European Conference on Optical Communication (ECOC) 2022 © Optica Publishing Group 2022

Power-Efficient and Robust Nonlinear Demapper for 64QAM using In-Memory Computing

Amro Eldebiky[(1)], Georg Böcherer[(2)], Grace li Zhang[(1)], Bing Li[(1)], Maximilian Schädler[(2)], Stefano Calabrò[(2)], Ulf Schlichtmann[(1)]

[(1)] Technical University of Munich, {amro.eldebiky, grace-li.zhang, b.li, ulf.schlichtmann}@tum.de
[(2)] Huawei Munich Research Center, {georg.boecherer, maximilian.schaedler, stefano.calabro}@huawei.com

Abstract *In-memory computing can trade computational accuracy for power saving. We consider the implementation of a nonlinear demapper for coherent optical transceivers and use Lipschitz constraints to increase robustness against device variations. Offline experiments demonstrate that for 64QAM we can recover the performance of a digital implementation. ©2022 The Authors*

Introduction

In reliable long-range and high-speed optical communications, nonlinear equalization and demapping are needed to compensate for transmission impairments. Soft demapping deploys soft-decision forward error correction (FEC) to translate a received symbol to its soft bits. The soft bits, expressed as log-likelihood ratio, represent the level of confidence of each bit being 0 or 1. Recently equalization and soft demapping of received symbols have been implemented using neural-networks (NNs)[1] due to their intrinsic parallelization and the availability of large amount of training data.

Previous work has introduced NN architectures for equalization and soft demapping in optical systems. In[2]–[4], several NN soft demappers are proposed for quadrature amplitude modulation (QAM) systems. In these methods, NN models calculate the log maximum of the posterior probability (log-MAP). In[5],[6], a NN demapper structure with memory taps similar to that of a Volterra series[7] is presented.

However, NNs require a huge number of multiply-and-accumulate (MAC) operations. Besides, one NN inference step on digital accelerators requires a huge number of parameters to be loaded and stored. The transfer of these parameters consumes a high energy[8]. To address such challenges, analog in-memory computing (IMC) platforms based on emerging devices, e.g., resistive RAM (RRAM), have been introduced[9]–[11]. In such accelerators, weights of NNs are represented with the conductances of RRAM cells. Such accelerators perform MAC operations based on Ohm's law and Kirchhoff's current law, so that a high computation and energy efficiency can be achieved. According to[9], the energy efficiency of IMC is 17 times of that of digital implementations.

Despite the advantages of the analog-based computing platforms, they suffer from manufacturing process variations and noise[12]. The physical parameter variations of RRAM cells lead to deviations of the programmed conductance from the nominal value and thus deviations of the corresponding weights of NNs. The perturbed weights cause the feature maps at the output of layers to become erroneous. The errors in feature maps are amplified as they go through subsequent layers. Accordingly, the accuracy of NNs implemented with IMC degrades significantly.

In the context of image processing, previous work addresses the performance degradation of NN models in IMC accelerators. For example, the effect of device variations is compensated by variation-aware training[13],[14]. Variations are modeled as functions of random variables to train NNs statistically[15]. The method in[16] uses a variation-aware mapping in which large weights are represented by RRAM cells with smaller variations. However, this method needs prior knowledge of the variation profile, which requires costly testing and measurement of each manufactured chip.

In this work, we investigate the effect of process variations in IMC on the performance of NN soft demappers. We adopt the NN architecture introduced in[5] as a soft demapper study case. A novel training technique which adopts Lipschitz constant constraints on each NN layer is introduced to mitigate the effect of weight variations in IMC accelerators. The Lipschitz constraints enhance NN robustness by preventing variation amplification inside NNs. The values of Lipschitz constants for each layer are determined by particle swarm optimization (PSO) to achieve a robust performance.

Variation Model and Performance Degradation

Various variation models have been proposed to model weight deviations. One of them is the log-normal model[13],[16],[17]:

$$w_{logNormal} = w_{nominal} * e^{\theta}, \quad \theta \sim N(0, \sigma^2) \qquad (1)$$

where $w_{nominal}$ is the nominal value of the trained weight, θ is an independent random variable following normal distribution for this weight, and σ is the standard deviation of this random variable. Another model is the normal model[14],[18],[19]:

$$w_{Normal} \sim N(w_{nominal}, (\sigma * w_{nominal})^2). \qquad (2)$$

Both models are used in the following experiments to demonstrate the performance degradation under variations and the effectiveness of the proposed method.

