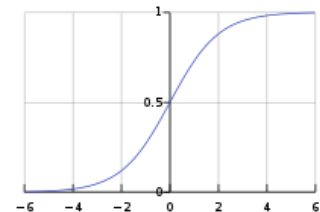


Activation function

In artificial neural networks, the **activation function** of a node defines the output of that node given an input or set of inputs. A standard integrated circuit can be seen as a digital network of activation functions that can be "ON" (1) or "OFF" (0), depending on input. This is similar to the linear perceptron in neural networks. However, only *nonlinear* activation functions allow such networks to compute nontrivial problems using only a small number of nodes, and such activation functions are called **nonlinearities**.^[1]



Logistic activation function

Contents

Classification of activation functions

Ridge activation functions

Radial activation functions

Folding activation functions

Comparison of activation functions

See also

References

Classification of activation functions

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

Ridge activation functions

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

- Linear activation: $\phi(\mathbf{v}) = \mathbf{a} + \mathbf{v}'\mathbf{b}$,
- ReLU activation: $\phi(\mathbf{v}) = \max(0, \mathbf{a} + \mathbf{v}'\mathbf{b})$,
- Heaviside activation: $\phi(\mathbf{v}) = 1_{\mathbf{a} + \mathbf{v}'\mathbf{b} > 0}$,
- 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.^[2] In its simplest form, this function is binary—that is, either the neuron is firing or not. The function looks like $\phi(\mathbf{v}) = U(\mathbf{a} + \mathbf{v}'\mathbf{b})$, where *U* is the Heaviside step function.

A line of positive slope may be used to reflect the increase in firing rate that occurs as input current increases. Such a function would be of the form $\phi(\mathbf{v}) = \mathbf{a} + \mathbf{v}'\mathbf{b}$.

Neurons also cannot fire faster than a certain rate, motivating sigmoid activation functions whose range is a finite interval.

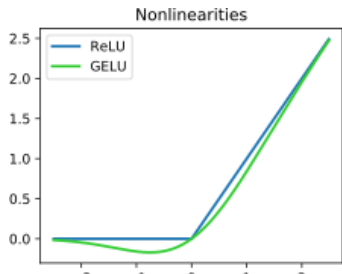
Radial activation functions

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 such as:

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

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

Folding activation functions



Rectified linear unit and Gaussian error linear unit 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.

Comparison of activation functions

There are numerous activation functions. Hinton et al.'s seminal 2012 paper on automatic speech recognition uses a logistic sigmoid activation function.^[3] The seminal 2012 AlexNet computer vision architecture uses the ReLU activation function, as did the seminal 2015 computer vision architecture ResNet. The seminal 2018 language processing model BERT uses a smooth version of the ReLU, the GELU.^[4]

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.^[5] 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

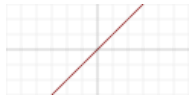


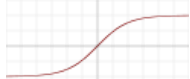
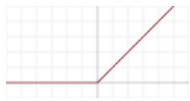
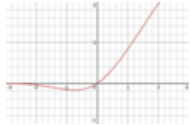
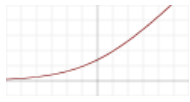




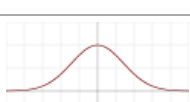
When the range of the activation function is finite, gradient-based training methods tend to be more stable, because pattern presentations significantly affect only limited weights. When the range is infinite, training is generally more efficient because pattern presentations significantly affect most 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.^[6]

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.

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

Name	Plot	Function, $f(x)$	Derivative of f , $f'(x)$	Range	Order of continuity
Identity		x	1	$(-\infty, \infty)$	C^∞
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$\begin{cases} 0 & \text{if } x \neq 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$\{0, 1\}$	C^{-1}
Logistic, sigmoid, or soft step		$\sigma(x) = \frac{1}{1 + e^{-x}}$ ^[1]	$f(x)(1 - f(x))$	$(0, 1)$	C^∞
Hyperbolic tangent (tanh)		$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - f(x)^2$	$(-1, 1)$	C^∞
Rectified linear unit (ReLU) ^[7]		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x \mathbf{1}_{x>0}$	$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$[0, \infty)$	C^0
Gaussian Error Linear Unit (GELU) ^[4]		$\frac{1}{2}x \left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right) \right)$ $= x\Phi(x)$	$\Phi(x) + x\phi(x)$	$(-0.17\dots, \infty)$	C^∞
Softplus ^[8]		$\ln(1 + e^x)$	$\frac{1}{1 + e^{-x}}$	$(0, \infty)$	C^∞
Exponential linear unit (ELU) ^[9]		$\begin{cases} \alpha(e^x - 1) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α	$\begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ 1 & \text{if } x = 0 \text{ and } \alpha = 1 \end{cases}$	$(-\alpha, \infty)$	$\begin{cases} C^1 & \text{if } \alpha = 1 \\ C^0 & \text{otherwise} \end{cases}$
Scaled exponential linear unit (SELU) ^[10]		$\lambda \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ 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
Leaky rectified linear unit (Leaky ReLU) ^[11]		$\begin{cases} 0.01x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$	$\begin{cases} 0.01 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-\infty, \infty)$	C^0
Parametric rectified linear unit (PReLU) ^[12]		$\begin{cases} \alpha x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameter α	$\begin{cases} \alpha & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-\infty, \infty)$ ^[2]	C^0
Sigmoid linear unit (SiLU) ^[4] Sigmoid shrinkage, ^[13] SiL, ^[14] or Swish-1 ^[15]		$\frac{x}{1 + e^{-x}}$	$\frac{1 + e^{-x} + xe^{-x}}{(1 + e^{-x})^2}$	$[-0.278\dots, \infty)$	C^∞
Mish ^[16]		$x \tanh(\ln(1 + e^x))$	$\frac{(e^x(4e^{2x} + e^{3x} + 4(1 + x) + e^x(6 + 4x)))}{(2 + 2e^x + e^{2x})^2}$	$[-0.308\dots, \infty)$	C^∞
Gaussian		e^{-x^2}	$-2xe^{-x^2}$	$(0, 1]$	C^∞
Growing Cosine Unit (GCU) ^[17]		$x \cos(x)$	$\cos(x) - x \sin(x)$	$(-\infty, \infty)$	C^∞

