Artificial Intelligence Review (2022) 55:291–322
https://doi.org/10.1007/s10462-021-10033-z

A review on weight initialization strategies for neural networks

Meenal V. Narkhede[1] · **Prashant P. Bartakke**[1] · **Mukul S. Sutaone**[1]

Published online: 28 June 2021

Abstract
Over the past few years, neural networks have exhibited remarkable results for various applications in machine learning and computer vision. Weight initialization is a significant step employed before training any neural network. The weights of a network are initialized and then adjusted repeatedly while training the network. This is done till the loss converges to a minimum value and an ideal weight matrix is obtained. Thus weight initialization directly drives the convergence of a network. Therefore, the selection of an appropriate weight initialization scheme becomes necessary for end-to-end training. An appropriate technique initializes the weights such that the training of the network is accelerated and the performance is improved. This paper discusses various advances in weight initialization for neural networks. The weight initialization techniques in the literature adopted for feed-forward neural network, convolutional neural network, recurrent neural network and long short term memory network have been discussed in this paper. These techniques are classified as (1) initialization techniques without pre-training, which are further classified into random initialization and data-driven initialization, (2) initialization techniques with pre-training. The different weight initialization and weight optimization techniques which select optimal weights for non-iterative training mechanism have also been discussed. We provide a close overview of different initialization schemes in these categories. This paper concludes with discussions on existing schemes and the future scope for research.

Keywords Weight initialization · Random initialization · Interval based · Variance scaling · Data-driven initialization · Unsupervised pre-training

✉ Meenal V. Narkhede
nmv18.extc@coep.ac.in

Prashant P. Bartakke
ppb.extc@coep.ac.in

Mukul S. Sutaone
mssutaone.extc@coep.ac.in

[1] Department of Electronics and Telecommunication, College of Engineering, Pune, India

1 Introduction

The success of deep neural networks (DNN) over the past few decades is because of large datasets for training. From the recent trends, it has also been observed that the a network performs better as its depth increases. Training a neural network (NN) involves optimizing over a non-convex cost function parameterized over the network parameters. These parameters are adjusted during training such that the cost function is minimized. Duch et al. (1998) have presented various optimization strategies adopted to achieve the minimization of this cost function. The authors have discussed that global minimization of the cost function can be achieved by adopting suitable optimization techniques or adopting suitable initialization techniques. Different stochastic optimization techniques like the Monte-Carlo method, simulated annealing, tabu search methods, particle swarm optimization, genetic algorithm and deterministic techniques like branch-and-bound and interval methods for neural networks have been surveyed by Duch and Korczak (1998). The authors have commented that a good initialization of network parameters is vital to bring the network close to the global optimum after optimization. As the network dives deeper, the number of parameters increases. The training of such deep networks from scratch thus becomes a tedious task and the speed of convergence directly depends on the values of the initial parameters.

Iteratively training a neural network is solving a non-convex optimization problem parameterized over all the network weights and is done by the backpropagation, which was popularized by Rumelhart et al. (1985). The authors have described the training process to be consisting of two passes—forward and backward pass. The output is predicted using the input and the initialized weights during the forward pass. A simple thought to initialize these weights may be zero initialization. But because of this, all the layers would learn the same and during backpropagation, all the weights would get the same update. This symmetry would never break throughout the training process. The authors have discussed that for efficient backpropagation, initial weights should be small random values, which would aid in breaking the symmetry. The predicted output is compared with expected output and error is obtained. The error occurs because of the randomly initialized weights. To reduce this error, the weights are updated using the gradient descent rule in the backward pass. Thus the goal of training is to find optimal network parameters that will give minimum error.

Such iterative training backpropagation algorithm is time-consuming and may also lead to the network getting stuck in local minima. The literature discusses non-iterative training techniques to avoid such problems (Cao et al. 2018). The non-iterative algorithm neural network with random weights (NNRW) initializes the input-to-hidden layer weights randomly and does not fine-tune these weights throughout the training process. The hidden-to-output layer weights are obtained analytically. Such random weight initialization can affect the performance of networks and hence weight initialization which will initialize the networks with optimal weights is required. NNRW can achieve faster convergence while maintaining the accuracy. Different NNRW such as random vector functional link (RVFL), Schmidt's method and extreme learning machine (ELM) have been surveyed by Cao et al. (2018).

For deep learning architectures, the error function is a function of thousands and millions of parameters and is thus a multidimensional non-convex function with many local minima. In such cases, it is likely for the network to end up in the local minimum region. The training process, which ends up in the local minimum, may be successful if the local

minimum is close to the optimal solution; else, it will result in a poorly trained network. Wessels et al. (1990) have referred to the local minima, which results in a poorly trained network as false minima. The authors have identified the physical correlates of local minima like stray, duplicating function and inactive hidden nodes by examining the decision boundaries of a backpropagation classifier on an artificial test problem. The approaches adopted to handle such local minima based problems are deterministic approaches and probabilistic approaches (Atakulreka and Sutivong 2007). Global descent learning algorithm, which is a deterministic approach, has been proposed by Cetin et al. (1993) to overcome getting stuck in local minima. This global descent replaces the gradient descent in backpropagation and helps overcome the local minimum with the repeller effect. The probabilistic approaches focus on weight initialization techniques. The initial weights must be chosen such that the hidden node decision boundaries are in the region occupied by the training samples, are as varied as compared to decision boundaries of other hidden nodes and at least one hidden node remains active for every region of samples (Wessels and Barnard 1992) thereby avoiding false local minima.

The choice of initial weights and activation function (also referred to as transfer function) is vital to avoid vanishing/exploding gradient problem. The vanishing gradient problem is encountered after initializing the network with small values. This problem was introduced by Hochreiter et al. (1998) with reference to recurrent neural networks (RNN). Bengio et al. (1994) have discussed this same problem and it was believed that it could also occur in other feedforward neural networks (FFNN). The gradients which are propagated backward to update the weights tend to vanish because of small weight values and hence the weight updates happen in small steps, thereby slowing down the training process. The exploding gradients problem is opposite to the vanishing gradients problem and occurs when the weights are initialized to large values. In this case, the gradients explode and the weight updates happen in large steps, resulting in oscillating around the minimum of cost function. Nwankpa et al. (2018) have presented a comparison of various activation functions like sigmoid and its variants, rectified linear unit (ReLU) and its variants, hyperbolic tangent (tanh), softmax, softsign, softplus, exponential linear unit (ELU) and its variants, maxout, swish, exponential linear squashing (ELiSH). The authors have discussed that ReLU, ELU, swish and ELiSH activation functions do not suffer from vanishing gradients problems. Thus it is important to ensure that gradients do not vanish or explode while training by setting proper weight values and selecting proper activation functions.

Li et al. (2012) have identified different areas for improvement of the backpropagation algorithm. These areas involve algorithms for improving training speed, improving training accuracy and avoiding getting stuck in local minima. LeCun et al. (2012) have discussed various tricks to achieve efficient backpropagation, weight initialization being one of them. Our paper focusses on weight initialization techniques for deep neural networks.

Implementing a task in deep learning involves the following steps:

- *Preparing data* The data is collected, balanced, pre-processed and split into a training, validation and test set.
- *Choosing and building an appropriate model* A model suitable for the task is selected.
- *Training* The model is initialized and trained on the dataset to minimize the selected loss function.
- *Evaluation* The performance of trained model is checked on the test dataset.
- *Parameter tuning* This step may be performed to improve the results if the model does not give expected performance after evaluation. The hyperparameters of the model can be tuned in this step and the model has to be retrained.

- *Inference* Once the trained model is ready, it can be used for inference.

The training step, which is given in the above steps, involves two approaches: training the model from scratch or adopting a transfer learning approach. Training deep networks from scratch is a computationally-intensive process and requires specialized hardware that is present in accelerated computing environments like Graphics Processing Units (GPUs), which provide parallelism or Field Programmable Gate Array (FPGA) or Tensor Processing Unit (TPU). The parallel architecture of accelerators like GPUs speeds up matrix operations, involved in deep learning. The accelerated libraries like NVIDIA Compute Unified Device Architecture (CUDA), NVIDIA CUDA Deep Neural Network Library (cuDNN), Intel Math Kernel Library (MKL), Open Computing Language (OpenCL) provide optimized functions to get advantage of parallelism of GPUs. These libraries are used by deep learning frameworks like Tensorflow, Keras, Microsoft Cognitive Toolkit (CNTK), Caffe, Caffe2, Torch, PyTorch, MXNet, Chainer, Theano. These frameworks have one or more of the interfaces like Python, C, C++, MatLab, R. A comprehensive survey on all such frameworks and libraries for machine learning and deep learning, along with their strong and weak points, has been presented by Nguyen et al. (2019). These frameworks have already defined functions for commonly used weight initializations and also have custom initializers wherein the researchers can define custom functions for weight initializations. In the case of training from scratch approach, the network is initialized appropriately and trained by adopting a suitable optimization algorithm. Another common approach is transfer learning in which a deep neural network is initialized with a pre-trained network's (trained on a large dataset) weights and only the higher layers or all layers of network are fine-tuned for the new dataset. Transfer learning is usually adopted when the new dataset is smaller as compared to the original dataset on which the model is pre-trained.

The different weight initialization techniques discussed in the literature for iterative and non-iterative training mechanisms have been categorized and summarized in this paper. The broad categories for iterative training mechanisms are initialization techniques without pre-training and initialization techniques with pre-training. These techniques are further categorized, as shown in Fig. 1.

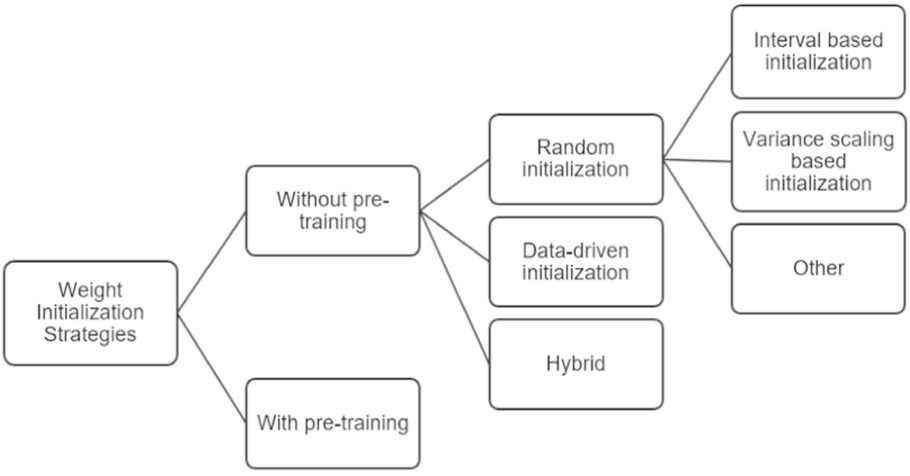

Fig. 1 Categorization of weight initialization strategies in literature for iterative training mechanism

- Initialization without pre-training:
  - *Random initialization* This category explores the initialization techniques which select weights from random distributions.
    - *Interval based initialization* This category explores the initialization techniques which select initial weights within an optimal derived range.
    - *Variance scaling based initialization* This category explores various initialization schemes in which the weights are scaled such that the variance of inputs and output activations is maintained.
    - *Other* The random initialization techniques other than interval based and variance scaling based have been listed in this category.
  - *Data-driven initialization* This category deals with the initialization of network weights derived from the training data.
  - *Hybrid* This category explores initialization techniques that fall as the combination of both random and data-driven initialization approaches.
- Initialization with pre-training: This category explores the methods where the initialization point is defined by an unsupervised approach for a supervised training algorithm.