For the demonstration of the variation effects and the

European Conference on Optical Communication (ECOC) 2022 © Optica Publishing Group 2022

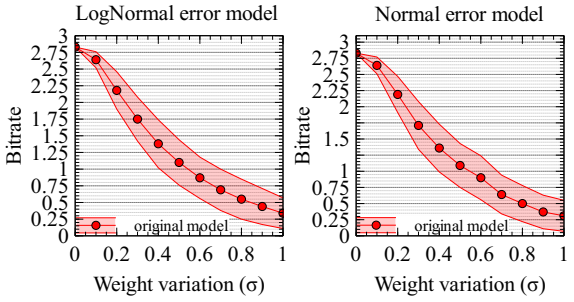

Fig. 1: NN soft demapper performance degradation

effectiveness of the proposed solution, the NN demapper architecture in[5] is adopted. The architecture of the NN demapper has 2 hidden layers, an input layer and an output layer. The input layer contains the currently received input signal and time delayed versions of the input signal in order to consider the memory effects of the channel and components. The architecture deploys a dual-side symmetric memory structure. The input layer has $2M+1$ neurons where M denotes the number of single side memory taps. The output layer has m neurons where m is the number of bits per real (imaginary) symbol, e.g., $m=3$ for the adopted 64-QAM. The number of neurons at each layer of the adopted NN is 17/16/10/3.

The performance metric is the achievable rate per real (imaginary) dimension. The achievable rate is calculated from the soft bit l and the actual transmitted bit b as follows[20]:

$$R = m - \min_{s\geq 0} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} \log_2[1+\exp(-s(2b_{ij}-1)l_{ij})] \quad (3)$$

where n is the total number of training symbols. l_{ij} and b_{ij} are the j-th soft bit output generated by the NN and the true output for the i-th input training symbol, respectively. The negative/positive sign of l_{ij} defines the decision of 0/1, respectively. The magnitude of l_{ij} defines the confidence level of the decision.

Figure 1 shows the mean values and the standard deviations of the achievable rate by the adopted model on the dataset, as described in the experimental setup section, under different levels of weight variations. The left figure shows the results with the log-normal model in equation (1) and the right figure shows the results with the normal model in equation (2). The solid lines in the middle of the ranges represent the mean values and the ranges represent the standard deviations. In this demonstration, 250 NN weights samples have been simulated at each σ value denoted on the x-axis. According to Figure 1, the achievable rate degrades significantly even with relatively small variations, which makes the NN demapper unusable in practice.

Proposed Robust IMC Accelerators

For a function f, the smallest value of the non-negative constant k for which the following constraint is satisfied:

$$|f(\mathbf{x_1}) - f(\mathbf{x_2})|_p \leq k\,|\mathbf{x_1}-\mathbf{x_2}|_p\,, \quad \forall \mathbf{x_1},\mathbf{x_2} \in X \quad (4)$$

is denoted as the Lipschitz constant of f, namely $L(f)=k$. f is called k-Lipschitz constrained[21],[22]. $|\cdot|_p$ is the p-distance metric between two vectors. The Lipschitz constant k describes how much a variation of an input is scaled at the output by this function. The composition of multiple Lipschitz constrained functions is also Lipschitz constrained as follows:

$$f = (f_l \circ f_{l-1} \circ ... \circ f_1)(x) \quad (5)$$

$$L(f) \leq k_l \cdot k_{l-1} \cdot ... \cdot k_1. \quad (6)$$

The forward propagation of a neural network can be considered a composition of successive layers. Accordingly, imposing Lipschitz constraints on each layer can prevent the variations in feature maps from being magnified when feature maps travel through layers in a neural network.

The transfer function of a layer in a NN can be written as the composition of f_i' and f_φ. $f_i' = \mathbf{w}\cdot\mathbf{x}+\mathbf{b}$ implements the multiplication of a weight matrix $\mathbf{w}$ and an input vector $\mathbf{x}$ and the addition with a bias vector $\mathbf{b}$. f_φ is the activation function, e.g., tanh. The Lipschitz constant of the tanh function used in[5] is equal to 1. Accordingly, we only need to restrict the Lipschitz constant of f_i' to suppress variation amplification through this layer as:

$$|(\mathbf{w}\cdot\mathbf{x_1}+\mathbf{b}) - (\mathbf{w}\cdot\mathbf{x_2}+\mathbf{b})|_p \leq\ k\,|\mathbf{x_1}-\mathbf{x_2}|_p \quad (7)$$

$$\Leftrightarrow \frac{|\mathbf{w}\cdot(\mathbf{x_1}-\mathbf{x_2})|_p}{|\mathbf{x_1}-\mathbf{x_2}|_p} \leq\ k \quad (8)$$

where x_1 is the nominal input to this layer and x_2 is the input affected by variations to this layer.

