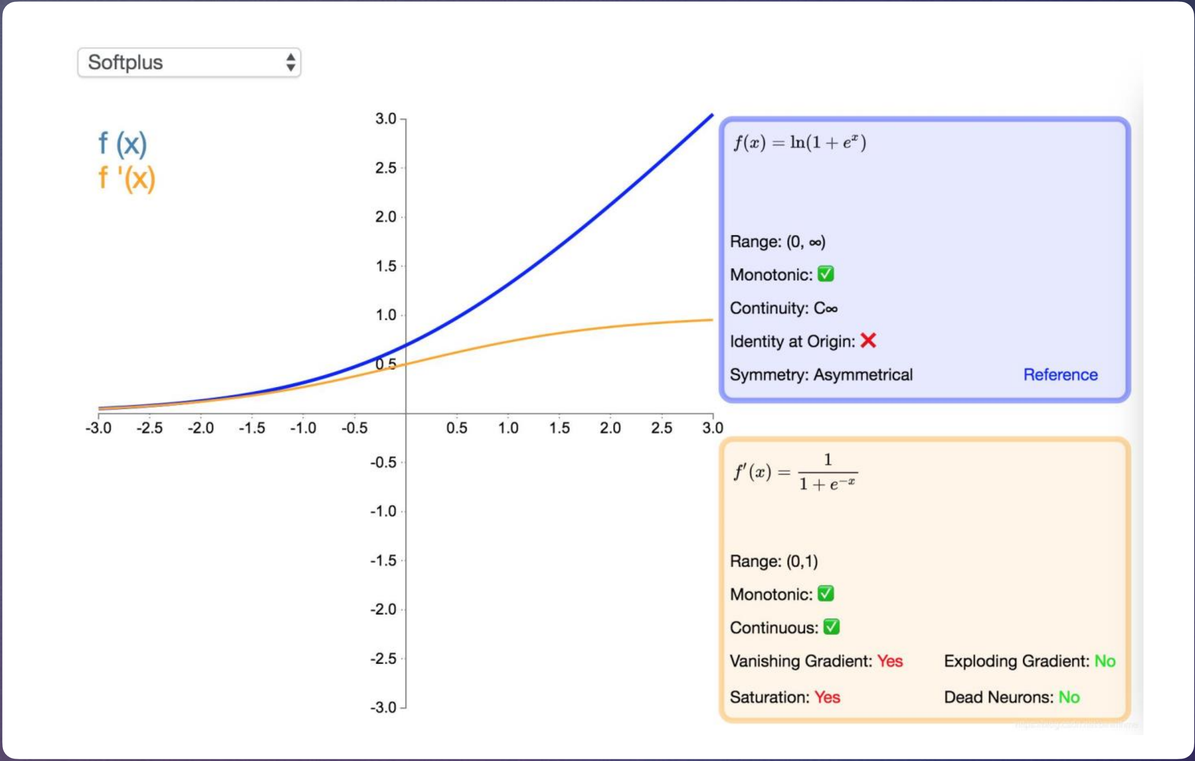
https://analyticsindiamag.com/the-number-game-behind-advanced-activation-functions-in-machine-learning/

Notorious Activation Functions

Softplus

Outputs produced by $\mathrm{sigmoid}$ and tanh functions have upper and lower limits, whereas $\mathrm{softplus}$ function produces outputs in the scale of $(0, +\infty)$. That is the essential difference. One might notice that the derivative is equal to sigmoid function. $\mathrm{Softplus}$ and $\mathrm{sigmoid}$ are like Russian dolls. They placed one inside another!

$$\mathrm{Softplus}\,(x) = \log\left(1 + \exp(x)\right)$$



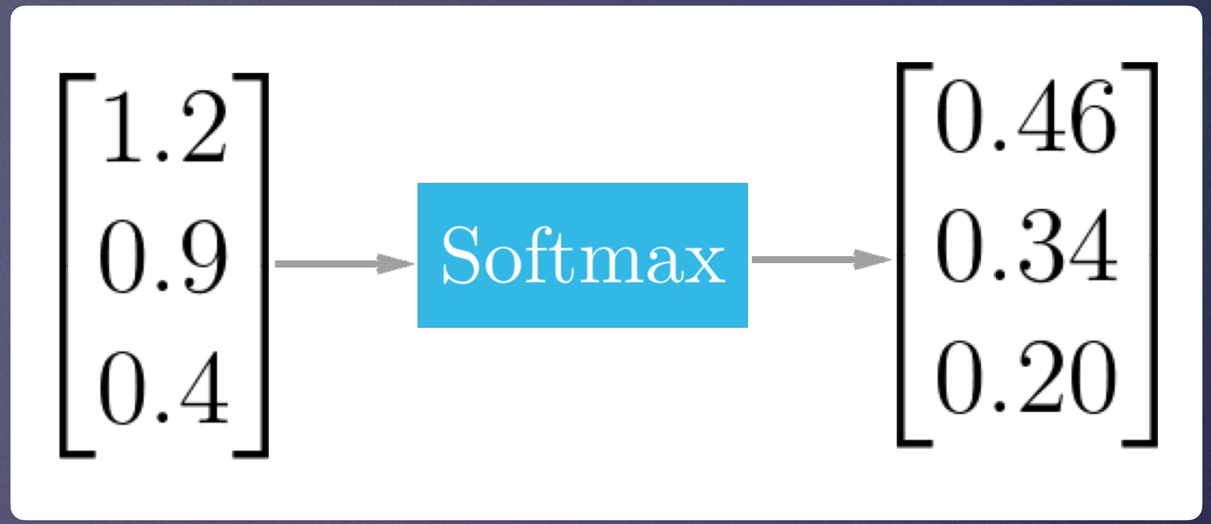
https://sefiks.com/2017/08/11/softplus-as-a-neural-networks-activation-function/

