Misconception on the Regularization Effect of Noise or Fault Injection : Theoretical Analysis

John Sum
Institute of Technology Management, National Chung Hsing Univesity
Taichung 40227, Taiwan {pfsum@nchu.edu.tw}

***Abstract*—Adding noise or fault during training has been a method to attain a neural network with better generalization. Yet, it is still unclear why it works. Some scholars have confused that the learning objective of adding noise or fault during the gradient descent learning is the desired measure – the expected mean square error (MSE) of the model with such noise or fault. Subsequently, the desired measure is used to interpret the regularization effect of noise injection. The purpose of this paper, together with an companion paper [1] is to clarify this misconception. It is shown that their equivalency depends on three factors: (i) the model of the neural network, (ii) the noise or fault model and (iii) the learning algorithm. They are equivalent if random weight fault is injected during gradient descent learning applying on either a MLP or RBF. If additive (resp. multiplicative) node noise is injected during gradient descent learning, the objective might not be the desired measure.**

***Index Terms*—Additive Node Noise, Multiplicative Node Noise, Regularization, Weight Fault.**

I. Introduction

Noise or fault injection is a classical method to improve the generalization of a neural network [2]–[7]. Various researches were then conducted to investigate the effect of such noise/fault on the performance of a neural network [8]–[11]. Learning algorithms were developed to synthesize a neural network that is able to tolerate such noise/fault [12]–[14]. The convergence properties, the learning objective functions and the regularization effects of applying gradient descent learning to train a FNN with such noise/fault were analyzed [15]–[24]. Recently, these ideas have be re-advocated in deep learning. Random node fault (i.e. dropout) [25]–[28], multiplicative node noise [27], [29], gradient noise [30] or input noise [31] is added during training a convolutionary neural network (CNN) or deep neural network (DNN).

A. Misconception

As mentioned in [21], there is a common misconception on the regularization effect of noise injection. It is confused that the objective function being minimized by noise injection-based training (denoted as $\mathcal{L}(\mathbf{w})$) is equivalent to the expected MSE of the model with the same type of noise (denoted as $\mathcal{J}(\mathbf{w})$), i.e. $\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$. Accordingly, $\mathcal{J}(\mathbf{w})$ is used to interpret the regularization effect of noise injection-based training [29], [32], [33]. This misconception could be due to the early analytical works on noise injection [8], [9], [11], [15], [16]. The objective function of training with input noise is given by $\mathcal{L}(\mathbf{w}) = V(\mathbf{w}) + \frac{S_I}{2}\sum_i \frac{\partial^2 V(\mathbf{w})}{\partial x_i^2}$, which is equivalent to the expected MSE $\mathcal{J}(\mathbf{w}) = E[\hat{V}(\mathbf{w})|\mathcal{D}] = V(\mathbf{w}) + \frac{S_I}{2}\sum_i \frac{\partial^2 V(\mathbf{w})}{\partial x_i^2}$. Furthermore, same conclusion is made for the case of additive weight noise [9], [21].

These equivalency results suggest that noise injection could be a cheap trick for implementing regularization. Suppose the noise variance S_I is known, minimizing $\mathcal{J}(\mathbf{w})$ by gradient descent, one needs to solve the following recursive equation :

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \mu_t \left\{ \frac{\partial V(\mathbf{w})}{\partial \mathbf{w}} + \frac{S_I}{2}\sum_i \frac{\partial}{\partial \mathbf{w}} \frac{\partial^2 V(\mathbf{w})}{\partial x_i^2} \right\}.$$

Clearly, the computation of the last term is far more expensive than adding input noise during training. In sequel, researchers started to confuse that the equivalency property could be applied to other noise models, [32, Section 7.5] and [33, Section 7.4.3]. Noise injection is a computationally cheap trick for minimizing $\mathcal{J}(\mathbf{w})$. On the contrary, the regularization effect of noise injection, like multiplicative Gaussian node noise [29], could be interpreted from $\mathcal{J}(\mathbf{w})$. In fact, it is not always true. Whether $\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$ depends on three factors: (i) the model of neural network, (ii) the noise or fault model and (iii) the learning algorithm, as depicted in Table I.

