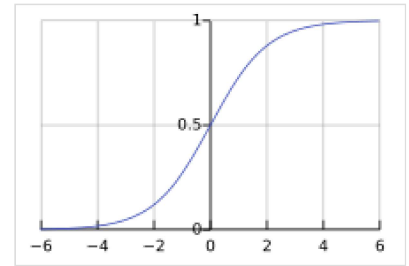




Activation function

The **activation function** of a node in an artificial neural network is a function that calculates the output of the node based on its individual inputs and their weights. Nontrivial problems can be solved using only a few nodes if the activation function is *nonlinear*.^[1] Modern activation functions include the smooth version of the ReLU, the GELU, which was used in the 2018 BERT model,^[2] the logistic (sigmoid) function used in the 2012 speech recognition model developed by Hinton et al,^[3] the ReLU used in the 2012 AlexNet computer vision model^{[4][5]} and in the 2015 ResNet model.



Logistic activation function

Comparison of activation functions

Aside from their empirical performance, activation functions also have different mathematical properties:

Nonlinear

When the activation function is non-linear, then a two-layer neural network can be proven to be a universal function approximator.^[6] This is known as the Universal Approximation Theorem. The identity activation function does not satisfy this property. When multiple layers use the identity activation function, the entire network is equivalent to a single-layer model.

Range

When the range of the activation function is finite, gradient-based training methods tend to be more stable, because pattern presentations significantly affect only limited weights. When the range is infinite, training is generally more efficient because pattern presentations significantly affect most of the weights. In the latter case, smaller learning rates are typically necessary.

Continuously differentiable

This property is desirable (ReLU is not continuously differentiable and has some issues with gradient-based optimization, but it is still possible) for enabling gradient-based optimization methods. The binary step activation function is not differentiable at 0, and it differentiates to 0 for all other values, so gradient-based methods can make no progress with it.^[7]

These properties do not decisively influence performance, nor are they the only mathematical properties that may be useful. For instance, the strictly positive range of the softplus makes it suitable for predicting variances in variational autoencoders.

Mathematical details

The most common activation functions can be divided into three categories: ridge functions, radial functions and fold functions.

An activation function f is **satürating** if $\lim_{|v| \rightarrow \infty} |\nabla f(v)| = 0$. It is **nonsatürating** if it is $\lim_{|v| \rightarrow \infty} |\nabla f(v)| \neq 0$.

Non-satürating activation functions, such as ReLU, may be better than satürating activation functions, because they are less likely to suffer from the vanishing gradient problem.^[8]

Ridge activation functions

Ridge functions are multivariate functions acting on a linear combination of the input variables. Often used examples include:

- Linear activation: $\phi(\mathbf{v}) = a + \mathbf{v}'\mathbf{b}$,
- ReLU activation: $\phi(\mathbf{v}) = \max(0, a + \mathbf{v}'\mathbf{b})$,
- Heaviside activation: $\phi(\mathbf{v}) = \mathbf{1}_{a + \mathbf{v}'\mathbf{b} > 0}$,

- Logistic activation: $\phi(\mathbf{v}) = (1 + \exp(-\mathbf{a} - \mathbf{v}'\mathbf{b}))^{-1}$.

In biologically inspired neural networks, the activation function is usually an abstraction representing the rate of action potential firing in the cell.^[9] In its simplest form, this function is binary—that is, either the neuron is firing or not. Neurons also cannot fire faster than a certain rate, motivating sigmoid activation functions whose range is a finite interval.

The function looks like $\phi(\mathbf{v}) = U(\mathbf{a} + \mathbf{v}'\mathbf{b})$, where U is the Heaviside step function.

If a line has a positive slope, on the other hand, it may reflect the increase in firing rate that occurs as input current increases. Such a function would be of the form $\phi(\mathbf{v}) = \mathbf{a} + \mathbf{v}'\mathbf{b}$.

