cases	doc_1	doc_2
authors	• V. I. Avrutskiy	• V.I. Avrutskiy
title	Avoiding overfitting of multilayer perceptrons by training derivatives	Avoiding overfitting of multilayer perceptrons by training derivatives
publication_date	2018-02-28 08:16:26+00:00	2018-02-28 00:00:00
source	SupportedSources.ARXIV	SupportedSources.INTERNET_ARCHIVE
journal	Proceedings of the Future Technologies Conference (FTC) 2019. Advances in Intelligent Systems and Computing, vol 1069	
volume		
doi	10.1007/978-3-030-32520-6_12	
urls	• http://arxiv.org/pdf/1802.10301v1 • http://dx.doi.org/10.1007/978-3-030-32520-6_12 • http://arxiv.org/abs/1802.10301v1 • http://arxiv.org/pdf/1802.10301v1	• https://web.archive.org/web/20191017025643/https://arxiv.org/pdf/1802.10301v1.pdf
id	id7992048420003122395	id7209914630581190656
abstract	Resistance to overfitting is observed for neural networks trained with extended backpropagation algorithm. In addition to target values, its cost function uses derivatives of those up to the 4^{th} order. For common applications of neural networks, high order derivatives are not readily available, so simpler cases are considered: training network to approximate analytical function inside 2D and 5D domains and solving Poisson equation inside a 2D circle. For function approximation, the cost is a sum of squared differences between output and target as well as their derivatives with respect to the input. Differential equations are usually solved by putting a multilayer perceptron in place of unknown function and training its weights, so that equation holds within some margin of error. Commonly used cost is the equation's residual squared. Added terms are squared derivatives of said residual with respect to the independent variables. To investigate overfitting, the cost is minimized for points of regular grids with various spacing, and its root mean is compared with its value on much denser test set. Fully connected perceptrons with six hidden layers and $2\cdot10^{4}$, $1\cdot10^{6}$ and $5\cdot10^{6}$ weights in total are trained with Rprop until cost changes by less than 10% for last 1000 epochs, or when the 10000^{th} epoch is reached. Training the network with $5\cdot10^{6}$ weights to represent simple 2D function using 10 points with 8 extra derivatives in each produces cost test to train ratio of 1.5, whereas for classical backpropagation in comparable conditions this ratio is $2\cdot10^{4}$.	Resistance to overfitting is observed for neural networks trained with extended backpropagation algorithm. In addition to target values, its cost function uses derivatives of those up to the 4^th order. For common applications of neural networks, high order derivatives are not readily available, so simpler cases are considered: training network to approximate analytical function inside 2D and 5D domains and solving Poisson equation inside a 2D circle. For function approximation, the cost is a sum of squared differences between output and target as well as their derivatives with respect to the input. Differential equations are usually solved by putting a multilayer perceptron in place of unknown function and training its weights, so that equation holds within some margin of error. Commonly used cost is the equation's residual squared. Added terms are squared derivatives of said residual with respect to the independent variables. To investigate overfitting, the cost is minimized for points of regular grids with various spacing, and its root mean is compared with its value on much denser test set. Fully connected perceptrons with six hidden layers and 2Â·10^4, 1Â·10^6 and 5Â·10^6 weights in total are trained with Rprop until cost changes by less than 10 reached. Training the network with 5Â·10^6 weights to represent simple 2D function using 10 points with 8 extra derivatives in each produces cost test to train ratio of 1.5, whereas for classical backpropagation in comparable conditions this ratio is 2Â·10^4.
versions		
decision	DUPLICATES	
id	315	