B. Goal of the Paper

The goal of this paper and the companion paper [1] is to clarify this misconception. Apart from the works depicted in Table I, additional results on injecting random weight fault, additive node noise or multiplicative node noise are presented. The desire measures of the model with such fault or noise are derived and the objective functions of these fault/noise injection training are derived. Empirical evidences on the node noise injection are presented in [1].

II. Training with Noise or Fault Injection

For a network with L hidden layers and one output layer, the network could be defined as follows : $\mathbf{f} = \mathbf{h}(\mathbf{z}^L, \mathbf{w}^L)$, $\mathbf{z}^l = \mathbf{h}(\mathbf{z}^{l-1}, \mathbf{w}^l)$ and $\mathbf{z}^1 = \mathbf{h}(\mathbf{x}, \mathbf{w}^1)$ for $l = 2, \cdots, L$, where $\mathbf{z}^l$ and $\mathbf{w}^l$ are respectively the node vector and the weight matrix of the l^{th} hidden layer. $\mathbf{z}^1 = \mathbf{h}(\mathbf{x}, \mathbf{w}^1)$ The transfer function $h(\cdot)$ is a nonlinear function.

Consider one output node, we could let $f(\mathbf{x}, \mathbf{z}, \mathbf{w})$ be the model, where $\mathbf{x} \in R^m$ is the input vector, $\mathbf{z} \in R^s$ is the hidden node vector and $\mathbf{w} \in R^n$ is the weight vector, i.e. $\mathbf{z} =$

The work is supported in part by grants from Taiwan MOST 105-2221-E-005-065-MY2 and 108-2221-E-005-036.

TABLE I
$\mathcal{L}(\mathbf{w})$ VERSUS $\mathcal{J}(\mathbf{w})$

Noise/Fault	NN Model	Learning	Equivalency	Ref.
Input Noise	MLP	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	[2], [8], [9], [11], [15]
Random Weight Fault	MLP	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	This paper
Additive Weight Noise	MLP	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	[9], [19], [21], [22]
Multiplicative Weight Noise	MLP	GD	$\mathcal{L}(\mathbf{w}) \neq \mathcal{J}(\mathbf{w})$	[19], [21], [22]
Random Node Fault	MLP	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	[20]
Additive Node Noise	MLP	GD	$\mathcal{L}(\mathbf{w}) \neq \mathcal{J}(\mathbf{w})$	This paper
Multiplicative Node Noise	MLP	GD	$\mathcal{L}(\mathbf{w}) \neq \mathcal{J}(\mathbf{w})$	[24], this paper
Input Noise	RBF	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	[18]
Random Weight Fault	RBF	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	This paper
Additive Weight Noise	RBF	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	[18]
Multiplicative Weight Noise	RBF	GD	$\mathcal{L}(\mathbf{w}) \neq \mathcal{J}(\mathbf{w})$	[18]
Random Node Fault	RBF	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	[17], [34]
Additive Node Noise	RBF	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	This paper
Multiplicative Node Noise	RBF	GD	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	This paper
Additive Weight Noise	BM	BL	$\mathcal{L}(\mathbf{w}) = \mathcal{J}(\mathbf{w})$	[23]

MLP: Multilayer perceptron; RBF: Radial basis function network
BM: Boltzmann machine; GD: Gradient descent; BL: Boltzmann learning

$\left(\mathbf{z}^L, \cdots, \mathbf{z}^1\right)$ and $\mathbf{w} = \left(\mathbf{w}^L, \cdots, \mathbf{w}^1\right)$. Thus, $\mathbf{z}$ is a function of $\mathbf{x}$ and $\mathbf{w}$, i.e. $\mathbf{z}(\mathbf{x}, \mathbf{w})$.