Notorious Activation Functions

Softmax

The $\mathrm{Softmax}$ function is used to map a K-dimensional vector of arbitrary real values to another K-dimensional vector of real values. Each vector element lies in the interval $(0, 1)$. All the elements sum up to 1. The $\mathrm{Softmax}$ function is often used as the output layer of a multi-class classification task.

$$\mathrm{Softmax}\,(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_j^K \exp(x_j)}$$



https://towardsdatascience.com/deep-learning-concepts-part-1-ea0b14b234c8

Notorious Activation Functions

Difference Between Sigmoid and Softmax functions

Softmax Function	Sigmoid Function
Used for multi-class classification in logistic regression model.	Used for binary classification in logistic regression model.
The probabilities sum will be 1.	The probabilities sum need not to be 1.
Used in different layers of neural networks.	Used as activation function while building neural networks.

https://towardsdatascience.com/deep-learning-concepts-part-1-ea0b14b234c8

02

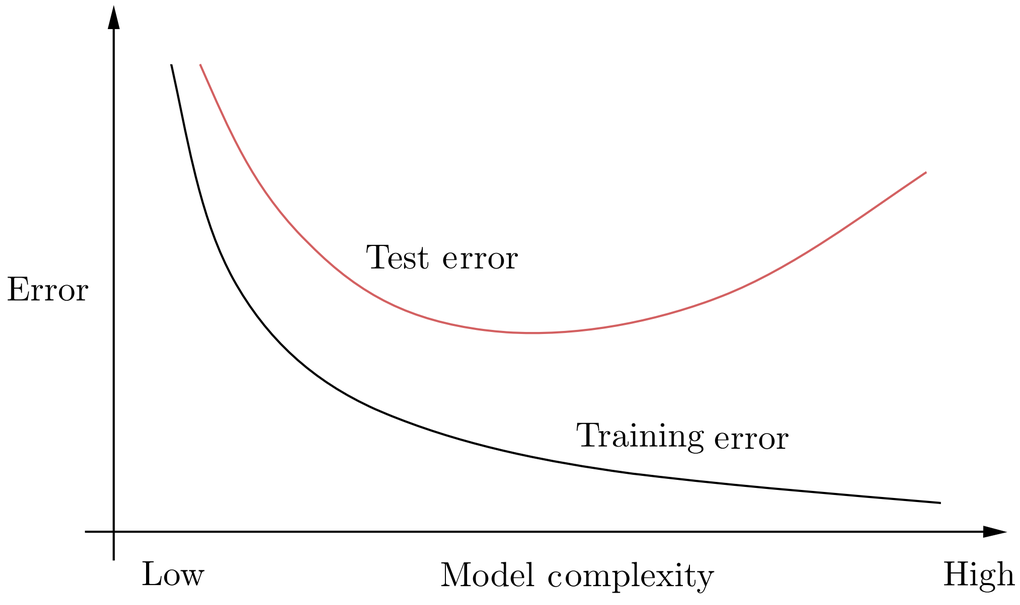
Controlling Complexity

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

Controlling Complexity in Neural Networks

- In describing the computational complexity of an algorithm, we are generally interested in the number of basic mathematical operations, such as additions, multiplications and divisions it requires, or in the time and memory needed on a computer.
- Although in the current context, the complexity issue emerges in a somewhat disguised form is usually defined in terms of the number of free parameters the model can learn, e.g., hidden units and layers and their connectivity, the form of activation functions, and free parameters of the learning algorithm itself.



THEODORIDIS, S. Machine Learning: A Bayesian and Optimization Perspective. Elsevier Science, 2020. ISBN 9780128188040.

- From this perspective, one can choose different functions and norms as terms in the model formulation to account such a complexity.

Controlling Complexity in Neural Networks

- Regularization is an important and effective technology to reduce generalization errors in machine learning. It is especially useful for deep learning models that tend to be over-fit due to a large number of parameters. Therefore, researchers have proposed many effective technologies to prevent over-fitting, including:
  - Adding constraints to parameters, such as L_1 and L_2 norms;
  - Expanding the training set, such as adding noise and transforming data;
  - Dropout;
  - Early stopping.