[^] Here, σ is the logistic function.
[^] $\alpha > 0$ for the range to hold true.

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

Name	Equation, $f_i(\vec{x})$	Derivatives, $\frac{\partial f_i(\vec{x})}{\partial x_j}$	Range	Order of continuity
Softmax	$\frac{e^{x_i}}{\sum_{j=1}^J e^{x_j}}$ for $i = 1, \dots, J$	$f_i(\vec{x}) (\delta_{ij} - f_j(\vec{x}))^{[3][4]}$	$(0, 1)$	C^∞
Maxout ^[18]	$\max_i x_i$	$\begin{cases} 1 & \text{if } j = \underset{i}{\operatorname{argmax}} x_i \\ 0 & \text{if } j \neq \underset{i}{\operatorname{argmax}} x_i \end{cases}$	$(-\infty, \infty)$	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.

See also

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

References

- Hinkelmann, Knut. "Neural Networks, p. 7" (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*.
- 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/jphysiol.1952.sp004764>). PMC 1392413 (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1392413>). PMID 12991237 (<https://pubmed.ncbi.nlm.nih.gov/12991237>).
- 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>).
- 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.LG>)].
- Cybenko, G. (December 1989). "Approximation by superpositions of a sigmoidal function". *Mathematics of Control, Signals, and Systems*. **2** (4): 303–314. doi:10.1007/BF02551274 (<https://doi.org/10.1007/BF02551274>). ISSN 0932-4194 (<https://www.worldcat.org/issn/0932-4194>). S2CID 3958369 (<https://api.semanticscholar.org/CorpusID:3958369>).
- 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_UKl7oC). Springer Science & Business Media. ISBN 978-0-387-24348-1.
- 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
- 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*.
- 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>)].
- 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>). Bibcode 2017arXiv170602515K (<https://ui.adsabs.harvard.edu/abs/2017arXiv170602515K>).
- 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>).
- 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>)].
- 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*, doi:10.1109/ICASSP.2008.4518347 (<https://doi.org/10.1109/ICASSP.2008.4518347>), S2CID 9959057 (<https://api.semanticscholar.org/CorpusID:9959057>)
- Elfwing, 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/j.neunet.2017.12.012>). PMID 29395652 (<https://pubmed.ncbi.nlm.nih.gov/29395652>). S2CID 6940861 (<https://api.semanticscholar.org/CorpusID:6940861>).

15. Ramachandran, Prajit; Zoph, Barret; Le, Quoc V (2017). "Searching for Activation Functions". [arXiv:1710.05941](https://arxiv.org/abs/1710.05941) (<https://arxiv.org/abs/1710.05941>) [cs.NE ([https://arxiv.org/archive/cs.NE](https://arxiv.org/archive/cs/NE))].
 16. Misra, Diganta (2020-08-13). "Mish: A Self Regularized Non-Monotonic Activation Function". [arXiv:1908.08681](https://arxiv.org/abs/1908.08681) (<https://arxiv.org/abs/1908.08681>) [cs.LG ([https://arxiv.org/archive/cs.LG](https://arxiv.org/archive/cs/LG))].
 17. Noel, Mathew Mithra; L, Arunkumar; Trivedi, Advait; Dutta, Praneet (2021-08-29). "Growing Cosine Unit: A Novel Oscillatory Activation Function That Can Speedup Training and Reduce Parameters in Convolutional Neural Networks" (<http://arxiv.org/abs/2108.12943>). *arXiv:2108.12943* [cs].
 18. 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) (<https://arxiv.org/abs/1302.4389>). Bibcode:2013arXiv1302.4389G (<https://ui.adsabs.harvard.edu/abs/2013arXiv1302.4389G>).
-

Retrieved from "https://en.wikipedia.org/w/index.php?title=Activation_function&oldid=1041641052"

This page was last edited on 31 August 2021, at 17:00 (UTC).

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.