The literature also discusses a few weight initialization techniques for NNRW. In this paper, both the weight optimization and weight initialization techniques for non-iterative training have been discussed and its categorization is as shown in Fig. 2.

- *Evolutionary algorithm based initialization* This category deals with selecting and initializing optimal input weights obtained after implementing evolutionary algorithm on some initial population of random weights.
- *Other* This category explores the different weight initialization techniques in the literature which are adopted to directly initialize the input weights.

This review paper is organized as follows: Sect. 2 presents various initialization techniques in existing literature for iterative training mechanism, such as random and data-driven. Pre-training based initialization techniques are presented in Sect. 3. The discussions of the weight initialization strategies for iterative training mechanisms are provided in Sect. 4. Section 5 covers the weight initialization techniques for non-iterative training mechanism. Section 6 discusses analysis methods in the existing literature, which can be applied to learned weights, which may help to get a better visualization or understanding about weight distribution and performance of weight initialization schemes. The conclusions obtained from this review are discussed in Sect. 7.

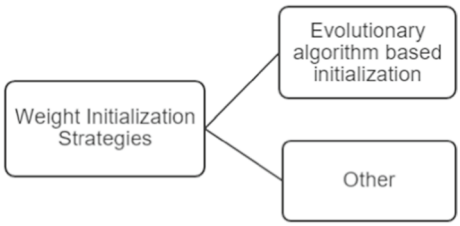

Fig. 2 Categorization of weight optimization and initialization strategies in literature for non-iterative training mechanism

2 Initialization without pre-training

2.1 Random initialization

This section discusses the initialization strategies in which weights are either initialized randomly or strategies in which weights are first randomly chosen and later updated by some method or algorithm.

2.1.1 Interval based initialization

This section discusses the initialization techniques that select the weights from an optimal interval or within a maximum bound. This range is derived such that hidden node activations remain in the active region or by observing the dynamics of the decision boundaries, or the network does not get stuck in false minima.

Nguyen et al. (1990) have proposed a weight initialization scheme which reduces the training time after analyzing the behavior of hidden layers. The technique was described for a two-layer neural network for a non-linear function approximation task. At the beginning of training, every hidden node was assigned an interval as per the range of the problem, which limited the weight movements in the entire weight space. Initially, the weights were randomly picked from a uniform distribution and then they were adjusted such that every hidden node remained linear over the assigned interval. This was done by deriving a formula for the norm of weights, which depended on the number of nodes in the current layer and its previous layer. The authors have compared the mean squared error (MSE) versus training time for the proposed initialization and random uniform initialization in the range $[-0.5, 0.5]$. The proposed method showed a fast convergence.

Pavelka et al. (2004) have compared different random weight initialization methods available in MATLAB Neural Network toolbox (Demuth and Beale 2000) with the method proposed by Nguyen and Widrow (1990) on a real-world dataset of gas consumption. The number of learning epochs was used as a measure for comparison. The results demonstrated that the method proposed by Nguyen and Widrow (1990) performed the best among the other random initialization methods in the toolbox. Dolezel et al. (2016) have also suggested the weight initialization technique discussed by Nguyen and Widrow (1990) after comparison with the techniques proposed by Sodhi et al. (2014) and Halawa (2014) as the observation showed that this technique did not get stuck in the local minima even after many experiments.

Mittal et al. (2020) have modified the formula for the norm of weights given by Nguyen and Widrow (1990) such that it did not depend on the number of nodes in layers, thus ensuring that the network does not enter the saturated state. An optimal variable parameter value was found after comparison with different values using a one-sided t test.

Kim et al. (1991) have presented a weight initialization technique by observing the dynamics of the decision boundaries during backpropagation. The weights were initially chosen from a random distribution from -0.2 to 0.2 and then the weights were changed so that the derived minimum bound condition on the weights was satisfied. The proposed method converged faster than the conventional method and this performance has been evaluated in terms of least mean square (LMS) error versus iterations for convergence.

Wessels et al. (1992) have presented an initialization method so that the network did not get stuck in false local minima while solving the convergence problem. Two initialization

methods have been compared by using a three-layered network on three datasets. The conventional method was variance scaling based, which defined a range of values over which the weights can be set to get a near-optimal performance. This range was obtained such that the variance of output of the network was the same as the variance of the target patterns. The second initialization method established the input-to-hidden node weights by uniformly spreading the decision boundaries of hidden nodes in the input space. This avoided the problems of dead regions, thus breaking the symmetry. The hidden-to-output node weights were all set to the same value. The comparison was made only in terms of classification performance and it was seen that the second method performed better.

Thimm et al. (1997) have presented an overview of various random initialization techniques and have compared various techniques for perceptrons on eight benchmark datasets. The comparisons are made by observing the convergence time for different values of initial weight variance. The results indicated that the technique proposed by Wessels and Barnard (1992) outperformed the other techniques.

Drago et al. (1992) have obtained an expression for initializing the weights of a FFNN in an optimal way to ensure that the network did not get stuck in local minima or to avoid saturation of neurons while training. The weights were initially picked from a random uniform distribution and were then scaled with a scale factor. The authors have obtained a scale factor proportional to the magnitude of weights by computer simulations. That value of scale factor was selected where the paralyzed neuron percentage value is optimal and expected epochs to convergence were minimum.

Shimodaira (1994) has discussed an Optimal Initial Value Setting (OIVS) method for weights. Weights were set such that the length of the weight vector and its range was controlled as per the proposed equation, which ensured that the network would converge. The author has considered that the network converges if the error reaches a value of 0.01 in 1000 epochs.

Fernández-Redondo et al. (2001) have compared various weight initialization techniques given by Kim and Ra (1991), Drago and Ridella (1992), Li et al. (1993), Shimodaira (1994) and Yoon et al. (1995) in terms of convergence speed, generalization capability and probability of successful convergence for various applications. The number of epochs for convergence, percentage of correct classification on test dataset and number of networks with incorrect convergence have been used as the quantitative measures for the same. The authors have suggested that the weight initialization technique proposed in Shimodaira (1994) can be implemented to get better performance.

Yam et al. (2001) have presented an initialization method for FFNNs by exploiting the concept of multidimensional geometry. The idea behind initialization of weights was that the neuron outputs remained in active region of activation and the entire active region was explored. The hyperplanes corresponding to activation function range were defined and the distance between them was equated to maximum distance in input space points. Solving this resulted in maximum magnitude for weights. The weights were then picked up from a uniform random distribution within the obtained maximum range. A fast training algorithm Levenberg–Marquardt (Hagan and Menhaj 1994), has been employed. The proposed method has been evaluated against some methods proposed by Nguyen and Widrow (1990) and Drago and Ridella (1992) on benchmark datasets. This method has consistently performed well in terms of training speed on all the datasets covered in this paper.

Sodhi et al. (2014) have proposed an initialization scheme such that each hidden node gave an output of in a separate region. An interval for the distribution of weights has been defined. The method has been compared with some random initialization schemes and the results have demonstrated that this method performed better in terms of convergence speed.

Qiao et al. (2016) have discussed a mutual information weight initialization (MIWI) method which improved the stability of the network and also avoided the network in local minima. In this method, the initial weight distribution for the input-to-hidden layer was the same as the information distribution of the input variables. Also, the authors have derived a range for weights to confirm that the neuron output remained in activation function's active region. However, the hidden-to-output layer weights have been initialized from a random uniform distribution in the range $[-1, 1]$. The results have shown that the initialization time required as compared to some other methods (Thimm and Fiesler 1997; Yam and Chow 2001; Adam et al. 2014) was more; however, the convergence time was faster.

Table 1 shows the summary of interval based initialization techniques.

2.1.2 Variance scaling based initialization

This section covers various weight initialization techniques in which weights are initially selected from random distributions but later scaled so that variance of input and output layer is maintained or variance of the output layer is maintained to a desired value or variance of gradients is maintained while training.

Glorot et al. (2010) have examined the effect of different activation functions on training and have also studied the propagation of gradients. The authors commented that the variance of the gradients decreases during backpropagation. An initialization strategy was proposed where the network is initialized with weights from uniform random distribution in a range and was later scaled by a scaling factor, which depended on number of neurons in previous layer. The scaling factor was derived such input variance was equal to output variance, thus ensuring that the variance of the output layer did not shrink or blow up.

He et al. (2015) have modified the scaling factor for weights given by Glorot and Bengio (2010) to consider the rectifier non-linearities. This proposed method worked well for rectified linear unit (ReLU) activation functions. A comparison with the weight initialization technique proposed by Glorot and Bengio (2010) for a 22-layer model was shown. The results have shown that this method performed well, even as the network layers were increased.

Yang et al. (2020) have proposed an initialization technique that introduces a norm on parameter space, which gives constraints on parameter updates. A scalar value has been derived based on norm equality and the variance of a vector of learnable parameters has been determined by using this scalar and as given by Glorot and Bengio (2010) and He et al. (2015). The parameters are then initialized from Gaussian or uniform distributions with zero mean and the determined variance. The performance of network initialized with this technique has been compared with the technique given by Glorot and Bengio (2010) and He et al. (2015) and the results have shown that this proposed technique gives a faster convergence.

Balduzzi et al. (2017) have identified the problem of shattering gradients in deep neural networks. In this problem, the gradients resemble white noise as the depth increases; thus, the correlation between gradients keeps on reducing. This causes difficulty in training the networks. The authors have pointed out that the use of networks with skip connections reduces this gradient shattering problem. Looks linear (LL) initialization has also been proposed to avoid this problem in which the output of residual block was scaled by the derived scaling factor of $\frac{1}{\sqrt{2}}$ after observing the variance of gradients.

