Review

Survey of Optimization Algorithms in Modern Neural Networks

Ruslan Abdulkadirov *, Pavel Lyakhov, Nikolay Nagornov

Posted Date: 20 April 2023

doi: 10.20944/preprints202304.0648.v1

Keywords: optimization methods; physics-informed neural networks; spiking neural networks; quantum neural networks; graph neural networks; information geometry; quasi-Newton methods; approximation; quantum computations; gradient free optimization; fractional order optimization; bilevel optimization

Article

Survey of Optimization Algorithms in Modern Neural Networks

Ruslan Abdulkadirov [1,*], Pavel Lyakhov [1,2] and Nagornov Nikolay [2]

[1] North-Caucasus Center for Mathematical Research, North-Caucasus Federal University, 355009 Stavropol, Russia; ljahov@mail.ru
[2] Department of Mathematical Modeling, North-Caucasus Federal University, 355009 Stavropol, Russia; sparta1392@mail.ru
* Correspondence: ruslanabdulkadirovstavropol@gmail.com

Abstract: Creating self-learning algorithms, developing deep neural networks and improving other methods that "learn" for various areas of human activity is the main goal of the theory of machine learning. It helps to replace the human with a machine, aiming to increase the quality of production. The theory of artificial neural networks, which already have replaced the humans in problems of detection of moving objects, recognition of images or sounds, time series prediction, big data analysis and numerical methods remains the most dispersed branch of the theory of machine learning. Certainly, for each area of human activity it is necessary to select appropriate neural network architectures, methods of data processing and some novel tools from applied mathematics. But the universal problem for all these neural networks with specific data is the achieving the highest accuracy in short time. Such problem can be resolved by increasing sizes of architectures and improving data preprocessing, where the accuracy rises with the training time. But there is a possibility to increase the accuracy without time growing, applying optimization methods. In this survey we demonstrate existing optimization algorithms of all types, which can be used in neural networks. There are presented modifications of basic optimization algorithms, such as stochastic gradient descent, adaptive moment estimation, Newton and quasi-Newton optimization methods. But the most recent optimization algorithms are related to information geometry, for Fisher-Rao and Bregman metrics. This approach in optimization extended the theory of classic neural networks to quantum and complex-valued neural networks, due to geometric and probabilistic tools. There are provided applications of all introduced optimization algorithms, what delighted many kinds of neural networks, which can be improved by including any advanced approaches in minimization of the loss function. Afterwards, we demonstrated ways of developing optimization algorithms in further researches, engaging neural networks with progressive architectures. Classical gradient based optimizers can be replaced by fractional order, bilevel and, even, gradient free optimization methods. There is a possibility to add such analogues in graph, spiking, complex-valued, quantum and wavelet neural networks. Besides the usual problems of image recognition, time series prediction, object detection, there are many are other tasks for modern theory of machine learning, such as solving problem of quantum computations, partial differential and integro-differential equations, stochastic processes and Brownian motion, making decisions and computer algebra.

Keywords: optimization methods; physics-informed neural networks; spiking neural networks; quantum neural networks; graph neural networks; information geometry; quasi-Newton methods; approximation; quantum computations; gradient free optimization; fractional order optimization; bilevel optimization

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Notations

θ	weight
α	learning rate
$f(\theta)$	loss function
g_t	gradient $\nabla f(\theta_t)$
λ	weight decay parameter, L^2 regularization factor
μ	momentum
G_t	sum of gradients $G_{t+1} = G_t + \nabla f(\theta_{t-1})$
m_t	exponential moving average of g_t
v_t	horizontal direction converging, exponential moving average of g_t^2
$E[g^2]_t$	running average $E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma)g_t^2$, where ρ is a decay rate parameter $\in (0,1)$.
η_t	schedule multiplier
ν	immediate discount factor
β	momentum buffer's discount factor
$\beta_0,\ \beta_1,\ \beta_2$	moments
ρ_t	variance $\rho_t = \rho_\infty - \frac{2t\beta_2^t}{1-\beta_2^t}$
r_t	variance rectification
ξ	DiffGrad friction coefficient (DFC)
$Hess(\theta_t)$	Hessian matrix $Hess(\theta_t) = \nabla^2 f(\theta_t)$
H_t	inverse BFGS Hessian approximation
(s_t, y_t)	curvature pairs
D_t	Hessian diagonal matrix
$\overline{D}_t$	Hessian diagonal matrix with momentum
$(\mathcal{M}^n, g)$	Riemannian manifold with n-dimensional topological space $\mathcal{M}^n$ and metric g
∇	affine connection, gradient
$T\mathcal{M}^n$	tangent bundle
$\Phi(\cdot,\cdot)$	proximity function
$B(\cdot,\cdot)$	Bregman

1. Introduction

The question of rising the accuracy of neural networks remains actual. There are many approaches applied for solving this problem: data augmentation [1], improving the mathematical model of neurons [2], adding complement neural network [3] and so on. Indeed, all this approaches resolve the problem of increasing the accuracy of neural networks particularly. But there exist universal methods for improving the quality of various types of networks, and one of them is the optimization of the loss function. The problem of minimization of such functions, which contain local extremes, is a descent to the neighborhood of the global minimum, which remains important in machine learning. First attempts to achieve the minimal value were realized by using stochastic gradient descent (SGD) [4]. Such approach is still widely used in modern neural networks and belongs to set of first order optimization methods. Later, there was stochastic gradient descent with momentum and Nesterov condition (SGDM Nesterov) [5], which increased the rate of convergence and gave the high accuracy for less number of iterations.

According to the structure of SGD, there were proposed AdaGrad [6], RMSProp [7], Adadelta [8] and Adam [9], which allow to attain the minimum of functions that contain little number of local extremes. But in case of Rastrigin, Rosenbrouck and Ackley functions from [10] these approaches are not able to reach the global minimum. It is explained by achieving the minimum taking into account only directions of gradients. Even presence of exponential moving averages in Adam do not suffice for minimization such test functions. The same technique is used in modifications of Adam and SGD. Adam-type algorithms, which updates the exponential moving averages of the gradient, were build to improve the process of minimization for multi-extreme functions. The advantage of these approaches is that the initialization of biases can be easily counteracted, resulting in bias-corrected estimates. However, they are able to minimize the loss function, but not always in global minimum.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Using Adam-type algorithms, it is possible to receive required accuracy without big time consumption. Therefore, for attaining higher accuracy one needs to apply second order optimization algorithms.

The main goal of second order optimization algorithms [11] is the achieving global minimum for short number of iterations, because, usually, they are slower compared with first order optimization algorithms. It can be explained by the fact, that second order algorithms take into account not only directions of gradients, but the convexity (curvature) of objective functions by Hessian matrix. Computations of inverted Hessian matrix make the second optimization more complex. This approach is called Newton optimization method [12]. In machine learning, where number of neurons can attain over one hundred, Newton optimization is an ineffective tool, but the approximation of inverted Hessian matrix makes possible to minimize the loss function for required time consumption. Such technique is called quasi-Newton optimization method [13]. Quasi-Newton optimization algorithms analyze the loss function from functional point of view, which allows to avoid local minimums and rapidly converge in global minimum. For increasing the quality of minimization, it is possible to observe the objective function from geometric and probabilistic points of view.

First attempt to geometrically minimize smooth functions was applied in [14], where authors engaged the properties of Rimannian geometry. Such optimization used the gradient flow for receiving the global minimum. Later, there was an idea to engage information geometry in [15], which is an intersection of Riemannian geometry and probability theory. Such approach is divided in two branches, which utilize Fisher-Rao [16] and Bregman [17] metrics, respectively. The optimization with Fisher-Rao metrics (Fisher information matrix) is called a natural gradient descent. In case of utilizing Bregman metrics, one receives the mirror descent. These optimization algorithms proved their abilities in image recognintion [18], time series prediction [19]. The main application they find in physics-informed neural networks [20], which are devoted to solve partial differential and integro-differential equations.

In this paper the main goal is the classification of optimization algorithms, which are used in machine learning, identification their properties, which allow to increase the rate of convergence for certain neural network, and providing ways of developing minimization methods in further researches. After gradient-based optimization methods, we will demonstrate alternative approaches in minimization of loss function: gradient-free and bilevel optimization.

All provided above optimization algorithms are presented in the Figure 1, which constructs the contests of this paper.

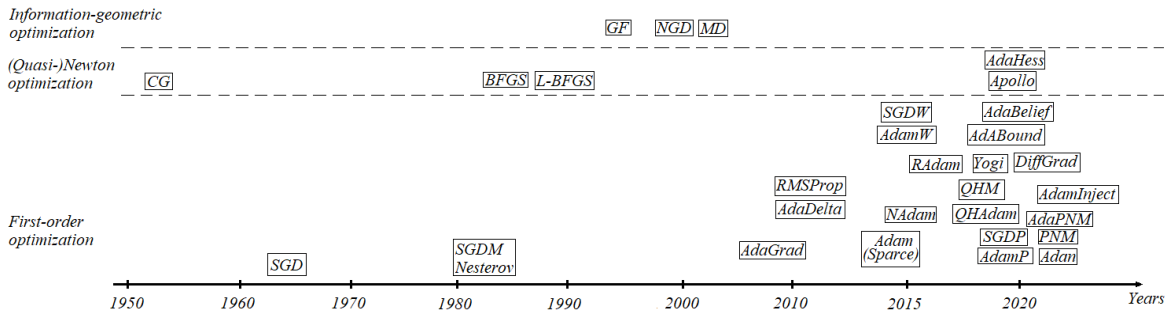

Figure 1. The historical tree of developing optimization algorithms from 1950 to 2022.

The remainder of the paper is organized as follows. Section 2 is devoted to first order optimization algorithms, which are described in 3 subsections: SGD-type, Adam-type and PNM-type optimization algorithms. Section 3 presents second order optimization techniques, such as Newton and Quasi-newton algorithms. Section 4 consists of representation of information-geometric methods, which are divided on Fisher-Rao and Bregman metrics. Such Part of this paper contains the most progressive approaches in modern theory of optimization methods. Section 5 represents applications of all optimization algorithms, introduced in Sections 2–4. In Section 6, we report the conclusions of the provided optimization algorithms and suggestions for their improving and further exploitation in neural networks.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

2. First Order Optimization Algorithms

2.1. SGD-Type Algorithms

How can we see from Figure 1, the earliest first order optimization algorithm is SGD [4], which can be described by the iterative formula

$$\theta_{t+1} = \theta_t - \alpha_t \nabla f(\theta_t), \tag{1}$$

where θ_t denotes weight, $f(\theta_t)$ is a loss function with its gradient $\nabla f(\theta_t)$ and α_t is a learning rate. Later such approach was modified to SGDM with Nestreov condition [5] presented by

$$\theta_{t+1} = \theta_t - \alpha_t(\nabla f(\theta_t) + \mu b_{t+1}), \tag{2}$$

where θ_0 is an initial point, $b^{(k+1)} = \mu b^{(k)} + (1-\tau)(\nabla f(\theta^{(k)}) + \lambda\theta^{(k)})$, where τ is a damping parameter and μ is a momentum. The algorithm of SGDM with Nesterov condition is constructed in [22]. This optimization method is still used in approximation theory and machine learning. Various back propagation methods ([23–26]) are based on calculating of local partial derivatives, which rectify the value of weights of neural networks using (2). But such approach can be modified to others more effective versions, which converge to minimum faster. Therefore, because of relatively small rate of convergence of SGDM with Nesterov condition and its step-size updating, There was introduced Adaptive gradient method (AdaGrad) in [6].

AdaGrad differs from SGDM with Nesterov condition by the adaptive step-size. This advantage allows to increase learning rate and, simultaneously, reduce time consumption. AdaGrad is described by the following formula

$$\theta_{t+1} = \theta_t - \frac{\alpha_t}{\sqrt{G_{t+1} + \epsilon}} \nabla f(\theta_t),$$

where the sum of gradients is presented as

$$G_{t+1} = G_t + \nabla f(\theta_{t-1}).$$

As was noted above, AdaGrad updates the stepsize α_t on the given information of all previous gradients observed along the way. But its disadvantage is similar, like in SGD, where minimization is based on directions of the gradients and step-size regulation, which does not guarantee the convergence in the neighborhood of global minimum. Afterwards, Adagrad was equipped with the gradient norm information, which improves the convergence rate. Such modification is called AdaGrad-Norm [6], presented as follows:

$$\theta_{t+1} = \theta_t - \frac{\alpha}{1 + (t-1)\eta} \frac{g_t}{\sqrt{G_t} + \epsilon}. \tag{3}$$

AdaGrad and AdaGrad-Norm adapt the step-size more accurately, compared with SGD and SGDM with Nesterov condition, what allows to increase the learning rate for growing the probability of reaching the global minimum. But accumulation of previous gradients does not resolve the problem of convergence in the neighborhood of global minimum. This approach, like SGD, descends towards negative directions of gradients. Such disadvantage caused researchers to come up with idea of adapting of the step-size using mean moments, what implied the root mean square propagation (RMSProp) optimization algorithm.

The RMSprop optimizer [7] partially uses the same technique as SGDM and limits the oscillations in the upright direction, what allows to increase the learning rate. It makes possible to take fast substantial step-sizes in the horizontal direction converging v_t. Such approach is described as

$$v_t = \gamma v_{t-1} + (1-\gamma) g_t^2,$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

$$\theta_{t+1} = \theta_t - \alpha_t \frac{g_t}{\sqrt{v_t} + \epsilon}, \tag{4}$$

where $\gamma \in (0, 1)$ is a moment. RMSProp is still actual algorithm in many neural networks, like SGDM with Nesterov condition, because it contains prerequisites of exponential moving averages, which later will have a lot of modifications, that give an opportunity to avoid some local minimums. Moreover, according to techniques in AdaGrad and RMSProp, there was introduce AdaDelta [8].

AdaDelta is a modification, based on techniques in SGD, RMSProp and AdaGrad, that separate dynamic learning rate per-dimension, requires no manual setting of a learning rate and takes minimal computation over gradient descent. Additionaly, it is insensitive to hyperparemeters and robust to blow-up gradient, noise and architecture choice. Unlike Adagrad such method reduces aggressive, monotonically decreasing learning rate. Adadelta restricts the window of accumulated past gradients to a fixed size, instead of accumulating all past squared gradients. The running average $E[g^2]_t$ depends only on the previous average and current gradient. Initial accumulation variables $E[g^2]_0$, $E[\Delta\theta^2]_0$ are equal to 0. AdaDelta algorithms is described as follows. Accumulate gradient:

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1 - \rho) g_t^2,$$

where ρ is a decay rate paremeter. Compute update:

$$\Delta\theta_t = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} g_t,$$

where

$$RMS[g]_t = \sqrt{E[g^2]_t + \epsilon}.$$

Accumulate updates:

$$E[\Delta\theta^2]_t = \rho E[\Delta\theta^2]_{t-1} + (1 - \rho)\Delta\theta_t^2,$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t. \tag{5}$$

The stochastic gradient descent was modified in other variations, which, like AdaGrad, AdaDelta and RMSProp, increase the test accuracy and accelerate the implementation of algorithm. Such modifications can improve the process of recognition, prediction, generation and making decision, what develops the theory of neural networks. But in deep convolutional neural networks modification (2)-(5) does not achieve the higher accuracy compared with SGDM with Nesterov condition, how can be seen in [27], where only some modified algorithms presented in next subsection give higher test accuracy of recognition. Therefore, one needs algorithms, which take into account gradient directions and values weight. First very important modification of SGDM is Nesterov Accelerated Gradient (NAG), presented in [28].

Nesterov's accelerated gradient descent is widely used in practice for training deep networks and other supervised learning models, because it often provides significant improvements over SGDM with Nesterov condition. Rigorously speaking, "fast gradient" methods have provable ammendments over gradient descent only for the deterministic case, where the gradients are exact. In the stochastic case, "fast gradients" partially mimic their exact gradient counterparts, resulting in some practical gain. Such approach is described as

$$v_t = \mu v_{t-1} + \alpha f(\theta - \mu v_{t-1}),$$

$$\theta_t = \theta_{t-1} + v_t. \tag{6}$$

As said before, there is an efficient technique of utilizing Lebesgue norm on weight of neurons for improving resulting test accuracy. Such approach is called L^2 regularization. It is well described and tested on SGDM with Nesterov condition in [29].

