Neural Networks: synthesis of back-propagation optimisation problems

Pieraut Francis, Caporossi Gilles, Bengio Yoshua

***Abstract*— Ours investigations to understand the reasons why huge neural networks seems to not be able to take advantage of their capacity to get much better learning result compare to smaller one, bring use to look at optimisation problems related to back-propagation algorithm. This article presents a synthesis of the optimisation problems of neural network using back-propagation for learning.**

***Index Terms*— Neural Networks, back-propagation, optimisation problems.**

I. Introduction

Beyond a certain capacity, experimental results show us that even if you increase the number of free parameters of a neural network, you won't get much better result on training error. Theoretically, a much higher number of free parameters should allow you to represent much complex function and get better result on training dataset. As mention earlier, it is not what we see in practice so the problem is not that we have not enough capacity but we think that it is related to optimisation problems.

II. Back-Propagation problems

Back-propagation is the most common learning algorithm to train neural networks. In the literature, we have finds three limitations of back-propagation: *the step size problem*, *the moving target problem*, *the attenuation and dilution of error signal* as it is propagates backward through the layers of the network. In our investigation, we have identify three others limitations: *the opposite gradient problem*, *the non-existence of specialisation parameters mechanism* and the *symmetry problem*. For convenience reasons, we gather these six problems under the term of *back-propagation problems*. Those six problems will be explain separately in the next sub-sections.

A. Step size problem

In standard back-propagation, we use a single learning rate or step size for all parameters (see algorithm 1). In theory, with a different learning rate for each parameters, the number of iterations for learning process can be much smaller. If all parameters are uncorrelated and if the cost function is a parabola, it will be possible to find the appropriate learning rate for each parameters to converge to the optimal solution in one step (Newton Algorithm).
Knowing that cost curve is not the same for each parameters, having a different step size for each parameters can insure constant learning speed for each of them. Unfortunately, a significant learning speedup of such an algorithm have never been demonstrated. Without second order derivative information, it is easier to chose a single learning rate for all parameters but step size problem appear.

Algorithm (stochastic back-propagation)

Initialisation n (learning rate), Θ, e=0, m=0
Faire e=e+1 (epoches)
  do $m \leftarrow m+1$ (all examples)
  x^m =randomly chosen pattern
  propagate inputs values
  compute δ_k related to cost function
  compute $\nabla w_{jk} = \delta_k \cdot y_j$
  compute $\nabla \delta_j ; \delta_j = [\sum_{k=1}^{c} w_{kj} \cdot \delta_k] \cdot g'_y(y_j)$
  compute $\nabla w_{ij} = \delta_j \cdot x_i$
  $w_{ij} \leftarrow w_{ij} - n \cdot \nabla w_{ij}$
  $w_{jk} \leftarrow w_{jk} - n \cdot \nabla w_{jk}$
  until $\|\nabla J\| < \Theta$

Fig. 1. Stochastic back-propagation algorithm

B. Moving target problem

Gradients for each parameters are compute independently of each other. Cost is not recompute each time we apply a single gradient. So, it is the same as if the target value is changing for each gradient computing. This problem limit of the optimisation process and become more important as the number of parameters increase. One common manifestation of *the moving target problem* is what we call the *herd effect*. Suppose we have 2 separate computational sub-tasks and knowing that units cannot communicate with one another, each unit must decide independently which of the two problem it will tackle.

C. Attenuation and dilution of error signal

Weights initialisation between outputs and hidden neurons are chosen according to the number of hidden units. Their random values are chosen from an uniform distribution of mean 0 and variance $h^{-1/2}$ (eq 1). A higher number of hidden units lead error signal to be diluted and attenuated.

$$w_{kj}\ U(0, h^{-1/2}) \tag{1}$$

D. Opposite gradient problem

This problem is happening in classification task. During computation of sensibility factor (eq 2) associate to each hidden neuron, contributions of outputs neurons can be opposite or of the same sign. This problems can be compare as increasing or decreasing learning rate associate to each

inputs weights of hidden neurons. Moreover, much higher is the number of outputs, higher is this problem. Knowing that we use only first order derivative, using a learning rate to small or to high can only reduce learning process speed.

$$\delta_j = [\sum_{k=1}^{c} w_{kj} \cdot \delta_k] \cdot g'_y(y_j) \tag{2}$$

E. Non-existence of specialisation parameters mechanism

Usually we back-propagate error on all parameters. If we are not doing online training and we want to do back-propagation only on some hidden neurones, we can keep outputs of fixed ones to reduce forward propagation time. Without any mechanism to specialise parameters, no parameters are fixed and forward propagation time cannot be reduce. Having a specialisation mechanism can be also use to apply divide and conquer technics. Instead of solving a complex problem, we can split it up in several smaller problems. Without specialisation mechanism we cannot specialise parameters, speedup iteration time and apply divide and conquer. This technics can help us to speed up learning process.

F. Symmetry problem

Before training a neural network, all parameters are initialised randomly and during training, all gradients are computed independently of each other. Those remarks indicate to us that it is possible for some neurons to learn the same thing. When this phenomena happen, parameters space is not really well exploited and many parameters can be remove without changing anything.

III. CONCLUSION

The goal of this paper was to give a synthesis of back-propagation problems. Six different problems have been identify :*the step size problem*, *the moving target problem*, *the attenuation and dilution of error signal*, *the opposite gradient problem*, *the non-existence of specialisation parameters mechanism* and *the symmetry problem*. For convenience reasons, we gather these six problems under the term of *back-propagation problems*. For convenience reasons, we gather these six problems under the term of *back-propagation problems*. Those problems introduce optimisation problem when using back-propagation in neural networks.