Regularization

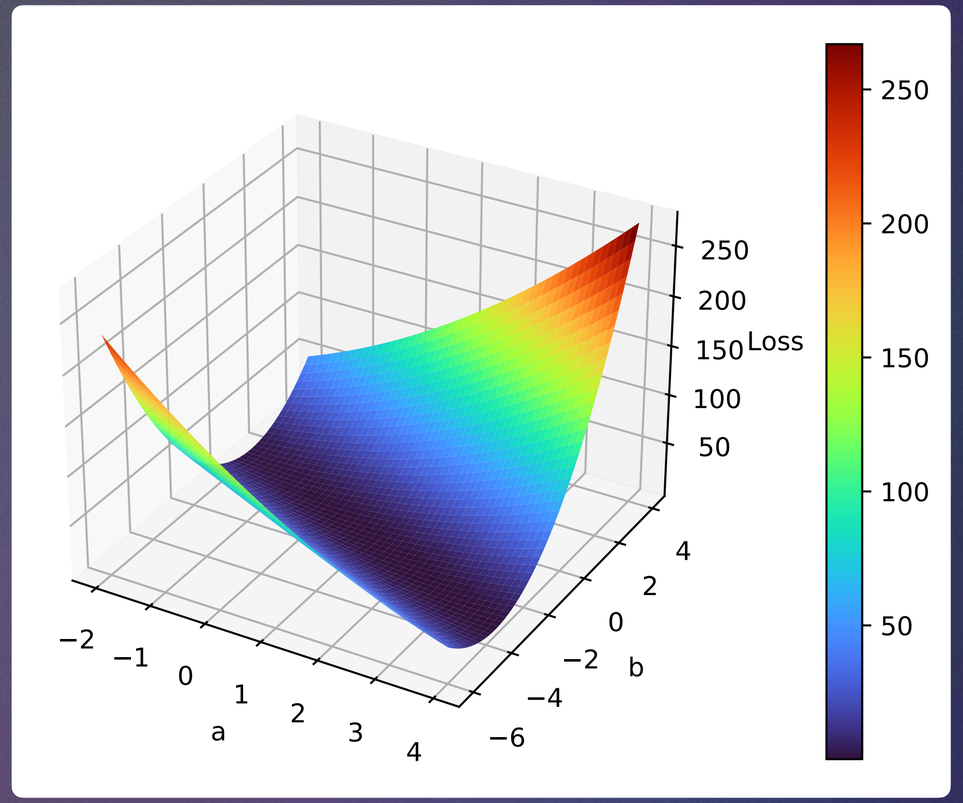
Recall the mean squared error loss we introduced in the previous, which we will denote as:

$$J(\mathbf{w}) = \frac{1}{n} \sum_i \left(y_i - \mathbf{w}^\intercal \mathbf{x}_i \right)^2.$$

Many regularization methods restrict the learning capability of models by adding a penalty parameter $\Omega(\cdot)$ to the objective function. Assume that the target function after regularization is $\tilde{J}$:

$$\tilde{J}(\mathbf{w}) = J(\mathbf{w}) + \alpha \Omega(\mathbf{w}) \,,$$

where $\alpha \in \mathbb{R}_{\geq 0}$ is a hyperparameter that weights the relative contribution of the norm penalty term $\Omega(\cdot)$ and the standard objective function $J(\cdot)$. If α is set to 0, no regularization is performed. The penalty in regularization increases with α.



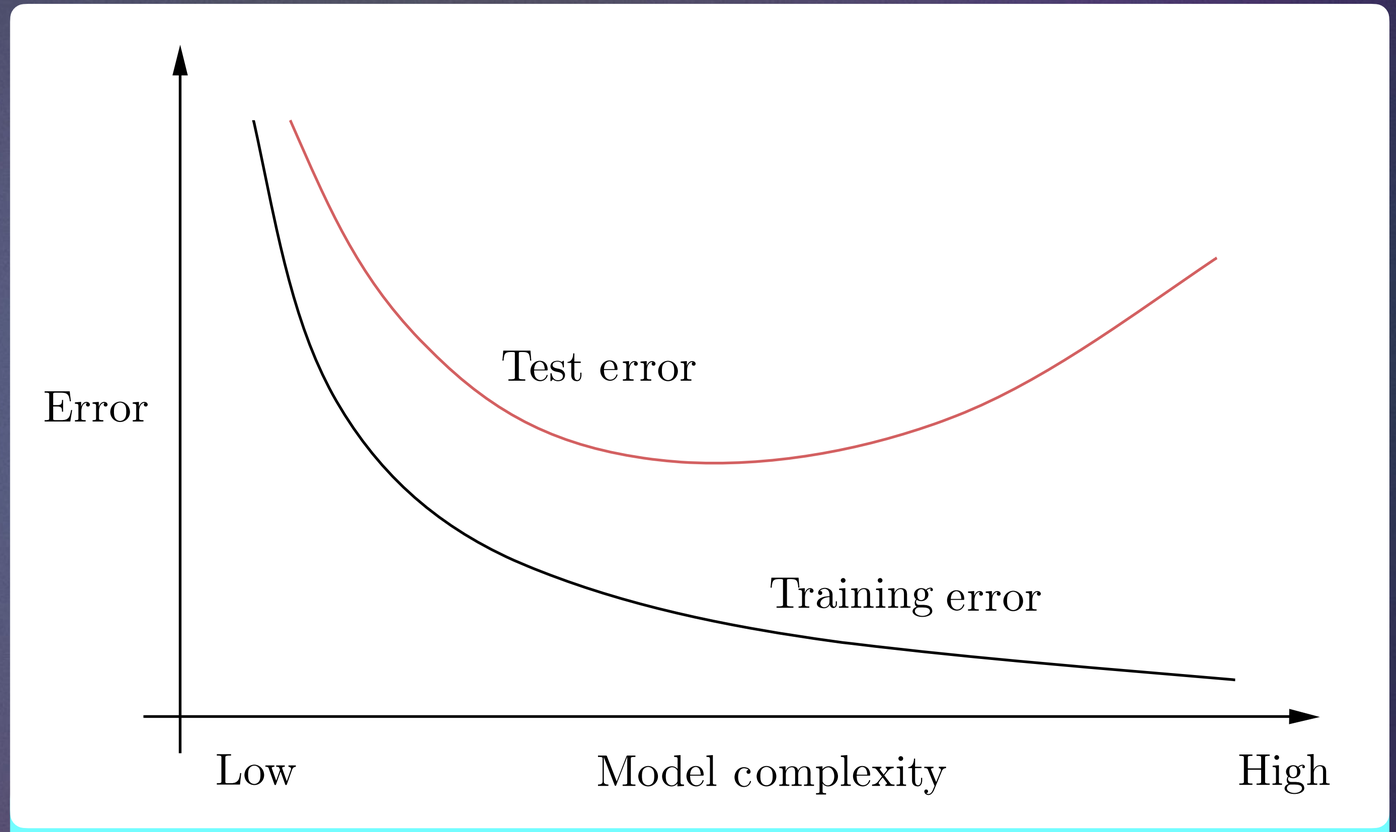
https://ml4a.github.io/ml4a/how_neural_networks_are_trained/