The function in (8) can be expressed further as

$$\sup\left(\frac{|\mathbf{w}\cdot(\mathbf{x_1}-\mathbf{x_2})|_p}{|\mathbf{x_1}-\mathbf{x_2}|_p}\right) = \|\ \mathbf{w}\ \|_p \leq k \quad (9)$$

where $\|\ .\ \|_p$ is the p-norm of a matrix. This function also shows that the variation propagation in a layer can be suppressed by constraining $\|\ \mathbf{w}\ \|_\mathbf{p}$. In the proposed method, we use the L^2 norm to bound $\mathbf{w}$ in each layer.

To provide flexibility for different layers in a neural network, we constrain the L^2 norm of the weight matrix for different layers with different values. For example, the L^2 norm of the ith layer is constrained with a value k_i during software training with the projection:

$$\phi(\mathbf{w_i}, k_i) = \frac{1}{max(1, \frac{\|\mathbf{w_i}\|_2}{k_i})}\mathbf{w_i} \quad (10)$$

where the max function is to guarantee that when the L^2 norm of $\mathbf{w_i}$ is larger than k_i, $\mathbf{w_i}$ will be scaled to make the L^2 norm of the scaled matrix to be k_i; Otherwise, $\mathbf{w_i}$ will remain unchanged. Since the rescaling of a weight matrix with (10) might lead to a performance degradation, the input signals are thus amplified to recover the performance of the NN.

To determine the best k for each layer, we use the particle swarm optimization (PSO) algorithm[23]. The PSO search is initiated with 20 particles and executed until the search converges. During the search, a candi-

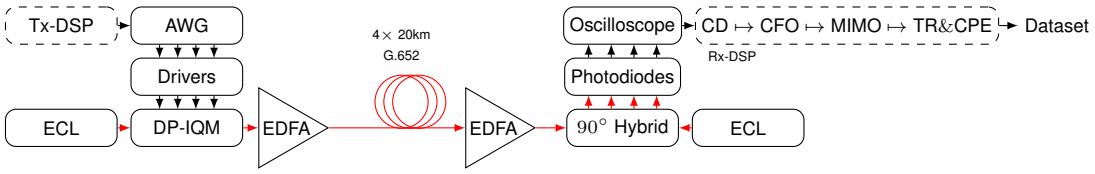

Fig. 2: Experimental setup over a 80 km G.652 fiber link at optimal launch power of 6.6 dBm. Chromatic disperion (CD) and carrier frequency offset (CFO) compensation, timing recovery (TR) and carrier phase estimation (CPE).

Tab. 1: Experimental results of Lipschitz constant constraint.

Error model	Bitrate		
	Original model		Proposed method
	$\sigma = 0$	$\sigma = 0.5$	$\sigma = 0.5$
LogNormal	2.83	1.1	2.54
Normal	2.83	1.09	2.51

date solution is evaluated by training the target NN with the Lipschitz constraints defined by the solution. The fitness of the solution is evaluated as follows:

$$fitness = \sum_{i=0,0.1,..,1} R_{\sigma=i} - std_{\sigma=i} \qquad (11)$$

where $R_{\sigma=i}$, and $std_{\sigma=i}$ are the mean and standard deviation of the achievable rate of the evaluated model at variation level $\sigma = i$, respectively. During each iteration, the particles are then updated until the search converges.

With the PSO search, we can obtain the best Lipschitz constraint setting and its trained model with respect to all variation levels. The performance of the Lipschitz constrained model for each variation level can be further boosted by fine-tuning the weights where variation-aware training at each specific σ value is conducted. During variation-aware training, weight variations are incorporated into the training process by sampling the variation model.

Experimental Setup / Data Acquisition

A coherent single carrier transmission over 80km G.652 fiber link at optimal launch power of 6.6 dBm is employed to experimentally evaluate the performance of the NN using IMC, as shown in Figure 2. The channel under test (CUT) carries a 80GBd DP-64QAM signal with gross data rate of 960Gb/s. Assuming 15% overhead for FEC and 3.47% overhead for training sequences, the net bit rate is 800Gb/s. At the transmitter, a constant amplitude zero auto-correlation (CAZAC) training sequence[24] is inserted for framing, carrier frequency offset and channel estimation. The electrical signals of the arbitrary waveform generator (AWG) are amplified by four 60GHz 3dB-bandwidth amplifiers. In the optical domain, two tunable 100 kHz external cavity lasers (ECLs) are used at the transmitter and receiver, respectively. The optically modulated signal is amplified by a booster EDFA. The receiver consists of an optical $90°$-hybrid and four 100GHz balanced photodiodes. The electrical signals are digitized by an oscilloscope with 256GSa/s and 110GHz 3dB-bandwidth.

