

01 Apr 2018

# Direct Error Driven Learning for Deep Neural Networks with Applications to Bigdata

R. Krishnan

Jagannathan Sarangapani
*Missouri University of Science and Technology*, sarangap@mst.edu

V. A. Samaranayake
*Missouri University of Science and Technology*, vsam@mst.edu



## Recommended Citation

R. Krishnan et al., "Direct Error Driven Learning for Deep Neural Networks with Applications to Bigdata," *Procedia Computer Science*, vol. 144, pp. 89-95, Elsevier, Apr 2018.
The definitive version is available at https://doi.org/10.1016/j.procs.2018.10.508

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 144 (2018) 89–95

Procedia Computer Science

www.elsevier.com/locate/procedia

INNS Conference on Big Data and Deep Learning 2018

# Direct Error Driven Learning for Deep Neural Networks with Applications to Bigdata

R. Krishnan[a,*], S. Jagannathan[a], V.A. Samaranayake[b]

[a]*Department of Electrical and Computer Engineering, Missouri University of Science and Technology, USA*
[b]*Department of Mathematics and Statistics, Missouri University of Science and Technology, USA*

**Abstract**

In this paper, generalization error for traditional learning regimes-based classification is demonstrated to increase in the presence of bigdata challenges such as noise and heterogeneity. To reduce this error while mitigating vanishing gradients, a deep neural network (NN)-based framework with a direct error-driven learning scheme is proposed. To reduce the impact of heterogeneity, an overall cost comprised of the learning error and approximate generalization error is defined where two NNs are utilized to estimate the costs respectively. To mitigate the issue of vanishing gradients, a direct error-driven learning regime is proposed where the error is directly utilized for learning. It is demonstrated that the proposed approach improves accuracy by 7 % over traditional learning regimes. The proposed approach mitigated the vanishing gradient problem and improved generalization by 6%.



*Keywords:* generalization error; vanishing gradients; bigdata; heterogeneity; noise

## 1. Introduction

Classification is the process of dividing data-points into categories which is comprised of learning and a prediction phase. During classification with big data, unique challenges are introduced that include: (1) noise of two types: noisy data - the presence of distortions, and noisy dimensions - those not contributing towards learning efficiency [3]; (2) statistical heterogeneity- dissimilarity between statistical properties in unique samples from a data-set [3]. Deep neural networks (deep NN) are capable of addressing a few of these challenges [1]. One of the common methodologies used for learning the NN weights is the stochastic gradient descent (SGD) [7].

In spite of many successes [1], SGD based regimes suffer from vanishing gradients where the learning signals tend to zero with an increase in the number of layers in the deep NN [13]. The issue is typically addressed using relu activation functions to keep the gradients constant with robust weight initializations [11] and L-1/L-2 regularization [4] methods to keep the magnitude of weights small. With these methods [11, 13, 4], no guarantee can be given that the issue of vanishing gradients will not be observed [13]. In fact, it has been reported that issue of vanishing gradients can be observed with [6] relu activation functions.

Furthermore, challenges such as noise and heterogeneity [3] can increase generalization error while learning with SGD. In the literature, generalization capacity has been addressed using techniques such as $L_2$ norm regularization

* Corresponding author.
*E-mail address:* krm9c@mst.edu

1877-0509 
10.1016/j.procs.2018.10.508

[11], dropouts [16] and adversarial regularization [4]. However, these methods [16, 4] are not systematic and require a comprehensive trial and error process.

Motivated by these challenges, the impact of big data challenges are addressed to; (1) reduce generalization error while mitigating the impact of heterogeneity and data-noise on it and (2) mitigate the learning inefficiency due to vanishing learning signals.