Regularization

Loss function (Error)

$$\tilde{J}(\mathbf{w}) = J(\mathbf{w}) + \alpha\Omega(\mathbf{w})$$

Complexity control



THEODORIDIS, S. Machine Learning: A Bayesian and Optimization Perspective. Elsevier Science, 2020. ISBN 9780128188040.

L_1 and L_2 Regularization

One can add L_1 norm constraint to model parameters, that is,

$$\tilde{J}(\mathbf{w}) = J(\mathbf{w}) + \alpha \parallel \mathbf{w} \parallel_1 ,$$

where

$$\parallel \mathbf{w} \parallel_1 = \sum_i |w_i| .$$

If a gradient method is used to resolve the value, the parameter gradient is:

$$w^{(t+1)} = w^{(t)} - \eta \nabla J\left(w^{(t)}\right) + \text{sign}\left(w^{(t)}\right) \alpha.$$

https://towardsdatascience.com/intuitions-on-l1-and-l2-regularisation-235f2db4c261

One can add norm penalty term L_2 to prevent overfitting,

$$\tilde{J}(\mathbf{w}) = J(\mathbf{w}) + \frac{1}{2}\alpha \parallel \mathbf{w} \parallel_2^2 ,$$

where

$$\parallel \mathbf{w} \parallel_2 = \sqrt{\sum_i |w_i|^2} .$$

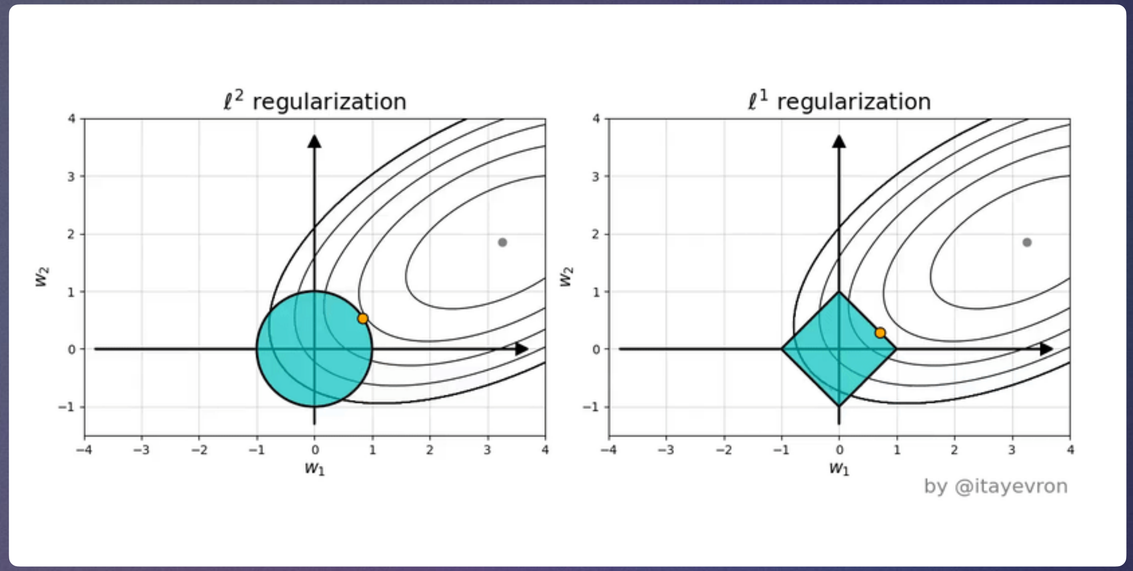
A parameter optimization method can be inferred using an optimization technology (such as a gradient method):

$$w^{(t+1)} = (1 - \eta\alpha) w^{(t)} + \eta \nabla J\left(w^{(t)}\right) .$$

L_1 v.s. L_2 Regularization

The major differences between L_2 and L_1:

- According to the preceding analysis, L_1 can generate a more sparse model than L_2. When the value of parameter w is small, L_1 regularization can directly reduce the parameter value to 0, which can be used for feature selection.
- From the probability perspective, many norm constraints are equivalent to adding prior probability distribution to parameters. In L_2, the parameter value complies with the Gaussian distribution, while in L_1, the parameter value complies with the Laplace distribution.



