WIKIPEDIA

# Vanishing gradient problem

In machine learning, the **vanishing gradient problem** is a difficulty found in training artificial neural networks with gradient-based learning methods and backpropagation. In such methods, each of the neural network's weights receives an update proportional to the partial derivative of the error function with respect to the current weight in each iteration of training. The problem is that in some cases, the gradient will be vanishingly small, effectively preventing the weight from changing its value. In the worst case, this may completely stop the neural network from further training. As one example of the problem cause, traditional activation functions such as the hyperbolic tangent function have gradients in the range $(-1, 1)$, and backpropagation computes gradients by the chain rule. This has the effect of multiplying $n$ of these small numbers to compute gradients of the "front" layers in an $n$-layer network, meaning that the gradient (error signal) decreases exponentially with $n$ while the front layers train very slowly.

Back-propagation allowed researchers to train supervised deep artificial neural networks from scratch, initially with little success. Hochreiter's diploma thesis of 1991[1][2] formally identified the reason for this failure in the "vanishing gradient problem", which not only affects many-layered feedforward networks,[3] but also recurrent networks.[4] The latter are trained by unfolding them into very deep feedforward networks, where a new layer is created for each time step of an input sequence processed by the network.

When activation functions are used whose derivatives can take on larger values, one risks encountering the related **exploding gradient problem**.

## Contents



## Solutions

### Multi-level hierarchy

To overcome this problem, several methods were proposed. One is Jürgen Schmidhuber's multi-level hierarchy of networks (1992) pre-trained one level at a time through unsupervised learning, fine-tuned through backpropagation.[5] Here each level learns a compressed representation of the observations that is fed to the next level.

#### Related approach

Similar ideas have been used in feed-forward neural networks for unsupervised pre-training to structure a neural network, making it first learn generally useful feature detectors. Then the network is trained further by supervised backpropagation to classify labeled data. The deep belief network model by Hinton et al. (2006) involves learning the distribution of a high level representation using successive layers of binary or real-valued latent variables. It uses a restricted Boltzmann machine to model each new layer of higher level features. Each new layer guarantees an increase on the lower-bound of the log likelihood of the data, thus improving the model, if trained properly. Once sufficiently many layers have been learned the deep architecture may be used as a generative model by reproducing the data when sampling down the model (an "ancestral pass") from the top level feature activations.[6] Hinton reports that his models are effective feature extractors over high-dimensional, structured data.[7]

## Long short-term memory

Another technique particularly used for recurrent neural networks is the long short-term memory (LSTM) network of 1997 by Hochreiter & Schmidhuber.[8] In 2009, deep multidimensional LSTM networks demonstrated the power of deep learning with many nonlinear layers, by winning three ICDAR 2009 competitions in connected handwriting recognition, without any prior knowledge about the three different languages to be learned.[9][10]

## Faster hardware

Hardware advances have meant that from 1991 to 2015, computer power (especially as delivered by GPUs) has increased around a million-fold, making standard backpropagation feasible for networks several layers deeper than when the vanishing gradient problem was recognized. Schmidhuber notes that this "is basically what is winning many of the image recognition competitions now", but that it "does not really overcome the problem in a fundamental way"[11] since the original models tackling the vanishing gradient problem by Hinton et al. (2006) were trained in a Xeon processor, not GPUs.[6]

## Residual networks

One of the newest and most effective ways to resolve the vanishing gradient problem is with residual neural networks, or ResNets[12] (not to be confused with recurrent neural networks).[13] It was noted prior to ResNets that a deeper network would actually have higher *training* error than the shallow network. This intuitively can be understood as data disappearing through too many layers of the network, meaning output from a shallow layer was diminished through the greater number of layers in the deeper network, yielding a worse result. Going with this intuitive hypothesis, Microsoft research found that splitting a deep network into three layer chunks and passing the input into each chunk straight through to the next chunk, along with the residual-output of the chunk minus the input to the chunk that is reintroduced, helped eliminate much of this disappearing signal problem. No extra parameters or changes to the learning algorithm were needed. ResNets[14] yielded lower training error (and test error) than their shallower counterparts simply by reintroducing outputs from shallower layers in the network to compensate for the vanishing data.[15]

Note that ResNets are an ensemble of relatively shallow nets and do not resolve the vanishing gradient problem by preserving gradient flow throughout the entire depth of the network – rather, they avoid the problem simply by constructing ensembles of many short networks together. (Ensemble by Construction[16])

## Other activation functions

Rectifiers such as ReLU suffer less from the vanishing gradient problem, because they only saturate in one direction.[17]

### Other

Behnke relied only on the sign of the gradient (Rprop) when training his Neural Abstraction Pyramid[18] to solve problems like image reconstruction and face localization.

