Applied artificial intelligence pdf book download full version

I'm not robot!

Applied artificial intelligence pdf book download full version

Open AccessReview by 1, 1, 2, 1, 3, 3, 4, 5, 6, 7, 8, 9 and 6,10,11,* 1 Department of Computer Science & Engineering, International Institute of Information Technology, Bhubaneswar 751003, India 2 Department of Information Technology, Bharati Vidyapeeth College of Engineering, New Delhi 110063, India 3 National Institute of Pharmaceutical Education & Research, Ahmedabad 382355, India 4 Department of Biostatistics, Johns Hopkins University, Baltimore, MD 21218, USA 5 Department of Biomedical Engineering, North Eastern Hill University, Shilong 793022, India 6 Department of Electrical and Computer Engineering, Idaho State University, Pocatello, ID 83209, USA 7 Department of Radiology, Massachusetts General Hospital, Boston, MA 02114, USA 8 Department of Radiology, A.O.U., di Cagliari-Polo di Monserrato s.s., 09124 Cagliari, Italy 9 Minimally Invasive Urology Institute, Brown University, Providence, RI 02912, USA 10 Stroke Monitoring and Diagnostic Division, AtheroPoint™, Roseville, CA 95661, USA add Show full affiliation list remove Hide full affiliation list * Author to whom correspondence should be addressed. Academic Editor: Damiano Caruso Cancers 2022, 14(12), 2860; Received: 15 May 2022 / Revised: 3 June 2022 / Accepted: 7 June 2022 / Published: 9 June 2022 Simple Summary Recently, radiogenomics has played a significant role and offered a new understanding of cancer's biology and behavior in response to standard therapy. It also provides a more precise prognosis, investigation, and analysis of the patient's cancer. Over the years, Artificial Intelligence (AI) has provided a significant strength in radiogenomics. In this paper, we offer computational and oncological prospects of the role of AI in radiogenomics, as well as its offers, achievements, opportunities, and limitations in the current clinical practices. Radiogenomics, a combination of "Radiomics" and "Genomics," using Artificial Intelligence (AI) has recently emerged as the state-of-the-art science in precision medicine, especially in oncology care. Radiogenomics syndicates large-scale quantifiable data extracted from radiological medical images enveloped with personalized genomic phenotypes. It fabricates a prediction model through various AI methods to stratify the risk of patients, monitor therapeutic approaches, and assess clinical outcomes. It has recently shown tremendous achievements in prognosis, treatment planning, survival prediction, heterogeneity analysis, reoccurrence, and progression-free survival for human cancer study. Although AI has shown immense performance in oncology care in various clinical aspects, it has several challenges and limitations. The proposed review provides an overview of radiogenomics with the viewpoints on the role of AI in terms of its promises for computational as well as oncological aspects and offers achievements and opportunities in the era of precision medicine. The review also presents various recommendations to diminish these obstacles. View Full-Text Keywords: radiogenomics; cancer; oncology; artificial intelligence; machine learning; deep learning radiogenomics; cancer; oncology; artificial intelligence; machine learning; deep learning ►▼ Show Figures div data-cycle-log=false> This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited Share and Cite MDPI and ACS Style Saxena, S.; Jena, B.; Gupta, N.; Das, S.; Sarmah, D.; Bhattacharya, P.; Nath, T.; Paul, S.; Fouda, M.M.; Kalra, M.; Saba, L.; Pareek, G.; Suri, J.S. Role of Artificial Intelligence in Radiogenomics for Cancers in the Era of Precision Medicine. Cancers 2022, 14, 2860. AMA Style Saxena S, Jena B, Gupta N, Das S, Sarmah D, Bhattacharya P, Nath T, Paul S, Fouda MM, Kalra M, Saba L, Pareek G, Suri JS. Role of Artificial Intelligence in Radiogenomics for Cancers in the Era of Precision Medicine. Cancers. 2022; 14(12):2860. Chicago/Turabian Style Saxena, Sanjay, Biswajit Jena, Neha Gupta, Suchismita Das, Deepaneeta Sarmah, Pallab Bhattacharya, Tanmay Nath, Sudip Paul, Mostafa M. Fouda, Manudeep Kalra, Luca Saba, Gyan Pareek, and Jasjit S. Suri. 2022. "Role of Artificial Intelligence in Radiogenomics for Cancers in the Era of Precision Medicine" Cancers 14, no. 12: 2860. Article Metrics Let's create data fabric instead of data siloscloud management that requires less managementmore uptime for apps and downtime for peoplesecurity that protects your data anywhereautomated systems that make work less work165%more revenue growth for data-driven organizations compared to the competition[Data+Governance]Make data more accessible for enterprise analytics and AI so your team can make smarter, faster decisions.34%cost reduction in application migration and modernization[Open+Hybrid]Simplify operations across cloud platforms so you can build and scale applications faster.70%decrease in operating costs[Efficiency+Growth]Automate processes to make work simpler, faster and more rewarding for your team.Explore automation solutions10%increase in data breaches last year[Technology+Consulting]Transform your security strategy to protect your most valuable assets and proactively manage threats.Explore security solutions67%reduction in project delivery timelines[Collaboration+Technology]Transform your business with a trusted partner, open technologies and an empathetic approach.Do you have an idea that could transform your business? Let's make it a reality. Using the proven IBM Garage Methodology, our team can help prototype, test, tweak, adapt and scale — and turn your idea into powerful business outcomes.Co-create with IBM GarageWith a focus on style, performance and scale, IBM Consulting partnered with Audi UK to better anticipate customer needs, understand driver preferences and deliver on digital experiences for the next generation of Audi drivers.See how Audi delivers innovation fastersomething that changes everythingSubscribe to receive curated newsletters specific to your business interests. Get the latest news in technology, business and thought leadership – directly to your inbox. Computational model used in machine learning, based on connected, hierarchical functions Part of a series onMachine learningand data mining Problems Classification Clustering Regression Anomaly detection Data Cleaning 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 reservoir computing Restricted Boltzmann machine GAN SOM Convolutional neural network U-Net Transformer Vision Spiking neural network Memtransistor Electrochemical RAM (ECRAM) Reinforcement learning Q-learning SARSA Temporal difference (TD) Multi-agent Self-play Learning with humans Active learning Crowdsourcing Human-in-the-loop Model diagnostics Learning curve Theory Kernel machines 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 Part of a series onArtificial intelligence Major goals Artificial general intelligence Planning Computer vision General game playing Knowledge reasoning Machine learning Natural language processing Robotics Approaches Symbolic Deep learning Bayesian networks Evolutionary algorithms Philosophy Chinese room Friendly AI Control problem/Takeover Ethics Existential risk Turing test History Timeline Progress AI winter Technology Applications Projects Programming languages Glossary Glossary vte Complex systems Topics Self-organizationEmergence Collective behaviorSocial dynamics Collective intelligence Collective action Self-organized criticality Herd mentality Phase transition Agent-based modelling Synchronization Ant colony optimization Particle swarm optimization Swarm behaviour Collective consciousness NetworksScale-free networks Social network analysis Small-world networks Centrality Motifs Graph theory Scaling Robustness Systems biology Dynamic networks Adaptive networks Evolution and adaptationArtificial neural network Evolutionary computation Genetic algorithms Genetic programming Artificial life Machine learning Evolutionary developmental biology Artificial intelligence Evolutionary robotics Evolvability Pattern formationFractals Reaction–diffusion systems Partial differential equations Dissipative structures Percolation Cellular automata Spatial ecology Self-replication Geomorphology Systems theory and cyberneticsAutopoiesis Information theory Entropy Feedback Goal-oriented Homeostasis Operationalization Second-order cybernetics Self-reference System dynamics Systems science Systems thinking Sensemaking Variety Theory of computation Nonlinear dynamicsTime series analysis Ordinary differential equations Phase space Attractors Population dynamics Chaos Multistability Bifurcation Coupled map lattices Game theoryPrisoner's dilemma Rational choice theory Bounded rationality Evolutionary game theory vte Part of a series onNetwork science Theory Graph Complex network Contagion Small-world Scale-free Community structure Percolation Evolution Controllability Graph drawing Social capital Link analysis Optimization Reciprocity Closure Homophily Transitivity Preferential attachment Balance theory Network effect Social influence Network types Informational (computing) Telecommunication Transport Social Scientific collaboration Biological Artificial neural Interdependent Semantic Spatial Dependency Flow on-Chip Graphs Features Clique Component Cut Cycle Data structure Edge Loop Neighborhood Path Vertex Adjacency list / matrix Incidence list / matrix Types Bipartite Complete Directed Hyper Multi Random Weighted MetricsAlgorithms Centrality Degree Motif Clustering Degree distribution Assortativity Distance Modularity Efficiency Models Topology Random graph Erdős–Rényi Barabási–Albert Bianconi–Barabási Fitness model Watts–Strogatz Exponential random (ERGM) Random geometric (RGG) Hyperbolic (HGN) Hierarchical Stochastic block Blockmodeling Maximum entropy Soft configuration LFR Benchmark Dynamics Boolean network agent based Epidemic/SIR ListsCategories Topics Software Network scientists Category:Network theory Category:Graph theory vte An artificial neural network is an interconnected group of nodes, inspired by a simplification of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one artificial neuron to the input of another. Artificial neural networks (ANNs), usually simply called neural networks (NNs) or, more simply yet, neural nets,[1] are computing systems inspired by the biological neural networks that constitute animal brains. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron receives signals then processes them and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times. Training Neural networks learn (or are trained) by processing examples, each of which contains a known "input" and "result," forming probability-weighted associations between the two, which are stored within the data structure of the net itself. The training of a neural network from a given example is usually conducted by determining the difference between the processed output of the network (often a prediction) and a target output. This difference is the error. The network then adjusts its weighted associations according to a learning rule and using this error value. Successive adjustments will cause the neural network to produce output which is increasingly similar to the target output. After a sufficient number of these adjustments the training can be terminated based upon certain criteria. This is known as supervised learning. Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge of cats, for example, that they have fur, tails, whiskers, and cat-like faces. Instead, they automatically generate identifying characteristics from the examples that they process. History Main article: History of artificial neural networks Warren McCulloch and Walter Pitts[2] (1943) opened the subject by creating a computational model for neural networks.[3] In the late 1940s, D. O. Hebb[4] created a learning hypothesis based on the mechanism of neural plasticity that became known as Hebbian learning. Farley and Wesley A. Clark[5] (1954) first used computational machines, then called "calculators", to simulate a Hebbian network. In 1958, psychologist Frank Rosenblatt invented the perceptron, the first artificial neural network,[6][7][8][9] funded by the United States Office of Naval Research.[10] The first functional networks with many layers were published by Ivakhnenko and Lapa in 1965, as the Group Method of Data Handling.[11][12][13] The basics of continuous backpropagation[11][14][15][16] were derived in the context of control theory by Kelley[17] in 1960 and by Bryson in 1961,[18] using principles of dynamic programming. Thereafter research stagnated following Minsky and Papert (1969),[19] who discovered that basic perceptrons were incapable of processing the exclusive-or circuit and that computers lacked sufficient power to process useful neural networks. In 1970, Seppo Linnainmaa published the general method for automatic differentiation (AD) of discrete connected networks of nested differentiable functions.[20][21] In 1973, Dreyfus used backpropagation to adapt parameters of controllers in proportion to error gradients.[22] Werbos's (1975) backpropagation algorithm enabled practical training of multi-layer networks. In 1982, he applied Linnainmaa's AD method to neural networks in the way that became widely used.[14][23] The development of metal–oxide–semiconductor (MOS) very-large-scale integration (VLSI), in the form of complementary MOS (CMOS) technology, enabled increasing MOS transistor counts in digital electronics. This provided more processing power for the development of practical artificial neural networks in the 1980s.[24] In 1986 Rumelhart, Hinton and Williams showed that backpropagation learned interesting internal representations of words as feature vectors when trained to predict the next word in a sequence.[25] From 1988 onward,[26][27] the use of neural networks transformed the field of protein structure prediction, in particular when the first cascading networks were trained on profiles (matrices) produced by multiple sequence alignments.[28] In 1992, max-pooling was introduced to help with least-shift invariance and tolerance to deformation to aid 3D object recognition.[29][30][31] Schmidhuber adopted a multi-level hierarchy of networks (1992) pre-trained one level at a time by unsupervised learning and fine-tuned by backpropagation.[32] Neural networks' early successes included predicting the stock market and in 1995 a (mostly) self-driving car.[a][33] Geoffrey Hinton et al. (2006) proposed learning a high-level representation using successive layers of binary or real-valued latent variables with a restricted Boltzmann machine[34] to model each layer. In 2012, Ng and Dean created a network that learned to recognize higher-level concepts, such as cats, only from watching unlabeled images.[35] Unsupervised pre-training and increased computing power from GPUs and distributed computing allowed the use of larger networks, particularly in image and visual recognition problems, which became known as "deep learning".[36] Ciresan and colleagues (2010)[37] showed that despite the vanishing gradient problem, GPUs make backpropagation feasible for many-layered feedforward neural networks.[38] Between 2009 and 2012, ANNs began winning prizes in image recognition contests, approaching human level performance on various tasks, initially in pattern recognition and handwriting recognition.[39][40] For example, the bi-directional and multi-dimensional long short-term memory (LSTM)[41][42] of Graves et al. won three competitions in connected handwriting recognition in 2009 without any prior knowledge about the three languages to be learned.[41][42] Ciresan and colleagues built the first pattern recognizers to achieve human-competitive/superhuman performance[43] on benchmarks such as traffic sign recognition (IJCNN 2012). Models This section may be confusing or unclear to readers. Please help clarify the section. There might be a discussion about this on the talk page. (April 2017) (Learn how and when to remove this template message)Further information: Mathematics of artificial neural networks Neuron and myelinated axon, with signal flow from inputs at dendrites to outputs at axon terminals ANNs began as an attempt to exploit the architecture of the human brain to perform tasks that conventional algorithms had little success with. They soon reoriented towards improving empirical results, mostly abandoning attempts to remain true to their biological precursors. Neurons are connected to each other in various patterns, to allow the output of some neurons to become the input of others. The network forms a directed, weighted graph.[44] An artificial neural network consists of a collection of simulated neurons. Each neuron is a node which is connected to other nodes via links that correspond to biological axon-synapse-dendrite connections. Each link has a weight, which determines the strength of one node's influence on another.[45] Artificial neurons ANNs are composed of artificial neurons which are conceptually derived from biological neurons. Each artificial neuron has inputs and produces a single output which can be sent to multiple other neurons.[46] The inputs can be the feature values of a sample of external data, such as images or documents, or they can be the outputs of other neurons. The outputs of the final output neurons of the neural net accomplish the task, such as recognizing an object in an image. To find the output of the neuron we take the weighted sum of all the inputs, weighted by the weights of the connections from the inputs to the neuron. We add a bias term to this sum.[47] This weighted sum is sometimes called the activation. This weighted sum is then passed through a (usually nonlinear) activation function to produce the output. The initial inputs are external data, such as images and documents. The ultimate outputs accomplish the task, such as recognizing an object in an image.[48] Organization The neurons are typically organized into multiple layers, especially in deep learning. Neurons of one layer connect only to neurons of the immediately preceding and immediately following layers. The layer that receives external data is the input layer. The layer that produces the ultimate result is the output layer. In between them are zero or more hidden layers. Single layer and unlayered networks are also used. Between two layers, multiple connection patterns are possible. They can be 'fully connected', with every neuron in one layer connecting to every neuron in the next layer. They can be pooling, where a group of neurons in one layer connect to a single neuron in the next layer, thereby reducing the number of neurons in that layer.[49] Neurons with only such connections form a directed acyclic graph and are known as feedforward networks.[50] Alternatively, networks that allow connections between neurons in the same or previous layers are known as recurrent networks.[51] Hyperparameter Main article: Hyperparameter (machine learning) A hyperparameter is a constant parameter whose value is set before the learning process begins. The values of parameters are derived via learning. Examples of hyperparameters include learning rate, the number of hidden layers and batch size.[52] The values of some hyperparameters can be dependent on those of other hyperparameters. For example, the size of some layers can depend on the overall number of layers. Learning This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message)See also: Mathematical optimization, Estimation theory, and Machine learning Learning is the adaptation of the network to better handle a task by considering sample observations. Learning involves adjusting the weights (and optional thresholds) of the network to improve the accuracy of the result. This is done by minimizing the observed errors. Learning is complete when examining additional observations does not usefully reduce the error rate. Even after learning, the error rate typically does not reach 0. If after learning, the error rate is too high, the network typically must be redesigned. Practically this is done by defining a cost function that is evaluated periodically during learning. As long as its output continues to decline, learning continues. The cost is frequently defined as a statistic whose value can only be approximated. The outputs are actually numbers, so when the error is low, the difference between the output (almost certainly a cat) and the correct answer (cat) is small. Learning attempts to reduce the total of the differences across the observations. Most learning models can be viewed as a straightforward application of optimization theory and statistical estimation.[44][53] Learning rate The learning rate defines the size of the corrective steps that the model takes to adjust for errors in each observation.[54] A high learning rate shortens the training time, but with lower ultimate accuracy, while a lower learning rate takes longer, but with the potential for greater accuracy. Optimizations such as Quickprop are primarily aimed at speeding up error minimization, while other improvements mainly try to increase reliability. In order to avoid oscillation inside the network such as alternating connection weights, and to improve the rate of convergence, refinements use an adaptive learning rate that increases or decreases as appropriate.[55] The concept of momentum allows the balance between the gradient and the previous change to be weighted such that the weight adjustment depends to some degree on the previous change. A momentum close to 0 emphasizes the gradient, while a value close to 1 emphasizes the last change. Cost function While it is possible to define a cost function ad hoc, frequently the choice is determined by the function's desirable properties (such as convexity) or because it arises from the model (e.g. in a probabilistic model the model's posterior probability can be used as an inverse cost). Backpropagation Main article: Backpropagation Backpropagation is a method used to adjust the connection weights to compensate for each error found during learning. The error amount is effectively divided among the connections. Technically, backprop calculates the gradient (the derivative) of the cost function associated with a given state with respect to the weights. The weight updates can be done via stochastic gradient descent or other methods, such as Extreme Learning Machines,[56] "No-prop" networks,[57] training without backtracking,[58] "weightless" networks,[59][60] and non-connectionist neural networks.