To address data-noise and heterogeneity, a framework of two deep NN is proposed. Both of the deep NNs learn the map from inputs to predictions. A sample of data is directly fed as inputs to the first deep NN to estimate learning error. Synthetic distortions are added to the input that is given to the second NN for approximating the generalization error. Subsequently, the overall cost, comprised of the cost due to learning and approximated generalization errors is minimized in the learning procedure. Thus, in contrast with [17, 16, 4], where heuristics are utilized, the impact of heterogeneity and data-noise is mitigated by minimizing the approximated cost due to generalization error.

To mitigate the issue of vanishing gradients, a direct error-driven learning scheme is introduced where the error is directly used for learning through a user-defined design matrix. In contrast with [11, 13] where the error is directly propagating through the layer-wise weights, learning signals do not vanish in EDL unless the error becomes zero.

A simulation study using four benchmarking datasets is presented. It is demonstrated that the proposed methodology indicates improvement over SGD in the presence of noise and heterogeneity. For notations, consider $\mathbb{R}$ to represent the set of real numbers. Let the superscript $i$ denote the index for the layer and the number of layers in the network is represented by $l$. All data-points in the prediction phase are denoted by $\boldsymbol{x}$ and in the learning phase are denoted by $\hat{\boldsymbol{x}}$.

The rest of the paper is organized as follows. The problem is described in Section II and the proposed framework with direct error-driven learning scheme is described in Section III. Finally, Section IV outlines simulation results for the paper while Section V provides the conclusions.

## 2. Problem Statement

Let a sample of data be denoted as $\boldsymbol{x} \in \mathbb{R}^{n\times p}$, where $n$ represents the number of sample points and $p$ is total number of attributes. The objective is to detect whether $\boldsymbol{x}$ belongs to the healthy case or at one of the faults $\mathcal{F}$, which is the problem of fault diagnostics. In a general problem of classification, that includes fault diagnostics, the objective is to predict the category for $\boldsymbol{x}$ by transforming $\boldsymbol{x}$ into $\boldsymbol{y}$ using $\psi(.)$, where the category for $\boldsymbol{x}$ is indicated by $\arg\max(\boldsymbol{y})$. Let a deep neural network be utilized to approximate $\psi(.)$ such that the estimate for $\psi(\boldsymbol{x})$ is given as

$$\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\boldsymbol{\theta}}) = g^{(l)}(\hat{\boldsymbol{W}}^{(l)} \cdots (g^{(1)}(\hat{\boldsymbol{W}}^{(1)}\boldsymbol{x}))), \tag{1}$$

with estimated weights $\hat{\boldsymbol{\theta}} = [\hat{\boldsymbol{W}}^{(1)} \cdots \hat{\boldsymbol{W}}^{(l)}]$. The bias is included in the weight matrix and the layer wise activation functions are denoted by $g^{(i)}, for \quad i = 1 \cdots l$. To learn the map, a data-set representing each of the faults/categories is required. The data is collected such that the following assumptions hold.

**Assumption 1.** *Samples from each fault are obtained such that they are independently and identically distributed.*

**Assumption 2.** *The distribution of each of the categories in the learning phase is similar to the distribution of the data in the prediction phase.*

Let $\boldsymbol{X}$ represent all the available data-points and let $\boldsymbol{Y}$ refer to the corresponding labels or the true categories in the data-set such that the data-points in $\boldsymbol{X}$ are collected across $p$ attributes and labels are of size $\mathbb{R}^{\mathcal{F}\times 1}$. Next, the learning objective is described.

