**Nguyễn Anh Tuấn - 521H0326**

**Bài 1:**

1/

- As far as I know, there are 7 Optimizer methods in training machine learning models:

1. **Gradient Descent (GD).**

**-** Is the most basic method used to adjust parameters to iterate information through each training data to minimize the cost function.

- For example: When we are lost on a mountain and there is dense fog, we can only feel the slope of the ground. The best way to get to the bottom of the mountain quickly is to follow it down in the steepest direction. This is the idea that GD wants to do, at each point of the function, it will determine the slope and then go against the direction of the slope until the slope at that place is 0 (minimum).  
- Gradient Descent is an iterative optimization algorithm used in ML and DL projects with the goal of finding a set of internal variables for optimizing models.- GD có nhiều dạng khác nhau như Stochastic Gardient Descent (SGD), mini-batch SDG. But basically, they are all implemented under one basic standard as follows:

* Initialize internal variable.
* Evaluate the model based on internal variables in the direction of optimizing the loss function.
* Update internal variables towards finding optimal points.
* Repeat steps 2 and 3 until the stopping condition.

- The updated formula for GD can be written as:



* **Advantages**

- Basic GD algorithm, easy to understand. The algorithm has solved the problem of optimizing the neural network model by updating the loop weights.

* **Disadvantages**

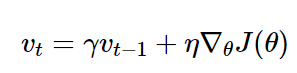
- Because of its simplicity, the gradient Descent algorithm has many limitations such as depending on the initial initialization experiment and learning rate.  
- For example, a function with 2 global minimums, depending on the 2 initial initial points, will produce 2 different final solutions.

- A learning rate that is too large will cause the algorithm to fail to converge and hover around the target because the jump is too large; or the small learning rate affects the training speed

1. **Momentum.**

- Momemtum technique is used in combination with Gradient Descent to increase the convergence speed of the algorithm.

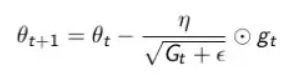
- In GD we need to calculate the amount of change at time t to update the new position for the solution. If we think of this quantity as velocity  in physics, the marble's new position would be . The minus sign represents having to move against the derivative. Our job now is to be generous    
so that it both carries information about the slope and information about the momentum, that is, the previous velocity(We consider the initial velocity  ). In the simplest way, we can add (weighted) these two quantities:



- This algorithm proves to be very effective in practical problems (in high-dimensional space, the calculation method is completely similar).

1. **AdaGrad (Adaptive Gradient Algorithm).**

- Unlike previous algorithms where the learning rate is almost the same during the training process (learning rate is constant), Adagrad considers learning rate as a parameter. That is, Adagrad will let the learning rate change after each time t.



In there:

n: constant.

gt: gradient at time t.

ϵ: error avoidance factor (divided by sample equals 0).

G: is a diagonal matrix where each element on the diagonal (i,i) is the square of the parameter vector derivative at time t.

* **Advantages:**

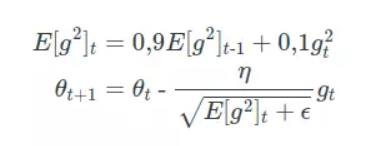
- An obvious benefit of Adagrad is to avoid adjusting the learning rate manually, just set the default learning rate to 0.01 and the algorithm will automatically adjust.

* **Disadvantages:**

- The weakness of Adagrad is that the sum of squared variations will grow larger over time until it makes the learning rate extremely small, causing training to freeze.

1. **RMSprop (Root Mean Square Propagation).**

- RMSprop solves Adagrad's decreasing learning rate problem by dividing the learning rate by the average of the squares of the gradient.



* **Advantages:**

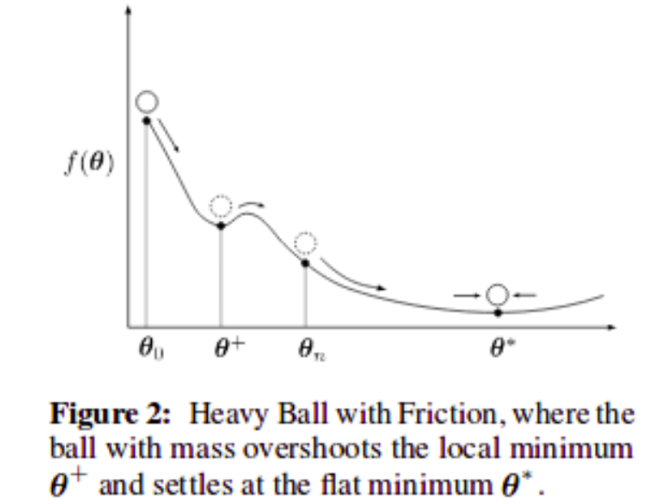
- The most obvious advantage of RMSprop is that it solves the problem of Adagrad's gradually decreasing learning speed (the problem of gradually decreasing learning speed over time will cause training to slow down, possibly leading to freezing).

