SciencePG
Science Publishing Group

Research and Application of Rectified-NAdam Optimization Algorithm in Data Classification

Zhu Zhixuan, Hou Zaien[*]

School of Mathematics and Data Science, Shanxi University of Science & Technology, Xi'an, China

Email address:
houze@sust.edu.cn (Hou Zaien)
[*]Corresponding author

Abstract: Data classification exists in various practical applications, such as the classification of words in natural language processing, classification of meteorological conditions, classification of environmental pollution degree, and so on. Artificial neural network is a basic method of data classification. A reasonable optimization algorithm will get better results for a loss function in the neural network. The research and improvement of these optimization algorithms has been a focus in this field. Because of the various optimizers developing in building the neural networks, an improved NAdam Algorithm (RNAdam) is proposed in this paper, on the basis of discussing and comparing several Algorithms with Adam Algorithm. This algorithm not only combines the advantages of RAdam algorithm, but also keeps the convergence of NAdam algorithm. A classification experiment is carried out on the data set composed of 300 sample points generated by the Make moon function. The experimental results show that the RNAdam algorithm is better than SGDM, Adam and Nadam algorithm in terms of the loss and accuracy between the output and the actual results, when the data are classified by the three-layer neural network. Therefore, the classification effect will be improved when this algorithm is applied to neural network for various practical data classification problems.

Keywords: Data Classification, Artificial Neural Network, Optimization Algorithm, Loss Function

1. Introduction

In the process of data classification with Artificial neural network, selecting an appropriate optimization algorithm to improve the loss function of neural network is an effective way to improve the classification in accuracy. Stochastic gradient descent method (SGDM) is one of the most commonly used optimization algorithm. Although it is easy to jump out of the singularity, but the path is sawtooth when the target value is small, resulting in slow convergence. It can be intuitively observed in the research that the adjustment of step size is very important in calculating the loss function. In practice, the desired optimization algorithm is that the step size can be longer in a flat place and smaller in a steep place. Kingma D [1] proposed Adam algorithm in combination with momentum and rmsprop. It can dynamically adjust the learning rate while using momentum. At present, it is one of the most commonly used optimization algorithm for loss

function in the neural networks. Sutskever I [2] show that Nesterov's accelerated gradient (NAG) algorithm performed better in some gradient descent cases. Dozat T [3] considered Nesterov's viewpoint and modified the momentum component of Adam algorithm to obtain the modified Adam algorithm (NAdam). It is found that this can improve the quality of learning model and improve the convergence speed. Since warmup is originally proposed to handle gradient variance for SGD algorithm [4-6]. In order to avoid getting local optimal results, the thermal heuristic method is used in Adam algorithm and its derivative algorithm. It uses a small learning rate in the initial stage of the algorithm, then gradually increases, reaches stability, and then slowly decreases [7, 8]. Although warmup has achieved good performance in the experimental results, it lacks certain theoretical support in previous studies. Liu L [9] studied the variance of adaptive

learning rate of Adam algorithm by using the method that exponential mean distribution (EMA) can be approximated to simple mean distribution (SMA) in economics. While presenting a theoretical proof for warmup, they proposed an Adam algorithm with improved adaptive learning rate (RAdam). Based on the discussion of the above random gradient descent method (SGD), Adam algorithm and improved Adam algorithm (NAdam, RAdam), this paper integrates the latter two improved algorithms, proposes an improved optimization algorithm (RNAdam) for loss function in the neural network, and compares the classification effect of the obtained algorithm with the above algorithm through an example of data set classification.

2. Classification Problem and Its Neural Network

2.1. Classification Problem

The data set to be classified in this paper is made of the data set (X, Y) of 300 sample points generated by the make_moon function, a simple dataset to visualize clustering and classification algorithms, in which X is composed of two columns which correspond to two coordinate of the point on the plane, Y is a column composed of two numbers 0 and 1, respectively indicating the label that the corresponding points belong to different type. The data (semi ring graph) generated by the make_moon function is shown in Figure 1 below.