A. Gradient Descent Learning

Given a set of N samples $\mathcal{D} = \{\mathbf{x}_k, y_k\}_{k=1}^N$, the performance measure of the model is given by

$$V(\mathbf{z}, \mathbf{w}) = \frac{1}{N} \sum_{k=1}^{N} \ell_k(\mathbf{z}(\mathbf{x}_k, \mathbf{w}), \mathbf{w}), \tag{1}$$

where $\ell_k(\mathbf{z}(\mathbf{x}_k, \mathbf{w}), \mathbf{w})$ is the measure of the network on the k^{th} sample. Then, the gradient descent (GD) learning is defined as follows :

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \mu_t \frac{\partial \ell_t(\mathbf{z}(\mathbf{x}_t, \mathbf{w}(t-1)), \mathbf{w}(t-1))}{\partial \mathbf{w}}, \tag{2}$$

where μ_t is the step size. The sample $\{\mathbf{x}_t, y_t\}$ is randomly picked from $\mathcal{D}$.

B. Learning with Weight Fault and Node Noise

With *weight fault*, the learning (2) is replaced by the following update.

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \mu_t \frac{\partial \ell_t(\mathbf{z}(\mathbf{x}_t, \tilde{\mathbf{w}}(t-1)), \tilde{\mathbf{w}}(t-1))}{\partial \mathbf{w}}, \tag{3}$$

where $\tilde{\mathbf{w}} = \mathbf{w} \otimes \mathbf{b}_W$ ($\otimes$ is the elementwise multiplication operator) and $\mathbf{b}_W$ is a random binary vector.

With *node noise*, the learning (2) is replaced by the following update.

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \mu_t \frac{\partial \ell_t(\tilde{\mathbf{z}}(\mathbf{x}_t, \mathbf{w}(t-1)), \mathbf{w}(t-1))}{\partial \mathbf{w}}, \tag{4}$$

$\tilde{\mathbf{z}} = \mathbf{z} + \mathbf{b}_N$ for additive noise, $\tilde{\mathbf{z}} = \mathbf{z} + \mathbf{z} \otimes \mathbf{b}_N$ for multiplicative noise and $\mathbf{b}_N$ is a mean zero Gaussian noise vector.

III. Desired Measures and Learning Objectives

For clarification, we let $\mathcal{J}_\star(\cdot)$ (where $\star = \{W, A, M\}$) be the desired measures of the model with weight fault, additive node noise and multiplicative node noise. Their corresponding learning objectives are denoted as $\mathcal{L}_\star(\cdot)$. The desired measure is defined as follows :

$$\mathcal{J}(\mathbf{w}) = E\left[V(\mathbf{z}, \mathbf{w})\right] = \frac{1}{N} \sum_{k=1}^{N} E\left[\ell_k(\mathbf{z}(\mathbf{x}_k, \mathbf{w}), \mathbf{w})\right]. \tag{5}$$

The expectation is taken over the probability space of $\mathbf{b}_\star$. The desired model could thus be obtained by the following algorithm :

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \mu_t \frac{\partial E[\ell_t]}{\partial \mathbf{w}}. \tag{6}$$

On the other hand, the model obtained either by (3) or (4) could be analyzed by the expected weight update, i.e.

$$E[\mathbf{w}(t) | \mathbf{w}(t-1)] = \mathbf{w}(t-1) - \mu_t E\left[\frac{\partial \ell_t}{\partial \mathbf{w}}\right]. \tag{7}$$

If $\partial E[\ell_t]/\partial \mathbf{w} = E[\partial \ell_t / \partial \mathbf{w}]$, the learning objective is the desired measure.