Radial activation functions

A special class of activation functions known as radial basis functions (RBFs) are used in RBF networks, which are extremely efficient as universal function approximators. These activation functions can take many forms, but they are usually found as one of the following functions:

- Gaussian: $\phi(\mathbf{v}) = \exp\left(-\frac{\|\mathbf{v} - \mathbf{c}\|^2}{2\sigma^2}\right)$
- Multiquadratics: $\phi(\mathbf{v}) = \sqrt{\|\mathbf{v} - \mathbf{c}\|^2 + a^2}$
- Inverse multiquadratics: $\phi(\mathbf{v}) = (\|\mathbf{v} - \mathbf{c}\|^2 + a^2)^{-\frac{1}{2}}$
- Polyharmonic splines

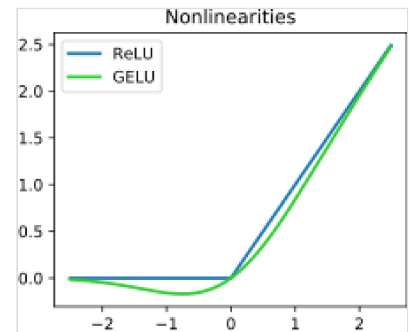
where \mathbf{c} is the vector representing the function *center* and \mathbf{a} and σ are parameters affecting the spread of the radius.

Folding activation functions

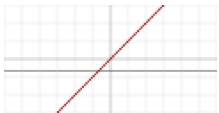
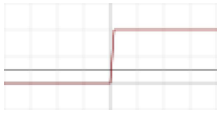
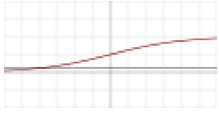
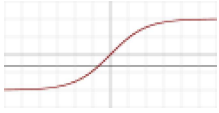
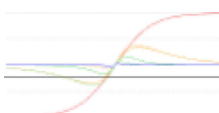
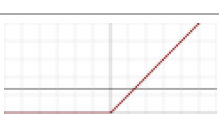

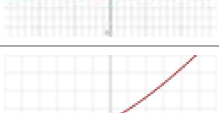

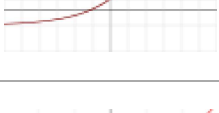

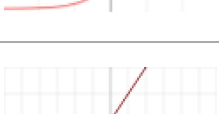
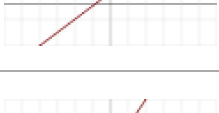
Folding activation functions are extensively used in the pooling layers in convolutional neural networks, and in output layers of multiclass classification networks. These activations perform aggregation over the inputs, such as taking the mean, minimum or maximum. In multiclass classification the softmax activation is often used.

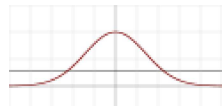
Table of activation functions

The following table compares the properties of several activation functions that are functions of one fold x from the previous layer or layers:



