Title: Extreme learning machine

URL: https://en.wikipedia.org/wiki/Extreme_learning_machine

PageID: 47378228

Categories: Category:Artificial neural networks

Supervised learning

Unsupervised learning

Semi-supervised learning

Self-supervised learning

Reinforcement learning

Meta-learning

Online learning

Batch learning

Curriculum learning

Rule-based learning

Neuro-symbolic AI

Neuromorphic engineering

Quantum machine learning

Classification

Generative modeling

Regression

Clustering

Dimensionality reduction

Density estimation

Anomaly detection

Data cleaning

AutoML

Association rules

Semantic analysis

Structured prediction

Feature engineering

Feature learning

Learning to rank

Grammar induction

Ontology learning

Multimodal learning

Apprenticeship learning

Decision trees

Ensembles Bagging Boosting Random forest

Bagging

Boosting

Random forest

k -NN

Linear regression

Naive Bayes

Artificial neural networks

Logistic regression

Perceptron

Relevance vector machine (RVM)

Support vector machine (SVM)

BIRCH

CURE

Hierarchical

k -means

Fuzzy

Expectation–maximization (EM)

DBSCAN

OPTICS

Mean shift

Factor analysis

CCA

ICA

LDA

NMF

PCA

PGD

t-SNE

SDL

Graphical models Bayes net Conditional random field Hidden Markov

Bayes net

Conditional random field

Hidden Markov

RANSAC

k -NN

Local outlier factor

Isolation forest

Autoencoder

Deep learning

Feedforward neural network

Recurrent neural network LSTM GRU ESN reservoir computing

LSTM

GRU

ESN

reservoir computing

Boltzmann machine Restricted

Restricted

GAN

Diffusion model

SOM

Convolutional neural network U-Net LeNet AlexNet DeepDream

U-Net

LeNet

AlexNet

DeepDream

Neural field Neural radiance field Physics-informed neural networks

Neural radiance field

Physics-informed neural networks

Transformer Vision

Vision

Mamba

Spiking neural network

Memtransistor

Electrochemical RAM (ECRAM)

Q-learning

Policy gradient

SARSA

Temporal difference (TD)

Multi-agent Self-play

Self-play

Active learning

Crowdsourcing

Human-in-the-loop

Mechanistic interpretability

RLHF

Coefficient of determination

Confusion matrix

Learning curve

ROC curve

Kernel machines

Bias–variance tradeoff

Computational learning theory

Empirical risk minimization

Occam learning

PAC learning

Statistical learning

VC theory

Topological deep learning

AAAI

ECML PKDD

NeurIPS

ICML

ICLR

IJCAI

ML

JMLR

Glossary of artificial intelligence

List of datasets for machine-learning research List of datasets in computer vision and image processing

List of datasets in computer vision and image processing

Outline of machine learning

v

t

e

Extreme learning machines are feedforward neural networks for classification , regression , clustering , sparse approximation , compression and feature learning with a single layer or multiple layers of hidden nodes, where the parameters of hidden nodes (not just the weights connecting inputs to hidden nodes) need to be tuned. These hidden nodes can be randomly assigned and never updated (i.e. they are random projection but with nonlinear transforms), or can be inherited from their ancestors without being changed. In most cases, the output weights of hidden nodes are usually learned in a single step, which essentially amounts to learning a linear model.

The name "extreme learning machine" (ELM) was given to such models by Guang-Bin Huang who originally proposed for the networks with any type of nonlinear piecewise continuous hidden nodes including biological neurons and different type of mathematical basis functions. [1] [2] The idea for artificial neural networks goes back to Frank Rosenblatt , who not only published a single layer Perceptron in 1958, [3] but also introduced a multilayer perceptron with 3 layers: an input layer, a hidden layer with randomized weights that did not learn, and a learning output layer. [4]

According to some researchers, these models are able to produce good generalization performance and learn thousands of times faster than networks trained using backpropagation . [5] In literature, it also shows that these models can outperform support vector machines in both classification and regression applications. [6] [1] [7]

History

From 2001-2010, ELM research mainly focused on the unified learning framework for "generalized" single-hidden layer feedforward neural networks (SLFNs), including but not limited to sigmoid networks, RBF networks, threshold networks, [8] trigonometric networks, fuzzy inference systems, Fourier series, [9] [10] Laplacian transform, wavelet networks, [11] etc. One significant achievement made in those years is to successfully prove the universal approximation and classification capabilities of ELM in theory. [9] [12] [13]