A. Weight Fault

With weight fault, it can be shown that $\mathcal{L}_W(\mathbf{w}) = \mathcal{J}_W(\mathbf{w})$ as the expected gradient vector in (3) is given by

$$E\left[\frac{\partial \ell_t}{\partial \mathbf{w}}\right] = \sum_{\mathbf{b}_W} \frac{\partial \ell_t}{\partial \mathbf{w}} P(\mathbf{b}_W) = \frac{\partial}{\partial \mathbf{w}} \left\{ \sum_{\mathbf{b}_W} \ell_t P(\mathbf{b}_W) \right\}.$$

The last equality is due to the fact that $\mathbf{b}_W$ are discrete random vector. As a result, the objective function of adding weight fault during gradient descent learning $\mathcal{L}_W(\mathbf{w})$ is equivalent to the desired measure $\mathcal{J}_W(\mathbf{w})$. Let $\mathbf{b}_W = (\alpha_1, \cdots, \alpha_n)$.

$$\mathcal{L}_W(\mathbf{w}) = \mathcal{J}_W(\mathbf{w}) = \sum_{\mathbf{b}_W} V(\mathbf{b}_W \otimes \mathbf{w}) P(\mathbf{b}_W), \tag{8}$$

where $P(\mathbf{b}_W) = \Pi_{i=1}^n p^{(1-\alpha_i)}(1-p)^{\alpha_i}$, p is the weight fault rate $P(\alpha_i = 0)$. Unfortunately, there is no simple close form for (8) in general, expect in some special cases.

B. Additive Node Noise

With additive node noise, it can be shown that the desired measure is given by

$$\mathcal{J}_A(\mathbf{w}) = E[V(\mathbf{w})] = V(\mathbf{w}) + \frac{S_A}{2}\sum_j \frac{\partial^2 V(\mathbf{w})}{\partial z_j^2}. \quad (9)$$

The update of weight w_i is thus be given by

$$w_i(t) = w_i(t-1) - \mu_t \left\{ \frac{\partial \ell_t}{\partial w_i} + \frac{S_A}{2}\sum_j \frac{\partial^3 \ell_t}{\partial w_i \partial z_j^2} \right\}. \quad (10)$$

On the other hand, one can derive the expected update of (4) as follows :

$$\begin{aligned} & E[w_i(t)|\mathbf{w}(t-1)] \\ = \; & w_i(t-1) - \mu_t \left\{ \frac{\partial \ell_t}{\partial w_i} + \frac{S_A}{2}\sum_j \frac{\partial^3 \ell_t}{\partial z_j^2 \partial w_i} \right\}. \end{aligned} \quad (11)$$

It is thus clear from (10) and (11) that $\mathcal{L}_A(\mathbf{w}) \neq \mathcal{J}_A(\mathbf{w})$ as z_j and w_i are not independent variables. One cannot claim that the third order derivatives are the same. From (11, it could be shown that the objective function is given by

$$\mathcal{L}_A(\mathbf{w}) = V(\mathbf{w}) + \frac{S_A}{2}\sum_i \sum_j \int \frac{\partial^3 V(\mathbf{w})}{\partial z_j^2 \partial w_i} dw_i. \quad (12)$$

Again, there is no simple close form for (12) expect in some special cases.

C. Multiplicative Node Noise

With multiplicative node noise, the desired measure is given by

$$\mathcal{J}_M(\mathbf{w}) = V(\mathbf{w}) + \frac{S_M}{2}\sum_j z_j^2 \frac{\partial^2 V(\mathbf{w})}{\partial z_j^2}. \quad (13)$$

The update of weight w_i is thus be given by

$$w_i(t) = w_i(t-1) - \mu_t \left\{ \frac{\partial \ell_t}{\partial w_i} + \frac{S_M}{2}\sum_j \frac{\partial}{\partial w_i} z_j(t)^2 \frac{\partial^2 \ell_t}{\partial z_j^2} \right\}. \quad (14)$$

Similar to additive node noise, the expected update of w_i is given by

$$\begin{aligned} & E[w_i(t)|\mathbf{w}(t-1)] \\ = \; & w_i(t-1) - \mu_t \left\{ \frac{\partial \ell_t}{\partial w_i} + \frac{S_M}{2}\sum_j z_j^2(t) \frac{\partial^3 \ell_t}{\partial z_j^2 \partial w_i} \right\} \end{aligned} \quad (15)$$

It is clear that $\frac{\partial}{\partial w_i} z_j(t)^2 \frac{\partial^2 \ell_t}{\partial z_j^2} \neq z_j^2(t) \frac{\partial^3 \ell_t}{\partial z_j^2 \partial w_i}$. Therefore, $\mathcal{L}_M(\mathbf{w}) \neq \mathcal{J}_M(\mathbf{w})$. From (15), we can get that

$$\mathcal{L}_M(\mathbf{w}) = V(\mathbf{w}) + \frac{S_M}{2}\sum_i \sum_j \int z_j^2(t) \frac{\partial^3 \ell_t}{\partial z_j^2 \partial w_i} dw_i. \quad (16)$$

Again, there is no simple close form for (16) expect in some special cases.

