Learning rate keras

Tuning parameter (hyperparameter) in optimization Part of a series onMachine learningand data mining Problems Classification Clustering Regression Anomaly detection AutoML Association rules Reinforcement learning Structured prediction Feature engineering Feature learning Online learning Semi-supervised learning Unsupervised learning Learning to rank Grammar induction Supervised learning(classification • regression) Decision trees Ensembles Bagging Boosting Random forest k-NN Linear regression Naive Bayes Artificial neural networks Logistic regression Perceptron Relevance vector machine (RVM) Support vector machine (SVM) Clustering BIRCH CURE Hierarchical k-means Expectation–maximization (EM) DBSCAN OPTICS Mean shift Dimensionality reduction Factor analysis CCA ICA LDA NMF PCA PGD t-SNE Structured prediction Graphical models Bayes net Conditional random field Hidden Markov Anomaly detection k-NN Local outlier factor Artificial neural network Autoencoder Cognitive computing Deep learning DeepDream Multilayer perceptron RNN LSTM GRU ESN Restricted Boltzmann machine GAN SOM Convolutional neural network U-Net Transformer Spiking neural network Memtransistor Electrochemical RAM (ECRAM) Reinforcement learning Q-learning SARSA Temporal difference (TD) Theory Bias–variance tradeoff Computational learning theory Empirical risk minimization Occam learning PAC learning Statistical learning VC theory Machine-learning venues NeurIPS ICML ML JMLR ArXiv:cs.LG Related articles Glossary of artificial intelligence List of datasets for machine-learning research Outline of machine learning vte In machine learning and statistics, the learning rate is a tuning parameter in an optimization algorithm that determines the step size at each iteration while moving toward a minimum of a loss function.[1] Since it influences to what extent newly acquired information overrides old information, it metaphorically represents the speed at which a machine learning model "learns". In the adaptive control literature, the learning rate is commonly referred to as gain.[2] In setting a learning rate, there is a trade-off between the rate of convergence and overshooting. While the descent direction is usually determined from the gradient of the loss function, the learning rate determines how big a step is taken in that direction. A too high learning rate will make the learning jump over minima but a too low learning rate will either take too long to converge or get stuck in an undesirable local minimum.[3] In order to achieve faster convergence, prevent oscillations and getting stuck in undesirable local minima the learning rate is often varied during training either in accordance to a learning rate schedule or by using an adaptive learning rate.[4] The learning rate and its adjustments may also differ per parameter, in which case it is a diagonal matrix that can be interpreted as an approximation to the inverse of the Hessian matrix in Newton's method.[5] The learning rate is related to the step length determined by inexact line search in quasi-Newton methods and related optimization algorithms.[6][7] When conducting line searches, mini-batch sub-sampling (MBSS) affect the characteristics of the loss function along which the learning rate needs to be resolved.[8] Static MBSS keeps the mini-batch fixed along a search direction, resulting in a smooth loss function along the search direction. Dynamic MBSS updates the mini-batch at every function evaluation, resulting in a point-wise discontinuous loss function along the search direction. Line searches that adaptively resolve learning rates for static MBSS loss functions include the parabolic approximation line (PAL) search.[9] Line searches that adaptively resolve learning rates for dynamic MBSS loss functions include probabilistic line searches,[10] gradient-only line searches (GOLS)[11] and quadratic approximations.[12] Learning rate schedule Initial rate can be left as system default or can be selected using a range of techniques.[13] A learning rate schedule changes the learning rate during learning and is most often changed between epochs/iterations. This is mainly done with two parameters: decay and momentum . There are many different learning rate schedules but the most common are time-based, step-based and exponential.[4] Decay serves to settle the learning in a nice place and avoid oscillations, a situation that may arise when a too high constant learning rate makes the learning jump back and forth over a minimum, and is controlled by a hyperparameter. Momentum is analogous to a ball rolling down a hill; we want the ball to settle at the lowest point of the hill (corresponding to the lowest error). Momentum both speeds up the learning (increasing the learning rate) when the error cost gradient is heading in the same direction for a long time and also avoids local minima by 'rolling over' small bumps. Momentum is controlled by a hyper parameter analogous to a ball's mass which must be chosen manually—too high and the ball will roll over minima which we wish to find, too low and it will not fulfil its purpose. The formula for factoring in the momentum is more complex than for decay but is most often built