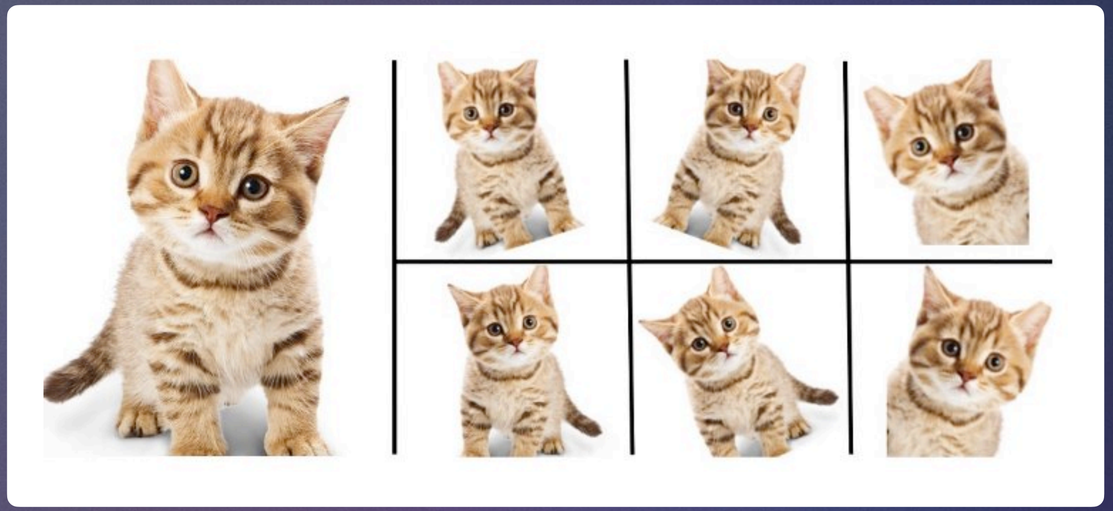
https://www.quora.com/What-is-the-intuition-that-the-l-1-norm-leads-to-sparse-solutions/answer/Itay-Evron-1

Dataset Expansion

The most effective way to prevent over-fitting is to add a training set. A larger training set has a smaller over-fitting probability. Dataset expansion is a time-saving method, but it varies in different fields.

➡ **A common method in the object recognition field is to rotate or scale images. Random noise is added to the input data in speech recognition.**

- Random noise is added to the input data in speech recognition.
- A common practice of natural language processing (NLP) is replacing words with their synonyms.
- Noise injection can add noise to the input or to the hidden layer or output layer.



https://www.kdnuggets.com/2020/02/easy-image-dataset-augmentation-tensorflow.html

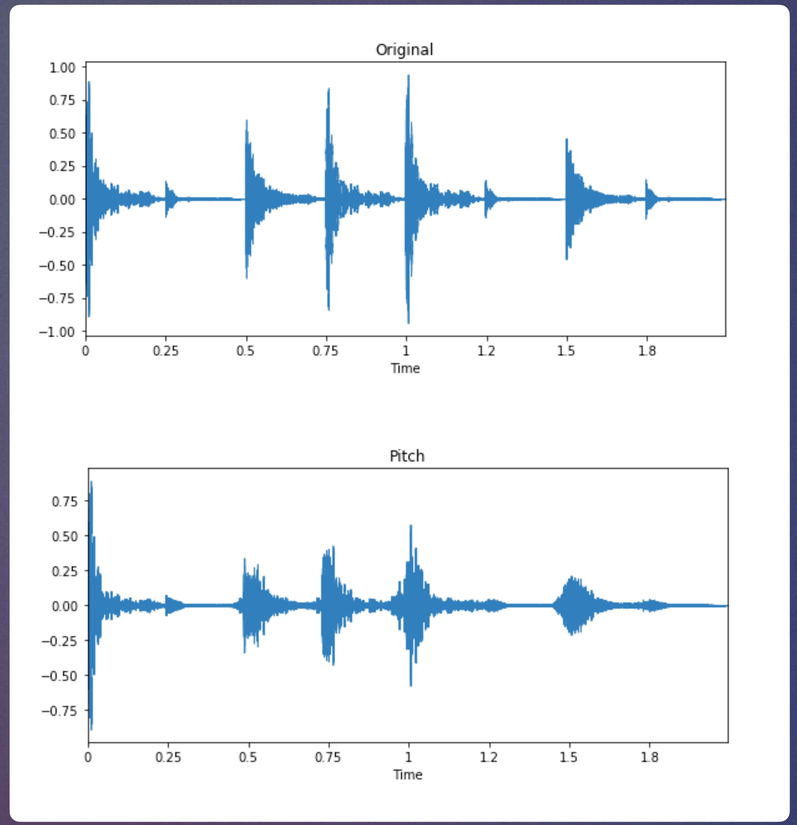
Dataset Expansion

The most effective way to prevent over-fitting is to add a training set. A larger training set has a smaller over-fitting probability. Dataset expansion is a time-saving method, but it varies in different fields.

- A common method in the object recognition field is to rotate or scale images. Random noise is added to the input data in speech recognition.