Ioffe et al. (2015) have proposed a batch normalizing technique for normalizing layer activations, which brought a significant speedup in the training process. Zhang et al. (2019)

Table 1 Summary of interval based initialization techniques

Initialization technique	Main characteristics	Goal
Nguyen and Widrow (1990)	Derived norm for weights such that hidden nodes remained linear in assigned interval	Achieve faster convergence
Mittal et al. (2020)	Modified the norm of weights to avoid saturation of network	Achieve faster convergence
Kim and Ra (1991)	Derived a minimum bound on weights	Achieve faster convergence
Wessels and Barnard (1992)	Derived weights by uniformly spreading decision boundary of hidden node in input space	Avoid false local minima
Drago and Ridella (1992)	Derived a scale factor for weights based on paralyzed neuron percentage	Avoid false local minima
Shimodaira (1994)	Controlled the length and range of weight vectors	Achieve faster convergence
Yam and Chow (2001)	Derived a maximum range for weights	Achieve faster convergence
Sodhi et al. (2014)	Derived an interval for weights such that output of each hidden node was in separate reion	Achieve faster convergence
Qiao et al. (2016)	Derived range for weights such that output of neurons remained in active region	Achieve faster convergence and avoid false local minima

have proposed a reliable method called fixup initialization for training residual networks without normalization. The weights of the last layer were initialized to zero, and layers other than the last layer were initialized by the technique given by He et al. (2015). A scaling factor has been derived and the layer weights inside residual branches were scaled accordingly. This method was compared with other methods given by Mishkin and Matas (2015), Balduzzi et al. (2017) and Ioffe and Szegedy (2015) against test accuracy versus depth of the residual network. This method outperformed other ones even on larger depths.

Ghazi et al. (2019) have proposed an initialization method for long short-term memory (LSTM) to consider the training instability. The weights were randomly initialized and were later scaled to maintain the input, output variances in the same range. The authors have derived the conditions on variance on traditional LSTMs as well as peehole LSTMs. This method was compared with methods proposed by Glorot and Bengio (2010) and Vorontsov et al. (2017) on univariate and multivariate time series and the results showed that this method performed better in terms of the number of epochs for convergence and MSE.

Table 2 shows the summary of variance scaling based initialization techniques.

2.1.3 Other random initialization techniques

This section discusses some custom random initialization techniques other than interval based and variance scaling based initializations.

Chen et al. (1991) have proposed two algorithms: forward and recurrent estimation algorithm for the estimation of initial weights in a three-layer neural network. The weights were initialized such that the error obtained was as near the global minimum, thereby reducing

Table 2 Summary of variance scaling based initialization techniques

Initialization technique	Main characteristics	Goal
Glorot and Bengio (2010)	Scaled weights such that variance of input is equal to variance of output	Achieve faster convergence, better accuracy
He et al. (2015)	Modified the scaling factor to consider rectifier non-linearities	Achieve depth independent performance
Yang et al. (2020)	Determined variance based on norm equality	Achieve faster convergence, better accuracy
Balduzzi et al. (2017)	Derived scale factor for weights based on gradient varaince	Achieve depth independent performance
Zhang et al. (2019)	Derived scale factor for weights in residual branches	Achieve depth independent performance
Ghazi et al. (2019)	Scaled weights such that input and output variance was maintained	Achieve faster convergence, better accuracy

the training time. The weights of the first layer were randomly initialized, whereas the weights for the next layer were obtained by employing pseudo-inverse method on hidden layer activation values. The number of epochs was used as a measure for convergence. The network was said to converge if the RMSE went below 0.1. The results showed a significant decrease in the number of epochs required for the proposed algorithms as compared to a fully randomized network.

Castro et al. (1998) have discussed an evolutionary approach based on a genetic algorithm (GA) to obtain initial weights. GA is applied in a randomly generated weight space and later, those regions of weight space are sampled, which may be near the optimal solution. This method has been compared with some non-evolutionary methods as uniform random initialization and those given by Nguyen and Widrow (1990) and Kim and Ra (1991). The results showed that the proposed techniques significantly reduced number of epochs for training to reach a pre-defined sum squared error of 0.1.

Saxe et al. (2013) have discussed an initialization method based on an orthonormal basis. Initially, the weights were picked up from a random normal distribution. These weights were later decomposed to an orthonormal basis with singular value decomposition (SVD). The authors have introduced a dynamical isometry condition in which the singular values of input–output Jacobian are near 1. This initialization method achieved dynamical isometry and thereby delivered depth-independent learning times. It was observed that random Gaussian initialization did not achieve this property.

Vorontsov et al. (2017) have addressed the problem of vanishing and exploding gradients on recurrent networks by putting soft constraints on weight matrices to ensure that orthogonality is maintained. The authors suggest that such soft constraints led to faster convergence and better performance.

Murru et al. (2016) have presented a Bayesian approach by customizing Kalman filter for initializing neural network weights. The initial state for weights is selected from a normal distribution and then the optimization is done by the Bayesian approach. This method has been compared with random initialization and initialization methods presented by Kim and Ra (1991), Drago and Ridella (1992), Shimodaira (1994) and Yoon et al. (1995) on different problems in terms of mean number of iterations to achieve convergence. The proposed method is useful for the convergence rate on a variety of applications.

Schneider (2020) have proposed a correlated initialization approach. In this method, a filter location was selected and the value at that location was randomly set. The other locations in the filter were assigned values based on the distance from the selected location. An uncorrelated filter is added with such an initialized filter to add noise independently to every location. The proposed method was seen to achieve faster convergence and it gets rid of zigzagging during training.

Table 3 shows a summary of initialization techniques discussed in this subsection.

2.2 Data-driven initialization

This section presents various weight initialization techniques in which the network weights are directly derived from the training data.

Duch et al. (1997) have reviewed various random initialization methods for multilayered perceptron. It has been suggested that initialization based on data clustering centers are appropriate for determining initial weights in case of classification tasks.

Lehtokangas et al. (1998) have proposed an initialization technique based on reference patterns, which are representative of the data. The reference patterns have been obtained by clustering and are assigned to hidden units. This ensures that the hidden units are within the active region. Also, the hidden units become sensitive to the reference pattern. The results of comparison of the proposed method with a random initialization method showed that this method allowed fast training.

Yam et al. (2002) have proposed a novel algorithm for multilayer perceptron for setting the optimal initial weights using independent component analysis (ICA). In this method, the initial hidden layer weights were those which captured important features of the input data. Thus, it boosted up the training speed as eventually, the learning algorithm also learns representations of the training data by adjusting weights. This method was based on the fixed-point ICA algorithm for calculating the separation matrix. This method also ensured that the output of neurons remained in the sigmoid activation function's active region. The proposed method showed significant speed up in convergence with respect to the mean number of iterations after comparison with methods proposed by Drago and Ridella (1992) and Yam and Chow (2000) on some benchmark datasets.

Gan et al. (2015) have presented a data-dependent weight initialization method where the filter weights of the convolutional neural network (CNN) were learned by principal component analysis (PCA). This network was called PCA-based convolutional network (PCN). For this, samples of a fixed size were extracted from the images in the training dataset and were represented in matrix form. PCA was applied to this obtained matrix and the eigenvectors formed filters for convolution. This method led to improved performance in terms of accuracy and reduced training time on standard image classification datasets.

Krähenbühl et al. (2015) have discussed two data-dependent techniques of weight initialization: PCA-based and k-means based. The weights were normalized such that the activations were equally distributed and the gradient ratio was maintained constant across all layers. K-means based initialization showed better performance than PCA-based initialization for both classification and detection tasks. The authors have compared the proposed initialization, which ensures that all the weights in a layer are learned at the same rate against existing PCA and k-means methods and it was observed that the pre-training time was reduced by three orders of magnitude for k-means based initialization.

Alberti et al. (2017) have employed the linear discriminant analysis (LDA) approach to initialize the layers of CNN for the task of image segmentation. The authors have presented

a comparison of performance of LDA-initialized and randomly initialized networks in terms of mean Intersection over Union (IU) versus epochs for different historical document images. The results showed that the network initialized with LDA is stable, performs better and converges faster.

Masden et al. (2020) have presented a weight initialization scheme for FFNN based on the LDA approach. The 'sorting game' approach presented aims to initialize the first layer filters of CNN, which, after training, extracts meaningful features. This method has helped to find a loss basin which is close to the global optimum as compared to random initialization schemes. This method has been compared with random initialization technique given by Saxe et al. (2013) on MNIST and fashion MNIST datasets and technique proposed by Glorot et al. (2010) on CIFAR-10 dataset. The results demonstrate that the proposed sorting game initialization gives a better training performance.

Hsiao et al. (2003) have given a method for initializing a backpropagation network with weights obtained from the partial least squares (PLS) algorithm. The PLS algorithm derives the loading matrices, which were used as weights. The model was then trained with these initial weights and generalized delta rule was adopted for updating the weights. The authors have compared this proposed method with backpropagation network with random initial weights and the results showed that this method gives faster convergence and also a lower root mean square error (RMSE).

Hasegawa et al. (2016) have proposed a partial least squares (PLS) based approach for initialization of convolutional layers in CNN. In this method, PLS has been applied to local crops taken from the training dataset. This method was compared with the PCA based initialization approach and has shown better performance in terms of accuracy for image classification tasks on MNIST and CIFAR-10 dataset.

Tang et al. (2017) have proposed a method of initializing the network based on k-means clustering. The k features were fed in the feature dictionary and were convolved with each image to obtain a feature map. The results showed that the model initialized using the k-means clustering method showed about 1.82% higher accuracy than random normally distributed weights. The variance and distribution of weights were also observed for both the types after optimization. K-means based method showed a small variance and regular distribution of weights.

Li et al. (2017) have presented an initialization technique for convolutional filters for sentiment analysis of movie reviews. The filters were initialized such that they captured semantic features of only useful n-grams from the text. The n-grams which had the value

Table 3 Summary of other random initialization techniques

Initialization technique	Main characteristics	Goal
Chen and Nutter (1991)	Applied pseudo-inverse on hidden layer activations	Achieve faster convergence
De Castro et al. (1998)	Genetic algorithm approach	Achieve faster convergence
Saxe et al. (2013)	Initialized weights with orthonormal matrices thereby achieving dynamical isometry	Achieve depth independent performance
Vorontsov et al. (2017)	Put soft constraints on weights to avoid vanishing and exploding gradients	Achieve faster convergence
Murru and Rossini (2016)	Kalman filter initialization approach	Achieve faster convergence
Schneider (2020)	Correlated initialization approach	Achieve faster convergence

of Naïve Bayes weight much greater than one were selected as important n-grams from the training data. The n-gram embeddings were later clustered using k-means algorithm and the center of the filters were initialized with centroid vectors. The proposed method has been compared on various datasets in terms of performance on text classification task and the results show that this method performed better than random initialization techniques.

Cachi et al. (2020) have proposed a weight initialization technique by using the input samples for initializing the connection weights in the context of competitive spiking neural networks (CSNN). Such a type of initialization helps to reduce the training time, as when the connection weights, which are initialized with input pixels, the neurons find prototypes that are close to the final prototypes as compared to the random initialization techniques. This proposed technique was evaluated on the MNIST dataset to observe convergence performance and testing accuracy. The results show that this technique has helped to achieve a reasonably faster convergence as compared to random initializations.

Table 4 gives a summary of data-driven initialization techniques.