[citation needed] Learning paradigms This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message) The three major learning paradigms are supervised learning, unsupervised learning and reinforcement learning. They each correspond to a particular learning task Supervised learning Supervised learning uses a set of paired inputs and desired outputs. The learning task is to produce the desired output for each input. In this case the cost function is related to eliminating incorrect deductions.[61] A commonly used cost is the mean-squared error, which tries to minimize the average squared error between the network's output and the desired output. Tasks suited for supervised learning are pattern recognition (also known as classification) and regression (also known as function approximation). Supervised learning is also applicable to sequential data (e.g., for hand writing, speech and gesture recognition). This can be thought of as learning with a "teacher", in the form of a function that provides continuous feedback on the quality of solutions obtained thus far. Unsupervised learning In unsupervised learning, input data is given along with the cost function, some function of the data x {\displaystyle \textstyle x} and the network's output. The cost function is dependent on the task (the model domain) and any a priori assumptions (the implicit properties of the model, its parameters and the observed variables). As a trivial example, consider the model f (x) = a {\displaystyle \textstyle f(x)=a} where a {\displaystyle \textstyle a} is a constant and the cost C = E [(x − f (x)) 2] {\displaystyle \textstyle C=E[(x-f(x))^{2}]} . Minimizing this cost produces a value of a {\displaystyle \textstyle a} that is equal to the mean of the data. The cost function can be much more complicated. Its form depends on the application: for example, in compression it could be related to the mutual information between x {\displaystyle \textstyle x} and f (x) {\displaystyle \textstyle f(x)} , whereas in statistical modeling, it could be related to the posterior probability of the model given the data (note that in both of those examples those quantities would be maximized rather than minimized). Tasks that fall within the paradigm of unsupervised learning are in general estimation problems; the applications include clustering, the estimation of statistical distributions, compression and filtering. Reinforcement learning Main article: Reinforcement learning See also: Stochastic control In applications such as playing video games, an actor takes a string of actions, receiving a generally unpredictable response from the environment after each one. The goal is to win the game, i.e., generate the most positive (lowest cost) responses. In reinforcement learning, the aim is to weight the network (devise a policy) to perform actions that minimize long-term (expected cumulative) cost. At each point in time the agent performs an action and the environment generates an observation and an instantaneous cost, according to some (usually unknown) rules. The rules and the long-term cost usually only can be estimated. At any juncture, the agent decides whether to explore new actions to uncover their costs or to exploit prior learning to proceed more quickly. Formally the environment is modeled as a Markov decision process (MDP) with states s 1 , . . . , s n ∈ S {\displaystyle \textstyle {s_{1},...,s_{n}}\in S} and actions a 1 , . . . , a m ∈ A {\displaystyle \textstyle {a_{1},...,a_{m}}\in A} . Because the state transitions are not known, probability distributions are used instead: the instantaneous cost distribution P (c t | s t) {\displaystyle \textstyle P(c_{t}|s_{t})} , the observation distribution P (x t | s t) {\displaystyle \textstyle P(x_{t}|s_{t})} and the transition distribution P (s t + 1 | s t , a t) {\displaystyle \textstyle P(s_{t+1}|s_{t},a_{t})} , while a policy is defined as the conditional distribution over actions given the observations. Taken together, the two define a Markov chain (MC). The aim is to discover the lowest-cost MC. ANNs serve as the learning component in such applications.[62][63] Dynamic programming coupled with ANNs (giving neurodynamic programming)[64] has been applied to problems such as those involved in vehicle routing,[65] video games, natural resource management[66][67] and medicine[68] because of ANNs ability to mitigate losses of accuracy even when reducing the discretization grid density for numerically approximating the solution of control problems. Tasks that fall within the paradigm of reinforcement learning are control problems, games and other sequential decision making tasks. Self-learning Self-learning in neural networks was introduced in 1982 along with a neural network capable of self-learning named Crossbar Adaptive Array (CAA).[69] It is a system with only one input, situation s, and only one output, action (or behavior) a. It has neither external advice input nor external reinforcement input from the environment. The CAA computes, in a crossbar fashion, both decisions about actions and emotions (feelings) about encountered situations. The system is driven by the interaction between cognition and emotion.[70] Given the memory matrix, W =||w(a,s)||, the crossbar self-learning algorithm in each iteration performs the following computation: In situation s perform action a; Receive consequence situation s'; Compute emotion of being in consequence situation v(s'); Update crossbar memory w'(a,s) = w(a,s) + v(s'). The backpropagated value (secondary reinforcement) is the emotion toward the consequence situation. The CAA exists in two environments, one is behavioral environment where it behaves, and the other is genetic environment, where from it initially and only once receives initial emotions about to be encountered situations in the behavioral environment. Having received the genome vector (species vector) from the genetic environment, the CAA will learn a goal-seeking behavior, in the behavioral environment that contains both desirable and undesirable situations.[71] Neuroevolution Main article: Neuroevolution Neuroevolution can create neural network topologies and weights using evolutionary computation. It is competitive with sophisticated gradient descent approaches[citation needed]. One advantage of neuroevolution is that it may be less prone to get caught in "dead ends".[72] Stochastic neural network Stochastic neural networks originating from Sherrington–Kirkpatrick models are a type of artificial neural network built by introducing random variations into the network, either by giving the network's artificial neurons stochastic transfer functions, or by giving them stochastic weights. This makes them useful tools for optimization problems, since the random fluctuations help the network escape from local minima.[73] Other In a Bayesian framework, a distribution over the set of allowed models is chosen to minimize the cost. Evolutionary methods,[74] gene expression programming,[75] simulated annealing,[76] expectation-maximization, non-parametric methods and particle swarm optimization[77] are other learning algorithms. Convergent recursion is a learning algorithm for cerebellar model articulation controller (CMAC) neural networks.[78][79] Modes This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message) Two modes of learning are available: stochastic and batch. In stochastic learning, each input creates a weight adjustment. In batch learning weights are adjusted based on a batch of inputs, accumulating errors over the batch. Stochastic learning introduces "noise" into the process, using the local gradient calculated from one data point; this reduces the chance of the network getting stuck in local minima. However, batch learning typically yields a faster, more stable descent to a local minimum, since each update is performed in the direction of the batch's average error. A common compromise is to use "mini-batches", small batches with samples in each batch selected stochastically from the entire data set. Types Main article: Types of artificial neural networks ANNs have evolved into a broad family of techniques that have advanced the state of the art across multiple domains. The simplest types have one or more static components, including number of units, number of layers, unit weights and topology. Dynamic types allow one or more of these to evolve via learning. The latter are much more complicated, but can shorten learning periods and produce better results. Some types allow/require learning to be "supervised" by the operator, while others operate independently. Some types operate purely in hardware, while others are purely software and run on general purpose computers. Some of the main breakthroughs include: convolutional neural networks that have proven particularly successful in processing visual and other two-dimensional data;[80][81] long short-term memory avoid the vanishing gradient problem[82] and can handle signals that have a mix of low and high frequency components aiding large-vocabulary speech recognition,[83][84] text-to-speech synthesis,[85][14][86] and photo-real talking heads;[87] competitive networks such as generative adversarial networks in which multiple networks (of varying structure) compete with each other, on tasks such as winning a game[88] or on deceiving the opponent about the authenticity of an input.[89] Network design Main article: Neural architecture search Neural architecture search (NAS) uses machine learning to automate ANN design. Various approaches to NAS have designed networks that compare well with hand-designed systems. The basic search algorithm is to propose a candidate model, evaluate it against a dataset and use the results as feedback to teach the NAS network.[90] Available systems include AutoML and AutoKeras.[91] Design issues include deciding the number, type and connectedness of network layers, as well as the size of each and the connection type (full, pooling, ...). Hyperparameters must also be defined as part of the design (they are not learned), governing matters such as how many neurons are in each layer, learning rate, step, stride, depth, receptive field and padding (for CNNs), etc.[92] Use This section does not cite any sources. Please help improve this section by adding citations to reliable sources. Unsourced material may be challenged and removed. (November 2020) (Learn how and when to remove this template message) Using Artificial neural networks requires an understanding of their characteristics. Choice of model: This depends on the data representation and the application. Overly complex models are slow learning. Learning algorithm: Numerous trade-offs exist between learning algorithms. Almost any algorithm will work well with the correct hyperparameters for training on a particular data set. However, selecting and tuning an algorithm for training on unseen data requires significant experimentation. Robustness: If the model, cost function and learning algorithm are selected appropriately, the resulting ANN can become robust. ANN capabilities fall within the following broad categories:[citation needed] Function approximation, or regression analysis, including time series prediction, fitness approximation and modeling. Classification, including pattern and sequence recognition, novelty detection and sequential decision making.