Proposed L^2 regularization in [30] acts similarly with the usual weight decay, which is used in SGD. Indeed, both approaches evaluate weights closer to zero with the same rate. The L^2 regularized

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

SGD (SGDW) is a modification of usual SGD, which differs by recovering original weight decay using decoupling weight decay from the optimization steps with respect to the loss function. However, for adaptive gradient algorithms the main difference is a presence of adapted sums of gradients of the loss function and the regularizer, whereas with decoupled weight decay, only gradients of the loss function are adapted. With L^2 regularization both types of gradients are normalized by their typical summed magnitudes, and therefore weights with large typical gradient magnitude are regularized by a smaller relative amount than other weights. The introduced SGDW method is described by the following iterative formula

$$g_t = \nabla f(\theta_t) + \lambda\theta_{t-1},$$

where λ is a weight decay parameter (L_2 regularization factor),

$$m_t = \beta_1 m_{t-1} + \eta_t \alpha g_t,$$

$$\theta_t = \theta_{t-1} - m_t - \eta_t \lambda \theta_{t-1}, \tag{7}$$

where η_t is a schedule multiplier. Unfortunately, such SGDW does not significantly improve usual SGD, especially in deep neural networks, auto-encoders and graph-neural networks. Because the architecture of these neural networks confuses the L^2 regularization and make the process of minimization too difficult for SGDW. But there are two techniques, which can improve the quality of minimization of the loss function using projection [31] and hyper-parameter methods [32].

Before introducing the projection technique for SGD, it is necessary to recall a batch normalization [33]. In practise, normalization techniques, such as batch normalization, play an important role in for modern deep learning. They allow weights converge more rapid with better generalization performances. The normalization-induced scale invariance among the weights gives advantages to SGD, such as the effective step-size automatic regularization and stabilizing the training procedure. In practise, one can notice that the including momentum in SGD-type optimizers reduces much step-sizes for scale-invariant weights. Such phenomenon is not yet studied and causes unwanted side effects in the process of minimization. This is a crucial issue because the vast majority of modern deep neural networks consist of SGD- and Adam-type optimizer, which contain momentum, and scale-invariant parameters. Therefore, there was proposed SGD with projection (SGDP), which removes the radial component, or the norm-increasing direction, at each optimizer step. Because of the scale invariance, this modification only alters the effective step-sizes without changing the effective update directions, thus satisfying the original convergence properties of GD optimizers.

SGDP can be presented as following iterative formulas:

$$\begin{aligned} p_t &= \mu p_{t-1} + \nabla f(\theta_{t-1}), \\ \theta_t &= \theta_{t-1} - \alpha p_t. \end{aligned} \tag{8}$$

This approach, compared with SGDW, increases the rate of convergence, but it is not enough to significantly advance the process of minimization of the loss function in global minimum. what can be seen in experiments of minimizing Rastrigin function in (https://github.com /jettify/pytorch-optimizer). But the problem of avoiding the local minimums was solved by introducing the hyper-parameter methods, which is called quasi-hyperbolic momentum algorithm (QHM).

Momentum-based acceleration of SGD is widely used in deep learning. There is presented QHM as an extremely simple alteration of momentum SGD, averaging a plain SGD step with a momentum step, what presented in [34]. This approach introduces the immediate discount factor ν, encapsulating plain SGD ($\nu = 0$) and momentum ($\nu = 1$). A self-evident interpretation of QHM is a ν-weighted average of the momentum update step and the plain SGD update step. The expressive power of QHM intuitively comes from decoupling the momentum buffer's discount factor β from the current

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

gradient's contribution to the update rule $1 - \nu\beta$. In contrast, momentum tightly couples the discount factor β and the current gradient's contribution $1 - \beta$.

Let us present QHM as the following iterative process:

$$g_t = \beta g_{t-1} + (1-\beta)\hat{g}_t,$$

$$\theta_t = \theta_{t-1} - \alpha\left[(1-\nu)\cdot\hat{g}_t + \nu g_t\right]. \tag{9}$$

Such method lets surmount the local minimums. This approach has no disadvantages, compared with demonstrated above algorithms. But, like others, it QHM does not suffice for achieving the highest accuracy in deep neural networks, because it still takes into account the direction of gradients and amendments with parameters and gradient normalization.

Before drawing conclusions, it should be noted that for any smooth objective function $f : \mathbb{R}^n \to \mathbb{R}$ provided optimization methods in above have almost the same regret bound [35,36]

$$R(T) = \sum_{t=1}^{T} f_t(x_t) - \min_{x\in\mathbb{R}^n} \sum_{t=1}^{T} f_t(x), \tag{10}$$

which measures the convergence rate and belongs to set of $O(\sqrt{T})$. Presented above algorithms can be improved by performing their regret bounds to $O(\log T)$, which was done in [37] and. Such algorithms are called strongly convex.

SGD-type algorithms are used in convolutional, recurrent, autoencoder, graph neural networks. But their huge disadvantage is insufficient information about the properties of loss function to reach the global minimum. SGD is led by the direction of gradient, which is not enough for increasing the test accuracy. In modern neural networks more preferred approaches are Adam and its modifications (Adam-type algorithms), which contain exponential moving averages of gradient and squared gradient, that significantly improves the quality of training neural networks. These methods improve the optimization by estimation of moments, which gives more information of about the global minimum. Moreover, such improvement allows to increase the accuracy of image recognition, time series prediction and object classification.

2.2. Adam-Type Algorithms

The adaptive moment estimate (Adam) [39] algorithm is a continuation of developing gradient based optimization, which significantly increases the accuracy in neural networks. Taking into account directions of gradients and means of moments, minimization of loss function with higher frequency converges in the neighborhood of global minimum. The iterative formula is presented as

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t,$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2,$$

$$\hat{m}_t = m_t/(1-\beta_1^t),\ \hat{v}_t = v_t/(1-\beta_2^t),$$

$$\theta_t = \theta_{t-1} - \alpha\hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon). \tag{11}$$

Note that β_1, β_2 are called moments, m_t and v_t are exponential moving averages of g_t and g_t^2, respectively.

According to the adaptive moment estimate algorithm (11), there were provided modified versions, which distinct by step-size adaptation and manipulation with exponential moving averages. Like SGDW, there was provided a modification in [40], which is called Adam algorithm with L^2 regularization. In the case of usual Adam, weights, that tend to have large gradients of f, are not regularized as much as they would with decoupled weight decay, since the gradient of the regularizer

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

is scaled. AdamW with the same exponential moving averages can be described as the following iteration formula:

$$\theta_t = \theta_{t-1} - \eta + t \left[\alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon) + \lambda \theta_t \right], \tag{12}$$

where η_t is a schedule multipliers and λ is a L^2 regularization factor. There is proved fact, that such modification with decoupled weight decay yields substantially better generalization performance than the common implementation of Adam with L^2 regularization, in [41]. Another approach for modification of Adam algorithms is projected Adam, which is called AdamP [42].

AdamP, like SGDP in (8), is based on projection of the sum of gradient and momentum vectors onto the tangent space of weights. Such approach allows to accelerate effective step sizes for scale-invariant weights. This technique with its applications is described in [43] and has the following representation:

$$g_t = \nabla f(\theta_{t-1}) + \lambda \theta_{t-1},$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t,$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2,$$

$$p_t = \frac{m_t}{\sqrt{v_t} + \epsilon}.$$

Then, according to cosine condition we obtain

$$if \ \cos(g_i) < \delta / \sqrt{dim(\theta)} : q_i = p_i - (\frac{\theta}{\|\theta\|_2} \cdot p_i) \frac{\theta}{\|\theta\|_2}$$

$$else: \ q_i = p_i$$

Afterwards, one obtain

$$\theta_t = \theta_{t-1} - \eta_t \left(\frac{\alpha m_t}{\sqrt{v_t} + \epsilon} + \lambda \theta_{t-1} \right). \tag{13}$$

According to notes in [44], it implies that momentum-based optimizers induce an excessive growth of the scale-invariant weight norms, which prematurely decay the effective optimization steps, leading to sub-optimal performances. The resulting AdamP, like SGDP in Section 2.1, successfully suppresses the weight norm growth and train a model at an unobstructed speed. Another approach to accelerate the convergence rate is the applying the quasi-hyperbolic momentum, like it was made for QHM in (9).

As for QHM, there was proposed QHAdam [45], which replaces both of Adam's moment estimators with quasi-hyperbolic terms. This approach is described as

$$g_{t+1} = \beta_1 g_t + (1 - \beta_1) \nabla f(\theta_t), \ g'_{t+1} = (1 - \beta_1^{t+1})^{-1} g_{t+1},$$

$$s_{t+1} = \beta_2 s_t + (1 - \beta_2) (\nabla f(\theta_t))^2, \ s'_{t+1} = (1 - \beta_2^{t+1})^{-1} s_{t+1},$$

$$\theta_{t+1} = \theta_t - \alpha \left[\frac{(1 - \nu_1) \nabla f(\theta_t) + \nu_1 g'_{t+1}}{\sqrt{(1 - \nu_2 (\nabla f(\theta_t))^2) + \nu_2 s'_{t+1}} + \epsilon} \right] \tag{14}$$

As was noted in previous subsection, there exist Nesterov trick NAG, which accelerates usual SGD in [46]. The same modification propagates to Adam and transform it to NAdam [47]. This technique with higher frecuency converge to the neighborhood of global minimum for less number of iteration, compared with Adam and its previous modifications, and constructed more simple than AdamW, AdamP and QHAdam. But for cases of Rastrigin functions and Rosenbrock functions, (11)-(13) do not converge in neighborhood of global minimum. As was said in Section 2.1, L^2 regularization, projection and quasi-hyperbolic parameters techniques influence to step-size, taking into account gradient directions. In cases, when included moment estimation, such modifications can accelerate the process of convergence, but not necessary in the neighborhood of global minimum. But there

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

are two techniques, which make the process of minimization 'smoother'. Such approaches are called Nesterov-accelerated (NAdam) and Rectified (RAdam) adaptive moment estimation.

Let us present the iterative formula of NAdam:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\left(\beta_1 \hat{m}_t + \frac{(1-\beta_1)g_t}{1-\beta_1^t}\right) \tag{15}$$

This method is a continuation of NAG due to the added exponential moving averages. Compared with Adam and its previous modifications, NAdam increases the accuracy of converging in deep convolutional neural networks and, simultaneously, make the minimization process faster by 'smoothing' the descent. But such technique is ineffective in physics-informed neural networks, because of the smoothing descent trajectory, which for partial differential equation solution gives more deviations. Such disadvantage was resolved by rectified Adam (RAdam).

Proposed in [48], RAdam differs from other optimization method by introducing a term to rectify the variance of the adaptive language modeling and learning rate. This modification proved its ability to receive higher test accuracy. Such optimization method has the following iterative formula:

$$\rho_t = \rho_\infty - 2t\beta_2^t/(1-\beta_2^t)$$

$$\rho_\infty = \frac{2}{1-\beta_2} - 1$$

If the variance is tractable ($\rho_t > 4$) then the adaptive learning rate is computed as:

$$l_t = \sqrt{(1-\beta_2^t)/v_t},$$

the variance rectification term is calculated as:

$$r_t = \sqrt{\frac{(\rho_t - 4)(\rho_t - 2)\rho_\infty}{(\rho_\infty - 4)(\rho_\infty - 2)\rho_t}}$$

and we update parameters with adaptive momentum:

$$\theta_t = \theta_{t-1} - \alpha_t r_t \hat{m}_t l_t. \tag{16}$$

If the variance is not tractable we update instead with:

$$\theta_t = \theta_{t-1} - \alpha_t \hat{m}_t. \tag{17}$$

Such method overtake NAdam and other algorithms, especilly in deep neural networks [49], such as AlexNet [50], ResNet [51], InceptionNet [52], GoogleNet [53] and Res-Next [54]. But RAdam adapts the learning rate too complex and, like previous analogues, can not converge in the neighborhood of global minimum of Rastrigin function. Moreover, there exist other simple ways to adjust the learning rate and make the process of minimization faster. One of them is difference gradient approach, which is called DiffGrad [55].

The difference gradient approach is based on moment estimate technique and, instead of complex manipulations with learning rate and weights, calculate only additional coefficient, which is called DiffGrad friction coefficient (DFC). The main distinction of DiffGrad is that such approach is based on the change in short-term gradients and controls the learning rate based on the need of dynamic adjustment of learning rate. This means that diffGrad follows the norm that the parameter update should be smaller in low gradient changing regions. The friction coefficient diffGrad (DFC) is designed

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

to control the learning rate using information about the short-term behavior of the gradient. The DFC is represented by ξ_t and defined as

$$\xi_t = \frac{1}{1 + \exp -|\Delta g_t|}, \tag{18}$$

where Δg_t is the change between previous and current gradients, given as

$$\Delta g_t = g_{t-1} - g_t.$$

In the proposed DiffGrad optimization method, the steps up to the computation of bias-corrected 1-st order moment m_t and bias-corrected 2-nd order moment v_t are the same as those of Adam optimization [38]. The DiffGrad optimization method updates θ_{t+1}, using the following update rule:

$$\theta_{t+1} = \theta_t - \frac{\alpha_t \xi_t \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}. \tag{19}$$

DiffGrad generate a high learning rate if the gradient change is more (i.e., the optimization is far from the optimum solution), and a low learning rate if the gradient changes minimally (i.e., the optimization is near to the optimum solution). Moreover, such technique lets to avoid some local minimums, which can be seen in minimization of Rastrigin and Rosenbrock functions [56]. This approach is suitable for deep convolutional neural networks due to moment estimation, analysing past and current gradient. Also, DiffGrad adjusts the learning rate very accurately for avoiding overshooting of the global minimum and reducing oscillation around it. Such algorithms is tested with the ResNet50 model for an image categorization task over the CIFAR10 and CIFAR100 datasets in [57]. According to result of recognition, it became clear that DiffGrad works better, compared with SGDM (2), AdaDelta (5) and Adam (11). But this approach does not guarantee the high accuracy in other neural networks, especially in quantum, spiking, complex valued and physics-informed neural networks. Such disadvantage is explained by the lack of analyzing the curvature of minimizing loss function. For that reason, there was introduced progressive optimization algorithm in [58], which is called Yogi.

The Yogi algorithms, like Adam, relies on scaling gradients down by the square root of exponential moving averages of past squared gradients and controls the increase in effective learning rate, leading to even better performance with similar theoretical guarantees on convergence in [59]. It allows to solve the problem of convergence failure in simple convex optimization settings, which Adam-type algorithms, like AdamW, AdamP, QHAdam, NAdam and RAdam, can not handle. The difference between v_t and v_{t-1} and its magnitude depend on v_{t-1} and g_t^2. When v_{t-1} is much larger than g_t^2, Yogi, like Adam, increase the effective learning rate, but such procedure is more controllable. This approach has the following describtion:

$$\begin{gathered} m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \\ v_t = v_{t-1} - (1-\beta_2) sign(v_{t-1} - g_t^2) g_t^2, \\ \hat{m}_t = m_t / (1-\beta_1^t),\ \hat{v}_t = v_t / (1-\beta_2^t), \\ \theta_t = \theta_{t-1} - \gamma \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon). \end{gathered} \tag{20}$$

This method shows better results in deep convolutional neural networks, compared with DiffGrad and other previous Adam-type algorithms. But authors from [60,61] defined new optimization methods for deep learning such as AdaBelief and AdaBound.

The main feature of AdaBelief is adapting the learning rate, according to the "belief" in the current gradient direction. There is a difference between AdaBelief and Adam in parameters v_t and s_t, which are defined as exponential moving averages of g_t^2 and $(g_t - m_t)^2$, respectively. According to s_t as the prediction of the gradient at the next time step, if the observed gradient greatly deviates from the prediction, then one distrust the current observation and take a small step; if the observed gradient is

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

close to the prediction, therefore one trust it and take a large step. This allows to achieve the high test accuracy in convolutional neural networks, that was presented in [61] on ImageNet and CIFAR10.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t,$$

$$s_t = \beta_2 s_{t-1} + (1-\beta_2)(g_t - m_t)^2 + \epsilon,$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t},\ \hat{s}_t = \frac{s_t}{1-\beta_2^t},$$

$$\theta_t = \theta_{t-1} - \gamma \hat{m}_i / (\sqrt{\hat{s}_i} + \epsilon). \tag{21}$$

It should be noticed that such approach has a modified version with fast gradient sign method (FGSM) presented in [62,63].

The AdaBound method gives an opportunity to restrict the learning rate between upper and lower continuous functions, which are called clips. Such technique reduces the probability of vanishing and blow-up gradient. This approach is defined as

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t,$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \ and\ V_t = diag(v_t),$$

$$\hat{\eta}_t = Clip(\alpha/\sqrt{V_t}, \eta_l(t), \eta_u(t)) \ and\ \eta_t = \hat{\eta}_t/\sqrt{t},$$

$$\theta_{t+1} = \theta_t - \eta_t \hat{m}_t / (\sqrt{\hat{v}_i} + \epsilon). \tag{22}$$

This approach is more complex, compared with DiffGrad, Yogi and AdaBelief, but it is capable to converge in the neighborhood of global minimum. There were made experiments of minimization of Rastrigin and Rosenbrock functions (https://github.com/jettify/pytorch-optimizer), where AdaBound achieved the highest accuracy, converging in the neighborhood of global minimum. But such approach is too complex for optimization and there exists much simple method, which is called AdamInject [64]. It reduces time consumption, preserving convergence rate.