Rectified linear unit and Gaussian error linear unit activation functions

Name	Plot	Function, $g(x)$	Derivative of g , $g'(x)$	Range	Order of continuity
<u>Identity</u>		x	1	$(-\infty, \infty)$	C^∞
<u>Binary step</u>		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	0	$\{0, 1\}$	C^{-1}
<u>Logistic, sigmoid, or soft step</u>		$\sigma(x) \doteq \frac{1}{1 + e^{-x}}$	$g(x)(1 - g(x))$	$(0, 1)$	C^∞
<u>Hyperbolic tangent (tanh)</u>		$\tanh(x) \doteq \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - g(x)^2$	$(-1, 1)$	C^∞
<u>Soboleva modified hyperbolic tangent (smht)</u>		$\text{smht}(x) \doteq \frac{e^{ax} - e^{-bx}}{e^{cx} + e^{-dx}}$		$(-1, 1)$	C^∞
<u>Rectified linear unit (ReLU)^[10]</u>		$(x)^+ \doteq \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max(0, x) = x \mathbf{1}_{x>0}$	$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \end{cases}$	$[0, \infty)$	C^0
<u>Gaussian Error Linear Unit (GELU)^[2]</u>		$\frac{1}{2}x \left(1 + \operatorname{erf} \left(\frac{x}{\sqrt{2}} \right) \right)$ $= x\Phi(x)$	$\Phi(x) + x\phi(x)$	$(-0.17\dots, \infty)$	C^∞
<u>Softplus^[11]</u>		$\ln(1 + e^x)$	$\frac{1}{1 + e^{-x}}$	$(0, \infty)$	C^∞
<u>Exponential linear unit (ELU)^[12]</u>		$\begin{cases} \alpha(e^x - 1) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α	$\begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x > 0 \end{cases}$	$(-\alpha, \infty)$	$\begin{cases} C^1 & \text{if } \alpha = 1 \\ C^0 & \text{otherwise} \end{cases}$
<u>Scaled exponential linear unit (SELU)^[13]</u>		$\lambda \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameters $\lambda = 1.0507$ and $\alpha = 1.67326$	$\lambda \begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-\lambda\alpha, \infty)$	C^0
<u>Leaky rectified linear unit (Leaky ReLU)^[14]</u>		$\begin{cases} 0.01x & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$	$\begin{cases} 0.01 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \end{cases}$	$(-\infty, \infty)$	C^0
<u>Parametric rectified linear unit (PReLU)^[15]</u>		$\begin{cases} \alpha x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameter α	$\begin{cases} \alpha & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-\infty, \infty)$	C^0
<u>Sigmoid linear unit (SiLU,^[2] Sigmoid shrinkage,^[16] SiL,^[17] or Swish-1^[18])</u>		$\frac{x}{1 + e^{-x}}$	$\frac{1 + e^{-x} + xe^{-x}}{(1 + e^{-x})^2}$	$[-0.278\dots, \infty)$	C^∞

Name	Plot	Function, $g(x)$	Derivative of g , $g'(x)$	Range	Order of continuity
<u>Gaussian</u>		e^{-x^2}	$-2xe^{-x^2}$	$(0, 1]$	C^∞

The following table lists activation functions that are not functions of a single fold x from the previous layer or layers:

Name	Equation, $g_i(\vec{x})$	Derivatives, $\frac{\partial g_i(\vec{x})}{\partial x_j}$	Range	Order of continuity
<u>Softmax</u>	$\frac{e^{x_i}}{\sum_{j=1}^J e^{x_j}}$ for $i = 1, \dots, J$	$g_i(\vec{x}) (\delta_{ij} - g_j(\vec{x}))^{[1][2]}$	$(0, 1)$	C^∞
Maxout ^[19]	$\max_i x_i$	$\begin{cases} 1 & \text{if } j = \underset{i}{\operatorname{argmax}} x_i \\ 0 & \text{if } j \neq \underset{i}{\operatorname{argmax}} x_i \end{cases}$	$(-\infty, \infty)$	C^0

[^] Here, δ_{ij} is the Kronecker delta.

[^] For instance, j could be iterating through the number of kernels of the previous neural network layer while i iterates through the number of kernels of the current layer.

Quantum activation functions

In quantum neural networks programmed on gate-model quantum computers, based on quantum perceptrons instead of variational quantum circuits, the non-linearity of the activation function can be implemented with no need of measuring the output of each perceptron at each layer. The quantum properties loaded within the circuit such as superposition can be preserved by creating the Taylor series of the argument computed by the perceptron itself, with suitable quantum circuits computing the powers up to a wanted approximation degree. Because of the flexibility of such quantum circuits, they can be designed in order to approximate any arbitrary classical activation function.^[20]

See also

- Logistic function
- Rectifier (neural networks)
- Stability (learning theory)
- Softmax function