### 2.1. Learning Objective

For learning $\psi(.)$, a cost $C_o$ [2] measuring the difference between the predicted categories $\hat{\boldsymbol{y}}(.)$ and the true categories $\boldsymbol{y}(.)$ is defined and the deep NN weights $\hat{\boldsymbol{\theta}}$ are estimated to minimize $C_o$. The overall cost $C_o(t)$ can be written as

$$C_o(k) = \underbrace{C_{emp} + C_{gen}}_{C(k)} + C_{apx}, \tag{2}$$

where the empirical cost is denoted as $C_{emp}(t) = E[\boldsymbol{e}_l]^T E[\boldsymbol{e}_l]$ such that $E[\boldsymbol{e}_l] = E_{\forall \hat{\boldsymbol{x}} \in \mathcal{X}, \boldsymbol{y} \in \boldsymbol{Y}}[\boldsymbol{y}(\boldsymbol{x}) - \hat{\boldsymbol{y}}(\boldsymbol{x}; \hat{\boldsymbol{\theta}})]$ is the learning error, while $\boldsymbol{y}(\boldsymbol{x})$ represents the target categories and $\hat{\boldsymbol{y}}(\boldsymbol{x}; \hat{\boldsymbol{\theta}})$ is a NN of the form in Eq. (1). The cost [2] introduced by generalization and approximation errors are written using $C_{gen}$ and $C_{apx}$ respectively. Generally, $C_{emp}$ is the only measured quantity and is therefore minimized during the learning phase under the assumption that the unknown quantities $C_{gen}$ and $C_{apx}$ are small and bounded [2] respectively.

However, due to bigdata challenges, deterioration in performance [3] is observed that is reflected in $C_{gen}$. For example, when big data is collected over a period of years from one attribute and stored at multiple locations, data from one location does not fully represent the underlying distribution of each attribute that is observed. As a result, if $\boldsymbol{x}$ is an outlier with respect to the subset of data from one location, $\boldsymbol{x}$ will be misclassified if only the data from one location is used to build the classification model. In addition, if, $\boldsymbol{x}$ is an outlier to the underlying distribution, one may observe erroneous predictions. In both these scenarios, $C_{gen}$ would increase. A methodology to approximate $C_{gen}$ and minimize it to improve learning effectiveness is therefore needed and presented next.

## 3. Direct Error-driven Learning (EDL)

A flow chart of the proposed approach is shown in Fig. 1b. The impact of heterogeneity and data-noise are measured in terms of generalization error and the associated cost is included in the learning problem. Two deep NNs are utilized to estimate the learning error and the generalization error respectively. Finally, a direct-error driven learning regime is introduced. The details are described next.

### 3.1. Mitigating Heterogeneity and Data-noise

Measuring generalization error is one way to quantify the impact of heterogeneity and data-noise on learning effectiveness. However, it is impossible to measure this error because the data-points that result in generalization error are not available during the learning phase. Thus, we aim to achieve an approximation of generalization error and its associated cost by introducing randomly generated perturbations denoted as $\Delta\hat{\boldsymbol{x}}$ into every data-point $\hat{\boldsymbol{x}}$.

The collection of these new data-points, achieved by introducing perturbations, represent a neighborhood to $\hat{\boldsymbol{x}}$ and all the data-points in the neighborhood belong to the same category as $\hat{\boldsymbol{x}}$. The neighborhood can therefore be considered as representing the data-points similar to $\hat{\boldsymbol{x}}$ but not available during the learning phase. Therefore, any error introduced by these data-points into the learning problem provide an approximation of the extent of generalization error introduced by the neighborhood. A collection of the neighborhood for each of the data-point, denoted as $\boldsymbol{X}_B$ represents the neighborhood of $\boldsymbol{X}$.

To approximate generalization error using $\boldsymbol{X}_B$, a second deep NN $\hat{\boldsymbol{y}}(\boldsymbol{x}_B; \hat{\boldsymbol{\Theta}})$ of the form shown in Eq. (1) is introduced. The second NN learns the map between $\boldsymbol{X}_B$ and the predictions, with the estimated parameters denoted as $\hat{\boldsymbol{\Theta}} = [\hat{\boldsymbol{V}}^{(1)} \cdots \hat{\boldsymbol{V}}^{(d)}]$. Using the additional NN, the approximated generalization error ($\boldsymbol{e}_{gen}$) is given as $E[\boldsymbol{e}_{gen}] = E_{\forall \hat{\boldsymbol{x}}_B \in \boldsymbol{\mathcal{X}}_B, \boldsymbol{y} \in \boldsymbol{\mathcal{Y}}}[\boldsymbol{y}(\boldsymbol{x}_B) - \hat{\boldsymbol{y}}(\boldsymbol{x}_B; \hat{\boldsymbol{\Theta}})]$.

