Area-Efficient Neural Network CD Equalizer for 4×200Gb/s PAM4 CWDM4 Systems

Bo Liu[1,*], Christian Bluemm[2], Stefano Calabrò[2], Bing Li[1], and Ulf Schlichtmann[1]

[1] *Technical University Munich, Chair for Electronic Design Automation, Munich 80333, Germany*
[2] *Huawei Technologies Duesseldorf GmbH, Munich 80992, Germany*

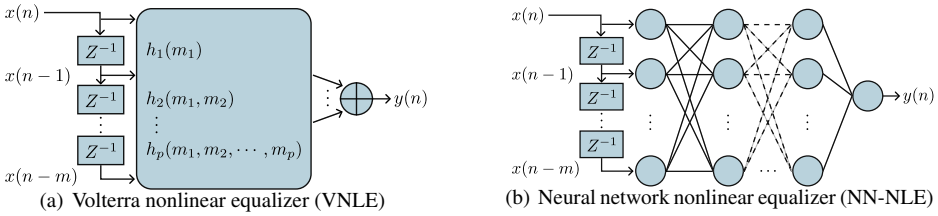
[*] *Author e-mail address: bo.liu@tum.de*

Abstract: We use a neural network trained jointly by multi-task learning on datasets acquired at multiple wavelengths to mitigate the impact of chromatic dispersion in 4×200Gb/s CWDM4 PAM4 transmission. By sharing a single set of weights among all involved wavelengths, while keeping the biases reconfigurable, we enable logic simplification of multipliers in the VLSI implementation of the neural network. Results show that the neural network equalizer achieves a similar BER compared with a Volterra equalizer with 71% reduction in hardware area.

1. Introduction

Transmission of optical signals in fibers suffers from non-linearity effects and chromatic dispersion (CD) in intensity modulation / direct detection (IM/DD) systems [1]. Such effects lead to intersymbol interference and bit errors. An effective countermeasure are Volterra nonlinear equalizers (VNLEs) with kernels $h_1 \cdots h_p$, as shown in Fig. 1(a). However, VNLEs suffer from huge computation complexity and numerical instability [2].

Recent literature has proposed neural network nonlinear equalizers (NN-NLEs) to realize nonlinear equalization and demapping [2, 3]. For instance, as shown in Fig. 1(b), in [3] NN-NLEs are used to compensate CD in IM/DD systems. Whereas the performance is promising, the complexity remains critical for high-speed communication systems. When the equalization target for a NN-NLE changes, for example when addressing different CD values, the NN-NLE requires reconfiguration. To allow for such reconfiguration, usually weights (multiplicative factors in neural network layers) are kept flexible, which is extremely costly in terms of chip area. To reduce the hardware cost of NN-NLEs, several authors have proposed solutions like pruning, weight clustering and quantization [4]. Here we propose a radical approach suited for VLSI (very large scale integration) implementation. We use multi-task learning (MTL) [5] to train a NN-NLE on datasets of multiple wavelengths together. We freeze and share the weights of the NN-NLE among all involved wavelengths to simplify the multipliers, and keep the biases reconfigurable for each wavelength to preserve the adaptability. We adopt our approach for the use case, considered in [3], of CD compensation in a CWDM4 4×200Gb/s system and achieve area savings in the order of 70%.



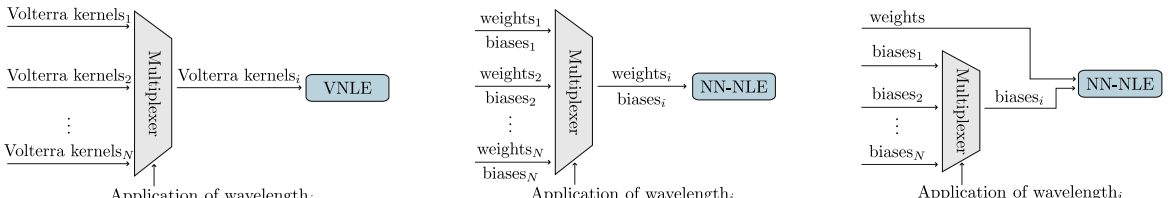
(a) Volterra nonlinear equalizer (VNLE)

(b) Neural network nonlinear equalizer (NN-NLE)

Fig. 1. Volterra nonlinear equalizer and neural network nonlinear equalizer.