2.3 Hybrid initialization techniques

This section discusses the initialization techniques, which fall into both the categories—random initialization and data-driven initialization.

Mishkin et al. (2015) have given a layer sequential unit variance (LSUV) approach for initialization of weights. This technique first chose random weights and converted them to orthonormal matrices, as given by Saxe et al. (2013). The variance of each output obtained using a batch of training data was computed and scaled to maintain the variance to one. The initialization methods given by Glorot and Bengio (2010), He et al. (2015), Saxe et al. (2013) and Mishkin and Matas (2015) were compared and the results showed that the LSUV method performed well in terms of accuracy and convergence except for tanh activation. This technique is thus a hybrid of data-driven and variance scaling based initialization.

Aguirre et al. (2019) have proposed an initialization technique for ReLU layers and output layers. This technique first initialized the weights with orthonormal matrices as given by Saxe et al. (2013). An active fraction hyperparameter which gives the likeliness of a ReLU unit to give a non-zero value was used for initializing the bias. The weights were then scaled similar to the method given by Mishkin and Matas (2015) to get a desired standard deviation for layer outputs. The output layer has been initialized with pseudo-inverse of last hidden layer output. This method has been compared with Glorot and Bengio (2010), He et al. (2015) and Mishkin and Matas (2015) and results show that this method observes a higher accuracy in a lesser number of epochs. This technique is thus a hybrid of data-driven and variance scaling based initialization.

Koturwar et al. (2017) have presented a weight initialization scheme exploiting the statistics of the data. The results of the strategies given by Glorot and Bengio (2010) and He et al. (2015) were studied on practical datasets and it was seen that these techniques did not work well for practical datasets. The authors have discussed that the weight initialization should be such that the dot product of weights and data is maximum when the data contains relevant information and minimum when it contains irrelevant information. The authors proposed an algorithm where the weights were initialized from a normal distribution with mean and covariance calculated from the random crops from the image and later were scaled by the method proposed by He et al. (2015). This technique is thus a hybrid

of data-driven as it uses data statistics and random as it picks random weights from the derived normal distribution.

Table 5 gives a summary of hybrid intialization techniques discussed in this section.

3 Initialization with pre-training

This section discusses various techniques in the literature which obtain the initialization values for a network defined by an unsupervised task. Such unsupervised pre-training techniques help to place the parameter values in a near-optimum place, thus achieving effective optimization. The pre-training approach also improves the generalization by learning quality features from the data (Erhan et al. 2010).

Bengio et al. (2007) have performed various experiments and have discussed that the unsupervised pre-training strategy helps in optimization of neural networks as this technique initializes weights close to the global optimum. Such unsupervised techniques may be used when it is difficult to understand the relation of the input distribution and the variable to be predicted.

Larochelle et al. (2009) have also discussed after experimentation that unsupervised weight initialization helped to select the parameters in a region of suitable local minimum and also acted as a regularization that offered good generalization. The authors have compared the models: stacked restricted Boltzmann machine (SRBM), stacked autoencoders (SAA), deep networks with supervised pre-training and no pre-training on MNIST dataset in terms of classification errors during validation and testing. The methods with unsupervised pre-training showed fewer errors as compared to methods with no pre-training. These

Table 4 Summary of data-driven initialization techniques

Initialization technique	Main characteristics	Goal
Lehtokangas and Saarinen (1998)	Initialized using reference patterns obtained by clustering	Achieve faster convergence
Yam et al. (2002)	ICA approach	Achieve faster convergence
Gan et al. (2015)	PCA approach	Achieve faster convergence, better accuracy
Krähenbühl et al. (2015)	PCA and k-means approach	Learn meaningful representations
Alberti et al. (2017)	LDA approach	Achieve faster convergence, better accuracy
Masden and Sinha (2020)	Sorting game approach based on LDA	Achieve faster convergence, better accuracy
Hsiao et al. (2003)	PLS approach	Achieve faster convergence
Hasegawa and Hotta (2016)	PLS approach	Achieve better accuracy or error rate
Tang et al. (2017)	k-means clustering approach	Achieve better accuracy or error rate
Li et al. (2017)	k-means clustering approach	Achieve better accuracy or error rate
Cachi et al. (2020)	Initialized with input pixels	Achieve faster convergence, better accuracy

experiments thus confirmed that unsupervised learning initializes parameters in the region of the optimum solution.

Li et al. (1993) have proposed a delta pre-training method to obtain initial weights for the backpropagation algorithm. The pre-training started by assuming initial weights to be zero. The weights obtained after pre-training the network were set as initial weights. The average number of epochs was used as a measure for convergence. The authors have discussed that this method of weight initialization has very less probability of getting stuck in local minima as compared to random weight initialization.

Huang et al. (2007) have proposed a method for learning invariant features with an encoder–decoder architecture that captures the features in an image along with there locations. These features have been trained in an unsupervised manner and later, a supervised classifier has been employed on these features. The features learned from the unsupervised method are locally shift invariant. This method was tested on MNIST and Caltech-101 dataset and it gave state-of-the-art results on the MNIST dataset.

Paine et al. (2014) have proposed a zero-bias convolutional autoencoder (CAE) to learn the parameters from the input data in an unsupervised way. A randomly initialized CNN has been compared with the proposed technique. The results demonstrated that the pre-trained model showed about 3.87% better accuracy on the STL-10 dataset than the randomly initialized CNN. The classification accuracy showed improvement when the ratio of unsupervised to supervised samples was increased.

Ruiz-Garcia et al. (2017) have employed stacked convolutional autoencoder (SCAE), which pre-trained the weights for CNN for the task of emotion recognition from facial recognition. The authors have compared the performance of random weight initialization and the presented technique. It was observed that the pre-training method using SCAE gave better accuracy and also reduced the training time.

Masci et al. (2011) have proposed unsupervised feature learning using stacked convolutional autoencoder. The results presented show that this method has outperformed unsupervised training methods on the CIFAR-10 dataset and also this method performed considerably better than randomly initialized networks in terms of recognition rate.

Tan et al. (2014) have presented an unsupervised pre-training based method on SCAE for steganalysis of digital images. This method as been compared with random initialized steganalyzer and results show that the proposed pre-training approach gave a significantly small detection error.

Ferreira et al. (2018) have presented an autoencoder based pre-training method for weight initialization to classify papillary thyroid carcinoma from gene expression. The authors have compared the performance in terms of F1-score for different autoencoders and the results have shown that pre-trained denoising autoencoder (DAE) followed by fine-tuning achieved a very good overall result.

Zhang et al. (2018) have proposed a technique by which the filters were initialized by values learned by the AdapDAE algorithm. The advantages of both denoising autoencoders and sparse autoencoders were combined for the filter initialization. Firstly the random patches from input data were picked up and a corrupted version of the data was obtained. The noise level for corrupting the data was adaptive and worked on the principle of annealing. The weights were learned in the same way as in autoencoders. The results showed improved classification accuracy when tested on benchmark datasets.

Wiehman et al. (2016) have employed unsupervised pre-training on U-net architecture, which is a fully convolutional architecture for image segmentation task. The authors have evaluated this proposed method using statistical tests for equality of means and variance. This method was compared with a fully supervised learning method and the results show

Table 5 Summary of hybrid initialization techniques

Initialization technique	Main characteristics	Goal
Mishkin and Matas (2015)	Scaled orthogonal weights such that variance of every layer is one	Achieve faster convergence
Aguirre and Fuentes (2019)	Scaled weights to maintain the desired standard deviation	Achieve faster convergence, better accuracy
Koturwar and Merchant (2017)	Derived parameters of normal distribution from training data	Achieve faster convergence

that it is more robust than random initialization as it aids to reduce the model variance, which supports the observation by Erhan et al. (2010).

Tu et al. (2017) have presented convolution sparse filter learning (CSFL) technique for face classification task. The discriminative features of the input data were obtained from the CSFL algorithm in an unsupervised way. The filters of the first layer of CNN were initialized using these features and CNN was then trained in a supervised way. This method showed better results in terms of MSE over epochs as compared to some random initialization techniques.

Dai et al. (2015) have proposed a semi-supervised approach for initialization of weights for recurrent networks. This approach is based on pre-training a sequence autoencoder to obtain weights, which can be later used as initialization for the recurrent network. The authors have tested this method for sentiment analysis, text classification and object classification on standard datasets using LSTM initialized with sequence autoencoder. The results showed that this method performed well for document classification tasks.

Trinh et al. (2019) have presented a self-supervised pre-training approach for initialization of Resnet-50 CNN architecture. An encoder–decoder module has been used in the pre-training phase. This encoder–decoder understands the global content of the image, which later helps in classifying images. The image is first sampled into patches. A few patches called as distractor patches are fed as an input to the decoder and are masked from the image given to the encoder. The encoder takes patches of an image as input and gives a vector representing the image as output. The decoder takes distractor patches of the image as input and is trained such that it finds the correct distractor patch for the masked location. The first three blocks of Resnet-50 architecture have been initialized from this pre-trained patch processing unit. The authors have compared this technique of initialization with random initialization on CIFAR-10 dataset and results demonstrate that this technique improves the generalization ability of the trained model.

Sudowe et al. (2016) have presented a self-supervised pre-training method to obtain the initial weights for image recognition. Random patches from the image are first extracted and each patch is classified using a network initialized with random weights as given by He et al. (2015). This pre-training phase thus encodes the patch structure from the training images. The final parameters are used as initial weights for the network performing the final task of image recognition. The authors have employed VGG16 on the PARSE-27k dataset. A comparative analysis of weight initializations given by He et al. (2015) and this paper was presented where this self-supervised method showed improved accuracy and also faster convergence.

Table 6 gives a summary of the pre-training based initialization techniques.

4 Discussions on goals of weight initialization strategies

The training speed of neural networks depends on initial weights, the learning rate and the depth of the network. The research carried out for determining an appropriate method for the selection of initial weights has been discussed in the previous sections. It is evident from the discussed literature that, researchers have proposed weight initialization strategies achieving either or some of the following:

- *Faster convergence* The time taken for network convergence is the time taken for updating the initial weights to reach optimum values such that the loss is minimized. This convergence time is measured in terms of number of epochs required to reach a desired value of the loss. The initialization techniques which were proposed with a goal of achieving faster convergence have been listed in Table 7 along with the details of experiments carried out and evaluation metrics.
- *Avoiding getting stuck in false minima* The conditions when the network gets stuck in false minima have been pointed out by Wessels and Barnard (1992). The initialization techniques, as listed in Table 8, ensure that the network does not get stuck in such false minima and are important for efficient network convergence.
- *Depth independent performance* Some initialization techniques as listed in Table 10 aim to achieve training times that do not depend on the depth of the neural network. Such schemes are important to achieve considerably faster convergence for deeper networks while also ensuring good performance in terms of accuracy.
- *Learning meaningful representations from data* The initial weights of the network are directly responsible for capturing meaningful features from the data, thereby improving performance. The initialization techniques which have focused on learning good features from data are listed in Table 11.
- *Better accuracy or error rate* Some research papers, as listed in Table 12, have evaluated the performance of proposed scheme by observing the accuracy or error rate on the testing dataset.