2.2. The Neural Network

In this paper, three-layer neural network is selected for the data classification. The structure diagram of neural network is shown in Figure 2.
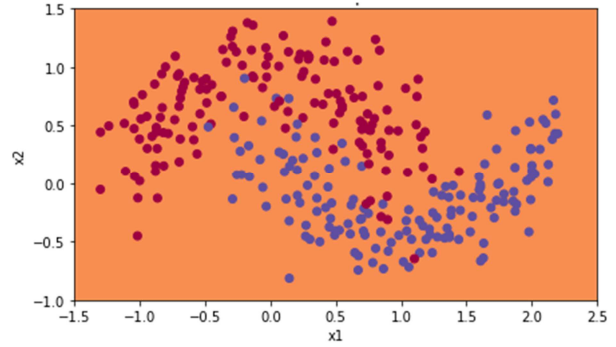

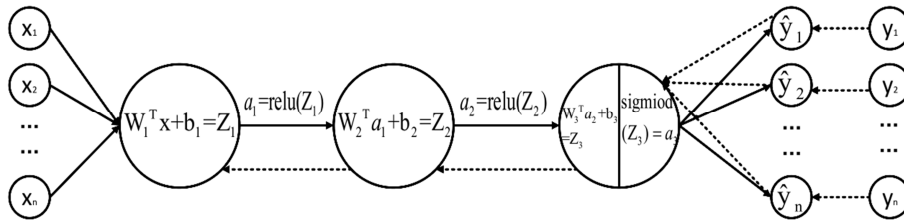**Figure 1.** *Diagram of red and blue data sets.*

Figure 2. *Three layer neural network structure diagram.*

The three-layer neural network structure formula used here are as fellows.

$$Z_1 = W_1^{\mathrm{T}} x + b_1$$

$$a_1 = relu(Z_1)$$

$$Z_2 = W_2^{\mathrm{T}} a_1 + b_2$$

$$a_2 = relu(Z_2)$$

$$Z_3 = W_3^{\mathrm{T}} a_2 + b_3$$

$$a_3 = sigmoid(Z_3)$$

The loss function is given by the cross entropy cost, which is defined as fellows.

$$J = -\frac{1}{n}\sum_x (y \ln a_3 + (1 - y)\ln(1 - a_3))$$

Where x is the input sample, $W_1, W_2, W_3, b_1, b_2, b_3$ are the parameters, relu and sigmoid are the activation function, a_3 is the output result, Y is the actual result and J is the loss function.

Derivatives of loss function J are as fellows.

$$\frac{\partial J}{\partial a_3} = -\frac{1}{n}\sum_x \left(\frac{y}{a_3} - \frac{1-y}{1-a_3}\right),$$

$$\frac{\partial J}{\partial Z_3} = \frac{\partial J}{\partial a_3} a_3(1 - a_3),$$

$$\frac{\partial J}{\partial a_L} = \frac{\partial J}{\partial Z_L} W_L^{\mathrm{T}}, L = 1,2$$

$$\frac{\partial J}{\partial Z_m} = \begin{cases} 0, a_m \le 0 \\ \frac{\partial J}{\partial a_m}, \ a_m > 0 \end{cases}, m = 1,2$$

$$\frac{\partial J}{\partial W_{L+1}} = \frac{\partial J}{\partial Z_{L+1}} a_L, L = 0,1,2, a_0 = x$$

$$\frac{\partial J}{\partial b_L} = \frac{\partial J}{\partial Z_L} \ L = 1,2,3$$

After the loss function is obtained, the optimization algorithm is used to solve the optimization problem $\min J$.

Thus, various suitable parameters are obtained, and then the parameters are brought back to the formula to obtain the loss value, so as to obtain the classification accuracy of the obtained artificial neural network.

3. Introduction to Optimization Algorithm

As mentioned above, this paper solves a class of unconstrained programming problems $\min f$

Let's discuss the optimization algorithm with f as the objective function of the optimization problem. In the later experiments, take $f = J$.

3.1. SGDM and NAG Algorithm

The stochastic gradient descent method (SGDM) with momentum is obtained by adding a first-order momentum to the stochastic gradient descent method, which can make the whole optimization process drop more in the place with large gradient and restrain the swing. The algorithm is as follows.

$$g_t \leftarrow \nabla_\theta f_t(\theta_t)$$
$$m_t \leftarrow \beta \cdot m_{t-1} + (1 - \beta) \cdot g_t$$
$$\theta_{t+1} \leftarrow \theta_t - \alpha_t m_t$$

Where t is the number of iterations, α_t is the learning rate, m_t is the updated biased first order, β is usually set to 0.9.