IV. Regularization Effects on RBF

While the objective functions of the respective noise/fault injection training have been revealed, their regularization effects are still unclear. Here, we consider a special case that the model is RBF network, i.e. $f(\mathbf{x}, \mathbf{w}) = \mathbf{h}(\mathbf{x})^T\mathbf{w}$, where $h_i(\mathbf{x})$ is a radial basis function. Moreover, $V(\mathbf{w})$ is the MSE, i.e. $V(\mathbf{w}) = \frac{1}{N}\sum_{k=1}^n \left(y_k - \mathbf{h}(\mathbf{x}_k)^T\mathbf{w}\right)^2$.

A. Weight Fault

With weight fault, the learning (3) could be re-written as follows :

$$\mathbf{w}(t) = \mathbf{w}(t-1) + \mu_t \left(y_t - \mathbf{h}(\mathbf{x}_t)^T \tilde{\mathbf{w}}(t-1)\right) \mathbf{h}(\mathbf{x}_t). \quad (17)$$

The expected update is given by

$$\begin{aligned} & E[\mathbf{w}(t)|\mathbf{w}(t-1)] \\ = \; & \mathbf{w}(t-1) + \mu_t \left(y_t - \mathbf{h}(\mathbf{x}_t)^T \mathbf{w}(t-1)\right) \mathbf{h}(\mathbf{x}_t) \\ & + \mu_t p \mathbf{h}(\mathbf{x}_t)\mathbf{h}(\mathbf{x}_t)^T \mathbf{w}(t-1). \end{aligned} \quad (18)$$

Therefore, the learning objective is given by

$$\mathcal{L}_W(\mathbf{w}) = V(\mathbf{w}) - \mathbf{w}^T \left(\frac{p}{N}\sum_k \mathbf{H}(\mathbf{x}_k) \right) \mathbf{w}, \quad (19)$$

where $\mathbf{H}(\mathbf{x}_k) = \mathbf{h}(\mathbf{x}_k)\mathbf{h}(\mathbf{x}_k)^T$. The additional term plays a role as a de-regularizer. Recall that $\mathcal{L}_W(\mathbf{w}) = \mathcal{J}(\mathbf{w})$. The equivalency applies to weight fault.

B. Additive Node Noise

With additive node noise, it can be shown that the desired measure is given by

$$\mathcal{J}_A(\mathbf{w}) = V(\mathbf{w}) + S_A \mathbf{w}^T\mathbf{w}. \quad (20)$$

On the contrary, the learning (4) could be re-written as follows :

$$\mathbf{w}(t) = \mathbf{w}(t-1) + \mu_t \left(y_t - \tilde{\mathbf{h}}(\mathbf{x}_t)^T \mathbf{w}(t-1)\right) \tilde{\mathbf{h}}(\mathbf{x}_t), \quad (21)$$

where $\tilde{\mathbf{h}} = \mathbf{h} + \mathbf{b}_A$. The expected update is given by

$$\begin{aligned} & E[\mathbf{w}(t)|\mathbf{w}(t-1)] \\ = \; & \mathbf{w}(t-1) + \mu_t \left(y_t - \mathbf{h}(\mathbf{x}_t)^T \mathbf{w}(t-1)\right) \mathbf{h}(\mathbf{x}_t) \\ & - \mu_t S_A \mathbf{w}(t-1). \end{aligned} \quad (22)$$

Thus, we can get the learning objective that

$$\mathcal{L}_A(\mathbf{w}) = V(\mathbf{w}) + S_A \mathbf{w}^T\mathbf{w}. \quad (23)$$

Compare (20) and (23, it is clear that $\mathcal{L}(\mathbf{w} = \mathcal{J}(\mathbf{w})$. The regularization effect is identical to the effect of weight decay.