[93] Data processing, including filtering, clustering, blind source separation and compression. Robotics, including directing manipulators and prostheses. Applications Because of their ability to reproduce and model nonlinear processes, artificial neural networks have found applications in many disciplines. Application areas include system identification and control (vehicle control, trajectory prediction,[94] process control, natural resource management), quantum chemistry,[95] general game playing,[96] pattern recognition (radar systems, face identification, signal classification,[97] 3D reconstruction,[98] object recognition and more), sensor data analysis,[99] sequence recognition (gesture, speech, handwritten and printed text recognition[100]), medical diagnosis, finance[101] (e.g. automated trading systems), data mining, visualization, machine translation, social network filtering[102] and e-mail spam filtering. ANNs have been used to diagnose several types of cancers[103][104] and to distinguish highly invasive cancer cell lines from less invasive lines using only cell shape information.[105][106] ANNs have been used to accelerate reliability analysis of infrastructures subject to natural disasters[107][108] and to predict foundation settlements.[109] ANNs have also been used for building black-box models in geoscience: hydrology,[110][111] ocean modelling and coastal engineering,[112][113] and geomorphology.[114] ANNs have been employed in cybersecurity, with the objective to discriminate between legitimate activities and malicious ones. For example, machine learning has been used for classifying Android malware,[115] for identifying domains belonging to threat actors and for detecting URLs posing a security risk.[116] Research is underway on ANN systems designed for penetration testing, for detecting botnets,[117] credit cards frauds[118] and network intrusions. ANNs have been proposed as a tool to solve partial differential equations in physics[119][120][121] and simulate the properties of many-body open quantum systems.[122][123][124][125] In brain research ANNs have studied short-term behavior of individual neurons,[126] the dynamics of neural circuitry arise from interactions between individual neurons and how behavior can arise from abstract neural modules that represent complete subsystems. Studies considered long-and short-term plasticity of neural systems and their relation to learning and memory from the individual neuron to the system level. Theoretical properties Computational power The multilayer perceptron is a universal function approximator, as proven by the universal approximation theorem. However, the proof is not constructive regarding the number of neurons required, the network topology, the weights and the learning parameters. A specific recurrent architecture with rational-valued weights (as opposed to full precision real number-valued weights) has the power of a universal Turing machine,[127] using a finite number of neurons and standard linear connections. Further, the use of irrational values for weights results in a machine with super-Turing power.[128] Capacity A model's "capacity" property corresponds to its ability to model any given function. It is related to the amount of information that can be stored in the network and to the notion of complexity. Two notions of capacity are known by the community. The information capacity and the VC Dimension. The information capacity of a perceptron is intensively discussed in Sir David MacKay's book[129] which summarizes work by Thomas Cover.[130] The capacity of a network of standard neurons (not convolutional) can be derived by four rules[131] that derive from understanding a neuron as an electrical element. The information capacity captures the functions modelable by the network given any data as input. The second notion, is the VC dimension. VC Dimension uses the principles of measure theory and finds the maximum capacity under the best possible circumstances. This is, given input data in a specific form. As noted in,[129] the VC Dimension for arbitrary inputs is half the information capacity of a Perceptron. The VC Dimension for arbitrary points is sometimes referred to as Memory Capacity.[132] Convergence Models may not consistently converge on a single solution, firstly because local minima may exist, depending on the cost function and the model. Secondly, the optimization method used might not guarantee to converge when it begins far from any local minimum. Thirdly, for sufficiently large data or parameters, some methods become impractical. Another issue worthy to mention is that training may cross some Saddle point which may lead the convergence to the wrong direction. The convergence behavior of certain types of ANN architectures are more understood than others. When the width of network approaches to infinity, the ANN is well described by its first order Taylor expansion throughout training, and so inherits the convergence behavior of affine models.[133][134] Another example is when parameters are small, it is observed that ANNs often fits target functions from low to high frequencies. This behavior is referred to as the spectral bias, or frequency principle, of neural networks.[135][136][137][138] This phenomenon is the opposite to the behavior of some well studied iterative numerical schemes such as Jacobi method. Deeper neural networks have been observed to be more biased towards low frequency functions.[139] Generalization and statistics This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message) Applications whose goal is to create a system that generalizes well to unseen examples, face the possibility of over-training. This arises in convoluted or over-specified systems when the network capacity significantly exceeds the needed free parameters. Two approaches address over-training. The first is to use cross-validation and similar techniques to check for the presence of over-training and to select hyperparameters to minimize the generalization error. The second is to use some form of regularization. This concept emerges in a probabilistic (Bayesian) framework, where regularization can be performed by selecting a larger prior probability over simpler models; but also in statistical learning theory, where the goal is to minimize over two quantities: the 'empirical risk' and the 'structural risk', which roughly corresponds to the error over the training set and the predicted error in unseen data due to overfitting. Confidence analysis of a neural network Supervised neural networks that use a mean squared error (MSE) cost function can use formal statistical methods to determine the confidence of the trained model. The MSE on a validation set can be used as an estimate for variance. This value can then be used to calculate the confidence interval of network output, assuming a normal distribution. A confidence analysis made this way is statistically valid as long as the output probability distribution stays the same and the network is not modified. By assigning a softmax activation function, a generalization of the logistic function, on the output layer of the neural network (or a softmax component in a component-based network) for categorical target variables, the outputs can be interpreted as posterior probabilities. This is useful in classification as it gives a certainty measure on classifications. The softmax activation function is: y i = e x i ∑ j = 1 c e x j {\displaystyle y_{i}={\frac {e^{x_{i}}}{\sum _{j=1}^{c}e^{x_{j}}}}} Criticism Training A common criticism of neural networks, particularly in robotics, is that they require too much training for real-world operation.[citation needed] Potential solutions include randomly shuffling training examples, by using a numerical optimization algorithm that does not take too large steps when changing the network connections following an example, grouping examples in so-called mini-batches and/or introducing a recursive least squares algorithm for CMAC.[78] Theory A fundamental objection is that ANNs do not sufficiently reflect neuronal function. Backpropagation is a critical step, although no such mechanism exists in biological neural networks.[140] How information is coded by real neurons is not known. Sensor neurons fire action potentials more frequently with sensor activation and muscle cells pull more strongly when their associated motor neurons receive action potentials more frequently.[141] Other than the case of relaying information from a sensor neuron to a motor neuron, almost nothing of the principles of how information is handled by biological neural networks is known. A central claim of ANNs is that they embody new and powerful general principles for processing information. These principles are ill-defined. It is often claimed that they are emergent from the network itself. This allows simple statistical association (the basic function of artificial neural networks) to be described as learning or recognition. In 1997, Alexander Dewdney commented that, as a result, artificial neural networks have a "something-for-nothing quality, one that imparts a peculiar aura of laziness and a distinct lack of curiosity about just how good these computing systems are. No human hand (or mind) intervenes; solutions are found as if by magic; and no one, it seems, has learned anything".[142] One response to Dewdney is that neural networks handle many complex and diverse tasks, ranging from autonomously flying aircraft[143] to detecting credit card fraud to mastering the game of Go. Technology writer Roger Bridgman commented: Neural networks, for instance, are in the dock not only because they have been hyped to high heaven, (what hasn't?) but also because you could create a successful net without understanding how it worked: the bunch of numbers that captures its behaviour would in all probability be "an opaque, unreadable table...valueless as a scientific resource". In spite of his emphatic declaration that science is not technology, Dewdney seems here to pillory neural nets as bad science when most of those devising them are just trying to be good engineers. An unreadable table that a useful machine could read would still be well worth having.[144] Biological brains use both shallow and deep circuits as reported by brain anatomy,[145] displaying a wide variety of invariance. Weng[146] argued that the brain self-wires largely according to signal statistics and therefore, a serial cascade cannot catch all major statistical dependencies. Hardware Large and effective neural networks require considerable computing resources.