The NAG algorithm combines SGD with nesterov. In the first step of derivation, it takes the last gradient change into account, which is equivalent to adding an approximate second-order derivative of the objective function. It has a correction effect in the calculation of the update direction. To a certain extent, it can make the calculation more accurate and converge faster. This predictive update is of great significance for the performance improvement of RNN [10]. This optimization method has faster convergence speed, which uses the temporarily updated gradient according to the historical gradient and directly jumps one more step to the next step. The algorithm is as follows.

$$g_t \leftarrow \nabla_\theta f_t(\theta_t - \alpha_t \cdot \beta \cdot m_{t-1})$$
$$m_t \leftarrow \beta \cdot m_{t-1} + (1 - \beta) \cdot g_t$$
$$\theta_{t+1} \leftarrow \theta_t - \alpha_t m_t$$

Where t is the number of iterations, α_t is the learning rate, m_t is the updated biased first order derivative, β is usually set to 0.9.

Then make $x_t = \theta_t - \alpha_t \cdot \beta \cdot m_{t-1}$, NAG can be changed to as fellows.

$$g_t \leftarrow \nabla_\theta f_t(x_t)$$
$$m_t \leftarrow \beta \cdot m_{t-1} + (1 - \beta) \cdot g_t$$
$$\theta_{t+1} \leftarrow \theta_t - \alpha_t m_t$$
$$\theta_{t+1} \leftarrow x_t - \alpha_t(1 - \beta) \cdot g_t$$
$$x_{t+1} \leftarrow x_t - \alpha_t(1 - \beta) \cdot g_t - \alpha_t \cdot \beta \cdot m_t$$

Expansion with SGD $\theta_{t+1} = \theta_t - \alpha_t(\beta \cdot m_{t-1} + (1 - \beta) \cdot g_t)$ In contrast, m_{t-1} is replaced by m_t.

3.2. Adam and RAdam Algorithm

Adam algorithm absorbs the momentum method, and its algorithm is as follows.

$$g_t \leftarrow \nabla_\theta f_t(\theta_t)$$
$$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$
$$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$
$$\widehat{m_t} \leftarrow m_t/(1 - \beta_1^t)$$
$$\widehat{v_t} \leftarrow v_t/(1 - \beta_2^t)$$
$$\theta_{t+1} \leftarrow \theta_t - \alpha_t \widehat{m_t}/(\sqrt{\widehat{v_t}} + \epsilon)$$

Where t is the number of iterations, α_t is the learning rate, m_t, v_t is the updated biased first-order and second-order moment estimation respectively $\widehat{m_t}$ is the first-order moment estimation of deviation correction, $\widehat{v_t}$ is the second original moment estimation of deviation correction, ϵ is normally set to 1×10^{-8}, β_1 is usually set to 0.9, β_2 is usually set to 0.999.

Adam's learning rate is controlled by second-order momentum, which can not guarantee the monotonic decline of learning rate, so the model may not converge in some cases [11]. Robert Nau [12] found through experiments that reducing the variance of adaptive learning rate can solve the convergence problem. They proposed that exponential moving average can be approximated as simple moving average, while the calculation method of v_t is the iterative form of exponential moving average, so Liu L [9] and others use this to approximate $\frac{1}{\sqrt{\widehat{v_t}}} = \sqrt{\frac{1-\beta_2^t}{(1-\beta_2)\Sigma_{i=1}^t \beta_2^{t-i} g_t^2}}$ to $\sqrt{\frac{t}{\Sigma_{i=1}^t g_i^2}}$, so as to obtain $\text{Var}\left[\frac{1}{\sqrt{\widehat{v_t}}}\right]$ with degrees of freedom ρ it decreases monotonically at the speed of $O\left(\frac{1}{\rho_t}\right)$. The first-order approximation is used to calculate the correction term [13], calculated $\rho_t \leq \rho_\infty = \lim_{t \to \infty} \rho_t = \frac{2}{1-\beta_2} - 1, r_t = \sqrt{\frac{(\rho_t-4)(\rho_t-2)\rho_\infty}{(\rho_\infty-4)(\rho_\infty-2)\rho_t}}$. Because $\rho_t \leq 4$, r_t may have an open negative number, so he replaced the algorithm at that time with SGDM. The RAdam algorithm is as follows.