Experimental results

To evaluate the effectiveness of the proposed framework, the NN demapper architecture in[5] is tested over the dataset as described previously. The variation models in (1), and (2) are used in the experiments. To evaluate the performance with the proposed method, weights in the neural network were sampled 250 times according to the variation models. In each sample, the achievable rate was evaluated with (3).

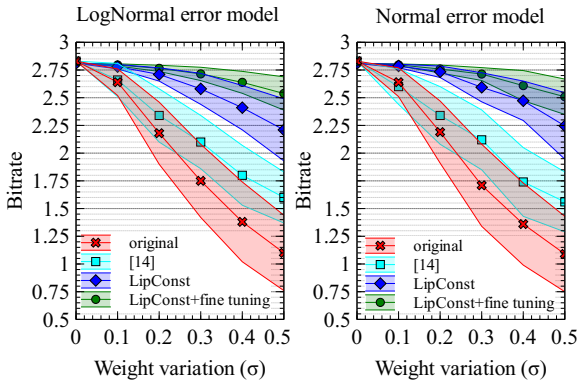

Fig. 3: Performance of the Lipschitz constrained model with and without fine-tuning vs the original NN and state of the art[14] at different variation levels

Table 1 summarizes the results showing the performance of the proposed method when σ in (1), and (2) was set to 0.5. This variation setting is already very large for RRAM cells[13],[14],[16]. The column $\sigma = 0$ in Table 1 shows the inference accuracy of the original NN model without variations. When variations with $\sigma = 0.5$ are applied to the weights in the original model, the achievable rate degrades from 2.83 down to as low as 1.09 in average. The last column shows the bitrate with the proposed method, which is similar to that of the original model without variations.

To demonstrate the effectiveness of the proposed method under different levels of variations, we compared the bitrate of the Lipschitz constrained network with fine-tuning with that achieved by the original network, the Lipschitz constrained network without fine-tuning and the method in[14]. The results are shown in Figure 3, where the solid lines represent the mean values and the ranges represent the standard deviations. According to this comparison, the proposed method can achieve the best performance under different variation levels.

Conclusion

A novel training method is proposed to counter device variations for a soft demapper NN implemented with IMC. The method is based on constraining the Lipschitz constants of NN layers. Experiments demonstrate that the achieved bit rate can be recovered from as low as 1.08 to 2.54 for the demapper NN in a 3 bit per symbol 64-QAM system.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. DOI: 10.1038/nature14539.

[2] O. Shental and J. Hoydis, ""Machine LLRning": Learning to softly demodulate", in *IEEE Globecom Workshops*, 2019. DOI: 10.1109/GCWkshps45667.2019.9024433.

[3] T. Koike-Akino, D. S. Millar, K. Parsons, and K. Kojima, "Fiber nonlinearity equalization with multi-label deep learning scalable to high-order DP-QAM", in *Signal Processing in Photonic Communications*, 2018. DOI: 10.1364/SPPCOM.2018.SpM4G.1.

[4] T. Koike-Akino, Y. Wang, D. S. Millar, K. Kojima, and K. Parsons, "Neural turbo equalization to mitigate fiber nonlinearity", in *European Conference on Optical Communication (ECOC)*, 2019. DOI: 10.1049/cp.2019.0803.

[5] M. Schädler, G. Böcherer, and S. Pachnicke, "Soft-demapping for short reach optical communication: A comparison of deep neural networks and volterra series", *Journal of Lightwave Technology*, vol. 39, no. 10, pp. 3095–3105, 2021. DOI: 10.1109/JLT.2021.3056869.

[6] M. Schaedler, S. Calabrò, F. Pittalà, C. Bluemm, M. Kuschnerov, and S. Pachnicke, "Neural network-based soft-demapping for nonlinear channels", in *Optical Fiber Communications Conference and Exhibition (OFC)*, 2020. DOI: 10.1364/OFC.2020.W3D.2.

[7] F. P. Guiomar, S. B. Amado, N. J. Muga, J. D. Reis, A. L. Teixeira, and A. N. Pinto, "Simplified volterra series nonlinear equalizer by intra-channel cross-phase modulation oriented pruning", in *European Conference and Exhibition on Optical Communication (ECOC)*, 2013. DOI: 10.1049/cp.2013.1450.