AdamInject is one of the most recent approaches in first order optimization algorithms, which, unlike the AdaBelief algorithm, modifies m_t, which is the exponential moving average of g_t, into s_t. Such parameter is equipped with the difference between the previous parameters θ_{t-1} and θ_{t-2}. AdamInject has the following description.

If $t = 1$:

$$s_t = \beta_1 s_{t-1} + (1-\beta_1) g_t.$$

Else:

$$\Delta\theta = \theta_{t-2} - \theta_{t-1}$$

$$s_t = \beta_1 s_{t-1} + (1-\beta_1)(g_t + \Delta g_t^2)/k.$$

Afterwards, one make usual calculations

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

$$\hat{s}_t = \frac{s_t}{1-\beta_1^t},\ \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

$$\theta_t = \theta_{t-1} - \gamma \hat{s}_t / (\sqrt{\hat{v}_t} + \epsilon)$$

This algorithms was tested on Rastrigin and Rosenbrock functions. Afterwards, there were trained on CIFAR10 VGG 16, ResNet, ResNext, SENet and DenseNet equipped with AdamInject, which presented the best results, compared with known analogues.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Introduced Adam-type optimization algorithms are used in deep convolutional neural networks, like Res-Net, Res-Next, InceptionNet, GoogleNet and so on. Also they find an application in recurrent and spiking neural networks due to their described above advantages, which SGD-type algorithms does not have. But in quantum, complex-valued and quaternion-valued neural networks such approaches loses in accuracy to usual SGDM with Nesterov condition. This problem caused researches in [65] come up with extending the number of moment from 2 to 3. This approach is called positive-negative momentum (PNM)

2.3. Positive-Negative Momentum

Developing SGD- and Adam-types optimization algorithms can not be infinite, there have to another approaches and techniques for extending class of first-order methods. This issue made the researches to consider the methods, which allow more than two exponential moving averages m_t and v_t. Because step-size regularization, according to moment estimation and introduced modification in Sections 2.1 and 2.2, has its limits, what can be seen in convolutional neural networks, such as ResNet18, GoogLeNet and DenseNet. In this cases there have to additional exponential moving average, that let descend to neighborhood of global minimum.

In the paper [66] introduced conventional momentum method, also called Heavy Ball (HB), which is described in [67]. Later there was proposed positive-negative momentum (PNM) optimization algorithms. In this approach the main feature is a positive-negative averaging, which is analogue of exponential moving average in Adam. This averaging is described as follows:

$$m_t = \sum_{k=0}^{t} \beta_3 \beta_1^{t-k} g_k.$$

Inspired by this simple idea, there were proposed the combining the positive-negative averaging with the conventional momentum method in [66]. The positive-negative average is described as

$$m_t = (1+\beta_0) m_t^{(odd)} + \beta_0 m_t^{even} = (1+\beta_0) \sum_{k=1,3,\dots,t} \beta_3 \beta_0^{t-k} g_k + \beta_0 \sum_{k=0,2,\dots,t} \beta_3 \beta_0^{t-k} g_k.$$

The stochastic gradient descent equipped with the conventional momentum estimates, which adjust the learning rate and value of the gradient, is written in the following formula

$$\begin{gathered} m_t = \beta_1^2 m_{t-2} + (1-\beta_1^2) g_t, \\ \theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{(1+\beta_0)^2 + \beta_0^2}} [(1+\beta_0) m_t - \beta_0 m_{t-1}]. \end{gathered} \tag{23}$$

This approach is an analogue of SGD-type algorithm, which differs by presence of positive-negative average m_t and step-size adaptation. There is also proposed Adam-type analogue of PNM algorithm, which is called AdaPNM, described as

$$\begin{gathered} m_t = \beta_1^2 m_{t-2} + (1-\beta_1^2) g_t, \\ \hat{m}_t = \frac{(1+\beta_0) m_t - \beta_0 m_{t-1}}{1-\beta_1^t}, \\ v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \\ v_{\max} = \max(v_t, v_{\max}),\ \hat{v}_t = \frac{v_{\max}}{1-\beta_2^t}, \\ \theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{(1+\beta_0)^2 + \beta_0^2}} \hat{m}_t. \end{gathered} \tag{24}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

The advantage of PNM and AdaPNM can be seen in [68], where were tested deep neural networks, such as ResNet, VGG, DenseNet and GoogleNet on image bases CIFAR10 and CIFAR100. These approaches gives higher test accuracy, compared with advances Adam-type optimization algorithms, like Yogi and AdaBound. If one equips AdaPNM and PNM with techniques contained in known analogues with two exponential moving averages, then it can improve the quality of optimization process. Besides these algorithms, there exists the extension of NAG and Nadam, which is called adaptive Nesterov momentum Algorithm (Adan).

In [69] proposed the Adaptive Nesterov momentum algorithm is devoted to effectively accelerate the training of deep neural networks. Adan first reformulates the vanilla Nesterov acceleration to develop a new Nesterov momentum estimation method, which reduces extra computations and memory overhead of computing gradient at the extrapolation point. Adan adopts the first and second order moments of the gradient in adaptive gradient algorithms for convergence acceleration. This approach has the following form:

$$
\begin{aligned}
m_t &= (1-\beta_0)m_{t-1} + \beta_0 g_t, \\
v_t &= (1-\beta_1)v_{t-1} + \beta_1(g_t - g_{t-1}), \\
n_t &= (1-\beta_2)n_{t-1} + \beta_2[g_t + (1-\beta_1)(g_t - g_{t-1})]^2, \\
\eta_t &= \eta/(\sqrt{n_t + \epsilon}), \\
\theta_{t+1} &= (1+\lambda\eta)^{-1}[\theta_t - \eta_t(m_t + (1-\beta_1))v_t].
\end{aligned} \tag{25}
$$

This method is the same generalization of SGD, Adam and NAdam. There is presented the advantage of this approach over AdaBelief, which shew third result of test accuracy after usual SGD. Regardless the modifications of Adam-type algorithms, usual SGD can give even better results, what claims to search other methods, that approve their modifications.

First order optimization methods are suitable for problem of image recognition, time series prediction and object classification. They do not consume much time of execution and power, what makes them actual in modern neural networks. But first order optimization algorithms, except PNM, AdaPNM and Adan, can not significantly increase the accuracy in neural networks with complex architecture, such as graph, complex-valued and quantum neural networks. In the solving differential equations they can not overtake the results of Adam, because physics-informed neural networks contain automatic differentiation, which work after multilayer perceptron.. Therefore, one needs second order optimization algorithms, which are able to significantly improve the process of minimization of the loss function.

3. Second Order Optimization Algorithms

First order optimization algorithms are not the most effective for receiving the global minimum of objective smooth function. Such approaches of minimization generally take into account the directions of gradient in every iterations. The customization and performing of class of first order optimization algorithms can only increase the accuracy and avoid some local minimums. Therefore, for improving the quality of minimization there are provided second order optimization algorithms. They reach the minimum taking into account not only directions of gradients, but the convexity (curvature) of objective function, which is measured with the Hessian. Such approach is called as Newton method.

3.1. Newton Algorithms

The main idea of the Newton optimization method [70] is based on gradient descent, containing the calculation of inverse Hessian of smooth objective function. Such approach increases the accuracy of

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

minimization of functions with multiple number of local minimums. Newton optimization algorithm is described as

$$\theta_{t+1} = \theta_t - Hess(\theta_t)^{-1}\nabla f(\theta_t), \tag{26}$$

where $t = 0, ..., M > 0$ and

$$Hess(\theta_t) = \nabla^2 f(\theta_t). \tag{27}$$

This iterative formula is received from second order Taylor expansion.

In [71] were introduced Newton-Minimum-Residual (Newton-MR) optimization method, which calculates the Hessian matrix by Moore-Penrose inverse [72] operator $[\cdot]^\dagger$ such as

$$\theta_{t+1} = \theta_t + p_t = \theta_t + \left[\nabla^2 f(\theta_t)\right]^\dagger \nabla f(\theta_t). \tag{28}$$

According to Newton and Newton-MR methods, there were suggested OverSketch Newton Fast convex optimization. Such approach is described in [73] with proper algorithms, which compute the update direction, taking into account the case of strongly-convexity. But there are other modifications of Newton's optimization methods, that differs by their simplicity in realization and implementation in neural networks.

There are Krylov methods, such as conjugate gradients (CG) [74], the minimal residual method (MINRES) [75], and the generalized minimal residual method (GMRES) [76]. GMRES applies to indefinite matrices, MINRES applies to symmetric indefinite matrices. Besides GD, MINRES and GMRES, there exists a generic stochastic inexact Newton-Krylov method, described in [77]. Such approach can be implemented and applied in physics-informed neural networks, because of suitable approximations for time-dependent equations with divergence operator $(\nabla\cdot)$. But for vast majority of neural networks CG is the most preferred.

One of the most dispersed Newton's optimization method is conjugate gradient. This approach comprises a class of unconstrained optimization algorithms, characterized by low memory requirements and strong local and global convergence properties. Such optimization algorithms is described as the following iterative formula:

$$\theta_{t+1} = \theta_t + \eta_t d_t,$$

$$d_{t+1} = -g_{t+1} + \beta_t d_t,\ d_0 = -g_0.$$

The main part of this formula is β_t, which is called CG update parameter, which are presented in Table 1.

Table 1. List of conjugate gradient update parameters.

CG Update Parameter	Authors	Year
$\beta_t^{HS} = \frac{g_{t+1}^T y_t}{d_t^T y_t}$	Hestenes and Stiefel [78]	1952
$\beta_t^{FR} = \frac{\|g_{t+1}\|_2^2}{\|g_t\|_2^2}$	Fletcher and Reeves [79]	1964
$\beta_t^{D} = \frac{g_{t+1}^T \nabla^2 f(\theta_t) d_t}{d_t^T \nabla^2 f(\theta_t) d_t}$	Daniel [80]	1967
$\beta_t^{PRP} = \frac{g_{t+1}^T y_t}{\|g_t\|_2^2}$	Polak and Ribière [81] and by Polyak [82]	1969
$\beta_t^{CD} = \frac{\|g_{t+1}\|_2^2}{-d_t^T g_t}$	Fletcher [83], CD stands for "Conjugate Descent"	1987
$\beta_t^{LS} = \frac{g_{t+1}^T y_t}{-d_t^T g_t}$	Liu and Storey [84]	1991
$\beta_t^{DY} = \frac{\|g_{t+1}\|_2^2}{d_t^T g_t}$	Dai and Yuan [85]	1999
$\beta_t^{N} = \left(y_t 2 d_t \frac{\|y_{t+}\|_2^2}{d_t^T g_t}\right)^T \frac{g_{t+1}}{d_t^T y_t}$	Hager and Zhang [86]	2005

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Despite the increasing the accuracy of minimization, Newton method optimization is slower comparing with the first order optimization. This disadvantage significantly impacts on time consumption, that slows the training of deep neural networks. For accelerating minimization process, there are developed many approximations of Hessian matrix. Second order optimization algorithms, which contain the approximation of Hessian matrix, are called quasi-Newton.

3.2. Quasi-Newton Algorithms

The class of quasi-Newton optimization algorithms shows approximately the same accuracy as Newton optimization algorithms, but ability to converge faster makes them useful in machine learning. Ones of the first quasi-newton optimization algorithms are BFGS [87], and L-BFGS [88].

BFGS method and its limited memory version, at the t-th iteration, is presented as the following iterative formula:

$$\theta_{t+1} = \theta_t - \alpha_t H_t \nabla f(\theta_t), \tag{29}$$

where α_t is the step length, $\nabla f(\theta_t)$ is the gradient and H_t is the inverse BFGS Hessian approximation, that is updated at every iteration by means of the formula

$$H_{t+1} = V_k^T H_t V_t + \rho_t s_t s_t^T, \tag{30}$$

$$\rho_t = \frac{1}{y_t^T s_t}, \; V_t = I - \rho_t y_t s_t^T, \tag{31}$$

where the curvature pairs (s_t, y_t) are defined as

$$s_t = \theta_t - \theta_{t-1}, \; y_t = \nabla f(\theta_t) - \nabla f(\theta_{t-1}). \tag{32}$$

The curvature pairs (s_t, y_t) are constructed sequentially at every iteration, and inverse Hessian approximation at the t-th iteration H_t depends on iterate and gradient information from past iterations.

The inverse BFGS Hessian approximations have to satisfy secant and curvature conditions:

$$H_{t+1} y_t = s_t, \; s_t^T y_t > 0, \tag{33}$$

Consequently, as a result, as long as the initial inverse Hessian approximation is positive definite, then all subsequent inverse BFGS Hessian approximations are also positive definite. Note, the inverse Hessian approximation H_{k+1} differs from the approximation H_k by a rank-2 matrix.

In the limited memory version, the matrix H_k is defined at each iteration, as the result of applying m BFGS updates to a multiple of the identity matrix, using the set of m most recent curvature pairs $\{s_t, y_t\}$ kept in storage. As a result, one does not need store four dense inverse Hessian matrices approximation, rather one can store two $m \times d$ matrices and compute the matrix-vector product via the two-loop recursion [89]. After the step has been computed, the oldest pair (s_t, y_t) is discarded and the new curvature pair is stored.

Analogically, there were proposed symmetric rank-one (SR-1) [90] formula and its limited memory version in [91]. At the t-th iteration, the SR1 method computes a new iterate by the formula

$$\theta_{k+1} = \theta_k + p_k, \tag{34}$$

where p_k is the minimizer of the following equation

$$\min_p m_k(p) = f(\theta_k) + \nabla f(\theta_k)^T p + \frac{1}{2} p^T B_k p, \tag{35}$$

$$\|p\|_2 \leq \Delta_k,$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

where Δ_k is the trust region and B_k is the SR1 Hessian approximation computed as

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}. \tag{36}$$

Similar to L-BFGS, in the limited memory version of SR1 the matrix B_k is defined at each iteration as the result of applying m SR1 updates to a multiple of the identity matrix, using a set of m correction pairs $\{s_i, y_i\}$ kept in storage.

Provided BFGS and SR-1 with their low memory versions preserve the rate of convergence and reduce the computational complexity. But this methods remain ineffective in deep neural networks, which makes calculations of matrices of extremely large dimension. Therefore, one needs to come with simplified versions of quasi-Newton methods. One of the most used quasi-Newton optimization method in neural networks is Apollo [92]. Apollo is a quasi-Newton method for nonconvex stochastic optimization, which dynamically incorporates the curvature of the loss function by approximating the Hessian via a diagonal matrix. Importantly, the update and storage of the diagonal approximation of Hessian is as efficient as adaptive first-order optimization methods with linear complexity for both time and memory. To handle nonconvexity, there is replaced the Hessian with its rectified absolute value, which is guaranteed to be positive definite. Experiments on deep neural networks, which recognize images from Cifar10 and ImageNet, show that Apollo achieves significant improvements over other stochastic optimization methods, including SGD and variants of Adam, in terms of both convergence speed and generalization performance. Such approach is described as:

$$\begin{aligned}
g_{t+1} &= \nabla f(\theta_t), \\
m_{t+1} &= \frac{\beta(1-\beta^t)}{1-\beta^{t+1}} m_t + \frac{1-\beta}{1-\beta^{t+1}} g_{t+1}, \\
\alpha &= \frac{d_t^T (m_{t+1} - m_t) + d_t^T B_t d_t}{(\|d_t\|_4 + \epsilon)^4}, \\
B_{t+1} &= B_t - \alpha \cdot diag(d_t^2), \\
D_{t+1} &= rectify(B_{t+1}, 1), \\
d_{t+1} &= D_{t+1}^{-1} m_{t+1}, \\
\theta_{t+1} &= \theta_t - \eta_{t+1} d_{t+1},
\end{aligned} \tag{37}$$

Apollo is a method, which is devoted to accelerate the process of minimization. But there is a problem of losing accuracy for achieving lower time consumption. Because of this disadvantage, authors in [93] proposed another often used quasi-Newton optimization method in neural networks, which is called AdaHessian.