➡ **Random noise is added to the input data in speech recognition.**

- A common practice of natural language processing (NLP) is replacing words with their synonyms.
- Noise injection can add noise to the input or to the hidden layer or output layer.



https://medium.com/@makcedward/data-augmentation-for-audio-76912b01fdf6

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

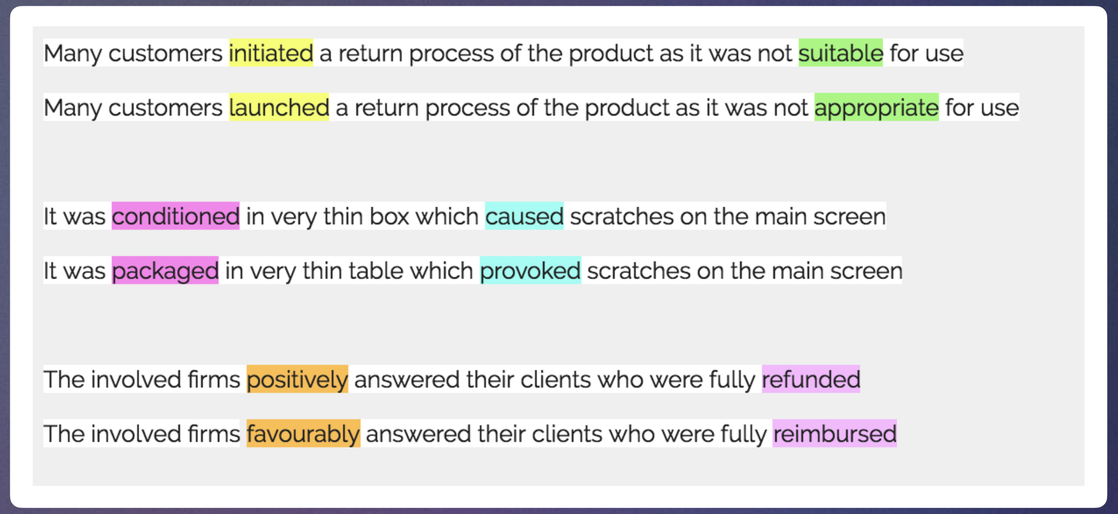
Dataset Expansion

The most effective way to prevent over-fitting is to add a training set. A larger training set has a smaller over-fitting probability. Dataset expansion is a time-saving method, but it varies in different fields.

- A common method in the object recognition field is to rotate or scale images. Random noise is added to the input data in speech recognition.
- Random noise is added to the input data in speech recognition.

➡ **A common practice of natural language processing (NLP) is replacing words with their synonyms.**

- Noise injection can add noise to the input or to the hidden layer or output layer.



https://medium.com/opla/text-augmentation-for-machine-learning-tasks-how-to-grow-your-text-dataset-for-classification-38a9a207f88d

Dataset Expansion

The most effective way to prevent over-fitting is to add a training set. A larger training set has a smaller over-fitting probability. Dataset expansion is a time-saving method, but it varies in different fields.

- A common method in the object recognition field is to rotate or scale images. Random noise is added to the input data in speech recognition.
- Random noise is added to the input data in speech recognition.
- A common practice of natural language processing (NLP) is replacing words with their synonyms.

➡ **Noise injection can add noise to the input or to the hidden layer or output layer.**