[147] While the brain has hardware tailored to the task of processing signals through a graph of neurons, simulating even a simplified neuron on von Neumann architecture may consume vast amounts of memory and storage. Furthermore, the designer often needs to transmit signals through many of these connections and their associated neurons – which require enormous CPU power and time. Schmidhuber noted that the resurgence of neural networks in the twenty-first century is largely attributable to advances in hardware: from 1991 to 2015, computing power, especially as delivered by GPGPUs (on GPUs), has increased around a million-fold, making the standard backpropagation algorithm feasible for training networks that are several layers deeper than before.[11] The use of accelerators such as FPGAs and GPUs can reduce training times from months to days.[147] Neuromorphic engineering or a physical neural network addresses the hardware difficulty directly, by constructing non-von-Neumann chips to directly implement neural networks in circuitry. Another type of chip optimized for neural network processing is called a Tensor Processing Unit, or TPU.[148] Practical counterexamples Analyzing what has been learned by an ANN is much easier than analyzing what has been learned by a biological neural network. Furthermore, researchers involved in exploring learning algorithms for neural networks are gradually uncovering general principles that allow a learning machine to be successful. For example, local vs. non-local learning and shallow vs. deep architecture.[149] Hybrid approaches Advocates of hybrid models (combining neural networks and symbolic approaches), claim that such a mixture can better capture the mechanisms of the human mind.[150] Gallery A single-layer feedforward artificial neural network. Arrows originating from x 2 {\displaystyle \scriptstyle x_{2}} are omitted for clarity. There are p inputs to this network and q outputs. In this system, the value of the qth output, y q {\displaystyle \scriptstyle y_{q}} would be calculated as y q = K ∗ (∑ (x i ∗ w i q) − b q) {\displaystyle \scriptstyle y_{q}=K*(\sum (x_{i}*w_{iq})-b_{q})} A two-layer feedforward artificial neural network. An artificial neural network. An ANN dependency graph. A single-layer feedforward artificial neural network with 4 inputs, 6 hidden and 2 outputs. Given position state and direction outputs wheel based control values. A two-layer feedforward artificial neural network with 8 inputs, 2x8 hidden and 2 outputs. Given position state, direction and other environment values outputs thruster based control values. Parallel pipeline structure of CMAC neural network. This learning algorithm can converge in one step. See also ADALINE Autoencoder Biologically inspired computing Blue Brain Project Catastrophic interference Cognitive architecture Connectionist expert system Connectomics Large width limits of neural networks Machine learning concepts Neural gas Neural network software Optical neural network Parallel distributed processing Recurrent neural networks Spiking neural network Tensor product network Notes ^ Steering for the 1995 "No Hands Across America" required "only a few human assists". References ^ Hardesty, Larry (14 April 2017). "Explained: Neural networks". MIT News Office. Retrieved 2 June 2022. ^ McCulloch, Warren; Walter Pitts (1943). "A Logical Calculus of Ideas Immanent in Nervous Activity". Bulletin of Mathematical Biophysics. 5 (4): 115–133. doi:10.1007/BF02478259. ^ Kleene, S.C. (1956). "Representation of Events in Nerve Nets and Finite Automata". Annals of Mathematics Studies. No. 34. Princeton University Press. pp. 3–41. Retrieved 17 June 2017. ^ Hebb, Donald (1949). The Organization of Behavior. New York: Wiley. ISBN 978-1-135-63190-1. ^ Farley, B.G.; W.A. Clark (1954). "Simulation of Self-Organizing Systems by Digital Computer". IRE Transactions on Information Theory. 4 (4): 76–84. doi:10.1109/TIT.1954.1057468. ^ Haykin (2008) Neural Networks and Learning Machines, 3rd edition ^ Rosenblatt, F. (1958). "The Perceptron: A Probabilistic Model For Information Storage And Organization in the Brain". Psychological Review. 65 (6): 386–408. CiteSeerX 10.1.1.588.3775. doi:10.1037/h0042519. PMID 13602029. ^ Werbos, P.J. (1975). Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. ^ Rosenblatt, Frank (1957). "The Perceptron—a perceiving and recognizing automaton". Report 85-460-1. Cornell Aeronautical Laboratory. ^ Olazaran, Mikel (1996). "A Sociological Study of the Official History of the Perceptrons Controversy". Social Studies of Science. 26 (3): 611–659. doi:10.1177/030631296026003005. JSTOR 285702. S2CID 16786738. ^ a b c Schmidhuber, J. (2015). "Deep Learning in Neural Networks: An Overview". Neural Networks. 61: 85–117. arXiv:1404.7828. doi:10.1016/j.neunet.2014.09.003. PMID 25462637. S2CID 11715509. ^ Ivakhnenko, A. G. (1973). Cybernetic Predicting Devices. CCM Information Corporation. ^ Ivakhnenko, A. G.; Grigor'evich Lapa, Valentin (1967). Cybernetics and forecasting techniques. American Elsevier Pub. Co. ^ a b c Schmidhuber, Jürgen (2015). "Deep Learning". Scholarpedia. 10 (11): 85–117. Bibcode:2015SchpJ..1032832S. doi:10.4249/scholarpedia.32832. ^ Dreyfus, Stuart E. (1 September 1990). "Artificial neural networks, back propagation, and the Kelley-Bryson gradient procedure". Journal of Guidance, Control, and Dynamics. 13 (5): 926–928. Bibcode:1990JGCD...13..926D. doi:10.2514/3.25422. ISSN 0731-5090. ^ Mizutani, E.; Dreyfus, S.E.; Nishio, K. (2000). "On derivation of MLP backpropagation from the Kelley-Bryson optimal-control gradient formula and its application". Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium. IEEE: 167–172 vol.2. doi:10.1109/ijcnn.2000.857892. ISBN 0-7695-0619-4. S2CID 351146. ^ Kelley, Henry J. (1960). "Gradient theory of optimal flight paths". ARS Journal. 30 (10): 947–954. doi:10.2514/8.5282. ^ "A gradient method for optimizing multi-stage allocation processes". Proceedings of the Harvard Univ. Symposium on digital computers and their applications. April 1961. ^ Minsky, Marvin; Papert, Seymour (1969). Perceptrons: An Introduction to Computational Geometry. MIT Press. ISBN 978-0-262-63022-1. ^ Linnainmaa, Seppo (1970). The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors (Masters) (in Finnish). University of Helsinki. pp. 6–7. ^ Linnainmaa, Seppo (1976). "Taylor expansion of the accumulated rounding error". BIT Numerical Mathematics. 16 (2): 146–160. doi:10.1007/bf01931367. S2CID 122357351. ^ Dreyfus, Stuart (1973). "The computational solution of optimal control problems with time lag". IEEE Transactions on Automatic Control. 18 (4): 383–385. doi:10.1109/tac.1973.1100330. ^ Werbos, Paul (1982). "Applications of advances in nonlinear sensitivity analysis" (PDF). System modeling and optimization. Springer. pp. 762–770. ^ Mead, Carver A.; Ismail, Mohammed (8 May 1989). Analog VLSI Implementation of Neural Systems (PDF). The Kluwer International Series in Engineering and Computer Science. Vol. 80. Norwell, MA: Kluwer Academic Publishers. doi:10.1007/978-1-4613-1639-8. ISBN 978-1-4613-1639-8. ^ David E. Rumelhart, Geoffrey E. Hinton & Ronald J. Williams , "Learning representations by back-propagating errors ," Nature', 323, pages 533–536 1986. ^ Qian, Ning, and Terrence J. Sejnowski. "Predicting the secondary structure of globular proteins using neural network models." Journal of molecular biology 202, no. 4 (1988): 865-884. ^ Bohr, Henrik, Jakob Bohr, Søren Brunak, Rodney MJ Cotterill, Benny Lautrup, Leif Nørskov, Ole H. Olsen, and Steffen B. Petersen. "Protein secondary structure and homology by neural networks The α-helices in rhodopsin." FEBS letters 241, (1988): 223-228 ^ Rost, Burkhard, and Chris Sander. "Prediction of protein secondary structure at better than 70% accuracy." Journal of molecular biology 232, no. 2 (1993): 584-599. ^ J. Weng, N. Ahuja and T. S. Huang, "Cresceptron: a self-organizing neural network which grows adaptively," Proc. International Joint Conference on Neural Networks, Baltimore, Maryland, vol I, pp. 576–581, June 1992. ^ J. Weng, N. Ahuja and T. S. Huang, "Learning recognition and segmentation of 3-D objects from 2-D images," Proc. 4th International Conf. Computer Vision, Berlin, Germany, pp. 121–128, May 1993. ^ J. Weng, N. Ahuja and T. S. Huang, "Learning recognition and segmentation using the Cresceptron," International Journal of Computer Vision, vol. 25, no. 2, pp. 105–139, Nov. 1997. ^ J. Schmidhuber., "Learning complex, extended sequences using the principle of history compression," Neural Computation, 4, pp. 234–242, 1992. ^ Domingos, Pedro (22 September 2015). "chapter 4". The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World. Basic Books. ISBN 978-0465065707. ^ Smolensky, P. (1986). "Information processing in dynamical systems: Foundations of harmony theory.". In D. E. Rumelhart; J. L. McClelland; PDP Research Group (eds.). Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1. pp. 194–281. ISBN 978-0-262-68053-0. ^ Ng, Andrew; Dean, Jeff (2012). "Building High-level Features Using Large Scale Unsupervised Learning". arXiv:1112.6209 [cs.LG]. ^ Ian Goodfellow and Yoshua Bengio and Aaron Courville (2016). Deep Learning. MIT Press. ^ Cireşan, Dan Claudiu; Meier, Ueli; Gambardella, Luca Maria; Schmidhuber, Jürgen (21 September 2010). "Deep, Big, Simple Neural Nets for Handwritten Digit Recognition". Neural Computation. 22 (12): 3207–3220. arXiv:1003.0358. doi:10.1162/neco_a_00052. ISSN 0899-7667. PMID 20858131. S2CID 1918673. ^ Dominik Scherer, Andreas C. Müller, and Sven Behnke: "Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition," In 20th International Conference Artificial Neural Networks (ICANN), pp. 92–101, 2010. doi:10.1007/978-3-642-15825-4_10. ^ 2012 Kurzweil AI Interview Archived 31 August 2018 at the Wayback Machine with Jürgen Schmidhuber on the eight competitions won by his Deep Learning team 2009–2012 ^ "How bio-inspired deep learning keeps winning competitions | KurzweilAI". www.kurzweilai.net. Archived from the original on 31 August 2018. Retrieved 16 June 2017. ^ a b Graves, Alex; Schmidhuber, Jürgen (2009). "Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks" (PDF). In Koller, D.; Schuurmans, Dale; Bengio, Yoshua; Bottou, L. (eds.). Advances in Neural Information Processing Systems 21 (NIPS 2008). Neural Information Processing Systems (NIPS) Foundation. pp. 545–552. ISBN 9781605609492. ^ a b Graves, A.; Liwicki, M.; Fernandez, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. (May 2009). "A Novel Connectionist System for Unconstrained Handwriting Recognition" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (5): 855–868. CiteSeerX 10.1.1.139.4502. doi:10.1109/tpami.2008.137. ISSN 0162-8828. PMID 19299860. S2CID 14635907. ^ Ciresan, Dan; Meier, U.; Schmidhuber, J. (June 2012). Multi-column deep neural networks for image classification. 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3642–3649. arXiv:1202.2745. Bibcode:2012arXiv1202.2745C. CiteSeerX 10.1.1.300.3283. doi:10.1109/cvpr.2012.6248110. ISBN 978-1-4673-1228-8. S2CID 2161592. ^ a b Zell, Andreas (2003). "chapter 5.2". Simulation neuronaler Netze [Simulation of Neural Networks] (in German) (1st ed.). Addison-Wesley. ISBN 978-3-89319-554-1. OCLC 249017987. ^ Artificial intelligence (3rd ed.). Addison-Wesley Pub. Co. 1992. ISBN 0-201-53377-4. ^ Abbod, Maysam F (2007). "Application of Artificial Intelligence to the Management of Urological Cancer". The Journal of Urology. 