AdaHessian is a second order stochastic optimization algorithm which dynamically incorporates the curvature of the loss function via adaptive estimates of the Hessian. Second order algorithms are among the most powerful optimization algorithms with superior convergence properties as compared to first order methods such as SGD and Adam. The main disadvantage of traditional second order methods is their heavier per-iteration computation and poor accuracy as compared to first order methods. But AdaHessian includes a fast Hutchinson based method to approximate the curvature matrix with low computational overhead, root-mean-square exponential moving average to smooth out variations of the Hessian diagonal across different iterations and a block diagonal averaging to reduce the variance of Hessian diagonal elements. This approximation of Hessian matrix makes the process of learning faster, preserving the reta of convergence.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Before presenting AdaHessian algorithm, it needs to note that gradient $g_t = \nabla f(\theta_t)$ and Hessian $H_t = \nabla^2 f(\theta_t)$. Let present the following matrix diagonalization of Hessian matrix as

$$D_t = diag(H) = \mathbb{E}[\theta_t \odot (H_t \theta_t)],$$

where $\odot$ is a componentwise multiplication of vector. Perform a simple spatial averaging on the Hessian diagonal as follows:

$$D^{(s)}[ib + j] = \frac{1}{b} \sum_{k=1}^{b} D[ib + k],$$

for $1 \leq j \leq b$, $0 \leq i \leq \frac{d}{b} - 1$. Hessian diagonal with momentum:

$$\overline{D}_t = \sqrt{\frac{(1-\beta_2)\sum_{i=1}^{t} \beta_2^{t-i} D_i^{(s)} D_i^{(s)}}{1-\beta_2^t}},$$

$$m_t = \frac{(1-\beta_1)\sum_{i=1}^{t} \beta_1^{t-i} g_i}{1-\beta_1^t},$$

$$v_t = (\overline{D}_t)^k,$$

$$\theta_t = \theta_{t-1} - \eta m_t / v_t. \tag{38}$$

Second order optimization algorithms have higher rate of convergence, especially in the neighborhood of global minimum. Unfortunately, they are not suitable in deep convolutional, recurrent and spiking neural networks, because of their high complexity. Therefore, one needs methods that satisfy the high rate of convergence and low time consumption. In 2012 there were proposed developing second order optimization algorithms using the smooth manifolds in [94]. But in [95] authors came up with idea of using probability distribution manifolds, instead of smooth. Such technique appeared on the intersection of geometry, probability, statistics and optimization.

4. Information-Geometric Optimization Methods

Second order optimization methods converge faster to the neighborhood of global minimum, compared with first order methods. But their time consumption is not fast enough for deep neural networks. Therefore, it is necessary to make quasi-Newton optimization algorithms faster. The most recent way is applying the information geometry. In this section we present a concise and modern view of the basic structures of information geometry, and report some applications in machine learning (statistical mixture clustering).

Initially, idea of using means of geometry in problems of optimization takes its beginning sicne 2010 in [96]. Hessian matrix, presented in second order optimization algorithms, does not contain full information about surface of the loss function, moreover, it makes process of descent too complex from computational point of view. But taking into account the geometric structure of the surface and corresponding spaces (Euclidean, hyperbolic, parabolic) allows to define the shortest way for descent and can reduce unnecessary computations. In [97] authors for defining shortest way applied gradient flow, that improved the quality of searching global minimum. This method applied smooth manifolds and generalized definition of gradient. But later, researches came up with idea to use probability distribution manifolds, which prove their advantage over smooth manifolds in problem of optimization. The intersection of probability theory and statistics with Riemannian geometry produces information geometry.

Analogically to information theory, considering the communication of messages over noisy transmission channels, one defines information sciences as the fields of studying connection between data and families of models, i.e. information sciences create methods to transform information from

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

data to models. Such transformation is made by means of probability, statistics and geometry, that allows to include it in the theory of machine learning.

There is another definition of information geometry given by F. Nielson in [98] as the geometry of decision making. It includes model fitting (inference) which can be interpreted as a decision problem, namely, deciding which model parameter to choose from a family of parametric models. This framework was advocated by Abraham Wald [99,100] who considered all statistical problems as statistical decision problems. Dissimilarities (also loosely called distances among others) play a crucial role not only for measuring the goodness-of-fit of data to model (likelihood in statistics, classifier loss functions in machine learning, objective functions in mathematical programming or operations research, etc.) but also for measuring the discrepancy (or deviance) between models.

In this section we show two distinguish optimization algorithms, based on means of information geometry – natural gradient descent [101] and mirror descent [102]. These methods differ from any other optimization algorithms by their ability to measure distances in non-Euclidean domains using Kullback-Leibler and Bregman divergences, resperctively.

4.1. Natural Gradient Descent

Initially, there was proposed gradient descent with gradient flow in [103]. This approach is a generalization of second-order optimization method on Riemannian manifolds, which, particularly, can be Euclidead or non-Euclidean. Let $(\mathcal{M}^n, g)$ be a Riemannian manifold, where $\mathcal{M}^n$ is a topological space, which can be expressed in local coordinate system of an atlas $\mathcal{A} = \{(U_i, x_i)\}_i$ of charts (U_i, x_i), and for tangent bundle $T\mathcal{M}^n$ Riemannian metric $g : T\mathcal{M}^n \otimes T\mathcal{M}^n \to \mathbb{R}$.

The Riemannian gradient glow dynamics $\theta(t)$ for the optimization problem $\min_{\theta \in \mathcal{M}^n} f(\theta)$, obtained by seeking an infinitesimal change in $\theta(t)$ that would lead to the best improvement in objective value, while controlling the length of the change in terms of the manifold geometry, that is,

$$\theta(t + dt) = argmin_\theta f(\theta)dt + \frac{1}{2} d\theta^T g(\theta, \theta + d\theta)d\theta. \tag{39}$$

For dt, utilizing $d\theta(t) = \theta(t + dt) - \theta(t)$, one can substitute $f(\theta(t)) + \langle d\theta, \nabla f(\theta(t))\rangle$ instead of $f(\theta)$ and receive

$$d\theta(t) = argmin_{d\theta} \langle d\theta, \nabla f(\theta(t))\rangle dt + \frac{1}{2} d\theta^T g(\theta, \theta + d\theta)d\theta.$$

Solving for $d\theta$, one get:

$$\frac{d\theta(t)}{dt} = -g(\theta, \theta + d\theta)^{-1} \nabla f(\theta(t)). \tag{40}$$

For better understanding, there are provided examples of Riemannian metric is the standard Euclidean manifold has the corresponding metric $g = I$, that reduces the gradient flow to usual gradient descent. But there is another very important example – probability distribution manifold with K-L divergence metric ([104,105]), that put the beginning of quantum neural networks.

There is noted the advantage of using probability distribution manifolds in [106], which can be equipped not necessary with KL-divergence, but Bergman, Jensen and others. There are presented extension of Rimannian manifolds with Levi-Civita connection $(\mathcal{M}^n, g, \nabla_{LC})$ to conjugate connection manifolds $(\mathcal{M}^n, g, \nabla, \nabla^*)$, where $\nabla_{LC} = \frac{\nabla + \nabla^*}{2}$. Conjugate connection manifolds are the private case of divergence manifolds, denoted as $(\mathcal{M}^n, D^g, \nabla^D), \nabla^{*D}$. In such manifolds D can be Kullback-Leibler or Bregman divergences. There are two ways to imply natural gradient descent using direct K-L divergence, like in [95], and more fundamental way, presented in [98]. Let $f \in C^\infty(\mathcal{M}^n)$ be a smooth loss function, $\exp_\theta : T_\theta \mathcal{M}^n \to \mathcal{M}^n$ be the Riemannian exponential map for updating the sequence of points θ_t on the manifold as follows:

$$\theta_{t+1} = \exp_{\theta_t}(-\alpha_t \nabla_M f(\theta_t)), \tag{41}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

where

$$\nabla_M f(\theta) = \nabla_v \left(f(\exp_\theta(v)) \right) \big|_{v=0} = \lim_{h \to 0} \frac{f(\theta + hv) - f(\theta)}{h}. \tag{42}$$

Returning to the formula of gradient flow, we receive:

$$\theta_{t+1} = R_{\theta_t}(-\alpha_t \nabla_\theta f(\theta_t)).$$

Using the retraction $R_\theta(v) = \theta + v$ which corresponds to a first-order Taylor approximation of the exponential map, one implies the natural gradient descent:

$$\theta_{t+1} = \theta_t - \alpha_t g^{-1}(\theta, \theta + d\theta) \nabla_\theta f(\theta_t). \tag{43}$$

Here $g(\theta, \theta + d\theta) = F(\theta_t)$ is called Fisher information matrix the natural gradient is defined as follows:

$$\nabla^{NG} f(\theta) = F^{-1}(\theta) \nabla_\theta f(\theta). \tag{44}$$

Afterwards, we come to natural gradient descent:

$$\theta_{t+1} = \theta_t - \alpha_t \nabla_\theta^{NG} f(\theta_t). \tag{45}$$

Natural gradient descent differs from the first- and second-order optimization algorithms presented in Sections 2 and 3, respectively, by ability to converge in global minimum for time consumption, suitable for deep learning. As was said, such approach created new branch in the theory of artificial intelligence – quantum machine learning. Application of neural networks for training quantum circuits required new structure of mathematical model of neurons, appropriate for quantum computations. Afterwards, vanilla natural gradient descent was replaced with quantum in [107]. Such networks are part of developing quantum information theory and quantum computers, that makes the process of training significantly faster. But in actual researches, vanilla natural gradient descent with Dirichlet distributions is already tested on Rastrigin and Rosebrock functions in [108] and exploited in convolutional, recurrent neural networks in [109]. The important thing in this method is selecting appropriate probability distribution, which can increase the rate of convergence and even accelerate learning process. There are presented probability distributions in Table 2, which can improve the quality of minimization of the loss functions.

As can be seen, this is second order optimization algorithm. But selecting appropriate probability distribution, like Gauss and Dirichelt, we can reduce the variable θ in Fisher infromation matrix, what makes possible to avoid its calculation in every iteration. Such approach is realized in [108–111]. The natural gradient descent, based on Fisher-Rao metrics, can replace second order optimization algorithms, due to their rate of convergence and time consumption. But there is another approach, that uses Bregman metrics, which is called mirror descent.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Table 2. List of Fisher information matrices for different probability distributions.

Probability Density Function	Fisher Information Matrix	Probability Distribution
$p(x;\mu,\sigma) = \frac{2\pi^{-\frac{n}{2}}}{\sqrt{\sigma_1^2...\sigma_n^2}} e^{-\frac{(x-\mu)^T diag(\sigma_1^2,...,\sigma_n^2)^{-1}(x-\mu)}{2}}$	$F_G = \begin{pmatrix} \frac{1}{\sigma_1^2} & 0 & ... & 0 & 0 \\ 0 & \frac{2}{\sigma_1^2} & ... & 0 & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ... & \frac{1}{\sigma_n^2} & 0 \\ 0 & 0 & ... & 0 & \frac{2}{\sigma_n^2} \end{pmatrix}$	Gauss [110]
$p(x;t) = \frac{n!}{x_1!...x_n!} t_1^{x_1}...t_n^{x_n}$	$F_M = \begin{pmatrix} \frac{t_1+t_n}{t_1 t_n} & \frac{1}{t_n} & ... & \frac{1}{t_n} \\ \frac{1}{t_n} & \frac{t_2+t_n}{t_2 t_n} & ... & \frac{1}{t_n} \\ ... & ... & ... & \\ \frac{1}{t_n} & \frac{1}{t_n} & ... & \frac{t_1+t_n}{t_{n-1} t_n} \end{pmatrix}$	Multinomial [111]
$p(x;\alpha) = \frac{1}{B(\alpha)} \prod_{i=1}^n x_i^{\alpha_i - 1}, B(\alpha) = \frac{\prod_i \Gamma(\alpha_i)}{\Gamma(\sum_i \alpha_i)}$	$F_D^{ii} = \psi'(\alpha_i) - \psi'(\sum_i \alpha_i)$, $F_D^{ij} = -\psi'(\sum_i \alpha_i)$ $i \neq j,\ i = 1,..,n$	Dirichlet [108,112]
$p(x;\alpha,\beta) = \prod_{i=1}^n \frac{x_i^{\alpha_i - 1}}{B(\alpha_i,\beta_i)} \left(1 - \sum_{j=1}^i x_j\right)^{\gamma_i}$ $\gamma_i = \beta_i - \alpha_{i+1} - \beta_{i+1}$ for $i = 1,...,n-1$ and $\gamma_n = \beta_{n-1}$	$F_{GD} = diag\left(F_D(\alpha_1,\beta_1),...,F_D(\alpha_1,\beta_1)\right)$ $\mathcal{O}$-zero matrix	Generalized Dirichlet [108]

4.2. Mirror Descent

Bregman metrics, as an alternative for Fisher-Rao metrics, is based on dual Hessian manifolds. This approach takes into account not only gradient direction, curvature of the minimizing loss function, but the duality. Dual spaces are defined as the set of maps from the topological spaces to their underlying field, which is usually presented as $\mathbb{R}$. This feature impacted on the name of such method. This technique is a reason continuation of second order optimization algorithms towards the information geometry. Moreover, this method contains a potential in the theory of quantum and complex-valued neural networks. Also it can be used in the physics-informed neural networks, where dual averaging procedure can increase the accuracy of final solutions.

Recall that the gradient descent can be extended by proximity function $\Phi(\cdot,\cdot)$ as follows:

$$\theta_{t+1} = argmin_\theta \left\{ \langle \theta, \nabla f(\theta_t) \rangle + \frac{1}{\alpha_t} \Phi(\theta, \theta_t) \right\}. \tag{46}$$

If $\Phi(\theta,\theta_t) = \frac{1}{2}\|\theta - \theta_t\|_2^2$ then we obtain usual gradient descent. For Bregman divergence the proximity function $\Phi(\theta,\theta_t) = D_\psi(\theta,\theta_t)$ and we get mirror descent

$$\theta_{t+1} = argmin_\theta \left\{ \langle \theta, \nabla f(\theta_t) \rangle + \frac{1}{\alpha_t} D_\psi(\theta, \theta_t) \right\}, \tag{47}$$

where

$$D_\psi(\theta,\theta_t) = \psi(\theta) - \psi(\theta_t) - \langle \theta - \theta_t, \nabla\psi(\theta_t) \rangle. \tag{48}$$

According to presented above divergence manifolds, it is possible to imply the equivalence between natural gradient descent and mirror descent. Also, this fact was proven in [98], but without geometric tools. The mirror descent can be interpreted as a natural gradient descent as follows: Bregman mirror descent on the Hessian manifold $(M, g = \nabla^2 F(\theta))$ is equivalent to natural gradient descent on the dual Hessian manifold $(M, g^* = \nabla^2 F(\eta))$, where F is a Bregman generator, $\eta = \nabla F(\theta)$ and $\theta = \nabla F^*(\eta)$.

Indeed, the mirror descent rule yields the following natural gradient update rule:

$$NG^* : \xi_{t+1} = \xi_t - \eta (g_\xi^*)^{-1}(\xi_t) \nabla_\xi f_\theta(\theta(\xi_t)), \tag{49}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

$$NG^*: \xi_{t+1} = \xi_t - \eta(g^*_\xi)^{-1}(\xi_t)\nabla_\xi f_\xi(\xi_t)$$

where $g^*_\xi(\xi) = \nabla^2 F^*(\xi) = (\nabla^2_\theta F(\theta))^{-1}$ and $\theta(\xi) = \nabla F^*$.

The method is called mirror descent because of performing the gradient step in the dual space, which plays the role of 'mirror'. It means that mirror descent search the global minimum, according to duality of probability distribution manifold, which is equivalent to Fisher infomration matrix for natural gradient descent.

Besides the usual mirror descent, there is proposed stochastic mirror descent (SMD) by Nemirovski and Yudin in [113]. This method presented high accuracy on ResNet18 recognising images from Cifar10. For a strictly convex differentiable function $\psi(\cdot)$, called the potential (proximity) function, stochastic mirror descent is presented by the following iterative formula

$$\nabla\psi(\theta_{t+1}) = \nabla\psi(\theta_t) - \eta\nabla f(\theta_t), \tag{50}$$

which is equivalent to the following expression

$$\theta_{t+1} = \arg\min_\theta \left\{ \eta\theta_t^T \nabla f(\theta_t) + D_\psi(\theta, \theta_t) \right\}, \tag{51}$$

where Bregman divergence can be presented as

$$D_\psi(\theta, \theta_t) = \psi(\theta) - \psi(\theta_t) - \nabla\psi(\theta_t)^T(\theta - \theta_t) \tag{52}$$

is the Bregman divergence with respect to the potential function ψ. Note that D_ψ is nonnegative, convex in its first argument, and that, due to strict convexity, $D_\psi(\theta, \theta') = 0$ if and only if $\theta = \theta'$.

Different choices of the potential function ψ yield different optimization algorithms, which will potentially have different implicit biases. A few examples follow in Table 3.