From 2010 to 2015, ELM research extended to the unified learning framework for kernel learning, SVM and a few typical feature learning methods such as Principal Component Analysis (PCA) and Non-negative Matrix Factorization (NMF). It is shown that SVM actually provides suboptimal solutions compared to ELM, and ELM can provide the whitebox kernel mapping, which is implemented by ELM random feature mapping, instead of the blackbox kernel used in SVM. PCA and NMF can be considered as special cases where linear hidden nodes are used in ELM. [14] [15]

From 2015 to 2017, an increased focus has been placed on hierarchical implementations [16] [17] of ELM. Additionally since 2011, significant biological studies have been made that support certain ELM theories. [18] [19] [20]

From 2017 onwards, to overcome low-convergence problem during training LU decomposition , Hessenberg decomposition and QR decomposition based approaches with regularization have begun to attract attention [21] [22] [23]

In 2017, Google Scholar Blog published a list of "Classic Papers: Articles That Have Stood The Test of Time". [24] Among these are two papers written about ELM which are shown in studies 2 and 7 from the "List of 10 classic AI papers from 2006". [25] [26] [27]

Algorithms

Given a single hidden layer of ELM, suppose that the output function of the i {\displaystyle i} -th hidden node is h i (x) = G (a i , b i , x) {\displaystyle h_{i}(\mathbf {x})=G(\mathbf {a} _{i},b_{i},\mathbf {x})} , where a i {\displaystyle \mathbf {a} _{i}} and b i {\displaystyle b_{i}} are the parameters of the i {\displaystyle i} -th hidden node. The output function of the ELM for single hidden layer feedforward networks (SLFN) with L {\displaystyle L} hidden nodes is:

f L (x) = ∑ i = 1 L β i h i (x) {\displaystyle f_{L}({\bf {x}})=\sum _{i=1}^{L}{\boldsymbol {\beta }}_{i}h_{i}({\bf {x}})} , where β i {\displaystyle {\boldsymbol {\beta }}_{i}} is the output weight of the i {\displaystyle i} -th hidden node.

h (x) = [h i (x) , . . . , h L (x)] {\displaystyle \mathbf {h} (\mathbf {x})=[h_{i}(\mathbf {x}),...,h_{L}(\mathbf {x})]} is the hidden layer output mapping of ELM. Given N {\displaystyle N} training samples, the hidden layer output matrix H {\displaystyle \mathbf {H} } of ELM is given as: H = [h (x 1) ■ h (x N)] = [G (a 1 , b 1 , x 1) ■ G (a L , b L , x 1) ■ ■ ■ G (a 1 , b 1 , x N) ■ G (a L , b L , x N)] {\displaystyle {\bf {H}}=\left[{\begin{matrix}{\bf {h}}({\bf {x}}_{1})\\\vdots \\{\bf {h}}({\bf {x}}_{N})\end{matrix}}\right]=\left[{\begin{matrix}G({\bf {a}}_{1},b_{1},{\bf {x}}_{1})&\cdots &G;({\bf {a}}_{L},b_{L},{\bf {x}}_{1})\\\vdots &\vdots &\vdots \\G({\bf {a}}_{1},b_{1},{\bf {x}}_{N})&\cdots &G;({\bf {a}}_{L},b_{L},{\bf {x}}_{N})\end{matrix}}\right]}

and T {\displaystyle \mathbf {T} } is the training data target matrix: T = [t 1 ■ t N] {\displaystyle {\bf {T}}=\left[{\begin{matrix}{\bf {t}}_{1}\\\vdots \\{\bf {t}}_{N}\end{matrix}}\right]}

Generally speaking, ELM is a kind of regularization neural networks but with non-tuned hidden layer mappings (formed by either random hidden nodes, kernels or other implementations), its objective function is:

Minimize: ■ β ■ p σ 1 + C ■ H β − T ■ q σ 2 {\displaystyle {\text{Minimize: }}\|{\boldsymbol {\beta }}\|_{p}^{\sigma _{1}}+C\|{\bf {H}}{\boldsymbol {\beta }}-{\bf {T}}\|_{q}^{\sigma _{2}}}

where σ 1 > 0 , σ 2 > 0 , p , q = 0 , 1 2 , 1 , 2 , ■ , + ∞ {\displaystyle \sigma _{1}>0,\sigma _{2}>0,p,q=0,{\frac {1}{2}},1,2,\cdots ,+\infty } .