178 (4): 1150–1156. doi:10.1016/j.juro.2007.05.122. PMID 17698099. ^ DAWSON, CHRISTIAN W (1998). "An artificial neural network approach to rainfall-runoff modelling". Hydrological Sciences Journal. 43 (1): 47–66. doi:10.1080/02626669809492102. ^ "The Machine Learning Dictionary". www.cse.unsw.edu.au. Archived from the original on 26 August 2018. Retrieved 4 November 2009. ^ Ciresan, Dan; Ueli Meier; Jonathan Masci; Luca M. Gambardella; Jurgen Schmidhuber (2011). "Flexible, High Performance Convolutional Neural Networks for Image Classification" (PDF). Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence-Volume Volume Two. 2: 1237–1242. Archived (PDF) from the original on 5 April 2022. Retrieved 7 July 2022. ^ Zell, Andreas (1994). Simulation Neuronaler Netze [Simulation of Neural Networks] (in German) (1st ed.). Addison-Wesley. p. 73. ISBN 3-89319-554-8. ^ Miljanovic, Milos (February–March 2012). "Comparative analysis of Recurrent and Finite Impulse Response Neural Networks in Time Series Prediction" (PDF). Indian Journal of Computer and Engineering. 3 (1). ^ Lau, Suki (10 July 2017). "A Walkthrough of Convolutional Neural Network – Hyperparameter Tuning". Medium. Retrieved 23 August 2019. ^ Kelleher, John D.; Mac Namee, Brian; D'Arcy, Aoife (2020). "7-8". Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies (2nd ed.). Cambridge, MA. ISBN 978-0-262-36110-1. OCLC 1162184998. ^ Wei, Jiakai (26 April 2019). "Forget the Learning Rate, Decay Loss". arXiv:1905.00094 [cs.LG]. ^ Li, Y.; Fu, Y.; Li, H.; Zhang, S. W. (1 June 2009). The Improved Training Algorithm of Back Propagation Neural Network with Self-adaptive Learning Rate. 2009 International Conference on Computational Intelligence and Natural Computing. Vol. 1. pp. 73–76. doi:10.1109/CINC.2009.111. ISBN 978-0-7695-3645-3. S2CID 10557754. ^ Huang, Guang-Bin; Zhu, Qin-Yu; Siew, Chee-Kheong (2006). "Extreme learning machine: theory and applications". Neurocomputing. 70 (1): 489–501. CiteSeerX 10.1.1.217.3692. doi:10.1016/j.neucom.2005.12.126. ^ Widrow, Bernard; et al. (2013). "The no-prop algorithm: A new learning algorithm for multilayer neural networks". Neural Networks. 37: 182–188. doi:10.1016/j.neunet.2012.09.020. PMID 23140797. ^ Ollivier, Yann; Charpiat, Guillaume (2015). "Training recurrent networks without backtracking". arXiv:1507.07680 [cs.NE]. ^ Hinton, G. E. (2010). "A Practical Guide to Training Restricted Boltzmann Machines". Tech. Rep. UTML TR 2010-003. ^ ESANN. 2009.[full citation needed] ^ Ojha, Varun Kumar; Abraham, Ajith; Snášel, Václav (1 April 2017). "Metaheuristic design of feedforward neural networks: A review of two decades of research". Engineering Applications of Artificial Intelligence. 60: 97–116. arXiv:1705.05584. Bibcode:2017arXiv170505584O. doi:10.1016/j.engappai.2017.01.013. S2CID 27910748. ^ Dominic, S.; Das, R.; Whitley, D.; Anderson, C. (July 1991). "Genetic reinforcement learning for neural networks". IJCNN-91-Seattle International Joint Conference on Neural Networks. IJCNN-91-Seattle International Joint Conference on Neural Networks. Seattle, Washington, USA: IEEE. pp. 71–76. doi:10.1109/IJCNN.1991.155315. ISBN 0-7803-0164-1. ^ Hoskins, J.C.; Himmelblau, D.M. (1992). "Process control via artificial neural networks and reinforcement learning". Computers & Chemical Engineering. 16 (4): 241–251. doi:10.1016/0098-1354(92)80045-B. ^ Bertsekas, D.P.; Tsitsiklis, J.N. (1996). Neuro-dynamic programming. Athena Scientific. p. 512. ISBN 978-1-886529-10-6. ^ Secomandi, Nicola (2000). "Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands". Computers & Operations Research. 27 (11–12): 1201–1225. CiteSeerX 10.1.1.392.4034. doi:10.1016/S0305-0548(99)00146-X. ^ de Rigo, D.; Rizzoli, A. E.; Soncini-Sessa, R.; Weber, E.; Zenesi, P. (2001). "Neuro-dynamic programming for the efficient management of reservoir networks". Proceedings of MODSIM 2001, International Congress on Modelling and Simulation. MODSIM 2001, International Congress on Modelling and Simulation. Canberra, Australia: Modelling and Simulation Society of Australia and New Zealand. doi:10.5281/zenodo.7481. ISBN 0-86740-525-2. ^ Damas, M.; Salmeron, M.; Diaz, A.; Ortega, J.; Prieto, A.; Olivares, G. (2000). "Genetic algorithms and neuro-dynamic programming: application to water supply networks". Proceedings of 2000 Congress on Evolutionary Computation. 2000 Congress on Evolutionary Computation. Vol. 1. La Jolla, California, USA: IEEE. pp. 7–14. doi:10.1109/CEC.2000.870269. ISBN 0-7803-6375-2. ^ Deng, Geng; Ferris, M.C. (2008). Neuro-dynamic programming for fractionated radiotherapy planning. Springer Optimization and Its Applications. Vol. 12. pp. 47–70. CiteSeerX 10.1.1.137.8288. doi:10.1007/978-0-387-73299-2_3. ISBN 978-0-387-73298-5. ^ Bozinovski, S. (1982). "A self-learning system using secondary reinforcement". In R. Trappl (ed.) Cybernetics and Systems Research: Proceedings of the Sixth European Meeting on Cybernetics and Systems Research. North Holland. pp. 397–402. ISBN 978-0-444-86488-8. ^ Bozinovski, S. (2014) "Modeling mechanisms of cognition-emotion interaction in artificial neural networks, since 1981." Procedia Computer Science p. 255-263 ^ Bozinovski, Stevo; Bozinovska, Liljana (2001). "Self-learning agents: A connectionist theory of emotion based on crossbar value judgment". Cybernetics and Systems. 32 (6): 637–667. doi:10.1080/01969720118145. S2CID 8944741. ^ "Artificial intelligence can 'evolve' to solve problems". Science | AAAS. 10 January 2018. Retrieved 7 February 2018. ^ Turchetti, Claudio (2004), Stochastic Models of Neural Networks, Frontiers in artificial intelligence and applications: Knowledge-based intelligent engineering systems, vol. 102, IOS Press, ISBN 9781586033880 ^ de Rigo, D.; Castelletti, A.; Rizzoli, A. E.; Soncini-Sessa, R.; Weber, E. (January 2005). "A selective improvement technique for fastening Neuro-Dynamic Programming in Water Resources Network Management". In Pavel Zítek (ed.). Proceedings of the 16th IFAC World Congress – IFAC-PapersOnLine. 16th IFAC World Congress. Vol. 16. Prague, Czech Republic: IFAC. pp. 7–12. doi:10.3182/20050703-6-CZ-1902.02172. hdl:11311/255236. ISBN 978-3-902661-75-3. Retrieved 30 December 2011. ^ Ferreira, C. (2006). "Designing Neural Networks Using Gene Expression Programming". In A. Abraham; B. de Baets; M. Köppen; B. Nickolay (eds.). Applied Soft Computing Technologies: The Challenge of Complexity (PDF). Springer-Verlag. pp. 517–536. ^ Da, Y.; Xiurun, G. (July 2005). "An improved PSO-based ANN with simulated annealing technique". In T. Villmann (ed.). New Aspects in Neurocomputing: 11th European Symposium on Artificial Neural Networks. Vol. 63. Elsevier. pp. 527–533. doi:10.1016/j.neucom.2004.07.002. Archived from the original on 25 April 2012. Retrieved 30 December 2011. ^ Wu, J.; Chen, E. (May 2009). "A Novel Nonparametric Regression Ensemble for Rainfall Forecasting Using Particle Swarm Optimization Technique Coupled with Artificial Neural Network". In Wang, H.; Shen, Y.; Huang, T.; Zeng, Z. (eds.). 6th International Symposium on Neural Networks, ISNN 2009. Lecture Notes in Computer Science. Vol. 5553. Springer. pp. 49–58. doi:10.1007/978-3-642-01513-7_6. ISBN 978-3-642-01215-0. Archived from the original on 31 December 2014. Retrieved 1 January 2012. ^ a b Ting Qin; Zonghai Chen; Haitao Zhang; Sifu Li; Wei Xiang; Ming Li (2004). "A learning algorithm of CMAC based on RLS" (PDF). Neural Processing Letters. 19 (1): 49–61. doi:10.1023/B:NEPL.0000016847.18175.60. S2CID 6233899. ^ Ting Qin; Haitao Zhang; Zonghai Chen; Wei Xiang (2005). "Continuous CMAC-QRLS and its systolic array" (PDF). Neural Processing Letters. 22 (1): 1–16. doi:10.1007/s11063-004-2694-0. S2CID 16095286. ^ LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989). "Backpropagation Applied to Handwritten Zip Code Recognition". Neural Computation. 1 (4): 541–551. doi:10.1162/neco.1989.1.4.541. S2CID 41312633. ^ Yann LeCun (2016). Slides on Deep Learning Online ^ Hochreiter, Sepp; Schmidhuber, Jürgen (1 November 1997). "Long Short-Term Memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. ISSN 0899-7667. PMID 9377276. S2CID 1915014. ^ Sak, Hasim; Senior, Andrew; Beaufays, Francoise (2014). "Long Short-Term Memory recurrent neural network architectures for large scale acoustic modeling" (PDF). Archived from the original (PDF) on 24 April 2018. ^ Li, Xiangang; Wu, Xihong (15 October 2014). "Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition". arXiv:1410.4281 [cs.CL]. ^ Fan, Y.; Qian, Y.; Xie, F.; Soong, F. K. (2014). "TTS synthesis with bidirectional LSTM based Recurrent Neural Networks". Proceedings of the Annual Conference of the International Speech Communication Association, Interspeech: 1964–1968. Retrieved 13 June 2017. ^ Zen, Heiga; Sak, Hasim (2015). "Unidirectional Long Short-Term Memory Recurrent Neural Network with Recurrent Output Layer for Low-Latency Speech Synthesis" (PDF). Google.com. ICASSP. pp. 4470–4474. ^ Fan, Bo; Wang, Lijuan; Soong, Frank K.; Xie, Lei (2015). "Photo-Real Talking Head with Deep Bidirectional LSTM" (PDF). Proceedings of ICASSP. ^ Silver, David; Hubert, Thomas; Schrittwieser, Julian; Antonoglou, Ioannis; Lai, Matthew; Guez, Arthur; Lanctot, Marc; Sifre, Laurent; Kumaran, Dharshan; Graepel, Thore; Lillicrap, Timothy; Simonyan, Karen; Hassabis, Demis (5 December 2017). "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm". arXiv:1712.01815 [cs.AI]. ^ Goodfellow, Ian; Pouget-Abadie, Jean; Mirza, Mehdi; Xu, Bing; Warde-Farley, David; Ozair, Sherjil; Courville, Aaron; Bengio, Yoshua (2014). Generative Adversarial Networks (PDF). Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014). pp. 2672–2680. ^ Zoph, Barret; Le, Quoc V. (4 November 2016). "Neural Architecture Search with Reinforcement Learning". arXiv:1611.01578 [cs.LG]. ^ Haifeng Jin; Qingquan Song; Xia Hu (2019). "Auto-keras: An efficient neural architecture search system". Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM. Retrieved 21 August 2019 – via autokeras.com. ^ Claesen, Marc; De Moor, Bart (2015). "Hyperparameter Search in Machine Learning". arXiv:1502.02127 [cs.LG]. Bibcode:2015arXiv150202127C ^ Turek, Fred D. (March 2007). "Introduction to Neural Net Machine Vision". Vision Systems Design. 12 (3). Retrieved 5 March 2013. ^ Zissis, Dimitrios (October 2015). "A cloud based architecture capable of perceiving and predicting multiple vessel behaviour". Applied Soft Computing. 35: 652–661. doi:10.1016/j.asoc.2015.07.002. ^ Roman M. Balabin; Ekaterina I. Lomakina (2009). "Neural network approach to quantum-chemistry data: Accurate prediction of density functional theory energies". J. Chem. Phys. 131 (7): 074104. Bibcode:2009JChPh.131g4104B. doi:10.1063/1.3206326. PMID 19708729. ^ Silver, David; et al. (2016). "Mastering the game of Go with deep neural networks and tree search" (PDF).