$$g_t \leftarrow \nabla_\theta f_t(\theta_t)$$
$$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$
$$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$
$$\widehat{m_t} \leftarrow m_t/(1 - \beta_1^t)$$
$$\rho_t \leftarrow \rho_\infty - \frac{2t\beta_2^t}{1-\beta_2^t}$$

When $\rho_t > 4$

$$\widehat{v_t} \leftarrow v_t/(1 - \beta_2^t)$$
$$r_t \leftarrow \sqrt{\frac{(\rho_t-4)(\rho_t-2)\rho_\infty}{(\rho_\infty-4)(\rho_\infty-2)\rho_t}}$$

$$\theta_{t+1} \leftarrow \theta_t - \alpha_t r_t \widehat{m_t}/\sqrt{\widehat{v_t}}$$

Other

$$\theta_{t+1} \leftarrow \theta_t - \alpha_t \widehat{m_t}$$

$$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$$

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

$$\widehat{m_t} \leftarrow \frac{\beta_1 \cdot m_t}{1-\beta_1^{t+1}} + \frac{(1-\beta_1) \cdot g_t}{1-\beta_1^t}$$

$$\rho_t \leftarrow \rho_\infty - \frac{2t\beta_2^t}{1-\beta_2^t}$$

When $\rho_t > 4$

$$\widehat{v_t} \leftarrow v_t/(1-\beta_2^t)$$

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

$$\theta_{t+1} \leftarrow \theta_t - \frac{\alpha_t r_t \widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon}$$

Other

$$\theta_{t+1} \leftarrow \theta_t - \alpha_t m_t/(1-\beta_1^t)$$

4. Rectified-Nadam Algorithm

Referring to the previous practice of adding NAG to Adam and using the current Nesterov momentum vector to replace the traditional momentum vector in Adam, Nadam is obtained. Here, we will try to combine NAdam and RAdam.

Due to $\rho_t \leftarrow \rho_\infty - \frac{2t\beta_2^t}{1-\beta_2^t}$, it is used in this algorithm $\beta_2 = 0.999$, when $t \geq 3$, $\rho_t > 4$. Therefore, under this setting in RAdam, only SGDM is used in the first two steps, which has little impact on the global situation. This setting is also used here. That is $lim_{t\to\infty} r_t = \sqrt{\frac{(\rho_t-4)(\rho_t-2)\rho_\infty}{(\rho_\infty-4)(\rho_\infty-2)\rho_t}} = 1$, when t tends to infinity, the effect of RAdam is similar to Adam, but it solves the problem of convergence through the analysis of square difference. Since changing the traditional momentum to Nesterov momentum will not affect the original convergence in Adam, the improvement of RAdam is also applicable to Nadam and can make it converge.

The algorithm Rectified-Nadam (RNAdam) is as follows.