in with deep learning libraries such as Keras. Time-based learning schedules alter the learning rate depending on the learning rate of the previous time iteration. Factoring in the decay the mathematical formula for the learning rate is: η n + 1 = η n 1 + d n {\displaystyle \eta _{n+1}={\frac {\eta _{n}}{1+dn}}} where η {\displaystyle \eta } is the learning rate, d {\displaystyle d} is a decay parameter and n {\displaystyle n} is the iteration step. Step-based learning schedules changes the learning rate according to some pre defined steps. The decay application formula is here defined as: η n = η 0 d f l o o r (1 + n r) {\displaystyle \eta _{n}=\eta _{0}d^{floor({\frac {1+n}{r}})}} where η n {\displaystyle \eta _{n}} is the learning rate at iteration n {\displaystyle n} , η 0 {\displaystyle \eta _{0}} is the initial learning rate, d {\displaystyle d} is how much the learning rate should change at each drop (0.5 corresponds to a halving) and r {\displaystyle r} corresponds to the droprate, or how often the rate should be dropped (10 corresponds to a drop every 10 iterations). The floor function here drops the value of its input to 0 for all values smaller than 1. Exponential learning schedules are similar to step-based but instead of steps a decreasing exponential function is used. The mathematical formula for factoring in the decay is: η n = η 0 e − d n {\displaystyle \eta _{n}=\eta _{0}e^{-dn}} where d {\displaystyle d} is a decay parameter. Adaptive learning rate The issue with learning rate schedules is that they all depend on hyperparameters that must be manually chosen for each given learning session and may vary greatly depending on the problem at hand or the model used. To combat this there are many different types of adaptive gradient descent algorithms such as Adagrad, Adadelta, RMSprop, and Adam[14] which are generally built into deep learning libraries such as Keras.[15] See also Hyperparameter (machine learning) Hyperparameter optimization Stochastic gradient descent Variable metric methods Overfitting Backpropagation AutoML Model selection Self-tuning References ^ Murphy, Kevin P. (2012). Machine Learning: A Probabilistic Perspective. Cambridge: MIT Press. p. 247. ISBN 978-0-262-01802-9. ^ Delyon, Bernard (2000). "Stochastic Approximation with Decreasing Gain: Convergence and Asymptotic Theory". Unpublished Lecture Notes. Université de Rennes. CiteSeerX 10.1.1.29.4428. ^ Buduma, Nikhil; Locascio, Nicholas (2017). Fundamentals of Deep Learning : Designing Next-Generation Machine Intelligence Algorithms. O'Reilly. p. 21. ISBN 978-1-4919-2558-4. ^ a b Patterson, Josh; Gibson, Adam (2017). "Understanding Learning Rates". Deep Learning : A Practitioner's Approach. O'Reilly. pp. 258–263. ISBN 978-1-4919-1425-0. ^ Ruder, Sebastian (2017). "An Overview of Gradient Descent Optimization Algorithms". arXiv:1609.04747. ^ Nesterov, Y. (2004). Introductory Lectures on Convex Optimization: A Basic Course. Boston: Kluwer. p. 25. ISBN 1-4020-7553-7. ^ Dixon, L. C. W. (1972). "The Choice of Step Length, a Crucial Factor in the Performance of Variable Metric Algorithms". Numerical Methods for Non-linear Optimization. London: Academic Press. pp. 149–170. ISBN 0-12-455650-7. ^ Kafka, Dominic; Wilke, Daniel N. (2021). "An empirical study into finding optima in stochastic optimization of neural networks". Information Sciences. 560: 235–255. arXiv:1903.08552. ^ Mutschler, Maximus; Zell, Andreas (2019). "Parabolic Approximation Line Search for DNNs". arXiv:1903.11991. ^ Mahsereci, Maren; Hennig, Phillip (2016). "Probabilistic Line Searches for Stochastic Optimization". arXiv:1502.02846v4. ^ Kafka, Dominic; Wilke, Daniel N. (2021). "Resolving learning rates adaptively by locating stochastic non-negative associated gradient projection points using line searches". Journal of Global Optimization. 79: 111–152. arXiv:2001.05113. ^ Chae, Younghwan; Wilke, Daniel N. (2019). "Empirical study towards understanding line search approximations for training neural networks". arXiv:1909.06893. ^ Smith, Leslie N. (4 April 2017). "Cyclical Learning Rates for Training Neural Networks". arXiv:1506.01186 [cs.CV]. ^ Murphy, Kevin (2021). Probabilistic Machine Learning: An Introduction. Probabilistic Machine Learning: An Introduction. MIT Press. Retrieved 10 April 2021. ^ Brownlee, Jason (22 January 2019). "How to Configure the Learning Rate When Training Deep Learning Neural Networks". Machine Learning Mastery. Retrieved 4 January 2021. Further reading Géron, Aurélien (2017). "Gradient Descent". Hands-On Machine Learning with Scikit-Learn and TensorFlow. O'Reilly. pp. 113–124. ISBN 978-1-4919-6229-9. Plagianakos, V. P.; Magoulas, G. D.; Vrahatis, M. N. (2001). "Learning Rate Adaptation in Stochastic Gradient Descent". Advances in Convex Analysis and Global Optimization. Kluwer. pp. 433–444. ISBN 0-7923-6942-4. External links de Freitas, Nando (February 12, 2015). "Optimization". Deep Learning Lecture 6. University of Oxford – via YouTube. Retrieved from "

learning rate keras tensorflow. learning rate keras r. learning rate keras lstm. learning rate keras example. learning rate keras python. learning rate keras cnn. learning rate keras callback. change learning rate keras