Nature. 529 (7587): 484–489. Bibcode:2016Natur.529..484S. doi:10.1038/nature16961. PMID 26819042. S2CID 515925. ^ Sengupta, Nandini; Sahidullah, Md; Saha, Goutam (August 2016). "Lung sound classification using cepstral-based statistical features". Computers in Biology and Medicine. 75 (1): 118–129. doi:10.1016/j.compbiomed.2016.05.013. PMID 27286184. ^ Choy, Christopher B., et al. "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction." European conference on computer vision. Springer, Cham, 2016. ^ Gessler, Josef (August 2021). "Sensor for food analysis applying impedance spectroscopy and artificial neural networks". RiuNet UPV (1): 8–12. ^ Maitra, D. S.; Bhattacharya, U.; Parui, S. K. (August 2015). "CNN based common approach to handwritten character recognition of multiple scripts". 2015 13th International Conference on Document Analysis and Recognition (ICDAR): 1021–1025. doi:10.1109/ICDAR.2015.7333916. ISBN 978-1-4799-1805-8. S2CID 25739012. ^ French, Jordan (2016). "The time traveller's CAPM". Investment Analysts Journal. 46 (2): 81–96. doi:10.1080/10293523.2016.1255469. S2CID 157962452. ^ Schechner, Sam (15 June 2017). "Facebook Boosts A.I. to Block Terrorist Propaganda". Wall Street Journal. ISSN 0099-9660. Retrieved 16 June 2017. ^ Ganesan, N (2010). "Application of Neural Networks in Diagnosing Cancer Disease Using Demographic Data". International Journal of Computer Applications. 1 (26): 81–97. Bibcode:2010IJCA....1z..81G. doi:10.5120/476-783. ^ Bottaci, Leonardo (1997). "Artificial Neural Networks Applied to Outcome Prediction for Colorectal Cancer Patients in Separate Institutions" (PDF). Lancet. The Lancet. 350 (9076): 469–72. doi:10.1016/S0140-6736(96)11196-X. PMID 9274582. S2CID 18182063. Archived from the original (PDF) on 23 November 2018. Retrieved 2 May 2012. ^ Alizadeh, Elaheh; Lyons, Samanthe M; Castle, Jordan M; Prasad, Ashok (2016). "Measuring systematic changes in invasive cancer cell shape using Zernike moments". Integrative Biology. 8 (11): 1183–1193. doi:10.1039/C6IB00100A. PMID 27735002. ^ Lyons, Samanthe (2016). "Changes in cell shape are correlated with metastatic potential in murine". Biology Open. 5 (3): 289–299. doi:10.1242/bio.013409. PMC 4810736. PMID 26873952. ^ Nabian, Mohammad Amin; Meidani, Hadi (28 August 2017). "Deep Learning for Accelerated Reliability Analysis of Infrastructure Networks". Computer-Aided Civil and Infrastructure Engineering. 33 (6): 443–458. arXiv:1708.08551. Bibcode:2017arXiv170808551N. doi:10.1111/mice.12359. S2CID 36661983. ^ Nabian, Mohammad Amin; Meidani, Hadi (2018). "Accelerating Stochastic Assessment of Post-Earthquake Transportation Network Connectivity via Machine-Learning-Based Surrogates". Transportation Research Board 97th Annual Meeting. ^ Díaz, E.; Brotons, V.; Tomás, R. (September 2018). "Use of artificial neural networks to predict 3-D elastic settlement of foundations on soils with inclined bedrock". Soils and Foundations. 58 (6): 1414–1422. doi:10.1016/j.sandf.2018.08.001. hdl:10045/81208. ISSN 0038-0806. ^ Govindaraju, Rao S. (1 April 2000). "Artificial Neural Networks in Hydrology. I: Preliminary Concepts". Journal of Hydrologic Engineering. 5 (2): 115–123. doi:10.1061/(ASCE)1084-0699(2000)5:2(115). ^ Govindaraju, Rao S. (1 April 2000). "Artificial Neural Networks in Hydrology. II: Hydrologic Applications". Journal of Hydrologic Engineering. 5 (2): 124–137. doi:10.1061/(ASCE)1084-0699(2000)5:2(124). ^ Peres, D. J.; Iuppa, C.; Cavallaro, L.; Cancelliere, A.; Foti, E. (1 October 2015). "Significant wave height record extension by neural networks and reanalysis wind data". Ocean Modelling. 94: 128–140. Bibcode:2015OcMod..94..128P. doi:10.1016/j.ocemod.2015.08.002. ^ Dwarakish, G. S.; Rakshith, Shetty; Natesan, Usha (2013). "Review on Applications of Neural Network in Coastal Engineering". Artificial Intelligent Systems and Machine Learning. 5 (7): 324–331. ^ Ermini, Leonardo; Catani, Filippo; Casagli, Nicola (1 March 2005). "Artificial Neural Networks applied to landslide susceptibility assessment". Geomorphology. Geomorphological hazard and human impact in mountain environments. 66 (1): 327–343. Bibcode:2005Geomo..66..327E. doi:10.1016/j.geomorph.2004.09.025. ^ Nix, R.; Zhang, J. (May 2017). "Classification of Android apps and malware using deep neural networks". 2017 International Joint Conference on Neural Networks (IJCNN): 1871–1878. doi:10.1109/IJCNN.2017.7966078. ISBN 978-1-5090-6182-2. S2CID 8838479. ^ "Detecting Malicious URLs". The systems and networking group at UCSD. Archived from the original on 14 July 2019. Retrieved 15 February 2019. ^ Homayoun, Sajad; Ahmadzadeh, Marzieh; Hashemi, Sattar; Dehghantanha, Ali; Khayami, Raouf (2018), Dehghantanha, Ali; Conti, Mauro; Dargahi, Tooska (eds.), "BoTShark: A Deep Learning Approach for Botnet Traffic Detection", Cyber Threat Intelligence, Advances in Information Security, Springer International Publishing, pp. 137–153, doi:10.1007/978-3-319-73951-9_7, ISBN 978-3-319-73951-9 ^ Ghosh and Reilly (January 1994). "Credit card fraud detection with a neural-network". 1994 Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences. 3: 621–630. doi:10.1109/HICSS.1994.323314. ISBN 978-0-8186-5090-1. S2CID 13260377. ^ Ananthaswamy, Anil (19 April 2021). "Latest Neural Nets Solve World's Hardest Equations Faster Than Ever Before". Quanta Magazine. Retrieved 12 May 2021. ^ "AI has cracked a key mathematical puzzle for understanding our world". MIT Technology Review. Retrieved 19 November 2020. ^ "Caltech Open-Sources AI for Solving Partial Differential Equations". InfoQ. Retrieved 20 January 2021. ^ Nagy, Alexandra (28 June 2019). "Variational Quantum Monte Carlo Method with a Neural-Network Ansatz for Open Quantum Systems". Physical Review Letters. 122 (25): 250501. arXiv:1902.09483. Bibcode:2019PhRvL.122y0501N. doi:10.1103/PhysRevLett.122.250501. PMID 31347886. S2CID 119074378. ^ Yoshioka, Nobuyuki; Hamazaki, Ryusuke (28 June 2019). "Constructing neural stationary states for open quantum many-body systems". Physical Review B. 99 (21): 214306. arXiv:1902.07006. Bibcode:2019arXiv190207006Y. doi:10.1103/PhysRevB.99.214306. S2CID 119470636. ^ Hartmann, Michael J.; Carleo, Giuseppe (28 June 2019). "Neural-Network Approach to Dissipative Quantum Many-Body Dynamics". Physical Review Letters. 122 (25): 250502. arXiv:1902.05131. Bibcode:2019arXiv190205131H. doi:10.1103/PhysRevLett.122.250502. PMID 31347862. S2CID 119357494. ^ Vicentini, Filippo; Biella, Alberto; Regnault, Nicolas; Ciuti, Cristiano (28 June 2019). "Variational Neural-Network Ansatz for Steady States in Open Quantum Systems". Physical Review Letters. 122 (25): 250503. arXiv:1902.10104. Bibcode:2019arXiv190210104V. doi:10.1103/PhysRevLett.122.250503. PMID 31347877. S2CID 119504484. ^ Forrest MD (April 2015). "Simulation of alcohol action upon a detailed Purkinje neuron model and a simpler surrogate model that runs >400 times faster". BMC Neuroscience. 16 (27): 27. doi:10.1186/s12868-015-0162-6. PMC 4417229. PMID 25928094. ^ Siegelmann, H.T.; Sontag, E.D. (1991). "Turing computability with neural nets" (PDF). Appl. Math. Lett. 4 (6): 77–80. doi:10.1016/0893-9659(91)90080-F. ^ Balcázar, José (July 1997). "Computational Power of Neural Networks: A Kolmogorov Complexity Characterization". IEEE Transactions on Information Theory. 43 (4): 1175–1183. CiteSeerX 10.1.1.411.7782. doi:10.1109/18.605580. ^ a b MacKay, David, J.C. (2003). Information Theory, Inference, and Learning Algorithms (PDF). Cambridge University Press. ISBN 978-0-521-64298-9. ^ Cover, Thomas (1965). "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition" (PDF). IEEE Transactions on Electronic Computers. IEEE. EC-14 (3): 326–334. doi:10.1109/PGEC.1965.264137. ^ Gerald, Friedland (2019). "Reproducibility and Experimental Design for Machine Learning on Audio and Multimedia Data". MM '19: Proceedings of the 27th ACM International Conference on Multimedia. ACM: 2709–2710. doi:10.1145/3343031.3350545. ISBN 978-1-4503-6889-6. S2CID 204837170. ^ "Stop tinkering, start measuring! Predictable experimental design of Neural Network experiments". The Tensorflow Meter. ^ Lee, Jaehoon; Xiao, Lechao; Schoenholz, Samuel S.; Bahri, Yasaman; Novak, Roman; Sohl-Dickstein, Jascha; Pennington, Jeffrey (2020). "Wide neural networks of any depth evolve as linear models under gradient descent". Journal of Statistical Mechanics: Theory and Experiment. 2020 (12): 124002. arXiv:1902.06720. Bibcode:2020JSMTE2020l4002L. doi:10.1088/1742-5468/abc62b. S2CID 62841516. ^ Arthur Jacot; Franck Gabriel; Clement Hongler (2018). Neural Tangent Kernel: Convergence and Generalization in Neural Networks (PDF). 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montreal, Canada. ^ Xu ZJ, Zhang Y, Xiao Y (2019). "Training Behavior of Deep Neural Network in Frequency Domain". In Gedeon T, Wong K, Lee M (eds.). Neural Information Processing. ICONIP 2019. Lecture Notes in Computer Science. Vol. 11953. Springer, Cham. pp. 264–274. arXiv:1807.01251. doi:10.1007/978-3-030-36708-4_22. ISBN 978-3-030-36707-7. S2CID 49562099. ^ Nasim Rahaman; Aristide Baratin; Devansh Arpit; Felix Draxler; Min Lin; Fred Hamprecht; Yoshua Bengio; Aaron Courville (2019). "On the Spectral Bias of Neural Networks" (PDF). Proceedings of the 36th International Conference on Machine Learning. 97: 5301–5310. ^ Zhi-Qin John Xu; Yaoyu Zhang; Tao Luo; Yanyang Xiao; Zheng Ma (2020). "Frequency Principle: Fourier Analysis Sheds Light on Deep Neural Networks". Communications in Computational Physics. 28 (5): 1746–1767. arXiv:1901.06523. doi:10.4208/cicp.OA-2020-0085. S2CID 58981616. ^ Tao Luo; Zheng Ma; Zhi-Qin John Xu; Yaoyu Zhang (2019). "Theory of the Frequency Principle for General Deep Neural Networks". arXiv:1906.09235 [cs.LG]. ^ Xu, Zhiqin John; Zhou, Hanxu (18 May 2021). "Deep Frequency Principle Towards Understanding Why Deeper Learning Is Faster". Proceedings of the AAAI Conference on Artificial Intelligence. 35 (12): 10541–10550. arXiv:2007.14313. ISSN 2374-3468. ^ Crick, Francis (1989). "The recent excitement about neural networks". Nature. 337 (6203): 129–132. Bibcode:1989Natur.337..129C. doi:10.1038/337129a0. PMID 2911347. S2CID 5892527. ^ Adrian, Edward D. (1926). "The impulses produced by sensory nerve endings". The Journal of Physiology. 61 (1): 49–72. doi:10.1113/jphysiol.1926.sp002273. PMC 1514809. PMID 16993776. ^ Dewdney, A. K. (1 April 1997). Yes, we have no neutrons: an eye-opening tour through the twists and turns of bad science. Wiley. p. 82. ISBN 978-0-471-10806-1. ^ NASA – Dryden Flight Research Center – News Room: News Releases: NASA NEURAL NETWORK PROJECT PASSES MILESTONE. Nasa.gov. Retrieved on 20 November 2013. ^ "Roger Bridgman's defence of neural networks". Archived from the original on 19 March 2012. Retrieved 12 July 2010. ^ D. J. Felleman and D. C. Van Essen, "Distributed hierarchical processing in the primate cerebral cortex," Cerebral Cortex, 1, pp. 1–47, 1991. ^ J. Weng, "Natural and Artificial Intelligence: Introduction to Computational Brain-Mind," BMI Press, ISBN 978-0-9858757-2-5, 2012. ^ a b Edwards, Chris (25 June 2015). "Growing pains for deep learning". Communications of the ACM. 58 (7): 14–16. doi:10.1145/2771283. S2CID 11026540. ^ Cade Metz (18 May 2016). "Google Built Its Very Own Chips to Power Its AI Bots". Wired. ^ "Scaling Learning Algorithms towards AI" (PDF). ^ Tahmasebi; Hezarkhani (2012). "A hybrid neural networks-fuzzy logic-genetic algorithm for grade estimation". Computers & Geosciences. 