Table 3. Potential function with corresponding Bregman divergences.

Potential Function $\psi(\theta)$	Bregman Divergence $D_\psi(\theta, \theta')$	Algorithm
$\frac{1}{2}\|\theta\|_2^2$	$\frac{1}{2}\|\theta - \theta'\|_2^2$	Gradient Descent
$\sum_j \theta_j \log\theta_j$	$\sum_j \theta_j \log\frac{\theta_j}{\theta'_j} - \sum_j \theta_j + \sum_j \theta'_j$	Exponentiated Gradient Descent

Potentially, this method can improve the process of minimization of the loss function in convolutional, graph and recurrent neural networks with huge architectures. Also mirror descent can be equipped with the adaptive moment estimation and other modifications proper for first-order optimization methods.

5. Application of Optimization Methods in Modern Neural Networks

Introduced optimization methods find their applications in various architectures of artificial neural networks. All presented above optimization algorithms show good results in some determined problems, which can be solved by means of machine learning.

First order optimization methods are reliable in problems of image recognition, time series prediction, voice detection and text analysis. In such cases one needs deep convolutional and recurrent neural networks, which can be equipped with meta-data [114]. Provided architectures train neural networks too long even with GPU, therefore one needs first optimization methods, which are classified in SGD- and Adam-type algorithms. Such approaches do not consume much time and power, that simplify the process of training. For convolutional neural networks, such as AlexNet, GoogleNet, ResNet, SqueezeNet [115] and VGG [116] it suffice to use SGD-type algorithms for achieving high test accuracy. In case of DenseNet [117], Xception [118], ShuffleNet [119] and GhostNet [120] it is necessary to use advanced Adam-type algorithms, such as DiffGrad, Yogi, AdaBelief, AdaBound,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

AdamInject and AdaPNM. The more complex deep neural network applied in problem of recognition, the more advanced first order optimization methods it requires. The same proposition holds for recurrent neural networks, where for SVR [121], XGBoost [122], LSTM [123] and GRU [124] is enought to apply SGD-type and Adam-type algorithms, such as Adam, RAdam, NAdam and QHAdam. Recurrent neural networks, containing CNN-LSTM [125], CNN-GRU [126], TCN-LSTM [127] layers, need advanced Adam-type algorithms.

Second-order optimization methods have a higher rate of convergence in the neighborhood of global minimum, but their time consumption grows. Even quasi-Newton methods, which are dedicated to reduce time consumption, preserving the rate of convergence, still consume more time resources compared with first-order optimization algorithms. But they can be used in some convolutional neural networks, such as AlexNet, Res-Net, VGG and SqueezeNet. For these architectures time and power consumption is not critical. The most appropriate algorithms for this architectures are Apollo and AdaHessian. In recurrent neural networks second-order optimization algorithms show better results, compared with first-order, but they rise time of training. But second order methods are good in physics-informed neural networks (PINN). In the problem of finding solution of some partial differential equation with initial and boundary conditions one needs to extensively analyse the loss function. In this case L-BFGS, SR1, Apollo and AdaHessian can reach higher accuracy compared with first-order optimization methods. Simple PINN, DeepONet [128], MFNN [129] are not large like CNN, what allows to use second-order optimization methods, which gives solution with the high accuracy. For Riemannian neural networks, which rise the accuracy by determining the geodesic, one prefers to apply Apollo and AdaHessian, because with gradient directions they analyze the curvature of loss function. But for cases of Riemannian convolutional neural networks [130] they can not reduce time and power consumption, and applying first-oder optimization algorithm one loses the test accuracy. Therefore, it is necessary to engage algorithms based on information geometry.

Optimization methods based on information geometry have advantages in speed and accuracy. They can be used in convolutional, recurrent, physics-informed and Rimannian neural networks. Moreover, according to their principle of work there are provided quantum neural networks, which correlate with complex-valued neural network. Natural gradient descent and mirror descent are able to achieve the same accuracy and consume low time resources, as second- and first-order optimization methods, respectively. Such possibilities allows to include such algorithms in convolutional, recurrent, graph and auto-encoder neural networks. Moreover, natural gradient and mirror descent extended the application of neural network in quantum computations, where vanilla natural gradient descent is modified into its quantum analogue. For physics-informed neural networks opportunity to converge in the neighborhood of global minimum, consuming time and power similar to first-order methods, can improve the process of solving partial differential and integral equations. This allows to compete with traditional finite difference, element and volume methods.

The summarizing of the above areas of applications is contained in Table 4.

Note, that CNN, RNN, GNN, PINN, SNN, CVNN, QNN are convolutional, recurrent, graph, physics-informed, spiking, complex-valued and quantum neural networks, respectively. These neural networks proved their ability to solve vast majority of problems, related to recognition, prediction, generation, processing, detecting and so on. All of them belong to set of neural networks with gradient based architectures. Recall, that machine learning is the theory, which studies self-learning algorithms, which also can be classified by means of learning. Gradient based neural networks present one of classes in machine learning. Meanwhile, there are many other ways train, which are advanced gradient and gradient free learning methods.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Table 4. Applications of optimization algorithm, divided in 4 types.

Type of Optimization Algorithm	Optimizer	Application
SGD-type	SGD	CNN [131], RNN [132], GNN [133], PINN [134], SNN [135], CVNN [136], AE [137]
	AdaGrad	CNN [138]
	AdaDelta	CNN[139], RNN [139], SNN [140]
	RMSProp	CNN [141], RNN [142], SNN [143]
	SGDW	CNN [30]
	SGDP	CNN [31]
	QHM	CNN [32]
	NAG	CNN [144], RNN [145]
Adam-type	Adam	CNN [146], RNN [147], SNN [148], PINN [150], GNN [151], CVNN [152]
	AdamW	CNN [40]
	AdamP	CNN [42]
	QHAdam	CNN [45]
	Nadam	CNN [47]
	Radam	CNN [48]
	DiffGrad	CNN [154], RNN [155], GNN [156]
	Yogi	CNN [157], RNN [158]
	AdaBelief	CNN [159], RNN [159], GNN [160]
	AdaBound	CNN [161], RNN [162]
	AdamInject	CNN [64]
PNM-type	PNM	CNN [68]
	AdaPNM	CNN [68]
	Adan	CNN [69]
Newton	Newton approach	CNN [163]
	CG	CNN [164], GNN [165]
Quasi-Newton	(L-)BFGS	PINN [166]
	SR1	CNN [167]
	Apollo	CNN [92]
	AdaHessian	CNN [93]
Information geometry	NGD	CNN [168], RNN [109], GNN [169], PINN [170], QNN [170]
	MD	CNN, RNN [172]

6. Challenges and Potential Research

Despite the advanced optimization methods, there exist problems, which concern the fundamental theory of machine learning. All algorithms presented above are utilizable for neural networks with gradient backpropagation, which is not the unique way to rectify weights. There are presented potential methods in [173], which let reduce gradient calculation, i.e. make the gradient-free error backpropagation process. Therefore, for minimizing loss function one needs to engage alternative optimization methods. On example Alternating Direction Method of Multipliers in [174] or ensemble neural network in [175], where gradient is reduced. For such models one need to use the following gradient free optimization methods in Table 5.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

Table 5. Types of gradient free optimization algorithm.

Type of Optimization Algorithm	Optimizer
Local optimization	Hill Climbing [176], Stochastic Hill Climbing [177], Simulated Annealing [178], Downhill Simplex Optimization [179]
Global optimization	Random Search [180], Grid Search [181], Random Restart Hill Climbing [182], Random Annealing [183], Pattern Search [184], Powell's Method [185]
Population-based optimization	Parallel Tempering [186], Particle Swarm Optimization [187], Spiral Optimization [188], Evolution Strategy [189]
Sequential model-based optimization	Bayesian Optimization [190], Lipschitz Optimization [191], Tree of Parzen Estimators [192]

Gradient free optimization methods attract the interest of many researches, because of their computational simplicity, absence of unwanted differentiation operators and convergence rate, that is not less compared with usual gradient descent. Such approach makes mathematicians to develop alternative mathematical models of neural networks, which show their effectiveness over gradient based neural networks in [193]. But it does not say, that one needs to refuse from the gradient-based optimization algorithms. They have worthy continuation, which is an engaging fractional derivatives for computing the gradient.

Fractional calculus, which is a reason continuation of classical calculus, put significant influence in theories of partial differential and integral equations, approximations, signal processing and optimizations. Attempts to generalize differential operators implied many various properties and helpful propositions, which concern the theory of machine learning. It can be seen by extending gradient based optimization methods by fractional derivatives. Let in Table 6 summarize all known fractional derivatives [194], which can improve the process of minimization of loss functions in neural networks.

Fractional optimization methods have proved their efficiency in [195], but there is a question: that chain rule for fractional derivatives makes the modeling of error propagation more complex? No, because there are implied generalized chain and Leibniz rules in [196], which do not differ much from usual versions. Therefore, it is possible to generalize first order optimization methods from SGD- to Adam-type algorithms.

Simultaneously, there is the problem of developing neural networks, containing bilevel optimization [197]. For neural networks equipped with meta-learning, utilization of provided optimization methods does not suffice. There have to be used bilevel optimization algorithms, such as BSA [198], TTSA [199], HOAG [200], AID-FP [201], AID-CG [202] and stocBiO [203]. Such approaches provide the theoretical guarantee in hyperparameter optimization and meta-learning. Potentionaly, these methods can be improved by means of information geometry or techniques from advanced first order optimization algorithms. Moreover, these algorithms can be useful in physics-informed neural networks.

Another problem in physics-informed neural networks is their extension on delay differential equations. The problem of solving such kind of equations with various type of delay is not solved yet, whereas there is found a model for solving fractional differential equations in [204]. In this case one rises the question about using different activation functions, what, consequently, impact on process of optimization of the loss function. Therefore, one can use optimizers based on information geometry. Afterwards, they can be modified in advanced algorithms, which can guarantee the higher

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

accuracy for minimal number of iterations. Especially, information-geometric methods are actual for complex-valued neural networks.

Table 6. Types of fractional derivatives on finite interval $[a, b]$ for gradient descent.

Type of fractional derivatives	Formulas
Riemann-Liouville	$({}_{RL}D^{\alpha}_{a+}f)(t) = \frac{1}{\Gamma(n-\alpha)}(\frac{d}{dt})^{\kappa}\int_a^t \frac{f(\tau)}{(\tau-t)^{\alpha-\kappa+1}}d\tau,$ $({}_{RL}D^{\alpha}_{b-}f)(t) = \frac{1}{\Gamma(n-\alpha)}(\frac{d}{dt})^{\kappa}\int_t^b \frac{f(\tau)}{(\tau-t)^{\alpha-\kappa+1}}d\tau,$ where $Re(\alpha) > 0$
Liouville-Sonine-Caputo	$({}_{LSC}D^{\alpha}_{a+}f)(t) = \frac{1}{\Gamma(\kappa-\alpha)}\int_a^t \frac{f^{(\kappa)}(\tau)}{(\tau-t)^{\alpha-\kappa+1}}d\tau,$ $({}_{LSC}D^{\alpha}_{b-}f)(t) = \frac{(-1)^{\kappa}}{\Gamma(\kappa-\alpha)}\int_t^b \frac{f^{(\kappa)}(\tau)}{(\tau-t)^{\alpha-\kappa+1}}d\tau,$ where $Re(\alpha) > 0$ and $\kappa = [Re(\alpha)] + 1$
Tarasov	$({}_{T}D^{\alpha}_{a+}f)(t) = \frac{\alpha}{\Gamma(1-\alpha)}\int_0^{\infty} \frac{f(\tau)-f(t-\tau)}{(\tau)^{\alpha+1}}d\tau,$ $({}_{T}D^{\alpha}_{b-}f)(t) = \frac{\alpha}{\Gamma(1-\alpha)}\int_0^{\infty} \frac{f(\tau)-f(t+\tau)}{(\tau)^{\alpha+1}}d\tau,$ where $0 < \alpha < 1$ and $a = 0, b = \infty$
Hadamard	$({}_{H}D^{\alpha}_{a+}f)(t) = \frac{1}{\Gamma(\alpha)}(t\frac{d}{dt})^{\kappa}\int_a^t (\log\frac{t}{\tau})^{\sigma} f(\tau)\frac{d\tau}{\tau},$ $({}_{H}D^{\alpha}_{b-}f)(t) = \frac{1}{\Gamma(\alpha)}(t\frac{d}{dt})^{\kappa}\int_a^t (\log\frac{\tau}{t})^{\sigma} f(\tau)\frac{d\tau}{\tau},$ where $Re(\alpha) > 0$, $\sigma \in \mathbb{C}$ and $\kappa = [Re(\alpha)] + 1$
Marchaud	$({}_{M}D^{\alpha}_{+}f)(t) = \frac{\alpha}{\Gamma(1-\alpha)}\int_{-\infty}^t \frac{f(t)-f(\tau)}{(t-\tau)^{\alpha+1}}d\tau,$ $({}_{M}D^{\alpha}_{-}f)(t) = \frac{\alpha}{\Gamma(1-\alpha)}\int_t^{\infty} \frac{f(t)-f(\tau)}{(t-\tau)^{\alpha+1}}d\tau,$ where $0 < Re(\alpha) < 1$
Liouville-Weyl	$({}_{LW}D^{\alpha}_{+}f)(t) = \frac{1}{\Gamma(\kappa-\alpha)}\frac{d^{\kappa}}{dt^{\kappa}}\int_{-\infty}^t \frac{f(\tau)}{(t-\tau)^{\alpha-\kappa+1}}d\tau,$ $({}_{LW}D^{\alpha}_{+}f)(t) = \frac{1}{\Gamma(\kappa-\alpha)}\frac{d^{\kappa}}{dt^{\kappa}}\int_t^{\infty} \frac{f(\tau)}{(\tau-t)^{\alpha-\kappa+1}}d\tau,$ where $0 < Re(\alpha)$ and $-\infty < x < b < \infty$
Sabzikar-Meerschaert-Chen	$({}_{SMC}D^{\alpha,\lambda}_{+}f)(t) = \frac{\alpha}{\Gamma(1-\alpha)}\int_0^{\infty} \frac{e^{-\lambda\tau}(f(t)-f(t-\tau))}{\tau^{\alpha+1}}d\tau,$ $({}_{SMC}D^{\alpha,\lambda}_{+}f)(t) = \frac{\alpha}{\Gamma(1-\alpha)}\int_0^{\infty} \frac{e^{-\lambda\tau}(f(t)+f(t-\tau))}{\tau^{\alpha+1}}d\tau,$ where $0 < Re(\alpha)$ and $\lambda \in \mathbb{C}$
Katugampola	$({}_{K}D^{\alpha,\lambda}_{a+,\sigma}f)(t) = \frac{\sigma^{\alpha-\kappa+1}e^{\lambda t}}{\Gamma(\kappa-\alpha)}\left(t^{1-\sigma}\frac{d}{dt}\right)^{\kappa}\int_a^t \frac{\tau^{\sigma-1}e^{-\lambda\tau}f(\tau)}{(t^{\sigma}-\tau^{\sigma})^{\alpha-\kappa+1}},$ $({}_{K}D^{\alpha,\lambda}_{b-,\sigma}f)(t) = \frac{\sigma^{\alpha-\kappa+1}e^{\lambda t}}{\Gamma(\kappa-\alpha)}\left(t^{1-\sigma}\frac{d}{dt}\right)^{\kappa}\int_t^b \frac{\tau^{\sigma-1}e^{-\lambda\tau}f(\tau)}{(\tau^{\sigma}-t^{\sigma})^{\alpha-\kappa+1}},$ where $0 < Re(\alpha)$ and $\lambda \in \mathbb{C}$

Complex-valued neural networks shew their efficiency in engineering areas such as antenna design, radar imaging, optical/lightwave information processing and quantum devices such as superconductive devices. All these areas of applications contain the mathematical models with rotational point, wave functions and integral transforms such as Fourier, Laplace and Hilbert. This kind of models suggest the future realization of intrinsically non-von Neumann computers including pattern-information representing devices. Conventional quantum computation is strictly limited in its treatable problems. Contrarily, CVNN-based quantum computation can deal with more general problems, which leads to wider applications of quantum computer science. Therefore, one needs to imply optimization methods, based on quantum and tensor computations. On example techniques, used in shampoo [205], can perform the optimization on tensor level.