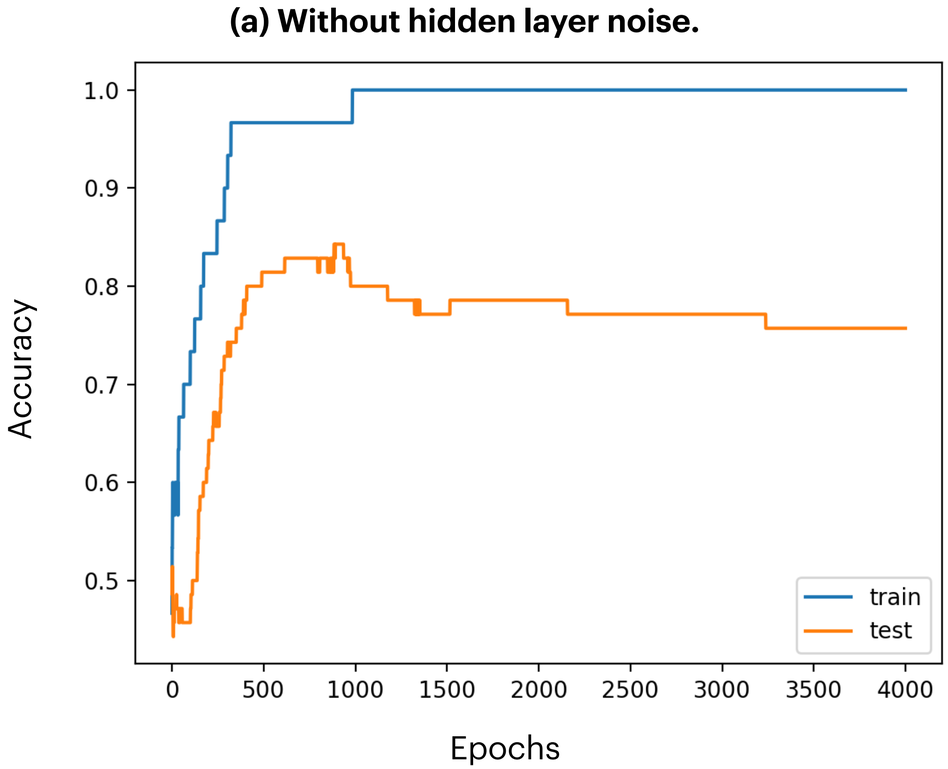
PYTHON: MLP WITH HIDDEN LAYER NOISE

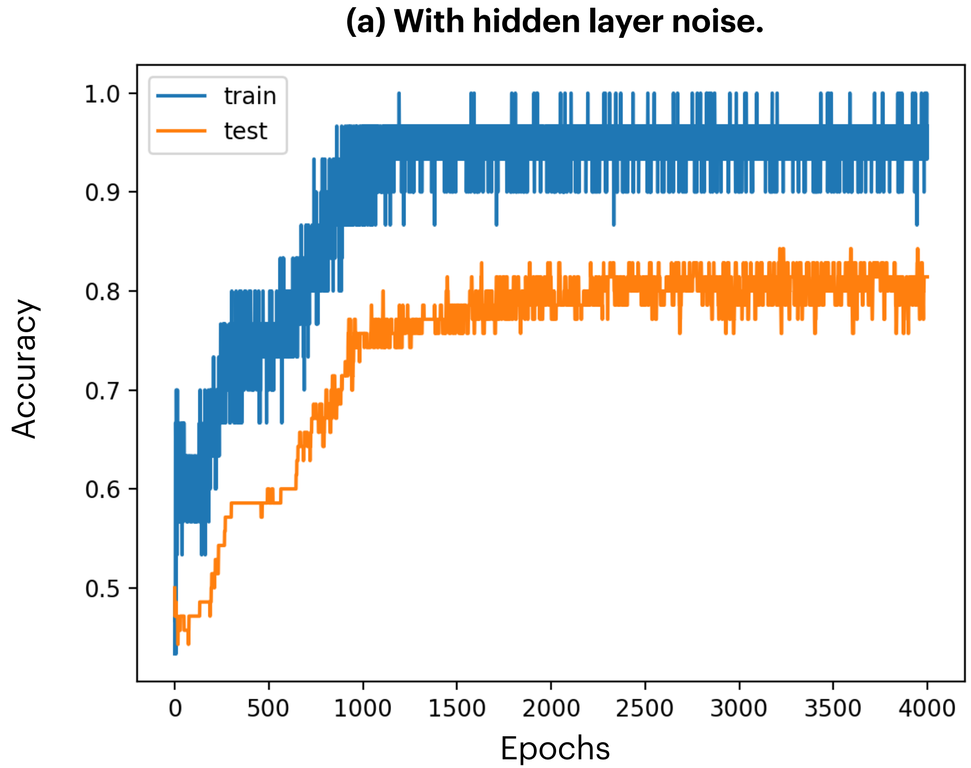
```
model = Sequential( )
model.add(Dense(500, input_dim=2))
model.add(GaussianNoise(0.1))
model.add(Activation('relu'))
model.add(Activation(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy',
                    optimizer='adam',
                    metrics=['accuracy'])
```

https://machinelearningmastery.com/how-to-improve-deep-learning-model-robustness-by-adding-noise/

Dataset Expansion: Hidden layer with noise

Train an test accuracy with and without hidden layer noise

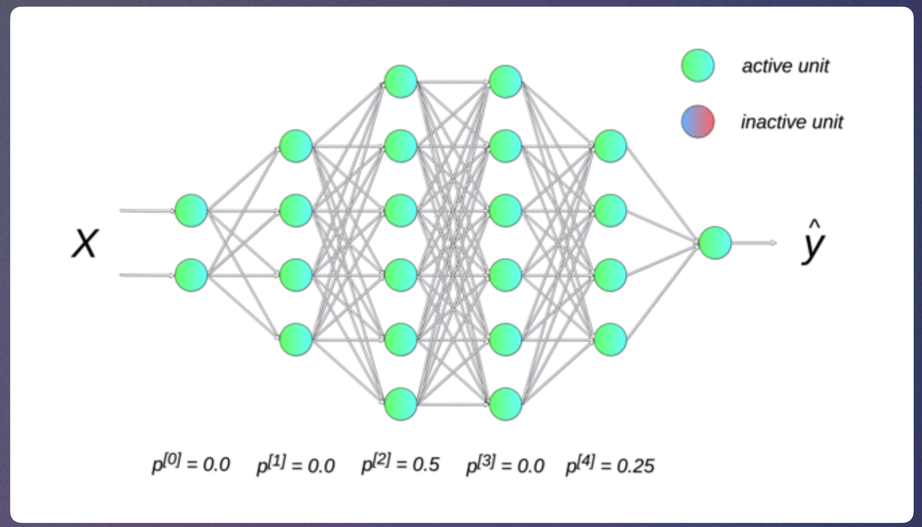




https://machinelearningmastery.com/how-to-improve-deep-learning-model-robustness-by-adding-noise/

Dropout

Dropout is a common and simple regularization method, which has been widely used since 2014. Simply put, Dropout randomly discards some inputs during the training process. In this case, the parameters corresponding to the discarded inputs are not updated. As an integration method, Dropout combines all sub-network results and obtains sub-networks by randomly dropping inputs.