References

- Hinkelmann, Knut. "Neural Networks, p. 7" (https://web.archive.org/web/20181006235506/http://didattica.cs.unicam.it/lib/exe/fetch.php?media=didattica:magistrale:kebi:ay_1718:ke-11_neural_networks.pdf) (PDF). *University of Applied Sciences Northwestern Switzerland*. Archived from the original (http://didattica.cs.unicam.it/lib/exe/fetch.php?media=didattica:magistrale:kebi:ay_1718:ke-11_neural_networks.pdf) (PDF) on 2018-10-06. Retrieved 2018-10-06.
- Hendrycks, Dan; Gimpel, Kevin (2016). "Gaussian Error Linear Units (GELUs)". *arXiv:1606.08415* (<https://arxiv.org/abs/1606.08415>) [[cs.LG](https://arxiv.org/archive/cs)] (<https://arxiv.org/archive/cs>).
- Hinton, Geoffrey; Deng, Li; Deng, Li; Yu, Dong; Dahl, George; Mohamed, Abdel-rahman; Jaitly, Navdeep; Senior, Andrew; Vanhoucke, Vincent; Nguyen, Patrick; Sainath, Tara; Kingsbury, Brian (2012). "Deep Neural Networks for Acoustic Modeling in Speech Recognition". *IEEE Signal Processing Magazine*. **29** (6): 82–97. doi:10.1109/MSP.2012.2205597 (<https://doi.org/10.1109/MSP.2012.2205597>). S2CID 206485943 (<https://api.semanticscholar.org/CorpusID:206485943>).

4. Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E. (2017-05-24). "ImageNet classification with deep convolutional neural networks" (<https://dl.acm.org/doi/10.1145/3065386>). *Communications of the ACM*. **60** (6): 84–90. doi:10.1145/3065386 (<https://doi.org/10.1145%2F3065386>). ISSN 0001-0782 (<https://www.worldcat.org/issn/0001-0782>).
5. King Abdulaziz University; Al-johania, Norah; Elrefaei, Lamiaa; Benha University (2019-06-30). "Dorsal Hand Vein Recognition by Convolutional Neural Networks: Feature Learning and Transfer Learning Approaches" (<http://www.inass.org/2019/2019063019.pdf>) (PDF). *International Journal of Intelligent Engineering and Systems*. **12** (3): 178–191. doi:10.22266/ijies2019.0630.19 (<https://doi.org/10.22266%2Fijies2019.0630.19>).
6. Cybenko, G. (December 1989). "Approximation by superpositions of a sigmoidal function" (<https://hal.archives-ouvertes.fr/hal-03753170/file/Cybenko1989.pdf>) (PDF). *Mathematics of Control, Signals, and Systems*. **2** (4): 303–314. doi:10.1007/BF02551274 (<https://doi.org/10.1007%2FBF02551274>). ISSN 0932-4194 (<https://www.worldcat.org/issn/0932-4194>). S2CID 3958369 (<https://api.semanticscholar.org/CorpusID:3958369>).
7. Snyman, Jan (3 March 2005). *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms* (https://books.google.com/books?id=0tFmf_UK17oC). Springer Science & Business Media. ISBN 978-0-387-24348-1.
8. Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E. (2017-05-24). "ImageNet classification with deep convolutional neural networks" (<https://doi.org/10.1145%2F3065386>). *Communications of the ACM*. **60** (6): 84–90. doi:10.1145/3065386 (<https://doi.org/10.1145%2F3065386>). ISSN 0001-0782 (<https://www.worldcat.org/issn/0001-0782>). S2CID 195908774 (<https://api.semanticscholar.org/CorpusID:195908774>).
9. Hodgkin, A. L.; Huxley, A. F. (1952-08-28). "A quantitative description of membrane current and its application to conduction and excitation in nerve" (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1392413>). *The Journal of Physiology*. **117** (4): 500–544. doi:10.1113/jphysiol.1952.sp004764 (<https://doi.org/10.1113%2Fjphysiol.1952.sp004764>). PMC 1392413 (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1392413>). PMID 12991237 (<https://pubmed.ncbi.nlm.nih.gov/12991237>).
10. Nair, Vinod; Hinton, Geoffrey E. (2010), "Rectified Linear Units Improve Restricted Boltzmann Machines" (<http://dl.acm.org/citation.cfm?id=3104322.3104425>), *27th International Conference on International Conference on Machine Learning, ICML'10, USA: Omnipress*, pp. 807–814, ISBN 9781605589077
11. Glorot, Xavier; Bordes, Antoine; Bengio, Yoshua (2011). "Deep sparse rectifier neural networks" (<http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf>) (PDF). *International Conference on Artificial Intelligence and Statistics*.
12. Clevert, Djork-Arné; Unterthiner, Thomas; Hochreiter, Sepp (2015-11-23). "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". *arXiv:1511.07289* (<https://arxiv.org/abs/1511.07289>) [cs.LG (<https://arxiv.org/archive/cs.LG>)].
13. Klambauer, Günter; Unterthiner, Thomas; Mayr, Andreas; Hochreiter, Sepp (2017-06-08). "Self-Normalizing Neural Networks". *Advances in Neural Information Processing Systems*. **30** (2017). *arXiv:1706.02515* (<https://arxiv.org/abs/1706.02515>).
14. Maas, Andrew L.; Hannun, Awni Y.; Ng, Andrew Y. (June 2013). "Rectifier nonlinearities improve neural network acoustic models". *Proc. ICML*. **30** (1). S2CID 16489696 (<https://api.semanticscholar.org/CorpusID:16489696>).
15. He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015-02-06). "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification". *arXiv:1502.01852* (<https://arxiv.org/abs/1502.01852>) [cs.CV (<https://arxiv.org/archive/cs.CV>)].
16. Atto, Abdourrahmane M.; Pastor, Dominique; Mercier, Grégoire (2008), "Smooth sigmoid wavelet shrinkage for non-parametric estimation" (https://hal.archives-ouvertes.fr/hal-02136546/file/ICASSP_ATTO_2008.pdf) (PDF), *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3265–3268, doi:10.1109/ICASSP.2008.4518347 (<https://doi.org/10.1109%2FICASSP.2008.4518347>), ISBN 978-1-4244-1483-3, S2CID 9959057 (<https://api.semanticscholar.org/CorpusID:9959057>)
17. Elfving, Stefan; Uchibe, Eiji; Doya, Kenji (2018). "Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning". *Neural Networks*. **107**: 3–11. *arXiv:1702.03118* (<https://arxiv.org/abs/1702.03118>). doi:10.1016/j.neunet.2017.12.012 (<https://doi.org/10.1016%2Fj.neunet.2017.12.012>). PMID 29395652 (<https://pubmed.ncbi.nlm.nih.gov/29395652>). S2CID 6940861 (<https://api.semanticscholar.org/CorpusID:6940861>).
18. Ramachandran, Prajit; Zoph, Barret; Le, Quoc V (2017). "Searching for Activation Functions". *arXiv:1710.05941* (<https://arxiv.org/abs/1710.05941>) [cs.NE (<https://arxiv.org/archive/cs.NE>)].
19. Goodfellow, Ian J.; Warde-Farley, David; Mirza, Mehdi; Courville, Aaron; Bengio, Yoshua (2013). "Maxout Networks". *JMLR Workshop and Conference Proceedings*. **28** (3): 1319–1327. *arXiv:1302.4389* (<https://arxiv.org/abs/1302.4389>).

20. Maronese, Marco; Destri, Claudio; Prati, Enrico (2022). "Quantum activation functions for quantum neural networks". *Quantum Information Processing*. **21** (4): 128. arXiv:2201.03700 (<https://arxiv.org/abs/2201.03700>). Bibcode:2022QuIP...21..128M (<https://ui.adsabs.harvard.edu/abs/2022QuIP...21..128M>). doi:10.1007/s11128-022-03466-0 (<https://doi.org/10.1007/s11128-022-03466-0>). ISSN 1570-0755 (<https://www.worldcat.org/issn/1570-0755>).
-

Retrieved from "https://en.wikipedia.org/w/index.php?title=Activation_function&oldid=1232764772"