Quantum neural networks start to attract attention of many researches, who research and develop advanced methods of machine learning. With appearance of quantum computers it was necessary to expand the theory of neural networks on quantum devices, which let to analyze quantum processes, and tensor-network circuits. For problems of image recognition, time series prediction and moving object detection one needs to use quanvolutional neural networks [206], which engage quantum natural gradient. This optimization algorithms presents a quantum analogue of vanilla natural gradient descent, but it uses the Fubini-Study tensor instead of the Fisher information matrix. Unlike the usual Fisher and Hessian matrices, such tensor considers quantum computations with wave functions, bra- and ket-vectors. Such method, can be equipped with exponential moving and positive-negative

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 April 2023 doi:10.20944/preprints202304.0648.v1

averages, that can increase the test accuracy for quantum neural networks. There are proposition of equipping the quantum neural networks with spiking neurons in recent researches, what will causes other tasks for developing optimization methods with memory.

Recall that neural networks of third generation are based on weights with memory. Such approach is more close model to biological neuron networks, what makes the process of training more accurate. Potentially, for such models one can use corresponding memory-based optimization algorithms, such as mixed stochastic gradient descent, denoted MEGA I and MEGA II in [207]. The ability of remembering past experiences while being trained on a continuum of tasks makes possible to significantly rise the test accuracy of spiking neural networks. This kind of neural networks experienced the extension from multi-layer perceptrons to various advanced networks, such as Feedback SNN, attice map SNN, deep convolutional SNN, spike timing dependent plasticity and many other networks. But one of the most recent architectures are graph networks, which without spiking neurons can successfully train the models.

Graph neural networks, which structure reminds the simple graph, are utilized in many tasks related to medicine and biology. As can be seen in neuroscience resource, biological neural networks do not present the linear, sequential and organized in determined order model, but construct another more progressive connections, which significantly impacted in mathematical model of neurons. If it is possible to create the model with corresponding structure, then one receives the network without unwanted layers and computations. But such model rise the problem of error propagation, which definitely differs from classical error backpropagation. Afterwards, there were proposed alternative error rectification methods, which are called neighbor aggregation and information update in [208]. For this methods are suitable to apply presented above first order optimization methods, which are based on information geometry, on example mirror descent. Such algorithms can be modified with new proximal functions, and extended version of Bregman divergence in further researches. Neverless, there are exists other divergence formulas, which can produce new information-geometric optimization methods. But, for modifying the structure of neural networks, one may use wavelet decompositions, which potentially can process input data more accurately.

Neural networks, based on wavelet, are dispersed in the problems of signal processing. Moreover, such tool is already utilized in numerical solving partial differential equations, which seems as an important advance of physics-informed neural networks. Wavelet decomposition was used in convolution, LSTM and GRU layers, what made the data processing more accurate, especially in cases of scaling input information. Moreover, there is presented model of graph-wavelet neural networks, which gave the best results of node classification in [209], compared with spectral CNN, ChebyNet, GCN and MoNet. For wavelet neural networks one can use algorithms presented in talbe 4, but there are presented two new approaches, which are called whale and butterfly optimization methods, in [210]. Also for such kind of network researches often utilize particle swarm method, which is gradient free. Therefore, there is a possibility in comparing gradient free optimization methods between each other and developing new approaches.

In summary, one needs to say, that developing of optimization algorithms correlates with evolution of architectures of neural network and challenges, posed before researches. Also such relation works in backward, what can be seen in case of quantum natural gradient descent and quantum neural networks. Note that improving neural network can be infinite on the problems of image recognition, therefore one needs to consider the problem of moving detection. In case of time series predictions, it should to consider the stochastic process and Brownian motion, which comes from statistical physics and thermodynamics. Such kind of challenges have a big influence to the theory not just neural networks, but the theory of artificial intelligence.

7. Conclusions

In conclusion of this review, we can say, that fundamental developing of the theory of neural networks allows to simplify the work of human in more scales. The developing of optimization

methods in machine learning allowed to achieve not only higher test accuracy for short time, but to imply necessary features and disadvantages of existing models, what through the time made researches to provide advanced architectures. First order optimization algorithms, presented in section 2, were improving and changing simultaneously with growing of neural networks in sizes and qualities. There were described every SGD-, Adam- and PNM-type algorithms, their approaches in optimization process, improving techniques and evolution. In section 3, we provided second order optimization algorithms, which are divided on Newton and quasi-Newton methods. There were described their applications in neural networks, simplifying their computation complexity by means of approximation theory, which produced other useful modifications and variations. Such approximation led to changing the Hessian by gradient flow tensor, and then by Fisher information matrix. This modification brought us to information geometry, that allowed to provide natural gradient and mirror descents, which equivalence was introduced in section 4. Afterwards, we summarized in Table 4 all type of neural networks, suitable for introduced optimization methods. There are provided modern networks, which already have used introduced optimization algorithms and those, that could have been used for increasing test accuracy. In section 6, we demonstrated ways of further developing and applications of optimization methods. There are provided gradient-free, fractional order and bilevel optimization algorithms, their existing versions and potentials for rising test accuracy for various type of neural networks. In the end, this survey can demonstrate to readers all type of existing optimization methods, their modifications, applications, which can help researches to comprehend the state of modern theory of optimization and machine learning. Such information allows to create advanced fundamental modifications and extend the area of applications for modern neural networks of all types.

Author Contributions: Conceptualization, R.A.; Formal analysis, R.A.; Funding acquisition, P.L., R.A. and N.N.; Investigation, R.A.; Methodology, P.L. and N.N.; Project administration, P.L.; Resources, R.A.; Supervision, P.L.; Writing—original draft, R.A.; Writing—review and editing, P.L. and N.N. All authors have read and agreed to the published version of the manuscript.

Funding: The section 2 is supported by North-Caucasus Center for Mathematical Research under agreement No 075-02-2022-892 with the Ministry of Science and Higher Education of the Russian Federation. The research in section 3 was supported by the Russian Science Foundation (Project No. 21-71-00017). The research in section 4 was supported by the Russian Science Foundation (Project No. 22-71-00009).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the North-Caucasus Federal University for supporting in the contest of projects competition of scientific groups and individual scientists of the North-Caucasus Federal University.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shorten, C.; Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. *J Big Data*, **2019**, *6*, 60.
2. Qian, K.; Pawar, A.; Liao, A. et al. Modeling neuron growth using isogeometric collocation based phase field method. *Sci Rep*, **2022**, *12*, 8120.
3. Liu, Y.; Shi, Y.; Mu, F.; Cheng, J.; Li, C.; Chen, X. Multimodal MRI Volumetric Data Fusion With Convolutional Neural Networks. *IEEE Transactions on Instrumentation and Measurement*, **2022**, *71*, 1-15.
4. Li, Q.; Xiong, D.; Shang, M. Adjusted stochastic gradient descent for latent factor analysis. *Information Sciences*, **2022**, *588*, 196-213.
5. Dogo, E. M.; Afolabi, O.J.; Nwulu, N.I.; Twala, B.; Aigbavboa, C.O. A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks. 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS), Belgaum, India, 2018, pp. 92-99.

6. Ward, R.; Wu, X.; Bottou, L. AdaGrad stepsizes: sharp convergence over nonconvex landscapes. *The Journal of Machine Learning Research*, **2020**, *21(1)*, 9047-9076.
7. Xu, D.; Zhang, S.; Zhang, H.; Mandic, D.P. Convergence of the RMSProp deep learning method with penalty for nonconvex optimization. *Neural Networks*, **2021**, *139*, 17-23.
8. Zeiler, M.D. Adadelta: an adaptive learning rate method, *aeXiv* **2012**, arXiv:1212.5701.
9. Singarimbun, R.N; Nababan, E.B.; Sitompul, O.S. Adaptive Moment Estimation To Minimize Square Error In Backpropagation Algorithm.2019 International Conference of Computer Science and Information Technology (ICoSNIKOM), Medan, Indonesia, 2019, pp. 1-7.
10. Seredynski, F.; Zomaya, A.Y.; Bouvry, P. Function Optimization with Coevolutionary Algorithms. *Intelligent Information Processing and Web Mining. Advances in Soft Computing*, **2003**, *22*, 13-22.
11. Osowski, S.; Bojarczak, P.; M. Stodolskia, M. Fast Second Order Learning Algorithm for Feedforward Multilayer Neural Networks and its Applications. *Neural Networks*, **1996**, *9(9)*, 1583-1596.
12. Tyagi, K.; Rane, C.; Irie, B. et al. Multistage Newton's Approach for Training Radial Basis Function Neural Networks. *SN COMPUT. SCI.*, **2021**, *2*, 366.
13. Likas, A. ; Stafylopatis, A. Training the random neural network using quasi-Newton methods. *European Journal of Operational Research*, **2000**, *126(2)*, 331-339.
14. Arbel, M.; Korba, A.; Salim, A.; Gretton, A. Maximum Mean Discrepancy Gradient Flow, *arXiv* **2019**, arXiv:1906.04370.
15. Ay, N.; Jost, N.J.; Lê, H.V.; Schwachhöfe, L. *Information Geometry*; Springer: Berlin, Heidelberg, Germany, 2008.
16. Gattone, S.A.; Sanctis, A.D.; Russo, T.; Pulcini, D. A shape distance based on the Fisher–Rao metric and its application for shapes clustering. *Physica A: Statistical Mechanics and its Applications*, **2017**, *487*, 93-102.
17. Hua, X.; Fan, H.; Cheng, Y.; Wang, H.; Qin, Y. Information Geometry for Radar Target Detection with Total Jensen–Bregman Divergence. *Entropy*, **2018**, *20*, 256.
18. Osawa, K.; Tsuji, Y.; Ueno, Y.; A. Naruse, A.; Foo. C. -S; Yokota, R. Scalable and Practical Natural Gradient for Large-Scale Deep Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **2022**, *44(1)*, 404-415.
19. Orabona, F.; Crammer, K.; Cesa-Bianchi, N. A generalized online mirror descent with applications to classification and regression. *Mach Learn*, **2015**, *99*, 411–435.
20. Lu, L.; Pestourie, R.; Yao, W.; Wang, Z.; Verdugo, F.; Johnson, S.G. Physics-Informed Neural Networks with Hard Constraints for Inverse Design. *SIAM Journal on Scientific Computing*, **2021**, *43(6)*, 1105 - 1132.
21. Shi, C.; Tan, C.; Wang, T.; Wang, L. A Waste Classification Method Based on a Multilayer Hybrid Convolution Neural Network. *Appl. Sci.*, **2021**, *11*, 8572.
22. Hacker, C.; Aizenberg, I.; Wilson, J. Gpu simulator of multilayer neural network based on multi-valued neurons. *2016 International Joint Conference on Neural Networks (IJCNN)*, **2016**, 4125–4132.
23. Chen, S.; McLaughlin, S.; Mulgrew, B. Complex-valued radial basis function network, part i: Network architecture and learning algorithms. *Signal Process.*, **1994**, *35*, 19–31.
24. Suzuki, Y.; Kobayashi, M. Complex-valued bidirectional auto-associative memory. he 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, TX, USA, 2013, pp. 1-7.
25. Traore, C; Pauwels, E. Sequential convergence of AdaGrad algorithm for smooth convex optimization. *Operations Research Letters*, **2021**, *49(4)*, 452-458.
26. Dogo, E.M.; Afolabi, O.J.; Nwulu, N.I.; Twala, B.; Aigbavboa, C.O. A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks. 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS), Belgaum, India, 2018, pp. 92-99.
27. Gu, P.; Tian, S.; Chen, Y. Iterative Learning Control Based on Nesterov Accelerated Gradient Method. *IEEE Access*, **2019**, *7*, 115836-115842.
28. Van Laarhoven, T. L_2 Regularization versus Batch and Weight Normalization, *arXiv* **2017**, arXiv:1706.05350.
29. Byrd, J.; Lipton, Z.C. What is the Effect of Importance Weighting in Deep Learning? *Proceedings of the 36th International Conference on Machine Learning, PMLR*, **2019**, *97*, 872-881.
30. Vrbančič, G.; Podgorelec, V. Efficient ensemble for image-based identification of Pneumonia utilizing deep CNN and SGD with warm restarts. *Expert Systems with Applications*, **2022**, *187*, 115834.

31. Heo, B.; Chun, S.; Oh, S.J.; Han, D.; Yun, S.; Kim, G.; Uh, Y.; Ha, J.-W. AdamP: Slowing Down the Slowdown for Momentum Optimizers on Scale-invariant Weights, *arXiv* **2021**, arXiv:2006.08217.
32. Sun, J.; Yang, Y.; Xun, G.; Zhang, A. Scheduling Hyperparameters to Improve Generalization: From Centralized SGD to Asynchronous SGD. *ACM Transactions on Knowledge Discovery from Data* (accepted paper).
33. Wu, S. et al. "L_1 -Norm Batch Normalization for Efficient Training of Deep Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, **2019**, *30(7), 2043-2051.*
34. *Yu, Z.; Sun, G.; Lv, J. A fractional-order momentum optimization approach of deep neural networks.* Neural Comput and Applic, **2022**, 34, *7091–7111.*
35. *Gokcesu, K.; Gokcesu, H. Regret Analysis of Global Optimization in Univariate Functions with Lipschitz Derivatives*, arXiv **2021**, *arXiv:2108.10859.*
36. *Gower, R.M.; Loizou, N.; Qian, X.; Sailanbayev, A.; Shulgin, E.; Richtárik, P. SGD: General Analysis and Improved Rates.* Proceedings of Machine Learning Research, **2019**, 97, *5200-5209.*
37. *Mukkamala, M.C.; Hein, M. Variants of RMSProp and Adagrad with Logarithmic Regret Bounds.* Proceedings of Machine Learning Research, **2017**, 70, *2545-2553.*
38. *Wang, G.; Lu, S.; Tu, W.; Zhang, L. Sadam: A variant of adam for strongly convex functions*, arXiv **2019**, *arXiv:1905.02957.*
39. *Kingma, D.P.; Ba. J. Adam: A Method for Stochastic Optimization*, arXiv **2015** *arXiv:1412.6980.*
40. *Loshchilov, I.; Hutter, F. Decoupled weight decay regularization*, arXiv **2017**, *arXiv:1711.05101.*
41. *Kalfaoglu, M.E.; Kalkan, S.; Alatan, A.A. (2020). Late Temporal Modeling in 3D CNN Architectures with BERT for Action Recognition.* Computer Vision – ECCV 2020 Workshops. ECCV 2020. Lecture Notes in Computer Science, **2020**, 12539, *731–747.*
42. *Herrera-Alcántara, O. Fractional Derivative Gradient-Based Optimizers for Neural Networks and Human Activity Recognition.* Appl. Sci. **2022**, 12, *9264.*
43. *Jia, X.; Feng, X.; Yong H.; Meng, D. Weight Decay With Tailored Adam on Scale-Invariant Weights for Better Generalization.* IEEE Transactions on Neural Networks and Learning Systems, **2022**, *1-12.*
44. *Heo, B.; Chun, S.; Oh, S.J.; Han, D.; Yun, S.; Kim, G.; Uh, Y.; Ha, J.-W. AdamP: Slowing Down the Slowdown for Momentum Optimizers on Scale-invariant Weights*, arXiv **2021**, *arXiv:2006.08217v3.*
45. *Ma, J.; Yarats, D. Quasi-hyperbolic momentum and Adam for deep learning*, arXiv **2019** *arXiv:1810.06801v4.*
46. *Tang, S.; Shen, C.; Wang, D.; Li, S.; Huang, W.; Zhu, Z. Adaptive deep feature learning network with Nesterov momentum and its application to rotating machinery fault diagnosis.* Neurocomputing, **2018**, 305, *1-14.*
47. *Li, L.; Xu, W.; Yu, H. Character-level neural network model based on Nadam optimization and its application in clinical concept extraction.* Neurocomputing, **2020**, 414, *182-190.*
48. *Melinte, D.O.; Vladareanu, L. Facial Expressions Recognition for Human–Robot Interaction Using Deep Convolutional Neural Networks with Rectified Adam Optimizer.* Sensors, **2020**, 20, *2393.*
49. *Gholamalinejad, H,; Khosravi, H. Whitened gradient descent, a new updating method for optimizers in deep neural networks.* Journal of AI and Data Mining, **2022**, 10(4), *467-477.*
50. *Shanthi, T.; Sabeenian, R.S. Modified Alexnet architecture for classification of diabetic retinopathy images.* Computers and Electrical Engineering, **2019**, 76, *56-64.*
51. *Wu, Z.; Shen, C.; Van Den Hengel, A. Wider or Deeper: Revisiting the ResNet Model for Visual Recognition.* Pattern Recognition, **2019**, 90, *119-133.*
52. *Das, D.; Santosh, K.C.; Pal, U. Truncated inception net: COVID-19 outbreak screening using chest X-rays.* Phys Eng Sci Med, **2020**, 43, *915–925.*
53. *Tang, P.; Wang, H.; Kwong, S. G-MS2F: GoogLeNet based multi-stage feature fusion of deep CNN for scene recognition.* Neurocomputing, **2017**, 225, *188-197.*
54. *Lin, L.; Liang, L.; Jin, L. R2-ResNeXt: A ResNeXt-Based Regression Model with Relative Ranking for Facial Beauty Prediction. 2018 24th International Conference on Pattern Recognition (ICPR), pp. 85-90, 2018.*
55. *Dubey, S.R.; Chakraborty, S.; Roy,S.K.; Mukherjee, S.; Singh, S.K.; Chaudhuri, B.B. "diffGrad: An Optimization Method for Convolutional Neural Networks.* IEEE Transactions on Neural Networks and Learning Systems, **2020**, 31(11), *4500-4511.*
56. *Panait, L.; Luke, S. A comparison of two competitive fitness functions. GECCO'02: Proceedings of the 4th Annual Conference on Genetic and Evolutionary ComputationJuly 2002, pp. 503–511.*