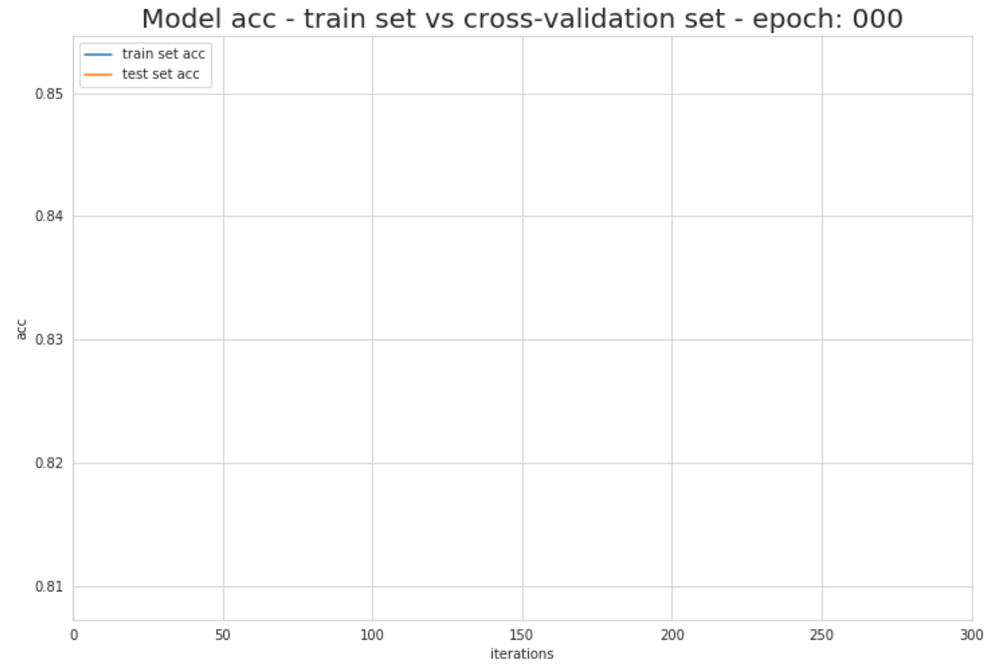
https://towardsdatascience.com/preventing-deep-neural-network-from-overfitting-953458db800a

Early Stopping

The graph shows the change in accuracy values calculated on the test and cross-validation sets during subsequent iterations of learning process. We see right away that the model we get at the end is not the best we could have possibly create.

To be honest, it is much worse than what we have had after 150 epochs. Why not interrupt the learning process before the model starts overfitting?

This observation inspired one of the popular overfitting reduction method, namely early stopping.



https://towardsdatascience.com/preventing-deep-neural-network-from-overfitting-953458db800a

References (1)

- Gene Kogan. How neural networks are trained. https://ml4a.github.io/ml4a/how_neural_networks_are_trained/. 2016, Accessed on Feb 2021.
- Irfan Danish. Introduction to Neural Networks and Their Key Elements (Part-C)—Activation Functions & Layers. https://laptrinhx.com/introduction-to-neural-networks-and-their-key-elements-part-c-activation-functions-layers-1195957448/. 2020, Accessed on Feb 2021.
- HUAWEI. Deep Learning Overview. 2020, Accessed on Feb 2021.
- Kishan Maladkar. The Number Game Behind Advanced Activation Functions In Machine Learning. Available at: https://analyticsindiamag.com/the-number-game-behind-advanced-activation-functions-in-machine-learning/. 2018, Accessed on Feb 2021.
- Sefik Ilkin Serengil. Softplus as a Neural Networks Activation Function. Available at: https://sefiks.com/2017/08/11/softplus-as-a-neural-networks-activation-function/, 2017. Accessed on Feb 2021.
- Manish Chablani. Deep learning concepts — PART 1. https://towardsdatascience.com/deep-learning-concepts-part-1-ea0b14b234c8. 2017, Accessed on Feb 2021.
- THEODORIDIS, S. Machine Learning: A Bayesian and Optimization Perspective. Elsevier Science, 2020. ISBN 9780128188040.

References (2)

- Raimi Karim. Intuitions on L1 and L2 Regularisation. https://towardsdatascience.com/intuitions-on-l1-and-l2-regularisation-235f2db4c261. 2018, Accessed on Feb 2021.
- Itay Evron. What is the intuition that the l-1 norm leads to sparse solutions? What is the intuition that the l-1 norm leads to sparse solutions?. 2017, Accessed on Feb 2021.
- Matthew Mayo. Easy Image Dataset Augmentation with TensorFlow. https://www.kdnuggets.com/2020/02/easy-image-dataset-augmentation-tensorflow.html. 2020, Accessed on Feb 2021.
- Edward Ma. Data Augmentation for Audio. https://medium.com/@makcedward/data-augmentation-for-audio-76912b01fdf6. 2019, Accessed on Feb 2021.
- Maâli Mnasri. Text augmentation for Machine Learning tasks: How to grow your text dataset for classification? https://medium.com/opla/text-augmentation-for-machine-learning-tasks-how-to-grow-your-text-dataset-for-classification-38a9a207f88d. 2019, Accessed on Feb 2021.
- Jason Brownlee. How to Improve Deep Learning Model Robustness by Adding Noise. https://machinelearningmastery.com/how-to-improve-deep-learning-model-robustness-by-adding-noise/. 2018, Accessed on Feb 2021.

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Ceará

HUAWEI

Thank you for your attention!

🙏☺️

Prof. Me. Saulo A. F. Oliveira
saulo.oliveira@ifce.edu.br