C. Multiplicative Node Noise

With multiplicative node noise, it can be shown that the desired measure is given by

$$\mathcal{J}_M(\mathbf{w}) = V(\mathbf{w}) + \frac{S_M}{N}\sum_{k=1}^{N}\sum_{i} h_i(\mathbf{x}_k)^2 w_i^2. \quad (24)$$

On the contrary, the learning (4) is re-written as follows :

$$\mathbf{w}(t) = \mathbf{w}(t-1) + \mu_t \left(y_t - \tilde{\mathbf{h}}(\mathbf{x}_t)^T \mathbf{w}(t-1)\right) \tilde{\mathbf{h}}(\mathbf{x}_t), \quad (25)$$

where $\tilde{\mathbf{h}} = \mathbf{h} + \mathbf{b}_M \otimes \mathbf{h}$. The expected update is given by

$$\begin{aligned} & E[\mathbf{w}(t)|\mathbf{w}(t-1)] \\ = \; & \mathbf{w}(t-1) + \mu_t \left(y_t - \mathbf{h}(\mathbf{x}_t)^T \mathbf{w}(t-1)\right) \mathbf{h}(\mathbf{x}_t) \\ & -\mu_t S_M \mathbf{G}(\mathbf{x}_t)\mathbf{w}(t-1), \end{aligned} \quad (26)$$

where $\mathbf{G}(\mathbf{x}_t)$ a diagonal matrix with the i^{th} diagonal element $(\mathbf{G}(\mathbf{x}_t))_{ii} = h_i(\mathbf{x}_t)^2$. Thus, the learning objective is given by

$$\mathcal{L}_M(\mathbf{w}) = V(\mathbf{w}) + \mathbf{w}^T \left(\frac{S_M}{N}\sum_{k=1}^{N} \mathbf{G}(\mathbf{x}_k)\right) \mathbf{w}. \quad (27)$$

One one hand, it is clear that $\mathcal{L}_M(\mathbf{w}) = \mathcal{J}_M(\mathbf{w})$. On the other hand, the regularization effect is similar to weighted weight decay.

V. Conclusions

In this paper, a misconception on noise or fault injection has been elucidated. By revealing the objective functions of the learning with weight fault and node noise injection, it is shown that the actual learning objective $\mathcal{L}(\mathbf{w})$ might not be the same as the desired measure $\mathcal{J}(\mathbf{w})$. Thus, the regularization effect of noise injection could not simply be interpreted from $\mathcal{J}(\mathbf{w})$. A lot more works have to be done in the future.

References

[1] J. Sum, "Misconception on the regularization effect of noise or fault injection : Empirical evidence," 2019, in submission.

[2] K. Matsuoka, "Noise injection into inputs in back-propagation learning," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 22, no. 3, pp. 436–440, May 1992.

[3] A. Murray, "Analogue noise-enhanced learning in neural network circuits," *Electronics Letters*, vol. 27, no. 17, pp. 1546–1548, 1991.

[4] ——, "Multilayer perceptron learning optimized for on-chip implementation: A noise-robust system," *Neural Computation*, vol. 4, no. 3, pp. 366–381, 1992.

[5] J. S. Judd and P. W. Munro, "Nets with unreliable hidden nodes learn error-correcting codes," in *Advances in Neural Information Processing Systems 5*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993, pp. 89–96.

[6] A. Murray and P. Edwards, "Synaptic weight noise during multilayer perceptron training: fault tolerance and training improvements," *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 722–725, 1993.

[7] ——, "Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training," *IEEE Transactions on Neural Networks*, vol. 5, no. 5, pp. 792–802, 1994.

[8] Y. Grandvalet and S. Canu, "Comments on "Noise injection into inputs in back propagation learning"," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 4, pp. 678–681, 1995.