57. *Khan, W.; Ali, S.; Muhammad, U.S.K.; Jawad, M.; Ali, M.; Nawaz, R. AdaDiffGrad: An Adaptive Batch Size Implementation Technique for DiffGrad Optimization Method. 2020 14th International Conference on Innovations in Information Technology (IIT), Al Ain, United Arab Emirates, 2020, pp. 209-214.*
58. *Valova, I.;Harris, C.; Mai, T.; Gueorguieva, N. Optimization of Convolutional Neural Networks for Imbalanced Set Classification.* Procedia Computer Science, **2020**, 176, *660-669.*
59. *Zaheer, M.; Reddi, S.; Sachan, D.; Kale, S.; Kumar, S. Adaptive Methods for Nonconvex Optimization.* Advances in Neural Information Processing Systems, **2018**, 31.
60. *Zhuang, J.; Tang, T.; Ding, Y.; Tatikonda, S.C.; Dvornek, N. Papademetris, X.; Duncan, J. AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients.* Advances in Neural Information Processing Systems, **2020**, 33.
61. *Liu, J.; Kong, J.; Xu, D.; Qi, M.; Lu, Y. Convergence analysis of AdaBound with relaxed bound functions for non-convex optimization.* Neural Networks, **2022**, 145, *300-307.*
62. *Wang, Y.; Liu, J.; Chang, X.; Wang, J.; Rodríguez, R.J. AB-FGSM: AdaBelief optimizer and FGSM-based approach to generate adversarial examples.* Journal of Information Security and Applications, **2022**, 68, *103227.*
63. *Wang, Y.; Liu, J.; Chang, X. Generalizing Adversarial Examples by AdaBelief Optimizer,* arXiv **2021**, *arXiv:2101.09930v1.*
64. *Dubey, S.R.; Basha, S.H.S.;Singh, S.K.; Chaudhuri, B.B. AdaInject: Injection Based Adaptive Gradient Descent Optimizers for Convolutional Neural Networks.* IEEE Transactions on Artificial Intelligence, **2022**, *1 - 10.*
65. *Li, G. A Memory Enhancement Adjustment Method Based on Stochastic Gradients. 2022 41st Chinese Control Conference (CCC), Hefei, China, 2022, pp. 7448-7453.*
66. *Xie, Z.; Yuan, L.; Zhu, Z.; Sugiyama, M. Positive-Negative Momentum: Manipulating Stochastic Gradient Noise to Improve Generalization.* Proceedings of the 38th International Conference on Machine Learning, PMLR, **2021**, 139, *11448-11458.*
67. *Zavriev, S.; Kostyuk, F. Heavy-ball method in nonconvex optimization problems.* Computational Mathematics and Modeling, **1993**, 4(4), *336–341.*
68. *Wright, L.; Demeure, N. Ranger21: a synergistic deep learning optimizer,* arXiv **2021**, *arXiv:2106.13731v2.*
69. *Xie, X.; Zhou, P.; Li, H.; Lin, Z.; Yan, S. Adan: Adaptive Nesterov Momentum Algorithm for Faster Optimizing Deep Models,* arXiv **2022**, *arXiv:2208.06677v3.*
70. *Burke, J.V.; Ferris, M.C. A Gauss—Newton method for convex composite optimization.* Mathematical Programming, **1995**, 71, *179–194.*
71. *Berahas, A.S.; Bollapragada, R.; Nocedal, J. An investigation of Newton-Sketch and subsampled Newton methods.* Optimization Methods and Software, **2020**, 35(4), *661-680.*
72. *Hartmann, W.M.; Hartwig, R.E. Computing the Moore–Penrose Inverse for the Covariance Matrix in Constrained Nonlinear Estimation.* SIAM Journal on Optimization, **1996**, 6(3), *727-747.*
73. *Gupta, V.; Kadhe, S.; Courtade, T.; Mahoney, M.W.; Ramchandran, K. OverSketched Newton: Fast Convex Optimization for Serverless Systems.* 2020 IEEE International Conference on Big Data (Big Data), **2020**, *288-297.*
74. *Yang, Z. Adaptive stochastic conjugate gradient for machine learning.* Expert Systems with Applications, **2022**, 206, *117719.*
75. *Faber, V.; Joubert, W.; Knill, E.; Manteuffel, T. Minimal Residual Method Stronger than Polynomial Preconditioning.* SIAM Journal on Matrix Analysis and Applications, **1996**, 17(4), *707-729.*
76. *Jia, Z.; Ng, M.K. Structure Preserving Quaternion Generalized Minimal Residual Method.* SIAM Journal on Matrix Analysis and Applications, **2021**, 42(2), *616-634.*
77. *Mang, A.; Biros, G. An Inexact Newton–Krylov Algorithm for Constrained Diffeomorphic Image Registration.* SIAM Journal on Imaging Sciences, **2015**, 8(2), *1030-1069.*
78. *Hestenes, M.R.; Stiefel, E.L. Methods of conjugate gradients for solving linear systems.* J. Research Nat. Bur. Standards, **1952**, 49, *409–436.*
79. *Fletcher, R.; Reeves, C. Function minimization by conjugate gradients.* Comput. J., **1964**, 7, *149–154.*
80. *Daniel, J.W. The conjugate gradient method for linear and nonlinear operator equations.* SIAM J. Numer. Anal., **1967**, 4, *10–26.*
81. *Polak, E.; Ribiere, G. Note sur la convergence de directions conjuge ´es.* Rev. Francaise Informat Recherche Opertionelle, 3e Ann´ee 16, **1969**, *35–43.*
82. *Polyak, B.T. The conjugate gradient method in extreme problems.* USSR Comp. Math.Math. Phys., **1969**,9, *94–112.*

83. *Fletcher, R.* Practical Methods of Optimization vol. 1: Unconstrained Optimization*; John Wiley and Sons: New York, USA, 1987.*
84. *Liu, Y.; Storey, C. Efficient generalized conjugate gradient algorithms.* J. Optim. Theory Appl., **1991**, 69, *129–137.*
85. *Dai, Y.H.; Yuan, Y. A nonlinear conjugate gradient method with a strong global convergence property.* SIAM J. Optim., **1999**, 10, *177–182.*
86. *Hager, W.W.; Zhang, H. A new conjugate gradient method with guaranteed descent and an efficient line search.* SIAM J. Optim., **2005**, 16(1), *170-192.*
87. *Dai, Y.-H. Convergence Properties of the BFGS Algoritm.* SIAM Journal on Optimization, **2002**, 13(3), *693–701.*
88. *Liu, D.C.; Nocedal, J. On the limited memory BFGS method for large scale optimization.* Mathematical Programming, **1989**, 45, *503–528.*
89. *Shi, H.-J. M.; Xie, Y.; Byrd, R.; J. Nocedal. A Noise-Tolerant Quasi-Newton Algorithm for Unconstrained Optimization.* SIAM Journal on Optimization, **2022**, 32(1), *29-55.*
90. *Byrd, R.H.; Khalfan, H.F.; Schnabel, R.B. Analysis of a Symmetric Rank-One Trust Region Method.* SIAM Journal on Optimization, **1996**, 6(4), *1025-1039.*
91. *Rafati, J.; Marcia, R.F. Improving L-BFGS Initialization for Trust-Region Methods in Deep Learning.* 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), **2018**, *501-508.*
92. *Ma, X. Apollo: An Adaptive Parameter-wise Diagonal Quasi-Newton Method for Nonconvex Stochastic Optimization*, arXiv **2021** , *arXiv:2009.13586v6.*
93. *Yao, Z.; Gholami, A.; Shen, S.; Mustafa, M.; Keutzer, K.; Mahoney, M. ADAHESSIAN: An Adaptive Second Order Optimizer for Machine Learning.* Proceedings of the AAAI Conference on Artificial Intelligence, **2021**, 35(12), *10665-10673.*
94. *Shen, J.; Wang, C.; Wang, X.; Wise, S.M. Second-order Convex Splitting Schemes for Gradient Flows with Ehrlich–Schwoebel Type Energy: Application to Thin Film Epitaxy.* SIAM Journal on Numerical Analysis, **2012**, 50(1), *105-125.*
95. *Martens, J. New insights and perspectives on the natural gradient method.* Journal of Machine Learning Research, **2020**, 21(146), *5776–5851.*
96. *Amari, Si. Information geometry in optimization, machine learning and statistical inference.* Front. Electr. Electron. Eng. China, **2010**, 5, *241–260.*
97. *Wang, S.; Teng, Y.; Perdikaris, P. Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks.* SIAM Journal on Scientific Computing, **2021**, 43(5), *3055-3081.*
98. *Nielsen, F. An Elementary Introduction to Information Geometry.* Entropy, **2020**, 22, *1100.*
99. *Wald, A. Statistical decision functions.* Ann. Math. Stat., **1949**, *165–205.*
100. *Wald, A.* Statistical Decision Functions*; Wiley: Chichester, U.K., 1950.*
101. *Rattray, M.; Saad, D.; Amari, S. Natural Gradient Descent for OnLine Learning.* Phys. Rev. Lett., **1998**, 81(24), *5461-5464.*
102. *Duchi, J.C.; Agarwal, A.; Johansson, M.; Jordan, M.I. Ergodic Mirror Descent.* SIAM Journal on Optimization, **2012**, 22(4), *1549-1578.*
103. *Wang, Y.; Li, W. Accelerated Information Gradient Flow.* J. Sci. Comput., **2022**, 90, *11.*
104. *Goldberger; Gordon; Greenspan. An efficient image similarity measure based on approximations of KL-divergence between two gaussian mixtures.* Proceedings Ninth IEEE International Conference on Computer Vision, **2003**, 1, *487-493.*
105. *Joyce, J.M.* Kullback-Leibler Divergence. In: Lovric, M. *(eds) International Encyclopedia of Statistical Science. Springer: Berlin, Heidelberg, Germany, 2011.*
106. *Nielsen, F. Statistical Divergences between Densities of Truncated Exponential Families with Nested Supports: Duo Bregman and Duo Jensen Divergences.* Entropy, **2022**, 24, *421.*
107. *Stokes, J.; Izaac, J.; Killoran, N.; Carleo, G. Quantum Natural Gradient.* Open journal for quantum science, **2020**, 4, *269-284.*
108. *Abdulkadirov, R.; Lyakhov, P.; Nagornov, N. Accelerating Extreme Search of Multidimensional Functions Based on Natural Gradient Descent with Dirichlet Distributions.* Mathematics, **2022**, 10, *3556.*
109. *Abdulkadirov R.I.; Lyakhov P.A. A new approach to training neural networks using natural gradient descent with momentum based on Dirichlet distributions.* Computer Optics, **2023**. 47(1), *160-170.*

110. *Lyakhov, P,; Abdulkadirov, R. Accelerating Extreme Search Based on Natural Gradient Descent with Beta Distribution. 2021 International Conference Engineering and Telecommunication (En&T), Dolgoprudny, Russian Federation, 2021, pp. 1-5.*
111. *Abdulkadirov, R.I.; Lyakhov, P.A. Improving Extreme Search with Natural Gradient Descent Using Dirichlet Distribution.* Mathematics and its Applications in New Computer Systems. MANCS 2021. Lecture Notes in Networks and Systems, **2022**, 424, *19-28.*
112. *Kesten, H.; Morse, N. A Property of the Multinomial Distribution.* The Annals of Mathematical Statistics, **1959**, 30(1), *120-127.*
113. *D'Orazio, R.; Loizou, N.; Laradji, I.; Mitliagkas, I. Stochastic Mirror Descent: Convergence Analysis and Adaptive Variants via the Mirror Stochastic Polyak Stepsize*, arXiv **2021**, *arXiv:2110.15412v2.*
114. *Gessert, N.; Nielsen, M.; Shaikh, M.; Werner, R.; Schlaefer, A. Skin lesion classification using ensembles of multi-resolution EfficientNets with meta data*, MethodsX, **2020**, 7, *100864.*
115. *Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size*, arXiv **2016**, *arXiv:1602.07360v4.*
116. *Ke, H.; Chen, D.; Li, X.; Tang, Y.; Shah, T.; Ranjan, R. Towards Brain Big Data Classification: Epileptic EEG Identification With a Lightweight VGGNet on Global MIC.* IEEE Access, **2018**, 6, *14722-14733.*
117. *Zhu, Y.; Newsam, S. DenseNet for dense flow. 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 2017, pp. 790-794.*
118. *Chollet, F. Xception: Deep Learning With Depthwise Separable Convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1251-1258*
119. *Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 6848-6856.*
120. *Paoletti, M.E.; Haut, J.M.; Pereira, N.S.; Plaza, J.; Plaza, A. Ghostnet for Hyperspectral Image Classification.* IEEE Transactions on Geoscience and Remote Sensing, **2021**, 59(12), *10378-10393.*
121. *Liu, Y. Novel volatility forecasting using deep learning–Long Short Term Memory Recurrent Neural Networks.* Expert Systems with Applications, **2019**, 132, *99-109.*
122. *Lai, CH.; Liu, DR.; Lien, KS. A hybrid of XGBoost and aspect-based review mining with attention neural network for user preference prediction.* Int. J. Mach. Learn. and Cyber., **2021**, 12, *1203–1217.*
123. *Sherstinsky, A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network.* Physica D: Nonlinear Phenomena, **2020**, 404, *132306.*
124. *Lynn, H.H.; Pan S.B.; Kim, P. A Deep Bidirectional GRU Network Model for Biometric Electrocardiogram Classification Based on Recurrent Neural Networks.* IEEE Access, **2019**, 7, *145395-145405.*
125. *Kim, T.Y.; Cho, S.B. Predicting residential energy consumption using CNN-LSTM neural networks.* Energy, **2019**, 182, *72-81.*
126. *Sajjad, M. et al. A Novel CNN-GRU-Based Hybrid Approach for Short-Term Residential Load Forecasting.* IEEE Access, **2020**,8, *143759-143768.*
127. *Hu, C,; Cheng, F.; Ma, L.; Li, B. State of Charge Estimation for Lithium-Ion Batteries Based on TCN-LSTM Neural Networks.* Journal of The Electrochemical Society, **2022**, 169, *0305544.*
128. *Lu, L.; Jin, P.; Pang, G. et al. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators.* Nat Mach Intell, **2021**, 3, *218–229.*
129. *Meng, X.; Karniadakis, G.T. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems.* Journal of Computational Physics, **2020**, 401, *109020.*
130. *Gao, C.; Lui, W.; Yang, X. Convolutional neural network and riemannian geometry hybrid approach for motor imagery classification.* Neurocomputing, **2022**, 507, *180-190. SGD*
131. *Hosseini, M.S.; Tuli, M.; Plataniotis, K.N. Exploiting Explainable Metrics for Augmented SGD. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 10296-10306*
132. *Singh, H.; Bhatt, P.; Jacob, S.; Kaur, A.; Vashist, A.; Vij. D. Stock Prediction on Historical Data based on SGD and LSTM. 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM), 2022, pp. 200-204.*
133. *Mu, Z.; Tang, S.; Zong, C.; Yu, D.; Zhuang, Y. Graph neural networks meet with distributed graph partitioners and reconciliations.* Neurocomputing, **2023**, 518, *408-417.*