To incorporate approximated generalization error into the learning problem, define $E[\boldsymbol{e}_{gen}]^T E[\boldsymbol{e}_{gen}]$ as the cost $\hat{C}_{gen}(k)$ and substitute $C_{gen}$ by $\hat{C}_{gen}(k)$ in Eq. (2) and simplify with $C(k)$ as $C$ to write the learning problem with respect to the estimated weights $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\Theta}}$ as

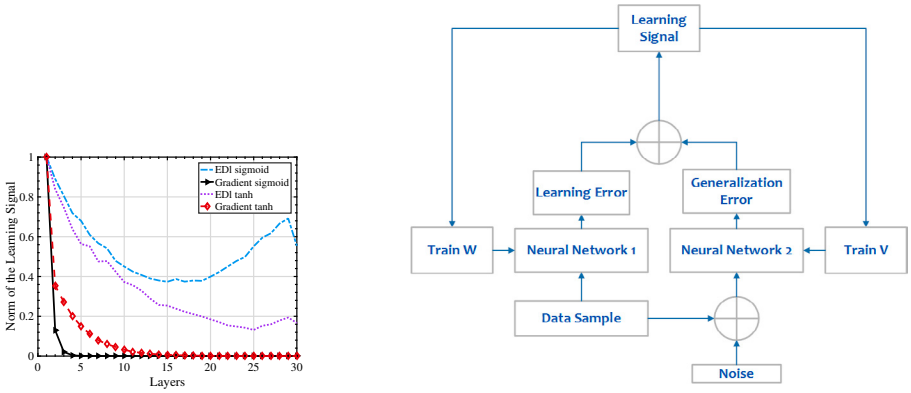
$$[\boldsymbol{\theta}^*, \boldsymbol{\Theta}^*] = \underset{\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\Theta}} \in \boldsymbol{\Omega}}{\arg\min}\, C, \tag{3}$$

where $C = \frac{1}{2} E[\boldsymbol{\epsilon}]^T E[\boldsymbol{\epsilon}]$, with $\boldsymbol{\epsilon} = [\boldsymbol{e}_l \quad \boldsymbol{e}_{gen}]$ being the overall error and $\boldsymbol{\Omega}$ representing the parameter space.

By minimizing Eq. (3), $\boldsymbol{\epsilon}$ would get minimized which in turn lead to minimization of $\boldsymbol{e}_{gen}$ that mitigates the impact of heterogeneity in the learning phase. Next, in order to optimize the overall cost, the following weight update law [2] with regularization term is utilized

$$\hat{\boldsymbol{W}}^{(i)}_{k+1} = (1 - \alpha\lambda)\hat{\boldsymbol{W}}^{(i)}_k + \alpha[\boldsymbol{\delta}^{(i)}_k(\boldsymbol{W})]^T. \tag{4}$$

Furthermore, at the learning instant $k$, the change in $\hat{\boldsymbol{W}}^{(i)}_k$ is given by $\boldsymbol{\delta}^{(i)}_k(\boldsymbol{W}) = \frac{E[\Lambda^{(i)}(\boldsymbol{W})]}{1 + \|E[\Lambda^{(i)}(\boldsymbol{W})]\|^2}$, where $\alpha > 0$ is a small learning rate and $\Lambda^{(i)}(\boldsymbol{W})$ is the learning signal [7]. The weight update for $\hat{\boldsymbol{V}}^{(i)}$ is defined similar to Eq. (4). In the




(a) (b)

Fig. 1: (a) Norm of the learning signal propagation with respect to layers in the deep NN and (b) Overall methodology.

following subsection, two methods of choosing $\Lambda^{(i)}$ are presented. First, traditional gradient descent (GD) is utilized for solving the optimization problem and then the direct error driven learning is introduced.