Different combinations of σ 1 {\displaystyle \sigma _{1}} , σ 2 {\displaystyle \sigma _{2}} , p {\displaystyle p} and q {\displaystyle q} can be used and result in different learning algorithms for regression, classification, sparse coding, compression, feature learning and clustering.

As a special case, a simplest ELM training algorithm learns a model of the form (for single hidden layer sigmoid neural networks):

where W 1 is the matrix of input-to-hidden-layer weights, σ {\displaystyle \sigma } is an activation function, and W 2 is the matrix of hidden-to-output-layer weights. The algorithm proceeds as follows:

Fill W 1 with random values (e.g., Gaussian random noise);

estimate W 2 by least-squares fit to a matrix of response variables Y , computed using the pseudoinverse ⋅ + , given a design matrix X : W 2 = σ (W 1 X) + Y {\displaystyle \mathbf {W} _{2}=\sigma (\mathbf {W} _{1}\mathbf {X})^{+}\mathbf {Y} }

Architectures

In most cases, ELM is used as a single hidden layer feedforward network (SLFN) including but not limited to sigmoid networks, RBF networks, threshold networks, fuzzy inference networks, complex neural networks, wavelet networks, Fourier transform, Laplacian transform, etc. Due to its different learning algorithm implementations for regression, classification, sparse coding, compression, feature learning and clustering, multi ELMs have been used to form multi hidden layer networks, deep learning or hierarchical networks. [16] [17] [28]

A hidden node in ELM is a computational element, which need not be considered as classical neuron. A hidden node in ELM can be classical artificial neurons, basis functions, or a subnetwork formed by some hidden nodes. [12]

Theories

Both universal approximation and classification capabilities [6] [1] have been proved for ELM in literature. Especially, Guang-Bin Huang and his team spent almost seven years (2001-2008) on the rigorous proofs of ELM's universal approximation capability. [9] [12] [13]

Universal approximation capability

In theory, any nonconstant piecewise continuous function can be used as activation function in ELM hidden nodes, such an activation function need not be differential. If tuning the parameters of hidden nodes could make SLFNs approximate any target function f (x) {\displaystyle f(\mathbf {x})} , then hidden node parameters can be randomly generated according to any continuous distribution probability, and lim L → ∞ ■ ∑ i = 1 L β i h i (x) − f (x) ■ = 0 {\displaystyle \lim _{L\rightarrow \infty }\left\|\sum _{i=1}^{L}{\boldsymbol {\beta }}_{i}h_{i}({\bf {x}})-f({\bf {x}})\right\|=0} holds with probability one with appropriate output weights β {\displaystyle {\boldsymbol {\beta }}} .

Classification capability

Given any nonconstant piecewise continuous function as the activation function in SLFNs, if tuning the parameters of hidden nodes can make SLFNs approximate any target function f (x) {\displaystyle f(\mathbf {x})} , then SLFNs with random hidden layer mapping h (x) {\displaystyle \mathbf {h} (\mathbf {x})} can separate arbitrary disjoint regions of any shapes.

Neurons

A wide range of nonlinear piecewise continuous functions G (a , b , x) {\displaystyle G(\mathbf {a} ,b,\mathbf {x})} can be used in hidden neurons of ELM, for example:

Real domain

Sigmoid function: G (a , b , x) = 1 1 + exp ■ (− (a ⋅ x + b)) {\displaystyle G(\mathbf {a} ,b,\mathbf {x})={\frac {1}{1+\exp(-(\mathbf {a} \cdot \mathbf {x} +b))}}}

Fourier function: G (a , b , x) = sin ■ (a ⋅ x + b) {\displaystyle G(\mathbf {a} ,b,\mathbf {x})=\sin(\mathbf {a} \cdot \mathbf {x} +b)}

Hardlimit function: G (a , b , x) = { 1 , if a ⋅ x − b ≥ 0 0 , otherwise {\displaystyle G(\mathbf {a} ,b,\mathbf {x})={\begin{cases}1,&{\text{if }}{\bf {a}}\cdot {\bf {x}}-b\geq 0\\0,&{\text{otherwise}}\end{cases}}}