[9] G. An, "The effects of adding noise during backpropagation training on a generalization performance," *Neural Computation*, vol. 8, pp. 643–674, 1996.

[10] K. Jim, C. Giles, and B. Horne, "An analysis of noise in recurrent neural networks: Convergence and generalization," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1424–1438, 1996.

[11] Y. Grandvalet, S. Canu, and S. Boucheron, "Noise injection: Theoretical prospects," *Neural Computation*, vol. 9, no. 5, pp. 1093–1108, 1997.

[12] C. Sequin and R. Clay, "Fault tolerance in feedforward artificial neural networks," *Neural Networks*, vol. 4, pp. 111–141, 1991.

[13] G. Bolt, "Fault tolerant in multi-layer perceptrons," Ph.D. dissertation, University of York, UK, 1992.

[14] J. L. Bernier, J. Ortega, E. Ros, I. Rojas, and A. Prieto, "A quantitative study of fault tolerance, noise immunity, and generalization ability of MLPs," *Neural Computation*, vol. 12, no. 12, pp. 2941–2964, 2000.

[15] C. Bishop, "Training with noise is equivalent to Tikhonov regularization," *Neural Computation*, vol. 7, pp. 108–116, 1995.

[16] R. Reed, R. M. II, and S. Oh, "Similarities of error regularization, sigmoid gain scaling, target smoothing, and training with jitter," *IEEE Transactions on Neural Networks*, vol. 6, no. 3, pp. 529–538, 1995.

[17] J. Sum, "On a multiple nodes fault tolerant training for RBF: Objective function, sensitivity analysis and relation to generalization," in *Proceedings of TAAI'05, Tainan, Taiwan*, 2005.

[18] K. Ho, C. Leung, and J. Sum, "Convergence and objective functions of some fault/noise injection-based online learning algorithms for RBF networks," *IEEE Transactions on Neural Networks*, vol. 21, no. 6, pp. 938–947, June 2010.

[19] ——, "Objective functions of the online weight noise injection training algorithms for MLP," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 317–323, Feb 2011.

[20] J. Sum, C.-S. Leung, and K. Ho, "Convergence analysis of on-line node fault injection-based training algorithms for MLP networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 2, pp. 211–222, Feb 2012.

[21] ——, "Convergence analyses on on-line weight noise injection-based training algorithms for MLPs," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 11, pp. 1827–1840, Nov 2012.

[22] ——, "A limitation of gradient descent learning," 2019, accepted for publication in *IEEE Transactions on Neural Networks and Learning Systems*.

[23] J. Sum and C.-S. Leung, "Learning algorithm for Boltzmann machines with additive weight and bias noise," *IEEE Transactions on Neural Networks and Learning Systems*, 2019, accepted for publication.

[24] ——, "Analysis on dropout regularization," 2019, in submission.

[25] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[26] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[28] H. Noh, T. You, J. Mun, and B. Han, "Regularizing deep neural networks by noise: Its interpretation and optimization," in *Advances in Neural Information Processing Systems*, 2017, pp. 5109–5118.

[29] E. Nalisnick, A. Anandkumar, and P. Smyth, "A scale mixture perspective of multiplicative noise in neural networks," *arXiv preprint arXiv:1506.03208*, 2015.

[30] A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens, "Adding gradient noise improves learning for very deep networks," *arXiv preprint arXiv:1511.06807*, 2015.

[31] K. Audhkhasi, O. Osoba, and B. Kosko, "Noise-enhanced convolutional neural networks," *Neural Networks*, vol. 78, pp. 15–23, 2016.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT press, 2016.

[33] B. Ghojogh and M. Crowley, "The theory behind overfitting, cross validation, regularization, bagging, and boosting: Tutorial," *arXiv preprint arXiv:1905.12787*, 2019.

[34] J. Sum, C.-S. Leung, and K. Ho, "On node-fault-injection training of an RBF network," in *International Conference on Neural Information Processing*. Springer, 2008, pp. 324–331.