### 3.2. Mitigating Vanishing Gradients

First, traditional gradient descent (GD) is utilized such that $\Lambda^{(i)}(\boldsymbol{W}) = \nabla_{\hat{\boldsymbol{W}}_k^{(i)}} C$. The term $\nabla_{\hat{\boldsymbol{W}}_k^{(i)}} C$ denotes the gradient of $C$ with respect to $\hat{\boldsymbol{W}}_k^{(i)}$. written as

$$\nabla_{\hat{\boldsymbol{W}}_t^{(i)}} C = \boldsymbol{G}^{(i)}(\boldsymbol{x}) + \boldsymbol{G}^{(i)}(\boldsymbol{x} + \Delta\boldsymbol{x}), \tag{5}$$

where $G^{(i)}(\boldsymbol{x}) = \nabla_{\hat{\boldsymbol{W}}_t^{(i)}} C_{emp}$ and $G^{(i)}(\boldsymbol{x} + \Delta x) = \nabla_{\hat{\boldsymbol{W}}_t^{(i)}} \hat{C}_{gen}$. Since, $\hat{C}_{gen}$ is the function of generalization error with parameters $\boldsymbol{V}$ thus $G^{(i)}(\boldsymbol{x} + \Delta x)$ is zero. Applying chain rule to get a generalized expression for $\boldsymbol{G}^{(i)}(\boldsymbol{x})$ [14], we get $\Lambda^{(i)}(\boldsymbol{W})$ as

$$\Lambda^{(i)}(\boldsymbol{W}) = -g^{(i-1)}(\boldsymbol{x}) \boldsymbol{e}_l^T [\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x})] diag(\nabla g^{(i)}(\boldsymbol{x})), \tag{6}$$

where $\prod_{j=l}^{i+1} diag(\nabla g^{(j)}(\boldsymbol{x})) \hat{\boldsymbol{W}}^{(j)}$ is denoted as $\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x})$. The issue of vanishing gradients arises when the updates in Eq. (6) are utilized because $\boldsymbol{e}_l$ has to propagate through $\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x})$ and the singular values of $\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x})$ vanish with the increase in the number of layers [13]. To demonstrate this issue, consider the MNIST digits recognition data-set with a NN for classification [8]. With an increase in the number of layers, the norm of the learning signal reduces and approaches zero as seen in Fig. 1a. Both sigmoid and tanh activation functions appear to give the same result.

To obviate this problem, we propose to use the error directly for learning using a user-defined feedback matrix $\boldsymbol{B}^{(i)}(\boldsymbol{x})$. The learning rule for EDL, therefore follows directly from Eq. (6) as

$$\Lambda^{(i)}(\boldsymbol{W}) = -g^{(i-1)}(\boldsymbol{x}) \boldsymbol{e}_l^T [\boldsymbol{B}^{(i)}(\boldsymbol{x})] diag(\nabla g^{(i)}(\boldsymbol{x})), \tag{7}$$

where $\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x})$ is replaced with $\boldsymbol{B}^{(i)}(\boldsymbol{x})$. Unlike in SGD, the learning signal in EDL does not vanish because $\boldsymbol{B}^{(i)}$ is chosen with non-zero singular values for learning effectiveness. As a result, $\Lambda^{(i)}(\boldsymbol{W})$ can only go to zero if $\boldsymbol{e}_l$ or $diag(\nabla f^{(i)}(\boldsymbol{x}))$ goes to zero. The factor $diag(\nabla f^{(i)}(\boldsymbol{x}))$ for a particular layer $i$ can be zero, if the activation function is saturated. In the literature, relu, sigmoid or tanh are popular choices for activation functions [13] and it is known that

[6] the issue of saturation exists in all of these functions. Proper initialization of weights with normalized data can be used to avoid the problem [4].