Neural networks can also be optimized by using a universal search algorithm on the space of neural network's weights, e.g., random guess or more systematically genetic algorithm. This approach is not based on gradient and avoids the vanishing gradient problem.[19]

## See also

- Spectral radius

## References


1. S. Hochreiter. Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut f. Informatik, Technische Univ. Munich, 1991.
2. S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In S. C. Kremer and J. F. Kolen, editors, A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press, 2001.
3. Goh, Garrett B.; Hodas, Nathan O.; Vishnu, Abhinav (15 June 2017). "Deep learning for computational chemistry". *Journal of Computational Chemistry*. **38** (16): 1291–1307. arXiv:1701.04503 (https://arxiv.org/abs/1701.04503). doi:10.1002/jcc.24764 (https://doi.org/10.1002%2Fjcc.24764). PMID 28272810 (https://www.ncbi.nlm.nih.gov/pubmed/28272810).
4. Pascanu, Razvan; Mikolov, Tomas; Bengio, Yoshua (21 November 2012). "On the difficulty of training Recurrent Neural Networks". arXiv:1211.5063 (https://arxiv.org/abs/1211.5063) [cs.LG (https://arxiv.org/archive/cs.LG)].
5. J. Schmidhuber., "Learning complex, extended sequences using the principle of history compression," *Neural Computation*, 4, pp. 234–242, 1992.
6. Hinton, G. E.; Osindero, S.; Teh, Y. (2006). "A fast learning algorithm for deep belief nets" (http://www.cs.toronto.edu/~hinton/absps/fastnc.pdf) (PDF). *Neural Computation*. **18** (7): 1527–1554. CiteSeerX 10.1.1.76.1541 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.76.1541). doi:10.1162/neco.2006.18.7.1527 (https://doi.org/10.1162%2Fneco.2006.18.7.1527). PMID 16764513 (https://www.ncbi.nlm.nih.gov/pubmed/16764513).
7. Hinton, G. (2009). "Deep belief networks". *Scholarpedia*. **4** (5): 5947. Bibcode:2009SchpJ...4.5947H (https://ui.adsabs.harvard.edu/abs/2009SchpJ...4.5947H). doi:10.4249/scholarpedia.5947 (https://doi.org/10.4249%2Fscholarpedia.5947).
8. Hochreiter, Sepp; Schmidhuber, Jürgen (1997). "Long Short-Term Memory". *Neural Computation*. **9** (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735 (https://doi.org/10.1162%2Fneco.1997.9.8.1735). PMID 9377276 (https://www.ncbi.nlm.nih.gov/pubmed/9377276).
9. Graves, Alex; and Schmidhuber, Jürgen; *Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks*, in Bengio, Yoshua; Schuurmans, Dale; Lafferty, John; Williams, Chris K. I.; and Culotta, Aron (eds.), *Advances in Neural Information Processing Systems 22 (NIPS'22), December 7th–10th, 2009, Vancouver, BC*, Neural Information Processing Systems (NIPS) Foundation, 2009, pp. 545–552
10. Graves, A.; Liwicki, M.; Fernandez, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. (2009). "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **31** (5): 855–868. CiteSeerX 10.1.1.139.4502 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.139.4502). doi:10.1109/tpami.2008.137 (https://doi.org/10.1109%2Ftpami.2008.137). PMID 19299860 (https://www.ncbi.nlm.nih.gov/pubmed/19299860).

11. Schmidhuber, Jürgen (2015). "Deep learning in neural networks: An overview". *Neural Networks*. **61**: 85–117. arXiv:1404.7828 (https://arxiv.org/abs/1404.7828). doi:10.1016/j.neunet.2014.09.003 (https://doi.org/10.1016%2Fj.neunet.2014.09.003). PMID 25462637 (https://www.ncbi.nlm.nih.gov/pubmed/25462637).
12. "Residual neural networks are an exciting area of deep learning research" (https://blog.init.ai/residual-neural-networks-are-an-exciting-area-of-deep-learning-research-acf14f4912e9). 28 April 2016.
13. http://www.fit.vutbr.cz/research/groups/speech/servite/2010/rnnlm_mikolov.pdf
14. "ResNets, HighwayNets, and DenseNets, Oh My! – Chatbot's Life" (https://chatbotslife.com/resnets-highwaynets-and-densenets-oh-my-9bb15918ee32). 14 October 2016.
15. He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015). "Deep Residual Learning for Image Recognition". arXiv:1512.03385 (https://arxiv.org/abs/1512.03385) [cs.CV (https://arxiv.org/archive/cs.CV)].
16. Veit, Andreas; Wilber, Michael; Belongie, Serge (20 May 2016). "Residual Networks Behave Like Ensembles of Relatively Shallow Networks". arXiv:1605.06431 (https://arxiv.org/abs/1605.06431) [cs.CV (https://arxiv.org/archive/cs.CV)].
17. Glorot, Xavier; Bordes, Antoine; Bengio, Yoshua (14 June 2011). "Deep Sparse Rectifier Neural Networks" (http://proceedings.mlr.press/v15/glorot11a.html). *PMLR*: 315–323.
18. Sven Behnke (2003). *Hierarchical Neural Networks for Image Interpretation* (http://www.ais.uni-bonn.de/books/LNCS2766.pdf) (PDF). Lecture Notes in Computer Science. **2766**. Springer.
19. "Sepp Hochreiter's Fundamental Deep Learning Problem (1991)" (http://people.idsia.ch/~juergen/fundamentaldeeplearningproblem.html). *people.idsia.ch*. Retrieved 7 January 2017.


---