[8] M. Verhelst and B. Moons, "Embedded deep neural network processing: Algorithmic and processor techniques bring deep learning to iot and edge devices", *IEEE Solid-State Circuits Magazine*, vol. 9, no. 4, pp. 55–65, 2017. DOI: 10.1109/MSSC.2017.2745818.

[9] C. Li, M. Hu, Y. Li, H. Jiang, N. Ge, E. Montgomery, J. Zhang, W. Song, N. Dávila, C. E. Graves, *et al.*, "Analogue signal and image processing with large memristor crossbars", *Nature Electronics*, vol. 1, no. 1, pp. 52–59, 2018. DOI: 10.1038/s41928-017-0002-z.

[10] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars", in *International Symposium on Computer Architecture (ISCA)*, 2016. DOI: 10.1145/3007787.3001139.

[11] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory", in *International Symposium on Computer Architecture (ISCA)*, 2016. DOI: 10.1145/3007787.3001140.

[12] D. Niu, Y. Chen, C. Xu, and Y. Xie, "Impact of process variations on emerging memristor", in *Design Automation Conference (DAC)*, 2010. DOI: 10.1145/1837274.1837495.

[13] B. Liu, H. Li, Y. Chen, X. Li, Q. Wu, and T. Huang, "Vortex: Variation-aware training for memristor X-bar", in *Design Automation Conference (DAC)*, 2015. DOI: 10.1145/2744769.2744930.

[14] Y. Long, X. She, and S. Mukhopadhyay, "Design of reliable DNN accelerator with un-reliable ReRAM", in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2019. DOI: 10.23919/DATE.2019.8715178.

[15] Y. Zhu, G. L. Zhang, T. Wang, B. Li, Y. Shi, T.-Y. Ho, and U. Schlichtmann, "Statistical training for neuromorphic computing using memristor-based crossbars considering process variations and noise", in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020. DOI: 10.23919/DATE48585.2020.9116244.

[16] L. Chen, J. Li, Y. Chen, Q. Deng, J. Shen, X. Liang, and L. Jiang, "Accelerator-friendly neural-network training: Learning variations and defects in RRAM crossbar", in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2017. DOI: 10.23919/DATE.2017.7926952.

[17] G. Charan, J. Hazra, K. Beckmann, X. Du, G. Krishnan, R. V. Joshi, N. C. Cady, and Y. Cao, "Accurate inference with inaccurate RRAM devices: Statistical data, model transfer, and on-line adaptation", in *Design Automation Conference (DAC)*, 2020. DOI: 10.1109/DAC18072.2020.9218605.

[18] M. J. Rasch, D. Moreda, T. Gokmen, M. Le Gallo, F. Carta, C. Goldberg, K. El Maghraoui, A. Sebastian, and V. Narayanan, "A flexible and fast Pytorch toolkit for simulating training and inference on analog crossbar arrays", in *International conference on artificial intelligence circuits and systems (AICAS)*, 2021. DOI: 10.1109/AICAS51828.2021.9458494.

[19] Y. Zhang, G. He, G. Wang, and Y. Li, "Efficient and robust RRAM-based convolutional weight mapping with shifted and duplicated kernel", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 40, no. 2, pp. 287–300, 2020. DOI: 10.1109/TCAD.2020.2998728.

[20] G. Böcherer, P. Schulte, and F. Steiner, "Probabilistic shaping and forward error correction for fiber-optic communication systems", *Journal of Lightwave Technology*, vol. 37, no. 2, pp. 230–244, 2019. DOI: 10.1109/JLT.2019.2895770.

[21] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, "Parseval networks: Improving robustness to adversarial examples", in *International Conference on Machine Learning (ICML)*, 2017.

[22] H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree, "Regularisation of neural networks by enforcing lipschitz continuity", *Machine Learning*, vol. 110, no. 2, pp. 393–416, 2021. DOI: 10.1007/s10994-020-05929-w.

[23] J. Kennedy and R. Eberhart, "Particle swarm optimization", in *International Conference on Neural Networks*, 1995. DOI: 10.1109/ICNN.1995.488968.

[24] F. Pittala, I. Slim, A. Mezghani, and J. A. Nossek, "Training-aided frequency-domain channel estimation and equalization for single-carrier coherent optical transmission systems", *JLT*, vol. 32, no. 24, pp. 4849–4863, 2014. DOI: 10.1109/JLT.2014.2358933.