However, for efficient learning, it is important to choose $\boldsymbol{B}^{(i)}$. Random sampling from a pre-selected distribution is the simplest way to choose $\boldsymbol{B}^{(i)}$. In random sampling, the learning directions are determined at random and the learning process can explore all modes of the weight matrix. By ensuring that $\boldsymbol{B}^{(i)}$ is chosen with all positive singular values, the learning signals will not vanish [12].

It can be empirically verified that the new update rule does not lead to vanishing gradients. Similar to the previous case, a NN is utilized for classification in the MNIST data-set. The norms in this case does not approach zero with an increase in the number of layers as seen in Fig. 1a.

Table 1: Summary of the update laws, where $\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x}) = \prod_{j=l}^{i+1} diag(\nabla g^{(j)}(\boldsymbol{x})\hat{\boldsymbol{W}}^{(j)}$ and $\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x}+\Delta\boldsymbol{x}) = \prod_{j=d}^{i+1} diag(\nabla g^{(j)}(\boldsymbol{x}+\Delta\boldsymbol{x})\hat{\boldsymbol{V}}^{(j)}$.

| $\Lambda^{(i)}(.)$ | **NN1** ($\boldsymbol{W}^{(i)}$) | **NN2** ($\boldsymbol{V}^{(i)}$) |
|---|---|---|
| Gradient descent | $-g^{(i-1)}(\boldsymbol{x})\boldsymbol{e}_l^T[\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x})]diag(\nabla g^{(i)}(\boldsymbol{x}))$ | $-g^{(i-1)}(\boldsymbol{x}+\Delta\boldsymbol{x})\boldsymbol{e}_{gen}^T[\boldsymbol{\mathcal{T}}^{(i)}(\boldsymbol{x}+\Delta\boldsymbol{x})]diag(\nabla g^{(i)}(\boldsymbol{x}+\Delta\boldsymbol{x}))$ |
| EDL | $-g^{(i-1)}(\boldsymbol{x})\boldsymbol{e}_l^T[\boldsymbol{B}^{(i)}(\boldsymbol{x})]diag(\nabla g^{(i)}(\boldsymbol{x}))$ | $-g^{(i-1)}(\boldsymbol{x}+\Delta\boldsymbol{x})\boldsymbol{e}_{gen}^T[\boldsymbol{B}^{(i)}(\boldsymbol{x}+\Delta\boldsymbol{x})]diag(\nabla g^{(i)}(\boldsymbol{x}+\Delta\boldsymbol{x}))$ |

Considering the update law in Eq (6) and Eq (7), the updates for $\boldsymbol{W}$ and $\boldsymbol{V}$ for gradient based updates as well as EDL is summarized in Table. 1. An overview of the proposed framework is given in Fig. 1b. Both the NN are considered to be of similar capacity in this paper, i.e., the number of hidden layers and the activation functions are kept same. Observe that the learning would progress until both $C_{emp}$ and $C_{gen}$ are minimized and the weight update for the two NNs are independent of each other. In the on-line learning phase, training is performed using mini-batches. In the off-line prediction phase, an average of the output from the two NNs is utilized for prediction. The performance of the proposed methodology on bench-marking datasets is detailed in the next section.

## 4. Results and Discussions

Four data-sets are used for analysis and the details for these data-sets are summarized in Table. 2. Rolling element bearing, sensorless and dexter are fault diagnostics data-set whereas MNIST is a classification data-set. In all the data-sets considered here, 80 % of the data is randomly chosen for training and 20 % for test. All the results presented in the section are on the test set.