Tables 7, 8, 9, 10, 11, 12 and 13 provides an overview of weight initialization techniques discussed in this paper based on the aim of technique, task and method of evaluation.

5 Initialization for non-iterative training mechanism

Weight initialization techniques play an important role in non-iterative training approaches as the weights are initialized randomly and weights of some layers are kept unchanged throughout the training process. This may lead to a significant impact on the model performance. A review of literature focussing on the analysis based on universal approximation of such mechanism has been presented by Wang and Cao (2018). NNRW have achieved success in various fields, however model initialization is a challenging task (Cao et al. 2019). This section provides a brief overview of the literature covering initialization techniques or optimal weight selection by optimization for non-iterative training mechanism for different NNRW. The goal of all the initialization techniques for non-iterative training mechanisms is to achieve faster convergence and better generalization.

5.1 Evolutionary algorithm based selection of input weights

This subsection discusses different evolutionary algorithm based methods used for obtaining optimal input weights for NNRW.

Zhu et al. (2005) have proposed evolutionary ELM (E-ELM) in which the input weights have been selected by differential evolutionary (DE) algorithm (Storn and Price 1997) and output weights have been obtained analytically. The initial weights in population were taken randomly in the range $[-1, 1]$. The fitness of individuals in this population was evaluated based on the validation RMSE. Then mutation, crossover and selection were carried out. The norm of output weights was also considered as a fitness criteria. This method was compared with gradient based backpropagation and method proposed by Ghosh and Verma (2003) on different classification and regression tasks using single hidden layer feed-forward networks (SLFN) and the results demonstrate that evolutionary ELM achieves better generalization and faster learning.

Cho et al. (2007) have proposed bacterial foraging algorithm (Passino 2002) for finding optimal input weights for ELM. The optimal weights were obtained from random initial weights with different steps in the algorithm such as chemotactic, reproduction and elimination-dispersal steps. This method shows better generalization on function approximation and a classification task when compared with backpropagation, support vector machines and ELM.

Han et al. (2011) have modified the particle swarm optimization (PSO) (Kennedy and Eberhart 1995) by considering validation RMSE and norm of output weights as fitness measures while weight optimization. This modified technique was compared with ELM, E-ELM (Zhu et al. 2005) and was seen to have better generalization performance.

De Oliveira et al. (2012) have presented a modified artificial fish swarm algorithm (MAFSA) combined with two fuzzy rules for selecting optimal weights for ELM. In the proposed method, the visual and step parameters of the AFSA and MAFSA were adjusted based on two fuzzy strategies. This method was compared with other ELMs on classification tasks in terms of mean accuracy errors. The proposed technique was seen to have performed well on test dataset.

Cao et al. (2012) have proposed self-adaptive E-ELM (SaE-ELM). The DE-LM (Subudhi and Jena 2008) and E-ELM (Zhu et al. 2005) manually choose control parameters and vector generation techniques. However, this proposed method chose these in a self-adaptive manner. The comparison results of SaE-ELM with DE-ELM and E-ELM showed that SaE-ELM outperformed the other two algorithms.

Matias et al. (2013) have proposed genetically optimized ELM (GO-ELM). The proposed methodology was tested on five benchmark datasets and compared with the techniques proposed by Leung et al. (2003) and Cao et al. (2012) and Levenberg–Marquardt algorithm. The results demonstrated that this proposed method is statistically better.

Pacifico et al. (2013) have proposed a particle swarm optimization (PSO) based approach alongwith population stereotyping using clustering for obtaining optimal input weights from initial random weights in the range $[-1, 1]$. Validation RMSE and norm of output weights was used as fitness criteria as presented by Han et al. (2011). The proposed technique has been evaluated on four classification tasks in terms of test accuracy.

Yang et al. (2015) have proposed quantum-behaved PSO approach for ELM (QPSO-ELM). This approach was based on minimizing both the empirical and structural risk ensuring that the ELM does not overfit. This approach was compared with ELM, E-ELM and PSO-ELM (Xu and Shu 2006) on function approximation and classification tasks. The

Table 6 Summary of pre-training based initialization techniques

Initialization technique	Main characteristics	Goal
Bengio et al. (2007)	Performed experiments to study advantages of unsupervised pre-training techniques	Learn meaningful representation
Larochelle et al. (2009)	Observed that unsupervised pre-training techniques initialize weights in suitable local minima and give better generalization	Achieve better accuracy or error rate
Li et al. (1993)	Proposed a delta pre-training approach	Avoid false local minima
Huang et al. (2007)	Proposed encoder–decoder architecture for unsupervised pre-training approach which captured shift invariant features	Learn meaningful representation
Paine et al. (2014)	Proposed zero-bias CAE approach for unsupervised pre-training	Achieve better accuracy or error rate
Ruiz Garcia	Used SCAE for unsupervised pre-training	Achieve better accuracy or error rate
Masci et al. (2011)	Used SCAE for unsupervised pre-training	Achieve better accuracy or error rate
Tan and Li (2014)	Used SCAE for unsupervised pre-training	Achieve better accuracy or error rate
Ferreira	Used DAE for unsupervised pre-training	Achieve better accuracy or error rate
Zhang and Zhang (2018)	Proposed AdapDAE algorithm for unsupervised pre-training	Achieve better accuracy or error rate
Tu et al. (2017)	Convolutional sparse filter learning approach	Achieve faster convergence, better accuracy
Dai and Le (2015)	Semi-supervised approach using autoencoder	Achieve better accuracy or error rate
Trinh et al. (2019)	Self-supervised pre-training approach using encoder–decoder	Achieve better accuracy or error rate
Sudowe and Leibe (2016)	Self-supervised pre-training approach	Achieve faster convergence, better accuracy

comparison results showed that QPSO-ELM are compact and have better generalization performance than the others compared.

Zhang et al. (2015) have proposed a memetic algorithm based ELM (M-ELM) for selection of optimal weights for ELM for classification task. The proposed memetic algorithm helped to get over with the problems of both global optimization and local search techniques. A variant of DE algorithm (Zhang and Sanderson 2009) was used for global optimization and simulated annealing (Rodriguez et al. 2012) is used for local search. The optimal individuals were obtained after the defined stopping criteria is reached. This proposed method was tested on 22 datasets for classification and compared with some state-of-the-art variants of ELM and the results showed that it performs better in terms of classification accuracy.

Mohapatra et al. (2015) have proposed an improvement to the traditional cuckoo search algorithm (Yang 2010). The improved algorithm modified the step size and probability factor such that the system is stable. This improved cuckoo search algorithm was combined

Table 7 Initialization techniques which aim to achieve faster convergence

Initialization technique	Architecture	Tasks	Evaluation metric
Lehtokangas and Saarinen (1998)	MLP	Two spirals problem, channel equalization problem	MSE and relative entropy versus epochs
Yam et al. (2002)	MLP	Prediction, classification	Mean number of iterations
Nguyen and Widrow (1990)	2-Layer NN	Function approximation	MSE versus training time (number of pattern presentations)
Kim and Ra (1991)	2-Layer NN	2 benchmark problems: XOR, parity 3	LMS error versus number of iterations
Shimodaira (1994)	2-Layer NN	XOR, random mapping problem	Mean value of SSE versus sweeps, mean and standard deviation of number of sweeps
Yam and Chow (2001)	2-Layer NN	Classification, prediction, function mapping	Number of iterations, number of networks getting stuck, generalization performance
Sodhi et al. (2014)	2-Layer NN	Function approximation	Number of epochs to reach a pre-defined MSE
Mittal et al. (2020)	3-Layer FFNN	Function approximation	Left tailed t test result for deriving optimal value of parameter α
Hsiao et al. (2003)	3-Layer FFNN	Multivariate analysis	RMSE versus training epochs
Chen and Nutter (1991)	3-Layer FFNN	6 benchmark problems: XOR, parity, majority, identity, decoder, symmetry	Number of epochs to reach a pre-defined RMSE
De Castro et al. (1998)	3-Layer FFNN	4 benchmark problems: XOR, parity 3, parity 4, encoder–decoder, function approximation	Number of epochs to reach a pre-defined SSE
Murru and Rossini (2016)	FFNN	Classification, regression	Percentage of correct recognition, number of epochs for convergence, number of networks that did not converge
Mishkin and Matas (2015)	CNN (FitNet)	Classification	Test accuracy (%) versus number of epochs for different activation functions
Koturwar and Merchant (2017)	CNN	Classification	Validation accuracy (%) versus epochs
Schneider (2020)	CNN (ResNet-10, VGG-10 variant)	Classification	Training and validation accuracy versus number of epochs
Vorontsov et al. (2017)	RNN	Classification, character prediction	MSE versus number of epochs

Table 8 Initialization techniques which avoid getting stuck in false minima

Initialization technique	Architecture	Tasks	Evaluation metric
Drago and Ridella (1992)	2-Layer NN	3 benchmark problems: parity, symmetry, encoder	Ratio of total successful epochs (RSS) and expected epochs to the solution (EES)
Li et al. (1993)	2-Layer NN	3 benchmark prroblems—encoder–decoder, compression problem, XOR problem	Average number of epochs
Wessels and Barnard (1992)	3-Layer FFNN	Classification	Average classification performance (%)

Table 9 Initialization techniques which aim to achieve faster convergence and avoid getting stuck in false minima

Initialization technique	Architecture	Tasks	Evaluation metric
Qiao et al. (2016)	2-Layer NN	Classification, function approximation, prediction	Initialization time in seconds, training time in seconds, RMSE

with ELM (ICSELM) to select the input weights and the performance was evaluated for four benchmark medical classification datasets.

Eshtay et al. (2018) have proposed a method for selecting the best set of parameters for ELM such that the generalization performance is improved. The input-to-hidden node weights in ELM were randomly initialized and output weights were obtained from Moore-Penrose inverse method (Huang et al. 2004). The authors have combined the competitive swarm optimization (CSO) algorithm with ELM. The CSO algorithm found the optimized set of input-to-hidden layer parameters such that the model generalized well. This method was evaluated for 15 medical datasets in terms of accuracy.

Table 14 summaries the evolutionary algorithm based weight selection techniques for non-iterative training mechanism.

5.2 Other weight initialization techniques

This subsection explores the weight initialization techniques in the literature which have been adopted to set the input weights directly for NNRW.

Javed et al. (2014) have presented a summation wavelet ELM (SW-ELM) to improve the generalization and fast learning ability of ELM. This SW-ELM holds two activation functions for hidden neurons which aids to overcome non-linearity. The SW-ELM was initialized by the technique as proposed by Nguyen and Widrow (1990). This technique was seen to have significant performance in time series prediction tasks.