$$g_t \leftarrow \nabla_\theta f_t(\theta_t)$$

5. Test Results

The data set are classified by the three-layer neural network with SGDM and RNAdam algorithm as shown in Figure 3 below. After 9000 iterations, the error value and accuracy between the output results and the actual results are obtained, and recorded in Table 1 and figure 4 for comparison.
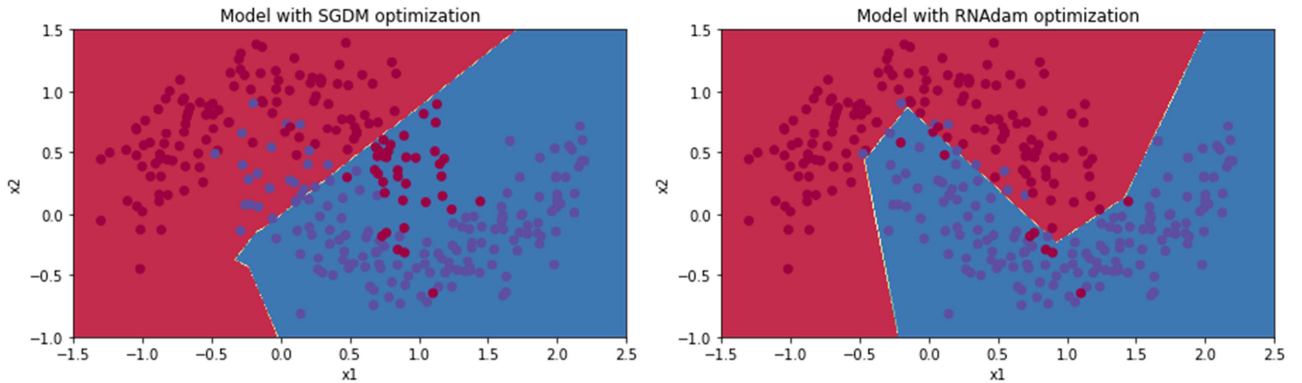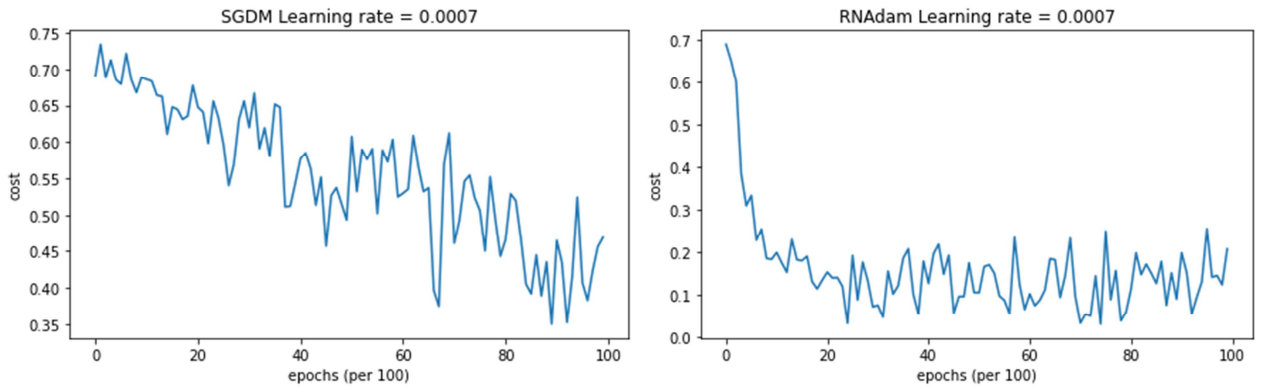

Figure 3. *Data distribution display.*

Figure 4. *Iterative results of four algorithms.*

Table 1. *Comparison of error and accuracy under various conditions.*

	SGDM	Adam	NAdam	RNAdam
cost	0.4647395967	0.1979400715	0.1977770164	0.1977480201
Accuracy	0.7966666666	0.94	0.94	0.94

6. Conclusion

This algorithm uses Nadam algorithm combined with a thermal heuristic and mathematical nterpretation proposed by RAdam to obtain an improved Nadam algorithm, which not only combines the advantages of rapid descent speed of radam algorithm, but also has the properties of Nadam. The application of this algorithm in natural language word classification will shorten the pre-processing time of natural language processing. In practical application, due to the high accuracy of the model, it can avoid over-fitting and get the best effect by adjusting the times of iteration before using.

Acknowledgements

References

[1] Kingma D, Ba J. Adam: A Method for Stochastic Optimization [J]. Computer Science, 2014.

[2] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Proceedings of the 30th International Conference on Machine Learning (ICML-13), pp. 1139–1147, 2013.

[3] Dozat T. Incorporating Nesterov Momentum into Adam. 2016.

[4] Goyal P, P Dollár, Girshick R, et al. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour [J]. 2017.

[5] Gotmare A, Keskar N S, Xiong C, et al. A Closer Look at Deep Learning Heuristics: Learning rate restarts, Warmup and Distillation [J]. 2018.

[6] Xiao L, Yu A W, Lin Q, et al. DSCOVR: Randomized Primal-Dual Block Coordinate Algorithms for Asynchronous Distributed Optimization [J]. 2017.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pp. 5998–6008, 2017.

[8] Popel M, Bojar O. Training Tips for the Transformer Model [J]. Prague Bulletin of Mathematical Linguistics, 2018, 110 (1): 43-70.

[9] Liu L, Jiang H, He P, et al. On the Variance of the Adaptive Learning Rate and Beyond [J]. 2019.

[10] Bengio, Y., Boulanger-Lewandowski, N., & Pascanu, R. (2012). Advances in Optimizing Recurrent Networks. Retrieved from http://arxiv.org/abs/1212.0901.

[11] Reddi S J, Kale S, Kumar S. On the Convergence of Adam and Beyond [J]. 2019.

[12] Robert Nau. Forecasting with moving averages. 2014.

[13] Wolter K M. Introduction to variance estimation | Clc [M].