In all the simulations for EDL, $\boldsymbol{B}^{(i)}$ is sampled from uniform distribution with support $[-1, 1]$. Noise is introduced in the learning phase for the proposed approach with both Gaussian and uniform distributions. Gaussian distributed noise is chosen with zero mean vector and covariance matrix given as $\sigma^2\boldsymbol{I}$, with $\boldsymbol{I}$ being an appropriate identity matrix and $\sigma^2$ being the variance of choice. Uniformly distributed noise is chosen by sampling between $[-\sigma^2, \sigma^2]$. Robustness to heterogeneity and data-noise with the MNIST data-set is first demonstrated. Software package Tensorflow with python is used for all the experiments in this paper and the results are averaged for hundred initial conditions.

### *4.1. MNIST data-set - Robustness to Heterogeneity and Data-noise*

To simulate the presence of heterogeneity in the data, randomly sampled noise is introduced in the data during the testing phase. First, the two NN in the proposed approach are considered with tanh and relu activation functions, respectively. Five hidden-layers are considered. Learning rates are kept at 0.01 for both the NNs. The NNs are trained until convergence. The NNs are trained with $\sigma^2 = 1$.

The proposed framework with gradient descent updates consistently performs better relative to regular SGD for the MNIST data-set as seen from Table 3. The results appear to hold for both Gaussian and uniformly-distributed noise. Furthermore, higher accuracies are observed for relu compared to tanh activation functions.

Table 2: Summary descriptions of the different data-sets used in this paper

| **Data-set** | **Dimensions** | **Data points** | **Classes** |
|---|---|---|---|
| Rolling [15] | 11 | 35000 | 4 |
| Sensorless [9] | 48 | 78000 | 11 |
| MNIST [8] | 784 | 72000 | 10 |
| Dexter[5] | 20000 | 300 | 2 |

Table 3: Mean test accuracies with the standard deviations for the proposed framework with $\sigma^2 = 1$ during the learning phase.

| | **SGD** | **PF+GD** |
|---|---|---|
| (Gaussian) relu | 0.83(0.07) | 0.91(0.093) |
| (Uniform) relu | 0.85(0.10) | 0.93(0.018) |
| (Gaussian) tanh | 0.82(0.006) | 0.87(0.003) |
| (Uniform) tanh | 0.86(0.10) | 0.89(0.018) |

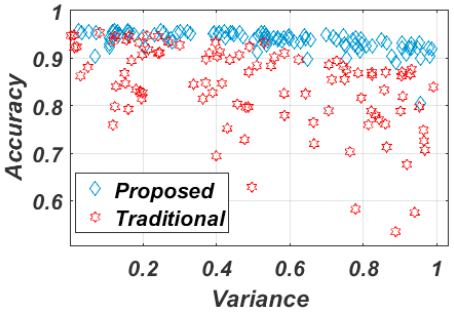



Fig. 2: Accuracy for change in $\sigma^2$ (variance) for the noise introduced during the test phase.

Table 4: Generalization error for all the data-sets with tanh activation functions.

| **Fault** | % Generalization Error | | |
|---|---|---|---|
| | **SGD** | **PF+GD** | **PF+EDL** |
| Rolling Element | 0.1 | 0.06 | 0.0 |
| Sensorless Drive Diagnostics | 0.11 | 0.008 | 0.002 |
| MNIST | 13 | 1 | 0.6 |
| Dexter | 2.31 | 1.1 | 1.00 |

Table 5: Mean test % accuracies

| Datasets | DFA | FA | SGD | EDL |
|---|---|---|---|---|
| **Rolling** | 99 | 99 | 99 | 99 |
| **Sensorless** | 94 | 94 | 95 | 94 |
| **MNIST** | 92 | 94 | 95 | 97 |
| **Dexter** | 71 | 77 | 81 | 80 |

The proposed framework and SGD are tested for hundred initial conditions of $\sigma^2$ and weights. The average accuracies are illustrated in Fig. 2. The proposed framework shows higher accuracies and less spread with respect to $\sigma^2$. The behavior suggests that the proposed approach is more robust to data-noise and heterogeneity than SGD.