Table 10 Initialization techniques which aim to achieve depth independent performance

Initialization technique	Architecture	Tasks	Evaluation metric
He et al. (2015)	CNN (VGG-19), Fast R-CNN	Classification, object detection	Top 5 error on testing dataset, mean average precision (mAP)
Balduzzi et al. (2017)	CNN	Classification	Test accuracy versus depth
Zhang et al. (2019)	CNN (WRN), Resnet, transformer	Classification, machine translation	First epoch test accuracy versus depth, test error (%), BLEU score
Saxe et al. (2013)	Deep linear NN	Classification	Number of epochs to reach a pre-defined error versus depth of network

Tapson et al. (2015) have presented a technique for computation of input weights for ELM. This technique produced input weights for ELM as randomized linear combinations of normalized training data. The proposed technique was compared with traditional ELM for MNIST classification task in term of accuracy. The results demonstrated that this method of computing initial weights outperformed the traditional ELM.

Huang et al. (2015) have presented a local receptive field based ELM (LRF-ELM) for object recognition task. The authors have first initialized the input weights randomly from Gaussian probability distribution. The weights were then orthogonalized using SVD. The effect of orthogonalization was also studied and it was seen that orthogonalized weights helped to extract complete set of features.

Pang et al. (2016) have presented deep CNN based ELM (DC-ELM) for the recognition of handwritten digits. The input-to-first convolutional layer weights were initialized same as presented in Huang et al. (2015). This algorithm was compared with ELM, LRF-ELM and state-of-the -art neural networks and it was observed that this algorithm performs well in terms of test accuracy.

Wang et al. (2017) have discussed different methods of generating the input weights for ELM such that the input weights are capable of sample structure preserving. The authors have proposed a random orthogonal projection method and compared it with Monte Carlo random sampling and Quasi-Monte Carlo method (Caflisch et al. 1998) in terms of error rates for 11 datasets. The results demonstrated that the proposed projection method aided ELM towards better generalization capability.

Table 15 gives a summary of initialization techniques discussed in this subsection.

6 Analysis of weights

Analysis of trained weights is vital to learn about the insights captured by weights after training. These captured insights may be utilized to propose a novel technique for initializing deep neural network weights. It may happen that the final loss function of a trained model may have different values close to the global optimum of loss function for different combinations of network parameters. For example, a network initialized with a random initialization technique with different random seeds learns different weights. However, if these networks are trained using the same optimization algorithm, then the networks will

Table 11 Initialization techniques which aim to learn meaningful representations

Initialization technique	Architecture	Tasks	Evaluation metric
Huang et al. (2007)	CNN	Classification	Classification error (%) on test set
Krähenbühl et al. (2015)	CNN (CaffeNet, GoogLeNet, VGG), Fast R-CNN	Classification, object detection	mAP
Bengio et al. (1994)	Deep belief network (DBN), Deep network	Classification, prediction	Classification error (%), MSE

have a similar quantitative performance. In neural networks, representations of the data are learned by the weights. As deeper the network goes, higher and complex representations of the data are learned. Thus the visualization and analysis of learned network weights and quantifying and finding useful representations is an emerging area of research. This section discusses the different types of analysis and visualization techniques presented by researchers.

Go et al. (1999) have investigated the distribution of trained weights to improve the training process. The authors have observed a set of weight points obtained after training a simple neural network in the weight space. The analysis is done based on Eigenvalues of the covariance matrix obtained from the set of weight points. The authors have suggested that results obtained from such an analysis may help select initial weights for any neural network.

Gabrielsson et al. (2018) have applied topological data analysis (TDA) to visualize the learned weights of CNN. The experiments with the MNIST, CIFAR10 dataset show that the topological properties captured by this analysis represent the input data by detecting edges and its rotations.

Fountain et al. (2019) have presented how varying training produces varying patterns in the weights of CNNs by visualization experiments. The results show that the filters trained for more number of iterations when applied to images generate feature maps with more softness.

Burnaev et al. (2016) have given a comparative analysis of some existing weight initialization techniques and have also developed three algorithms for the initialization of regression model parameters. A particular initialization method was considered efficient in terms of error by observing the logarithm of the ratio of median errors for that method and a reference method. The reference method was as given by Nguyen and Widrow (1990). The metric used to quantify this error was MSE, mean absolute error (MAE) and 95 and 99 percentiles of absolute error. To figure out a method with fast learning speed, the number of iterations was used.

Statistical analysis techniques are important as to analyze the results of an algorithm for a certain task and identify whether an algorithm is better than the other suggested algorithms. Demšar (2006) has given a detailed description of various statistical tests which can help to compare two classifiers. The author has discussed paired t test and ANOVA in the category of parametric tests and Wilcoxon signed-rank test and Friedman test in non-parametric tests.

A comprehensive explanation of non-parametric statistical techniques has been presented by García et al. (2010). The authors have presented non-parametric tests, namely

Table 12 Initialization techniques which aim to achieve better accuracy or error rate

Initialization technique	Architecture	Tasks	Evaluation metric
Paine et al. (2014)	CAE	Classification	Classification accuracy (%)
Ruiz-Garcia et al. (2017)	SCAE	Emotion recognition from facial expressions	Confusion matrix, accuracy (%)
Masci et al. (2011)	SCAE	Classification	Classification performance
Tan and Li (2014)	SCAE	Steganalysis	Detection error
Ferreira et al. (2018)	DAE	Classification	Validation loss versus epochs
Hasegawa and Hotta (2016)	CNN	Classification	Accuracy (%)
Tang et al. (2017)	CNN	Weed identification	Accuracy (%)
Zhang and Zhang (2018)	CNN-Adap-DAE	Classification	Testing accuracy (%)
Li et al. (2017)	CNN	Natural language processing (NLP)—text classification	Testing accuracy (%)
Trinh et al. (2019)	CNN (ResNet 50)	Classification	Testing accuracy (%)
Dai and Le (2015)	LSTM	Text classification, image recognition	Test error rate
Larochelle et al. (2009)	SRBM, SAA, Stacked logistic autoregressions network	Classification	Test classification error
Wiehman et al. (2016)	U-Net	Image segmentation	Rand score

Friedman test, multiple sign-test and contrast estimation based on medians and post hoc tests based on p values and adjusted p values. Such techniques are useful in carrying out a comparison of the performance of various algorithms for a common task and thus, there is a need to explore them in the context of weight initialization techniques as well as to figure out a technique that gives better performance.

Masood et al. (2015) have analyzed different weight initialization techniques for different tasks based on the statistical measure one-sided tailed t test, mean and standard deviation of error.

Ramos et al. (2017) have presented an evaluation technique to identify a statistically superior weight initialization scheme among various other weight initialization schemes. The authors have recorded the probability of correct classification for a network with a different sampling of weights for each initialization scheme. This probability was considered to be a random variable and the probability density function of this random variable

Table 13 Initialization techniques which aim to achieve faster convergence and better accuracy

Initialization technique	Architecture	Tasks	Evaluation metric
Glorot and Bengio (2010)	FFNN	Classification	Test error (%) versus number of examples seen while online training
Masden and Sinha (2020)	FFNN	Classification	Accuracy versus number of epochs, validation error (%)
Gan et al. (2015)	PCN	Classification	Accuracy (%), training time in seconds
Alberti et al. (2017)	CNN	Image segmentation	Mean IoU versus number of epochs, accuracy
Sudowe and Leibe (2016)	CNN (VGG 16)	Classification	mAP and number of epochs
Ghazi et al. (2019)	LSTM	Regression	Training loss versus number of epochs, MSE on test dataset
Tu et al. (2017)	CNN	Face classification	Training and testing accuracy (%), training MSE versus number of epochs
Yang et al. (2020)	CNN	Classification	Loss versus number of iterations
Aguirre and Fuentes (2019)	CNN	Classification	Accuracy versus number of epochs, test accuracy and loss
Cachi et al. (2020)	CSNN	Classification	Training accuracy versus number of samples, testing accuracy

has been studied to find out if the initialization schemes being compared are statistically different.

Tao et al. (2016) have analyzed the performance of extreme learning machines initialized by random weights and biases with different variances for classification tasks. The performance for initializations with different variances has been compared based on Wilcoxon signed-rank and Friedman test.

Cao et al. (2017a) have carried out experimentation to check whether the randomization range of $[-1, 1]$ for input weights and [0,1] for hidden biases is optimal for ELM. For experimentation, the authors have selected eight benchmark functions. The performance in terms of mean training and testing RMSE and standard deviation was checked for different ranges for input weights and hidden biases. The results of the experimentation showed that the range of $[-1, 1]$ for input weights and [0,1] for hidden biases may not prove to be optimal. The authors have suggested to use some random search techniques for selection of optimal parameters for ELM.

Zhang et al. (2018) have presented the results of experimentation carried out to study the generalization of ELM for 24 different random weight initializations for 30 classification tasks. The authors have considered 24 continuous probability distributions from bell-shaped, heavy-tailed and light-tailed categories. The performance of ELM for these initializations was presented in terms of testing accuracy and Wilcoxon signed-rank and Friedman tests. The authors have concluded from the observations that the initializations from heavy-tailed distributions help to achieve good generalization.

Table 14 Summary of optimization techniques for selecting optimal weights by optimization for non-iterative training mechanisms

Technique	Main characteristics	Architecture	Tasks	Evaluation metric
Zhu et al. (2005)	Used differential evolution based algorithm for ELM	SLFN	Regression, classification	RMSE, Testing accuracy (%)
Cho et al. (2007)	Used bacterial foraging algorithm for ELM	SLFN	Function approximation, classification	RMSE, success rate
Han et al. (2011)	Modified particle swarm optimization for ELM	SLFN	Function approximation, classification	RMSE, accuracy (%)
de Oliveira and Ludermir (2012)	Employed fuzzy rules to find parameters of MAFSA for ELM	SLFN	Classification	Mean validation accuracy (%)
Cao et al. (2012)	Proposed self-adaptive evolutionary ELM	SLFN	Regression, classification	Testing RMSE, testing accuracy (%)
Matias et al. (2013)	Used genetic optimization for ELM	SLFN	Regression	Mean testing RMSE
Pacifico and Ludermir (2013)	Used particle swarm optimization and clustering for ELM	SLFN	Classification	Average test accuracy (%)
Yang et al. (2015)	Used quantum behaved particle swarm optimization for ELM	SLFN	Function approximation, classification	RMSE, Accuracy
Zhang et al. (2015)	Used memetic algorithm which is a combination of global optimization and local search method	SLFN	Classification	Accuracy (%)
Mohapatra et al. (2015)	Improved cuckoo search algorithm for ELM	SLFN	Classification	Accuracy, sensitivity, specificity, confusion matrix, Gmean, F-score
Eshtay et al. (2018)	Used competitive swarm optimization for ELM	SLFN	Function approximation, classification	RMSE, accuracy

Cao et al. (2017b) have presented a study of different probability distributions for initialization of RVFL. The authors have studied the effect of uniform, Gauusian and gamma distribution on RVFL for 10 benchmark datasets.