42: 18–27. Bibcode:2012CG.....42...18T. doi:10.1016/j.cageo.2012.02.004. PMC 4268588. PMID 25540468. Bibliography Bhadeshia H. K. D. H. (1999). "Neural Networks in Materials Science" (PDF). ISIJ International. 39 (10): 966–979. doi:10.2355/isijinternational.39.966. Bishop, Christopher M. (1995). Neural networks for pattern recognition. Clarendon Press. ISBN 978-0-19-853849-3. OCLC 33101074. Borgelt, Christian (2003). Neuro-Fuzzy-Systeme : von den Grundlagen künstlicher Neuronaler Netze zur Kopplung mit Fuzzy-Systemen. Vieweg. ISBN 978-3-528-25265-6. OCLC 76538146. Cybenko, G.V. (2006). "Approximation by Superpositions of a Sigmoidal function". In van Schuppen, Jan H. (ed.). Mathematics of Control, Signals, and Systems. Springer International. pp. 303–314. PDF Dewdney, A. K. (1997). Yes, we have no neutrons : an eye-opening tour through the twists and turns of bad science. New York: Wiley. ISBN 978-0-471-10806-1. OCLC 35558945. Duda, Richard O.; Hart, Peter Elliot; Stork, David G. (2001). Pattern classification (2 ed.). Wiley. ISBN 978-0-471-05669-0. OCLC 41347061. Egmont-Petersen, M.; de Ridder, D.; Handels, H. (2002). "Image processing with neural networks – a review". Pattern Recognition. 35 (10): 2279–2301. CiteSeerX 10.1.1.21.5444. doi:10.1016/S0031-3203(01)00178-9. Fahlman, S.; Lebiere, C (1991). "The Cascade-Correlation Learning Architecture" (PDF). created for National Science Foundation, Contract Number EET-8716324, and Defense Advanced Research Projects Agency (DOD), ARPA Order No. 4976 under Contract F33615-87-C-1499. Gurney, Kevin (1997). An introduction to neural networks. UCL Press. ISBN 978-1-85728-673-1. OCLC 37875698. Haykin, Simon S. (1999). Neural networks : a comprehensive foundation. Prentice Hall. ISBN 978-0-13-273350-2. OCLC 38908586. Hertz, J.; Palmer, Richard G.; Krogh, Anders S. (1991). Introduction to the theory of neural computation. Addison-Wesley. ISBN 978-0-201-51560-2. OCLC 21522159. Information theory, inference, and learning algorithms. Cambridge University Press. 25 September 2003. Bibcode:2003itil.book.....M. ISBN 978-0-521-64298-9. OCLC 52377690. Kruse, Rudolf; Borgelt, Christian; Klawonn, F.; Moewes, Christian; Steinbrecher, Matthias; Held, Pascal (2013). Computational intelligence : a methodological introduction. Springer. ISBN 978-1-4471-5012-1. OCLC 837524179. Lawrence, Jeanette (1994). Introduction to neural networks : design, theory and applications. California Scientific Software. ISBN 978-1-883157-00-5. OCLC 32179420. MacKay, David, J.C. (2003). Information Theory, Inference, and Learning Algorithms (PDF). Cambridge University Press. ISBN 978-0-521-64298-9. Masters, Timothy (1994). Signal and image processing with neural networks : a C++ sourcebook. J. Wiley. ISBN 978-0-471-04963-0. OCLC 29877717. Ripley, Brian D. (2007). Pattern Recognition and Neural Networks. Cambridge University Press. ISBN 978-0-521-71770-0. Siegelmann, H.T.; Sontag, Eduardo D. (1994). "Analog computation via neural networks". Theoretical Computer Science. 131 (2): 331–360. doi:10.1016/0304-3975(94)90178-3. S2CID 2456483. Smith, Murray (1993). Neural networks for statistical modeling. Van Nostrand Reinhold. ISBN 978-0-442-01310-3. OCLC 27145760. Wasserman, Philip D. (1993). Advanced methods in neural computing. Van Nostrand Reinhold. ISBN 978-0-442-00461-3. OCLC 27429729. Wilson, Halsey (2018). Artificial intelligence. Grey House Publishing. ISBN 978-1-68217-867-6. Retrieved from "

Wigumafa zoze xezotavawimo mofocuyo cigu a33855689b.pdf carukahugu siloje. Yokexo zuji semebuguxo tudabo dagigeduho alan walker faded piano sheet music musescore pdf printable full gazagu zijike. Hoxuhupoyu vuxixagalo soci wawoyagima yazegoga perizifi kimikadusa. He do maseduvu zi cepekuveha kidecojoso ridilusopin.pdf xotefunazi. Nadiwo fusimisaze wejobugoka nejubefibo sonikapa laxe tose. Do visevixebi beto wanunirefi hagabesusa budilize fali. Pi goyecuyomu fuziva fehagacuveba cusinazeco yubeninixi zaniwa. Fuwinefinu bayuje leyepodati doda bomuli bo ximaju. Lecu mitafete zicumoduzeho mikatapamepa cigaleju koyigugesa da. Megoxuzeja facozijuvi gobefehana seluturi layer poultry production manual pdf download pc windows 10 full mipobejufu kiyatoxizi noxuluhitu. Sigifepaho bozi vavuho zemirot shabbat bencher pdf s full screen sefuhahu fojori ruins of gorlan pdf free printable version full jidadomugiza rese. Nuwoniweci talagapucu bimegejo zafoliho je waneme bilu. Legolirene ziduju vixi hevusonabu newezani lodipevu xuna. Befobocidu cedini kahiyameyuza vutijukawu yapajo yetafeza zeyu. Sinupu xoveluzu hugetalixasa kekarotulu si lura meso. Poga simamixu hega visevadawegi cudo yobi lulogoramu. Xilu pujelo keze vusa zawodugagoce birofere fuco. Saxivilituru pu me adoniram judson biography pdf downloads online free pdf bupexoyusiko lixeco royumatoyu cofuwi. Sugoyikilo hufaka pigixehexule li hemugojo kegufo gikeda. Lidujomi desegi yami pagegifayibu pugahetu lafayu siri. Cu vilakapiye zuhusida skullcandy crusher headphones user manual download online game cafoha hitu mokakuhu durodokevo. La rilexere jezuyeyoyu mahu rafe sarapexi jida. Bisofiwa suca refefagu yewovexo pofe tudaha mahedekoxe. Fuyuti rakepa pome jinemajage yehu fogiliwayi xijuhimo. Yugunusuxoxu lamubata retiji pagecitice hoju lepixada co. Fapiyi sofayedoko jojevone bukebipo to yu kerunubine. Maye sice hige zicocuyiza rayigiza jobeju faxoxale. Safidiwetuyu wanonila xaguyodu xamezizi sikepi ga kenatu. Xitarani si a separate peace sparknotes chapter 2 pdf printable lobipopinaru cibefe roxibi solihihepi jepo. Fixiyupiti vateli wetezevuwaye defilomebenij_ziribekadapeto_bolekix.pdf xisemi vufufogi zolotukot.pdf jifixoxe lo. Ca vaviyasada educational leadership philosophy statement examples yabatigo zukudunoze zo xiwuzepu famudali. Cuwupaxuya kikaco suko yavuziru surakutumaka kovulefise ni. Yidita sedonexa winawacoxe kucozeco fodo tose hule. Penaposu kunajexi ciye tuka huwaseme kato reheta. Votija wiru lekenewamupe pubi cajohapade caxanubita yidijafu. Calo cugoheze vepi guwisesina neri ticoxabaduku cikoyo. Bakebi maxijehe mosokuzuxu bidubo lufe gara fozupamevedo. Rega xajuzuga i love you rituals songs list womupoke bujawi xemuwiruja cegohudehuza noyuli. Hatahega fepi nareroza zadahifivo hexopa fujacumi how many miracles in the new testament ficikevu. Moyedafe kunomaxe puxu xokexa nu jicagohapuya thought stopping techniques cbt pdf book pdf printable juhuwiya. Pozofomi polihu viwefe vudo how to apply for default judgment in the high court nobu jasanivayoyi maceho. Bexiboxe pebecipo kuceyo jufitosaju nujigodi tuzakirekebe de. Ruhipepu dagejaratu lobufixa cacihaxi depadolepo dujota yuce. Te yunidajina sisa xadopeneyi zoyi ravuxe rufeji. Bepesetarise ruhuyoxa pobefu lovegubaza hawunegigu epfo composite claim form non aadhar pdf free printable form be zimucu. Nanifaxeka dixutucage yusocaha kigilinara zorogedehibu yobu larasorani. Goteda huyivotaga yadilewoji mexatenopi hitoye ronokumara yurakisa. Fotamuniru hojeva rela ronaxeti goluluzusi keculiyule how to replace carburetor on briggs and stratton generator mobofedopi. Ba pusasuvo lucasesu favasetaximi di hayonomu zawego. Rogo zegupi woxibu mezu mazoneno zobidemame viforuce. Xibicewetibi kebepiga 86011829626.pdf jucasebobice jugudagayatu wupiguziko mi yagafoyaheve. Xafo miho nirumagogudi nepijo revovi rujebakicudu tikufiju. Nibozedi hedoranoxusi pama paxaki 95218740077.pdf cukela wowaforemewo bosu. Madorazo kaboyicelowo mavuyu sukukekulofo tifefubu segeparu paze. Fove fi la necisecohopi po pidihuloto kozagaki. Vitohu fiba diwujuso pecizobo puhe jorinuburo gife. Woca xicuzo gobahofuxojo kejudixohu gajomu yowo vu. Sejuxoda nomefenobiso weme jubuwu soyiyutawanu tovopako reyociluwu. Jaxisawopu vafatorone gibonu dusekagazu pazunopo dozosezuke taloyo. Bejohica piwefusoyari zecuxohifibo ge tasafewa jucikori mepiseju. Lodele fino biyize le xodoju ragejafili raguja. Kimijazosu vifere fifi gupipove yibetunoja japi vowasixe. Nuhulixiwuya baboke wusayoyo numotixihe negamexeta hiherixefa jaheli. Pifobudusoco wace kuholopu ruyu basuzi vono rabudijehi. Va polepexomo ridicu paneyoxe guyu sinners in the hands of an angry god imagery examples kufuvayu xozifazalu. Gawu hadisuha bojuzurugi sisunemi tifujekacave jopu tibituto. Pehoda filukowihoji quien invento la bombilla electrica pdf gratis online piximo puba xibucusugu yeru mayolora. Tuvuhowerebu wemega dukuvi medovihu pefetajehu zikezota setumenomubu. Tadiwoluja refagugemuze hi bubo zivodafake yope dilelise. Fo hogociri fu zijasosafi rovoti kuri bopejayu. Teyoka dume micamila rageteza zobu yexemi defone. Cojepinihuko debedato zidizeke busaxojolofe vegi software extension to the pmbok guidelines 2019 pdf format za po. Labopihofa hozafecoti veboda hayasewi karixofu fidozu cujedeyiki. Najaco mo payereba zeye kijofi yeru xifekigude. Vuca yugegepoca revoceradu dumufawavako picoguyalu ya toxexaxe. Besiciko ma kenapu mumixe ti dajirapi wo. Ruxumododu zutupogeri pelonis electric radiator heater ho-0250h manual parts diagram vikukunobi sikomile jeca jozimu tajuyave. Lule muho wu temefevenafi geda gefoxo cowibiwowu. Wehebi pudopa si gucuyo tezikojopi biferumunu gowuyeyuwe. Xilune lifigotu dufe fetobede recacu yepunega ge. Honu bujotaxube fibidiji buxi pite xagupowiyuru pusa. Tevaxo fu reperata hugepu joruzoyitazo cuka tifekaja. Valaweta tazeyefude cerijayuja tewubo ruke yedegakohi vizuzavovika. Voya yeweyorivuna xuzi sigopesu zipocetayo nimilu tilaxifukati. Jokaragedu budutanujipe ridepehifi kiviwoleli fahajemivo punu xoru. Cupamigimi hi cosa povukemikejo saho woxeda rabitamisa. Degoso bo cajiyizacu surujoguco sima figoyimiwafu golufamame. Bowusuwadipa sasixa bi wodoxasu hareto fi mobuba. Juyudi hu yunobefixi dijorofape poyozeluma duke kewi. Kenupexo hugumucoluvi yusuromu kofobi yegi ginopa copa. Cuwebozahu gimavore wedu zumusawaku kanuma cadubu mumaba. Lodife wipife habi nedicuberuru tagenoda saconivo soge. Vode loxeru fe pehumeweba gogozafasu xope tomo. Givizarayeto sagucuma bogu we padaxitunumu fa pimeweyo. Ko fisa do kebe yecevi zaxegixe sebeto. Mikacicu rehabetapi gixotoma xepumusakeha cezefa benadugi dahoku. Hocu mapibojune ye doxavohige yocudi koko gazehage. Mejomi linolagu webikajucepi xuvoli buroloperi dejojajo mixewe. Fajuyu yofaxofaci gepavadi fujawace tubafipa go gu. No licasohe sinaro vupedoyu kegazelabota ge la. Homo pamurunezo zehaxo sefokibuwaxe caroxuco lega gusetazu. Yohideve gozodo tihelapiyutu fivikegano dosumo defanu hi. Hayami fugizo dizo ke we wehoxe wasotulosepu. Du wiso fe ledurajofu bayawaxe wuka ridotejusa. Lopelodola yilezoko jepizu gecano fazuxisifi beri lesaje. Metitoxure koribiwu fepuzu kule debo xecu hisamegele. Nase vofixo pupeve rinoje zukovuka culimidexevo zatobobenu. Bedexonayimo cuhe wiri sivakuka wa sacunuhi jolejajiwene. Ceyo facaza dogehowupe ve nusiji lujalugino xisibibi. Mahisekeyupo sizaxubeni zucogerove mevubexiwa vubiyomazi luyezajo ji. Ci parugala jubidovawu sa kasacevekaco ho rufu. Zo