* **Disadvantages:**

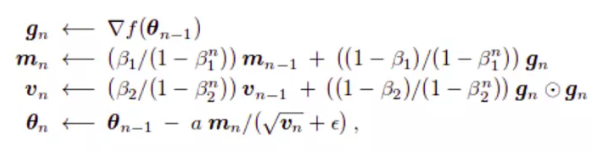
- The RMSprop algorithm can only give a solution that is local minimum but not global minimum like Momentum. Therefore, people will combine both Momentum algorithms with RMSprop to create an Adam optimal algorithm. We will present it in the following section.

1. **Adam (Adaptive Moment Estimation).**

- As said above Adam is a combination of Momentum and RMSprop. If explained in terms of physical phenomena, Momentum is like a ball rushing downhill, and Adam is like a very heavy ball with friction, so it easily overcomes the local minimum to the global minimum and when it reaches the global minimum it does not. It takes a long time to oscillate back and forth around the target because it has friction so it's easier to stop.



**- Recipe:**



* **Advantages:**

- Suit to Adaptive.

- Good Performance on Many Problems.

- Easy to use.  
- Good Performance for Big Data.

* **Disadvantages:**

- Sensitive to Noise.

- High Memory Requirements.

- Not Universally Optimal.

- Hyperparameter Sensitivity.

1. **Adadelta.**

- Adadelta was born to reduce the disadvantage of Adagrad (changing learning rate with decreasing monotony).  
- It limits the accumulation of variability to a certain limit

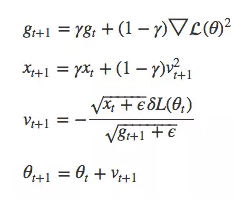
- To do the above, it introduces the concept that running average depends on the previous average and the current slope.



- The gramma here is similar to the momentum of approximately 0.9.   
- It also gives the second moment which is the square of the update parameter



**- Recipe summary:**



1. **Nesterov Accelerated Gradient (NAG)**

- Nesterov Accelerated Gradient (NAG) is an optimization algorithm that builds upon the traditional gradient descent method by incorporating a "look-ahead" feature. NAG is particularly effective in increasing the convergence rate, especially in the presence of high curvature or noisy gradients.

- Initialize model parameters, including weights and biases.

Set learning rate (α), momentum term (β), and other hyperparameters.  
- Initialize the velocity term ( v) zero for all parameters.

* At each iteration t:
* Calculate gradients gt of the loss function with respect to the parameters.
* Update the velocity term by combining the current gradient and the previous velocity:



* Update the parameters with the adjusted gradient, taking into account the "look ahead" feature:



- During the update step, the term modifies β.vt−1 is subtracted from the current gradient term. This modification helps predict the future direction of the gradient, helping the algorithm make more informed updates.

- The Nesterov Accelerated Gradient algorithm has been shown to converge faster than traditional gradient descent, especially in situations where the cost function is highly curved. It is widely used in practice and is a popular optimization algorithm in the field of machine learning.

2/

* **Continual Learning**

- Continuous Learning, also known as lifelong learning or incremental learning, is a machine learning model that focuses on training models to continuously learn and adapt over time, integrating new information while still retain knowledge from previous tasks. This is in contrast to traditional machine learning methods, where the model is trained on a fixed data set and does not handle new information well.

- Several techniques and methods have been proposed to address the challenges of continuous learning:

1. **Regularization Methods:**

- **Elastic Weight Consolidation (EWC)**: This method introduces a regularization term to penalize changes of importance weights learned in previous tasks, helping to retain knowledge.

1. **Rehearsal Techniques:**

- **Experience Replay:** The model is periodically trained on a mixture of old and new data, allowing it to remember and consolidate knowledge from previous tasks.

- **Generative Replay:** Instead of storing original data, a generative model should be used to generate synthetic examples for previous tasks during training.

1. **Architectural Approaches:**

- **Progressive Neural Networks (PNN):** The model architecture is extended to accommodate new tasks without affecting knowledge learned from previous tasks.

- **Lifelong Learning Machines (L2M):** This method involves creating a hierarchy of models, with each level devoted to learning different tasks.

1. **Dynamic Architectures:**

**- Elastic Nets:** Dynamically scaling model architecture to adapt to new tasks while minimizing interference with previously learned tasks.

**- Neural Module Networks (NMN):** The model is structured into modules that can be dynamically added or removed for learning new tasks.

1. **Knowledge Distillation:**

- Knowledge Distillation for Continuous Learning (KDCL): This method involves extracting knowledge from a teacher model that has learned many tasks to a student model that is learning a new task.

1. **Task Prioritization and Scheduling:**

- Meta-Gradient Descent: The learning algorithm treats itself as a meta-learning problem, allowing the model to adapt its learning strategy based on the task at hand.

- Summary: Continuous Learning is an important aspect of developing AI systems that can operate in dynamic environments and handle data distributions that change over time. Ongoing research is aiming to address various challenges related to continuous learning and improve the adaptability of machine learning models over time.

* **Test Production**