Cao et al. (2019) have proposed a similar study as Cao et al. (2017b), to study the distribution of various attributes in the dataset and relating it with initialization of NNRW to find appropriate initial weights. For the study, the authors have used seven datasets with attributes having different distributions such as uniform, gamma and Gaussian distribution. The NNRW was also initialized with these three distributions with different parameters. The experimentation was carried out for both ELM and RVFL. The results for different distributions were compared in terms of training RMSE versus number of hidden nodes.

Cao et al. (2020) have studied the relation between performance of ELM and RVFL and input data rank. A dispersion degree of matrix information distribution (DDMID) was also discussed which can be used for evaluation of initialization technique and feature mapping.

Table 15 Summary of initialization techniques for non-iterative training mechanisms

Technique	Main characteristics	Architecture	Tasks	Evaluation metric
Javed et al. (2014)	Proposed a summation wavelets based ELM for ELM	SLFN	Time series prediction	R2 score, coefficient of variation of RMSE
Tapson et al. (2015)	Proposed a method for computation of input weights based on linear combination of training data for ELM	SLFN	Classification	% Error
Huang et al. (2015)	Used random orthogonalized input weights for ELM	CNN	Classification for ELM	Error rate
Pang and Yang (2016)	Used random orthogonalized input-to-first convolutional layer weights for ELM	CNN	Classification	Accuracy
Wang and Liu (2017)	Proposed random orthogonal projection method for ELM	SLFN	Regression, classification	RMSE, error rate

7 Conclusions and outlook

Weight initialization is an important step in training any deep neural network. This paper has offered an extensive review of the research done in the field of weight initialization for neural networks for iterative as well as non-iterative training mechanisms. The existing techniques have been categorized based on the initialization method. The methods for the analysis of the learned weights and evaluation of weight initialization strategies available in the literature have also been presented. Based on the literature review carried out, the broad goals which have been achieved by researchers after proposing the weight initialization strategies have been discussed.

Deep neural networks have achieved great success in various tasks of machine learning and computer vision. This is because such networks learn the representations of input data and as the depth of the network increases, better representations are learned. In spite of this success, there is a need to provide strong theoretical reasoning for the behavior of this representation learning. We hope that understanding the insights captured by weights and studying the relations between the input data and learned weights would aid in giving relevant reasoning for the same.

A few initialization techniques suitable for non-iterative training mechanisms have been discussed in the literature. There is a need to explore this potential research area and propose initialization techniques which would prove suitable for such mechanism.

Finally, for better performance, the dataset size is increasing and the networks are going deeper. Thus, there is a need to evaluate various initialization techniques on real-world datasets on state-of-the-art architectures. An initialization scheme that will achieve all of the discussed goals and which will significantly speed up the tedious iterative training

process is needed. We hope that this paper provides a clear understanding of weight initialization, its importance and also encourages further research in this field.

Acknowledgements The authors would like to thank Dr. Shrinivas P. Mahajan, Head of Department, E&TC, College of Engineering, Pune for encouraging to carry the research work at the department. The authors would also like to thank Center of Excellence in Signal and Image Processing (CoE-S&IP) at College of Engineering, Pune for providing the necessary resources for this research work. The authors would also like to thank the reviewers for their valuable comments.

References

Adam SP, Karras DA, Magoulas GD, Vrahatis MN (2014) Solving the linear interval tolerance problem for weight initialization of neural networks. Neural Netw 54:17–37

Aguirre D, Fuentes O (2019) Improving weight initialization of relu and output layers. In: International conference on artificial neural networks. Springer, pp 170–184

Alberti M, Seuret M, Pondenkandath V, Ingold R, Liwicki M (2017) Historical document image segmentation with LDA-initialized deep neural networks. In: Proceedings of the 4th international workshop on historical document imaging and processing, pp 95–100

Atakulreka A, Sutivong D (2007) Avoiding local minima in feedforward neural networks by simultaneous learning. In: Australasian joint conference on artificial intelligence. Springer, pp 100–109

Balduzzi D, Frean M, Leary L, Lewis J, Ma KWD, McWilliams B (2017) The shattered gradients problem: if resnets are the answer, then what is the question? In: Proceedings of the 34th international conference on machine learning, vol 70, JMLR. org, pp 342–350

Bengio Y, Simard P, Frasconi P et al (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw 5(2):157–166

Bengio Y, Lamblin P, Popovici D, Larochelle H (2007) Greedy layer-wise training of deep networks. In: Advances in neural information processing systems, pp 153–160

Burnaev E, Erofeev P (2016) The influence of parameter initialization on the training time and accuracy of a nonlinear regression model. J Commun Technol Electron 61(6):646–660

Cachi PG, Ventura S, Cios KJ (2020) Fast convergence of competitive spiking neural networks with sample-based weight initialization. In: International conference on information processing and management of uncertainty in knowledge-based systems. Springer, pp 773–786

Caflisch RE et al (1998) Monte Carlo and quasi-Monte Carlo methods. Acta Numer 1998:1–49

Cao J, Lin Z, Huang GB (2012) Self-adaptive evolutionary extreme learning machine. Neural Process Lett 36(3):285–305

Cao W, Gao J, Ming Z, Cai S (2017a) Some tricks in parameter selection for extreme learning machine. In: IOP conference series: materials science and engineering. IOP Publishing, vol 261, p 012002

Cao W, Gao J, Ming Z, Cai S, Zheng H (2017b) Impact of probability distribution selection on RVFL performance. In: International conference on smart computing and communication. Springer, pp 114–124

Cao W, Wang X, Ming Z, Gao J (2018) A review on neural networks with random weights. Neurocomputing 275:278–287

Cao W, Patwary MJ, Yang P, Wang X, Ming Z (2019) An initial study on the relationship between meta features of dataset and the initialization of NNRW. In: 2019 international joint conference on neural networks (IJCNN). IEEE, pp 1–8

Cao W, Hu L, Gao J, Wang X, Ming Z (2020) A study on the relationship between the rank of input data and the performance of random weight neural network. Neural Comput Appl 32:1–12

Cetin BC, Burdick JW, Barhen J (1993) Global descent replaces gradient descent to avoid local minima problem in learning with artificial neural networks. In: IEEE international conference on neural networks. IEEE, pp 836–842

Chen CL, Nutter RS (1991) Improving the training speed of three-layer feedforward neural nets by optimal estimation of the initial weights. In: [Proceedings] 1991 IEEE international joint conference on neural networks. IEEE, pp 2063–2068

Cho JH, Lee DJ, Chun MG (2007) Parameter optimization of extreme learning machine using bacterial foraging algorithm. J Korean Inst Intell Syst 17(6):807–812

Dai AM, Le QV (2015) Semi-supervised sequence learning. In: Advances in neural information processing systems, pp 3079–3087

De Castro LN, Iyoda EM, Von Zuben FJ, Gudwin R (1998) Feedforward neural network initialization: an evolutionary approach. In: Proceedings 5th Brazilian symposium on neural networks (Cat. No. 98EX209). IEEE, pp 43–48
de Oliveira JFL, Ludermir TB (2012) An evolutionary extreme learning machine based on fuzzy fish swarms. In: Proceedings on the international conference on artificial intelligence (ICAI). The Steering Committee of The World Congress in Computer Science, Computer, p 1
Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
Demuth H, Beale M (2000) Neural network toolbox user's guide. The MathWorks, Inc, Portola Valley
Dolezel P, Skrabanek P, Gago L (2016) Weight initialization possibilities for feedforward neural network with linear saturated activation functions. IFAC-PapersOnLine 49(25):49–54
Drago GP, Ridella S (1992) Statistically controlled activation weight initialization (SCAWI). IEEE Trans Neural Netw 10(1109/72):143378
Duch W, Korczak J (1998) Optimization and global minimization methods suitable for neural networks. Neural Comput Surv 2:163–212
Duch W, Adamczak R, Jankowski N (1997) Initialization and optimization of multilayered perceptrons. In: Third conference on neural networks and their applications, pp 99–104
Emmett F, Joe R (2019) The effect of varying training on neural network weights and visualizations. J Emerg Investig 2(1)
Erhan D, Bengio Y, Courville A, Manzagol PA, Vincent P, Bengio S (2010) Why does unsupervised pre-training help deep learning? J Mach Learn Res 11:625–660
Eshtay M, Faris H, Obeid N (2018) Improving extreme learning machine by competitive swarm optimization and its application for medical diagnosis problems. Expert Syst Appl 104:134–152
Fernández-Redondo M, Hernández-Espinosa C (2001) Weight initialization methods for multilayer feedforward. In: ESANN, pp 119–124
Ferreira MF, Camacho R, Teixeira LF (2018) Autoencoders as weight initialization of deep classification networks applied to papillary thyroid carcinoma. In: 2018 IEEE international conference on bioinformatics and biomedicine (BIBM). IEEE, pp 629–632
Gabrielsson RB, Carlsson G (2018) A look at the topology of convolutional neural networks. arXiv:181003234
Gan Y, Liu J, Dong J, Zhong G (2015) A PCA-based convolutional network. arXiv:150503703
García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180(10):2044–2064
Ghazi MM, Nielsen M, Pai A, Modat M, Cardoso MJ, Ourselin S, Sørensen L (2019) On the initialization of long short-term memory networks. In: International conference on neural information processing. Springer, pp 275–286
Ghosh R, Verma B (2003) A hierarchical method for finding optimal architecture and weights using evolutionary least square based learning. Int J Neural Syst 13(01):13–24
Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp 249–256
Go J, Lee C (1999) Analyzing weight distribution of neural networks. In: IJCNN'99. International joint conference on neural networks. Proceedings (Cat. No. 99CH36339). IEEE, vol 2, pp 1154–1157
Hagan MT, Menhaj MB (1994) Training feedforward networks with the Marquardt algorithm. IEEE Trans Neural Netw 5(6):989–993
Halawa K (2014) A new multilayer perceptron initialisation method with selection of weights on the basis of the function variability. In: International conference on artificial intelligence and soft computing. Springer, pp 47–58
Han F, Yao HF, Ling QH (2011) An improved extreme learning machine based on particle swarm optimization. In: International conference on intelligent computing. Springer, pp 699–704
Hasegawa R, Hotta K (2016) Plsnet: a simple network using partial least squares regression for image classification. In: 2016 23rd international conference on pattern recognition (ICPR). IEEE, pp 1601–1606
He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034
Hochreiter S (1998) The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int J Uncertain Fuzziness Knowl-Based Syst 6(02):107–116
Hsiao TCR, Lin CW, Chiang HK (2003) Partial least-squares algorithm for weights initialization of back-propagation network. Neurocomputing 50:237–247

Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541). IEEE, vol 2, pp 985–990
Huang FJ, Boureau YL, LeCun Y, et al. (2007) Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: 2007 IEEE conference on computer vision and pattern recognition. IEEE, pp 1–8
Huang GB, Bai Z, Kasun LLC, Vong CM (2015) Local receptive fields based extreme learning machine. IEEE Comput Intell Mag 10(2):18–29
Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv:150203167
Javed K, Gouriveau R, Zerhouni N (2014) Sw-elm: a summation wavelet extreme learning machine algorithm with a priori parameter initialization. Neurocomputing 123:299–307
Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks. IEEE, vol 4, pp 1942–1948
Kim Y, Ra J (1991) Weight value initialization for improving training speed in the backpropagation network. In: [Proceedings] 1991 IEEE international joint conference on neural networks. IEEE, pp 2396–2401
Koturwar S, Merchant S (2017) Weight initialization of deep neural networks (DNNS) using data statistics. arXiv:171010570
Krähenbühl P, Doersch C, Donahue J, Darrell T (2015) Data-dependent initializations of convolutional neural networks. arXiv:151106856
Larochelle H, Bengio Y, Louradour J, Lamblin P (2009) Exploring strategies for training deep neural networks. J Mach Learn Res 10:1–40
LeCun YA, Bottou L, Orr GB, Müller KR (2012) Efficient backprop. In: Neural networks: tricks of the trade. Springer, pp 9–48
Lehtokangas M, Saarinen J (1998) Weight initialization with reference patterns. Neurocomputing 20(1–3):265–278
Leung FHF, Lam HK, Ling SH, Tam PKS (2003) Tuning of the structure and parameters of a neural network using an improved genetic algorithm. IEEE Trans Neural Netw 14(1):79–88
Li G, Alnuweiri H, Wu Y, Li H (1993) Acceleration of back propagation through initial weight pretraining with delta rule. In: IEEE international conference on neural networks. IEEE, pp 580–585
Li J, Cheng Jh, Shi Jy, Huang F (2012) Brief introduction of back propagation (BP) neural network algorithm and its improvement. In: Advances in computer science and information engineering. Springer, pp 553–558
Li S, Zhao Z, Liu T, Hu R, Du X (2017) Initializing convolutional filters with semantic features for text classification. In: Proceedings of the 2017 conference on empirical methods in natural language processing, pp 1884–1889
Masci J, Meier U, Cireşan D, Schmidhuber J (2011) Stacked convolutional auto-encoders for hierarchical feature extraction. In: International conference on artificial neural networks. Springer, pp 52–59
Masden M, Sinha D (2020) Linear discriminant initialization for feed-forward neural networks. arXiv:200712782
Masood S, Doja M, Chandra P (2015) Analysis of weight initialization methods for gradient descent with momentum. In: 2015 International conference on soft computing techniques and implementations (ICSCTI). IEEE, pp 131–136
Matias T, Araújo R, Antunes CH, Gabriel D (2013) Genetically optimized extreme learning machine. In: 2013 IEEE 18th conference on emerging technologies and factory automation (ETFA). IEEE, pp 1–8
Mishkin D, Matas J (2015) All you need is a good init. arXiv:151106422
Mittal A, Singh AP, Chandra P (2020) A modification to the Nguyen-Widrow weight initialization method. In: Intelligent systems, technologies and applications. Springer, pp 141–153
Mohapatra P, Chakravarty S, Dash PK (2015) An improved cuckoo search based extreme learning machine for medical data classification. Swarm Evol Comput 24:25–49
Murru N, Rossini R (2016) A Bayesian approach for initialization of weights in backpropagation neural net with application to character recognition. Neurocomputing 193:92–105
Nguyen D, Widrow B (1990) Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In: 1990 IJCNN international joint conference on neural networks. IEEE, pp 21–26
Nguyen G, Dlugolinsky S, Bobák M, Tran V, García ÁL, Heredia I, Malík P, Hluchỳ L (2019) Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey. Artif Intell Rev 52(1):77–124

Nwankpa C, Ijomah W, Gachagan A, Marshall S (2018) Activation functions: Comparison of trends in practice and research for deep learning. arXiv:181103378
Pacifico LD, Ludermir TB (2013) Evolutionary extreme learning machine based on particle swarm optimization and clustering strategies. In: The 2013 international joint conference on neural networks (IJCNN). IEEE, pp 1–6
Paine TL, Khorrami P, Han W, Huang TS (2014) An analysis of unsupervised pre-training in light of recent advances. arXiv:14126597
Pang S, Yang X (2016) Deep convolutional extreme learning machine and its application in handwritten digit classification. Comput Intell Neurosci 2016:3049632
Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst Mag 22(3):52–67
Pavelka A, Procházka A (2004) Algorithms for initialization of neural network weights. In: In Proceedings of the 12th annual conference, MATLAB, pp 453–459
Qiao J, Li S, Li W (2016) Mutual information based weight initialization method for sigmoidal feedforward neural networks. Neurocomputing 207:676–683
Ramos EZ, Nakakuni M, Yfantis E (2017) Quantitative measures to evaluate neural network weight initialization strategies. In: 2017 IEEE 7th annual computing and communication workshop and conference (CCWC). IEEE, pp 1–7
Rodriguez FJ, Garcia-Martinez C, Lozano M (2012) Hybrid metaheuristics based on evolutionary algorithms and simulated annealing: taxonomy, comparison, and synergy test. IEEE Trans Evol Comput 16(6):787–800
Ruiz-Garcia A, Elshaw M, Altahhan A, Palade V (2017) Stacked deep convolutional auto-encoders for emotion recognition from facial expressions. In: 2017 international joint conference on neural networks (IJCNN). IEEE, pp 1586–1593
Rumelhart DE, Hinton GE, Williams RJ (1985) Learning internal representations by error propagation. California Univ San Diego La Jolla Inst for Cognitive Science. Technical report
Saxe AM, McClelland JL, Ganguli S (2013) Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. arXiv:13126120
Schneider J (2020) Correlated initialization for correlated data. arXiv:200304422
Shimodaira H (1994) A weight value initialization method for improving learning performance of the backpropagation algorithm in neural networks. In: Proceedings sixth international conference on tools with artificial intelligence. TAI 94, IEEE, pp 672–675
Sodhi SS, Chandra P, Tanwar S (2014) A new weight initialization method for sigmoidal feedforward artificial neural networks. In: 2014 international joint conference on neural networks (IJCNN). IEEE, pp 291–298
Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
Subudhi B, Jena D (2008) Differential evolution and levenberg marquardt trained neural network scheme for nonlinear system identification. Neural Process Lett 27(3):285–296
Sudowe P, Leibe B (2016) Patchit: self-supervised network weight initialization for fine-grained recognition. BMVC 1:24–25
Tan S, Li B (2014) Stacked convolutional auto-encoders for steganalysis of digital images. In: Signal and information processing association annual summit and conference (APSIPA), 2014 Asia-Pacific. IEEE, pp 1–4
Tang J, Wang D, Zhang Z, He L, Xin J, Xu Y (2017) Weed identification based on k-means feature learning combined with convolutional neural network. Comput Electron Agric 135:63–70
Tao X, Zhou X, He YL, Ashfaq RAR (2016) Impact of variances of random weights and biases on extreme learning machine. JSW 11(5):440–454
Tapson J, De Chazal P, van Schaik A (2015) Explicit computation of input weights in extreme learning machines. In: Proceedings of ELM-2014 vol 1. Springer, pp 41–49
Thimm G, Fiesler E (1997) High-order and multilayer perceptron initialization. IEEE Trans Neural Netw 8(2):349–359
Trinh TH, Luong MT, Le QV (2019) Selfie: self-supervised pretraining for image embedding. arXiv:190602940
Tu S, Huang Y, Liu G et al (2017) Csfl: a novel unsupervised convolution neural network approach for visual pattern classification. AI Commun 30(5):311–324
Vorontsov E, Trabelsi C, Kadoury S, Pal C (2017) On orthogonality and learning recurrent networks with long term dependencies. In: Proceedings of the 34th international conference on machine learning-volume 70, JMLR. org, pp 3570–3578

Wang X, Cao W (2018) Non-iterative approaches in training feed-forward neural networks and their applications
Wang W, Liu X (2017) The selection of input weights of extreme learning machine: a sample structure preserving point of view. Neurocomputing 261:28–36
Wessels LF, Barnard E (1992) Avoiding false local minima by proper initialization of connections. IEEE Trans Neural Netw 3(6):899–905
Wessels L, Barnard E, Van Rooyen E (1990) The physical correlates of local minima. In: International neural network conference
Wiehman S, Kroon S, De Villiers H (2016) Unsupervised pre-training for fully convolutional neural networks. In: 2016 Pattern recognition association of south africa and robotics and mechatronics international conference (PRASA-RobMech). IEEE, pp 1–6
Xu Y, Shu Y (2006) Evolutionary extreme learning machine–based on particle swarm optimization. In: International symposium on neural networks. Springer, pp 644–652
Yam JYF, Chow TWS (2000) A weight initialization method for improving training speed in feedforward neural network. Neurocomputing. https://doi.org/10.1016/S0925-2312(99)00127-7
Yam JY, Chow TW (2001) Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients. IEEE Trans Neural Netw 12(2):430–434
Yam YF, Leung CT, Tam PK, Siu WC (2002) An independent component analysis based weight initialization method for multilayer perceptrons. Neurocomputing 48(1–4):807–818
Yang XS (2010) Nature-inspired metaheuristic algorithms. Luniver Press, Bristol
Yang Z, Wen X, Wang Z (2015) Qpso-elm: An evolutionary extreme learning machine based on quantum-behaved particle swarm optimization. In: 2015 seventh international conference on advanced computational intelligence (ICACI). IEEE, pp 69–72
Yang H, Ding X, Chan R, Hu H, Peng Y, Zeng T (2020) A new initialization method based on normed statistical spaces in deep networks. Inverse Probl Imaging 15:147
Yoon HS, Bae CS, Min BW (1995) Neural networks using modified initial connection strengths by the importance of feature elements. In: 1995 IEEE international conference on systems, man and cybernetics. Intelligent Systems for the 21st century. IEEE, vol 1, pp 458–461
Zhang J, Sanderson AC (2009) Jade: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958
Zhang Q, Zhang L (2018) Convolutional adaptive denoising autoencoders for hierarchical feature extraction. Front Comput Sci 12(6):1140–1148
Zhang Y, Cai Z, Wu J, Wang X, Liu X (2015) A memetic algorithm based extreme learning machine for classification. In: 2015 international joint conference on neural networks (IJCNN). IEEE, pp 1–8
Zhang X, Lin X, Ashfaq RAR (2018) Impact of different random initializations on generalization performance of extreme learning machine. JCP 13(7):805–822
Zhang H, Dauphin YN, Ma T (2019) Fixup initialization: residual learning without normalization. arXiv:190109321
Zhu QY, Qin AK, Suganthan PN, Huang GB (2005) Evolutionary extreme learning machine. Pattern Recognit 38(10):1759–1763