The generalization capacity of the proposed approach is studied and the results are shown in Table 4. Even for the case of gradient-based updates with the proposed framework, there is a significant reduction in generalization error and the lowest generalization error is achieve with EDL consistently.

### *4.2. Classification Performance*

With a total of four data-sets, four learning methodologies namely DFA (Direct Feedback Alignment) [12], SGD (Stochastic Gradient Descent) [7], FA (Feedback Alignment) [10] and EDL (Error-driven Learning), are tested. The two NN are chosen with ten hidden layers and relu activation functions. For EDL, $\sigma^2 = 1$ as the variance parameter.

The proposed methodology is seen to provide acceptable accuracies as seen in Table 5 for all the four data-sets. For the dexter data-set, improvement over DFA and FA is observed, but the results for SGD and EDL were similar. Overall, the performance for the proposed framework is better than DFA and FA in all data-sets considered here.

## 5. Conclusions

In this paper, a classifier design in the presence of challenges such as data-noise, heterogeneity and vanishing gradients is presented. By minimizing the approximated cost due generalization error in the learning phase, the impact of heterogeneity and data-noise was mitigated. Overall, the proposed approach appears to provide a 7% reduction in generalization error and a 6 % improvement in accuracy over SGD in the presence of noise. Theoretical implications of the proposed framework and the learning scheme are left as part of the future work.

## Acknowledgment

This research was supported in part by an NSF I/UCRC award IIP 1134721 and Intelligent Systems Center.

## References

[1] Bengio, Y., Courville, A., Vincent, P., 2013. Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence 35, 1798–1828.

[2] Bishop, C.M., 2006. Pattern recognition and machine learning. springer.
[3] Fan, J., Han, F., Liu, H., 2014. Challenges of big data analysis. National science review 1, 293–314.
[4] Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y., 2016. Deep learning. volume 1. MIT press Cambridge.
[5] Guyon, I., Gunn, S., Ben-Hur, A., Dror, G., 2005. Result analysis of the nips 2003 feature selection challenge, in: Advances in neural information processing systems, pp. 545–552.
[6] Hanin, B., 2018. Which neural net architectures give rise to exploding and vanishing gradients? arXiv preprint arXiv:1801.03744 .
[7] Hardt, M., 2015. 3.12 train faster, generalize better: Stability of stochastic gradient descent. Mathematical and Computational Foundations of Learning Theory , 64.
[8] LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. nature 521, 436.
[9] Lichman, M., 2013. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.
[10] Lillicrap, T.P., Cownden, D., Tweed, D.B., Akerman, C.J., 2016. Random synaptic feedback weights support error backpropagation for deep learning. Nature communications 7, 13276.
[11] Mishkin, D., Matas, J., 2015. All you need is a good init. arXiv preprint arXiv:1511.06422 .
[12] Nøkland, A., 2016. Direct feedback alignment provides learning in deep neural networks, in: Advances in Neural Information Processing Systems, pp. 1037–1045.
[13] Pascanu, R., Mikolov, T., Bengio, Y., 2013. On the difficulty of training recurrent neural networks, in: International Conference on Machine Learning, pp. 1310–1318.
[14] Rojas, R., 1996. The backpropagation algorithm, in: Neural networks. Springer, pp. 149–182.
[15] Soylemezoglu, A., Jagannathan, S., Saygin, C., 2010. Mahalanobis taguchi system (mts) as a prognostics tool for rolling element bearing failures. Journal of Manufacturing Science and Engineering 132, 051014.
[16] Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. Journal of machine learning research 15, 1929–1958.
[17] Yu, Z., Li, L., Liu, J., Han, G., 2015. Hybrid adaptive classifier ensemble. IEEE Transactions on Cybernetics 45, 177–190. doi:10.1109/TCYB.2014.2322195.