134. *Li, J.; Chen, J.; Li, B. Gradient-optimized physics-informed neural networks (GOPINNs): a deep learning method for solving the complex modified KdV equation.* Nonlinear Dyn, **2022**, 107, *781–792.*
135. *Volinski, A.; Zaidel,Y.; Shalumov, A.; DeWolf, T.; Supic, L.; Tsur, E.E. Data-driven artificial and spiking neural networks for inverse kinematics in neurorobotics.* Patterns, **2022**, 3(1), *100391.*
136. *Wang, R.; Liu, Z.; Zhang, B. et al. Few-Shot Learning with Complex-Valued Neural Networks and Dependable Learning.* Int J Comput Vis, **2023**, 131, *385–404.*
137. *Chen, M.; Shi, X.; Zhang, Y.; Wu, D.; Guizani, M. Deep Feature Learning for Medical Image Analysis with Convolutional Autoencoder Neural Network.* IEEE Transactions on Big Data, **2021**, 7(4), *750-758. AdaGrad*
138. *Taqi, A.M.; Awad, A.; Al-Azzo, F.; Milanova, M. The Impact of Multi-Optimizers and Data Augmentation on TensorFlow Convolutional Neural Network Performance. 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2018, pp. 140-145. AdaDelta*
139. *Qu, Z.; Yuan, S.; Chi, R.; Chang, L.; Zhao, L. Genetic Optimization Method of Pantograph and Catenary Comprehensive Monitor Status Prediction Model Based on Adadelta Deep Neural Network.* IEEE Access, **2019**, 7, *23210-23221.*
140. *Huang, Y.; Peng, H.; Liu, Q.; Yang, Q.; Wang, J.; Orellana-Martin, D.; Perez-Jimenez, M.J. Attention-enabled gated spiking neural P model for aspect-level sentiment classification.* Neural Networks, **2023**, 157, *437-443. RMSProp*
141. *Taqi, A.M.; Awad, A.; Al-Azzo, F.; Milanova, M. The Impact of Multi-Optimizers and Data Augmentation on TensorFlow Convolutional Neural Network Performance. 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2018, pp. 140-145.*
142. *Huk, M. Stochastic Optimization of Contextual Neural Networks with RMSprop.* Lecture Notes in Computer Science, **2020**, 12034, *343–352.*
143. *Gautam, A.; Singh, V. CLR-based deep convolutional spiking neural network with validation based stopping for time series classification.* Appl Intell, **2020**, 50, *830–848. NAG*
144. *Liu, B.; Zhang, Y.; He, D.; Li, Y. Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks.* Symmetry, **2018**, 10, *11.*
145. *Kisvari, A.; Lin, Z.; Liu, X. Wind power forecasting – A data-driven method along with gated recurrent neural network.* Renewable Energy, **2021**, 163, *1895-1909. Adam*
146. *Kim, K.-S.; Choi, Y.-S. HyAdamC: A New Adam-Based Hybrid Optimization Algorithm for Convolution Neural Networks.* Sensors, **2021**, 21, *4054.*
147. *Shankar, K.; Kumar, S.; Dutta, A.K.; Alkhayyat, A.; Jawad, A.J.M.; Abbas, A.H.; Yousif, Y.K. An Automated Hyperparameter Tuning Recurrent Neural Network Model for Fruit Classification.* Mathematics, **2022**, 10, *2358.*
148. *Wu, J.; Chua, Y.; Zhang, M.; Yang, Q.; Li, G.; Li, H. Deep Spiking Neural Network with Spike Count based Learning Rule. 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1-6.*
149. *Gong, M.; Zhou, H.; Qin, A.K.; Liu, W.; Zhao, Z. Self-Paced Co-Training of Graph Neural Networks for Semi-Supervised Node Classification.* IEEE Transactions on Neural Networks and Learning Systems, **2022**,*1 - 14.*
150. *Bararnia, H.; Esmaeilpour, M. On the application of physics informed neural networks (PINN) to solve boundary layer thermal-fluid problems.* International Communications in Heat and Mass Transfer, **2022**, 132, *105890.*
151. *Lu, S.; Sengupta, A. Exploring the Connection Between Binary and Spiking Neural Networks.* Front. Neurosci., **2020**, 14, *535.*
152. *Freire, P.J. et al. Complex-Valued Neural Network Design for Mitigation of Signal Distortions in Optical Links.* Journal of Lightwave Technology, **2021**, 39(6), *1696-1705.*
153. *Jiang, J.; Ren, H.; Zhang, M. A Convolutional Autoencoder Method for Simultaneous Seismic Data Reconstruction and Denoising.* IEEE Geoscience and Remote Sensing Letters, **2022**, 19, *1-5. DiffGrad*
154. *Khan, W.;Ali, S.; Muhammad, U.S.K.; Jawad, M.; Ali, M.; Nawaz, R. AdaDiffGrad: An Adaptive Batch Size Implementation Technique for DiffGrad Optimization Method. 2020 14th International Conference on Innovations in Information Technology (IIT), 2020, pp. 209-214.*
155. *Sun, W.; Wang, Y.; Chang, K.; Meng, K. IdiffGrad: A Gradient Descent Algorithm for Intrusion Detection Based on diffGrad. 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2021, pp. 1583-1590.*
156. *Roy, S.K.; Manna, S.; Dubey, S.R.; Chaudhuri, B.B. LiSHT: Non-parametric linearly scaled hyperbolic tangent activation function for neural networks*, arXiv **2022**, *arXiv:1901.05894v3. Yogi*

157. *Valova, I.; Harris, C.; Mai,T.; Gueorguieva, N. Optimization of Convolutional Neural Networks for Imbalanced Set Classification.* Procedia Computer Science, **2020**, 176, *660-669.*
158. *Yogi, S.C.; Tripathi, V.K.; Behera, L. Adaptive Integral Sliding Mode Control Using Fully Connected Recurrent Neural Network for Position and Attitude Control of Quadrotor.* IEEE Transactions on Neural Networks and Learning Systems, **2021**, 32(12), *5595-5609. AdaBelief*
159. *Shi, H.; Wang, L.; Scherer, R.; Woźniak, M.; Zhang, P.; Wei, W. Short-Term Load Forecasting Based on Adabelief Optimized Temporal Convolutional Network and Gated Recurrent Unit Hybrid Neural Network.* IEEE Access, **2021**, 9, *66965-66981.*
160. *Guo, J.; Liu, Q.; Guo, H.; Lu, X. Ligandformer: A Graph Neural Network for Predicting Ligand Property with Robust Interpretation*, arXiv **2022**, *arXiv:2202.10873v3. AdaBound*
161. *Wu, D.; Yuan, Y.; Huang, J.; Tan, Y. Optimize TSK Fuzzy Systems for Regression Problems: Minibatch Gradient Descent With Regularization, DropRule, and AdaBound (MBGD-RDA).* IEEE Transactions on Fuzzy Systems, **2020**, 28(5), *1003-1015.*
162. *Demertzis, K.; Iliadis, L.; Pimenidis, E. Large-Scale Geospatial Data Analysis: Geographic Object-Based Scene Classification in Remote Sensing Images by GIS and Deep Residual Learning.* Proceedings of the International Neural Networks Society, **2020**, 2, *274-291. Newton approach*
163. *Wang, C. et al. Distributed Newton Methods for Deep Neural Networks.* Neural Computation, **2018**, 30(6), *1673-1724. CG*
164. *Kim, H., Wang, C.; Byun, H.; Hu, W.; Kim, S.; Jiao, Q.; Lee, T.H. Variable three-term conjugate gradient method for training artificial neural networks.* Neural Networks, 2022 *(accepted paper).*
165. *Peng, C.-C.; Magoulas, G.D. Adaptive Nonmonotone Conjugate Gradient Training Algorithm for Recurrent Neural Networks. 19th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2007), 2007, pp. 374-381. (L-)BFGS*
166. *Franklin, T.S.; Souza, L.S.; Fontes, R.M.; Martins, M.A.F. A Physics-Informed Neural Networks (PINN) oriented approach to flow metering in oil wells: an ESP lifted oil well system as a case study.* Digital Chemical Engineering, **2022**, 5, *100056. SR-1*
167. *Koshimizu, H.; Kojima, R.; Kario, K.; Okuno, Y. Prediction of blood pressure variability using deep neural networks.* International Journal of Medical Informatics, **2020**, 136, *104067. NGD*
168. *Osawa, K.; Tsuji, Y.; Ueno, Y.; Naruse, A.; Foo, C.-S.; Yokota, R. Scalable and Practical Natural Gradient for Large-Scale Deep Learning.* IEEE Transactions on Pattern Analysis and Machine Intelligence, **2022**, 44(1), *404-415.*
169. *Sun, F.; Sun, J.; Zhao, Q. A deep learning method for predicting metabolite–disease associations via graph neural network.* Briefings in Bioinformatics, **2022**, 23(4), .
170. *Boso, F.; Tartakovsky, D.M. Information geometry of physics-informed statistical manifolds and its use in data assimilation.* Journal of Computational Physics ,**2022**, 467, *111438.*
171. *Becker, S.; Li, W. Quantum Statistical Learning via Quantum Wasserstein Natural Gradient.* J Stat Phys, **2021** 182(7), *1-26. mirror descent*
172. *You, J.-K. ; Cheng, H.-C.; Li, Y.-H. Minimizing Quantum Rényi Divergences via Mirror Descent with Polyak Step Size. 2022 IEEE International Symposium on Information Theory (ISIT), 2022, pp. 252-257.*
173. *Chen, Y.; Chang, H.; Meng, J.; Zhang, D. Ensemble Neural Networks (ENN): A gradient-free stochastic method.* Neural Networks, **2019**, 110, *170-185.*
174. *Han, D.; Yuan, X. A Note on the Alternating Direction Method of Multipliers.* J Optim Theory Appl, **2012**, 155, *227–238.*
175. *Zhang, S.; Liu, M.; Yan, J. The Diversified Ensemble Neural Network.* Advances in Neural Information Processing Systems 33, *bf 2020.*
176. *Dominic, S.; Das, R.; Whitley, D.; Anderson, C. Genetic reinforcement learning for neural networks.* IJCNN-91-Seattle International Joint Conference on Neural Networks, **1991**, 2, *71-76.*
177. *Kanwar, S.; Awasthi, L.K.; Shrivastava, V. Feature Selection with Stochastic Hill-Climbing Algorithm in Cross Project Defect Prediction. 2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), pp. 632-635, 2022.*
178. *Sexton, R.S.; Dorsey, R.E.; Johnson, J.D. Optimization of neural networks: A comparative analysis of the genetic algorithm and simulated annealing.* European Journal of Operational Research, **1999**, 114(3), *589-601.*

179. *Maehara, N.; Shimoda, Y. Application of the genetic algorithm and downhill simplex methods (Nelder–Mead methods) in the search for the optimum chiller configuration.* Applied Thermal Engineering, **2013**, 61(2), *433-442.*
180. *Huang, G.B.; Chen, L. Enhanced random search based incremental extreme learning machine.* Neurocomputing, **2008**, 71(16-18), 3460-3468.
181. *Pontes, F.J.; Amorim, G.F.; Balestrassi, P.P.; Paiva, A.P.; Ferreira, J.R. Design of experiments and focused grid search for neural network parameter optimization.* Neurocomputing, **2016**, 186, *22-34.*
182. *Farfán, J.F.; Cea, L. Improving the predictive skills of hydrological models using a combinatorial optimization algorithm and artificial neural networks.* Model. Earth Syst. Environ., **2022**.
183. *Zerubia, J.; Chellappa, R. Mean field annealing using compound Gauss-Markov random fields for edge detection and image estimation.* IEEE Transactions on Neural Networks, **1993**, 4(4), *703-709.*
184. *Ihme, M.; Marsden, A.L.; Pitsch, H. Generation of Optimal Artificial Neural Networks Using a Pattern Search Algorithm: Application to Approximation of Chemical Systems.* Neural Computation, **2008**, 20(2), *573–601.*
185. *Vilovic, I.; Burum, N.; Sipus, Z. Design of an Indoor Wireless Network with Neural Prediction Model. The Second European Conference on Antennas and Propagation, EuCAP 2007, pp. 1-5, 2007.*
186. *Bagherbeik, M.; Ashtari, P.; Mousavi, S.F.; Kanda, K.; Tamura, H.; Sheikholeslami, A. A Permutational Boltzmann Machine with Parallel Tempering for Solving Combinatorial Optimization Problems.* Lecture Notes in Computer Science, **2020**, 12269, *317–331.*
187. *Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization.* Swarm Intell, **2007**, 1, *33–57.*
188. *Wang, Q.; Perc, M.; Duan, Z.; Chen, G. Delay-enhanced coherence of spiral waves in noisy Hodgkin–Huxley neuronal networks.* Physics Letters A, **2008**, 372(35), *5681-5687.*
189. *Fernandes Jr, F.E.; Yen, G.G. Pruning deep convolutional neural networks architectures with evolution strategy.* Information Sciences, **2021**, 552, *29-47.*
190. *Cho, H.; Kim, Y.; Lee, E.; Choi, D.; Lee, Y.; Rhee, W. Basic Enhancement Strategies When Using Bayesian Optimization for Hyperparameter Tuning of Deep Neural Networks.* IEEE Access, **2020**, 8, *52588-52608.*
191. *Pauli, P.; Koch, A.; Berberich, J.; Kohler, P.; Allgöwer, F. Training Robust Neural Networks Using Lipschitz Bounds.* IEEE Control Systems Letters, **2022**, 6, *121-126.*
192. *Rong, G.; Li, K.; Su, Y.; Tong, Z.; Liu, X.; Zhang, J.; Zhang, Y.; Li, T. Comparison of Tree-Structured Parzen Estimator Optimization in Three Typical Neural Network Models for Landslide Susceptibility Assessment.* Remote Sens., **2021**, 13, *4694.*
193. *Chen, Y.; Chang, H.; Meng, J.; Zhang, F. Ensemble Neural Networks (ENN): A gradient-free stochastic method.* Neural Networks, **2019**, 110, *170-185.*
194. *Yang, X.-J.* General Fractional Derivatives. Theory, Methods and Applications*; Boca Raton : CRC Press, Taylor and Francis Group, 2019.*
195. *Wang, J.; Wen, Y.; Gou, Y.; Ye, Z.; Chen, H. Fractional-order gradient descent learning of BP neural networks with Caputo derivative.* Neural Networks,**2017**, 89, *19-30.*
196. *Sales Teodoro, G.; Tenreiro Machado, J.A.; Capelas de Oliveira, E. A review of definitions of fractional derivatives and other operators.* Journal of Computational Physics, **2019**, 388, *195-208.*
197. *Louati, H.; Bechikh, S.; Louati, A.; Hung, C.C.; Said, L.B. Deep convolutional neural network architecture design as a bi-level optimization problem.* Neurocomputing, **2021**, 439, *44-62.*
198. *Yang, J.; Ji, K.; Liang, Y. Provably Faster Algorithms for Bilevel Optimization.* Part of Advances in Neural Information Processing Systems 34, **2021**.
199. *Hong, M.; Wai, H.T.; Wang, Z.; Yang, Z. A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic*, arXiv, **2020**, *arXiv:2007.05170.*
200. *Khanduri, P.; Zeng, S.; Hong, M.; Wai, H.-T.; Wang, Z.;Yang, Z. A Near-Optimal Algorithm for Stochastic Bilevel Optimization via Double-Momentum*, Part of Advances in Neural Information Processing Systems 34, **2021**.
201. *Grazzi, R.; Franceschi, L.; Pontil, M.; Salzo, S. On the iteration complexity of hypergradient computation. In International Conference on Machine Learning (ICML), pp. 3748–3758, 2020.*
202. *Sow, D.; Ji, K.; Liang, Y. Es-based jacobian enables faster bilevel optimization*, arXiv **2021** *arXiv:2110.07004.*
203. *Ji, K.; Yang, J.; Liang, Y. Bilevel Optimization: Convergence Analysis and Enhanced Design.* Proceedings of the 38th International Conference on Machine Learning, PMLR, **2021**, 139, *4882-4892.*
204. *Pang, G.; Lu, L.; Karniadakis, G.E. fPINNs: Fractional physics-informed neural networks.* SIAM Journal on Scientific Computing, **2019**, 41(4), *2603-2626.*

205. *Gupta, V.; Koren, T.; Singer, Y. Shampoo: Preconditioned Stochastic Tensor Optimization.* Proceedings of Machine Learning Research, **2018**, 80, *1842-1850.*
206. *Henderson, M.; Shakya, S.; Pradhan, S. et al. Quanvolutional neural networks: powering image recognition with quantum circuits.* Quantum Mach. Intell., **2020**, 2, *2.*
207. *Guo, Y.; Liu, M.; Yang, T.; Rosing, T. Improved Schemes for Episodic Memory-based Lifelong Learning.* Part of Advances in Neural Information Processing Systems 33, **2020**.
208. *Zhang, D.; Liu, L.; Wei, Q.; Yang, Y.; Yang, P.; Liu, Q. Neighborhood Aggregation Collaborative Filtering Based on Knowledge Graph.* Appl. Sci., **2020**, 10, *3818.*
209. *Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications.* AI Open, **2020**, 1, *57-81.*
210. *Wang, G.; Deb, S.; Cui, Z. Monarch butterfly optimization.* Neural Comput and Applic, **2019**, 31, *1995–2014.*