Gaussian function: G (a , b , x) = exp ■ (− b ■ x − a ■ 2) {\displaystyle G(\mathbf {a} ,b,\mathbf {x})=\exp(-b\|\mathbf {x} -\mathbf {a} \|^{2})}

Multiquadrics function: G (a , b , x) = (■ x − a ■ 2 + b 2) 1 / 2 {\displaystyle G(\mathbf {a} ,b,\mathbf {x})=(\|\mathbf {x} -\mathbf {a} \|^{2}+b^{2})^{1/2}}

Wavelet: G (a , b , x) = ■ a ■ − 1 / 2 Ψ (x − a b) {\displaystyle G(\mathbf {a} ,b,\mathbf {x})=\|a\|^{-1/2}\Psi \left({\frac {\mathbf {x} -\mathbf {a} }{b}}\right)} where Ψ {\displaystyle \Psi } is a single mother wavelet function.

Complex domain

Circular functions:

tan ■ (z) = e i z − e − i z i (e i z + e − i z) {\displaystyle \tan(z)={\frac {e^{iz}-e^{-iz}}{i(e^{iz}+e^{-iz})}}}

sin ■ (z) = e i z − e − i z 2 i {\displaystyle \sin(z)={\frac {e^{iz}-e^{-iz}}{2i}}}

Inverse circular functions:

arctan ■ (z) = ∫ 0 z d t 1 + t 2 {\displaystyle \arctan(z)=\int _{0}^{z}{\frac {dt}{1+t^{2}}}}

arccos ■ (z) = ∫ 0 z d t (1 − t 2) 1 / 2 {\displaystyle \arccos(z)=\int _{0}^{z}{\frac {dt}{(1-t^{2})^{1/2}}}}

Hyperbolic functions:

tanh ■ (z) = e z − e − z e z + e − z {\displaystyle \tanh(z)={\frac {e^{z}-e^{-z}}{e^{z}+e^{-z}}}}

sinh ■ (z) = e z − e − z 2 {\displaystyle \sinh(z)={\frac {e^{z}-e^{-z}}{2}}}

Inverse hyperbolic functions:

arctanh (z) = ∫ 0 z d t 1 − t 2 {\displaystyle {\text{arctanh}}(z)=\int _{0}^{z}{\frac {dt}{1-t^{2}}}}

arcsinh (z) = ∫ 0 z d t (1 + t 2) 1 / 2 {\displaystyle {\text{arcsinh}}(z)=\int _{0}^{z}{\frac {dt}{(1+t^{2})^{1/2}}}}

Reliability

The black-box character of neural networks in general and extreme learning machines (ELM) in particular is one of the major concerns that repels engineers from application in unsafe automation tasks. This particular issue was approached by means of several different techniques. One approach is to reduce the dependence on the random input. [29] [30] Another approach focuses on the incorporation of continuous constraints into the learning process of ELMs [31] [32] which are derived from prior knowledge about the specific task. This is reasonable, because machine learning solutions have to guarantee a safe operation in many application domains. The mentioned studies revealed that the special form of ELMs, with its functional separation and the linear read-out weights, is particularly well suited for the efficient incorporation of continuous constraints in predefined regions of the input space.

Controversy

There are two main complaints from academic community concerning this work, the first one is about "reinventing and ignoring previous ideas", the second one is about "improper naming and popularizing", as shown in some debates in 2008 and 2015. [33] In particular, it was pointed out in a letter [34] to the editor of IEEE Transactions on Neural Networks that the idea of using a hidden layer connected to the inputs by random untrained weights was already suggested in the original papers on RBF networks in the late 1980s; Guang-Bin Huang replied by pointing out subtle differences. [35] In a 2015 paper, [1] Huang responded to complaints about his invention of the name ELM for already-existing methods, complaining of "very negative and unhelpful comments on ELM in neither academic nor professional manner due to various reasons and intentions" and an "irresponsible anonymous attack which intends to destroy harmony research environment", arguing that his work "provides a unifying learning platform" for various types of neural nets, [1] including hierarchical structured ELM. [28] In 2015, Huang also gave a formal rebuttal to what he considered as "malign and attack." [36] Recent research replaces the random weights with constrained random weights. [6] [37]

Open sources

Matlab Library

Python Library [38]

See also

Reservoir computing

Random projection

Random matrix

References