2. Neural Network Nonlinear Equalizer

As explored in [2, 3], NN-NLEs proved to outperform VNLEs. Such VNLEs (NN-NLEs) are trained on datasets of different wavelengths separately as shown in Fig. 2(a) (Fig. 2(b)), leading to multiple sets of kernels (weights and biases). For a specific wavelength, a specific set of kernels (weights and biases) should be loaded into the hardware. Therefore, full hardware implementation of multipliers should be used to enable different parameters for different wavelengths. In addition, large on-chip memory is required to store the parameters.



(a) Wavelength-specific VNLEs require to switch between matching pre-defined sets of Volterra kernels.

(b) Wavelength-specific NN-NLEs conventionally train sets of matching weights/biases separately and switch between them.

(c) MTL-trained NN-NLE can match specific wavelengths by switching merely between biases and sharing a fixed set of weights.

Fig. 2. Wavelength-matching with a) VNLEs, b) weight/bias-switching NN-NLEs and c) bias-switching NN-NLEs with fixed weights.

3. Multi-Task Learning for Neural Network Nonlinear Equalizer

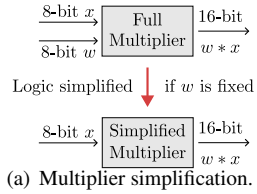
To reduce the hardware cost, as shown in Fig. 2(c), we use MTL to train a single NN-NLE on datasets of multiple wavelengths jointly to share the weights and keep the biases still flexible. Flexible biases equip the NN-NLE with the adaptability for different wavelengths, which is essential to achieve a low BER (bit error rate). On the other side, the shared weights are fixed to simplify the multipliers.

During MTL training, data from the different wavelengths are fed into the NN-NLE simultaneously. For each wavelength, the CD-specific loss L_i is weighted by a coefficient γ_i and the total loss is defined as

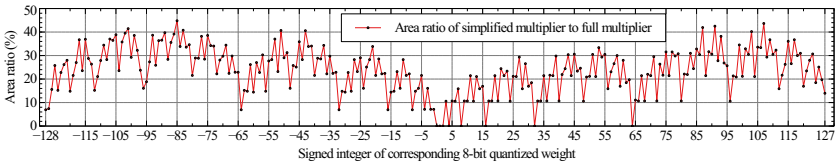
$$L_{total} = \sum_{i=1}^{N} \gamma_i \cdot L_i \tag{1}$$

where N is the number of different CD targets. We apply GradNorm [6] to adjust γ_i in each batch step. As shown in Fig. 2(c), the MTL-trained NN-NLE has only one set of weights and multiple sets of biases. Weights are shared among all involved wavelengths. On the other hand, biases are CD-specific for each wavelength. The flexibility allows specific neurons to be activated by adjusting their biases. Therefore, the NN-NLE with flexible biases can adapt to different CD values and achieve a lower BER. An interpretation of this approach is that we overlay multiple logic NN-NLEs on a single physical NN-NLE and we use the biases to enable the sub-NN-NLE we need. The key point is that flexible biases do not impose much area cost, because they are inputs to the adders, which in VLSI are much simpler than multipliers.

Our MTL-trained NN-NLE generates a single set of shared weights among all involved wavelengths. Since these weights are inputs to multipliers implementing the NN-NLE, the multipliers can be simplified significantly when the weights are fixed, as shown in Fig. 3(a). Fig. 3(b) demonstrates hardware area reduction of an 8-bit multiplier, with an average saving of 76.22%. The x-axis shows the frozen weight, and the y-axis shows the area ratio of the simplified multiplier to the full multiplier. These area savings were obtained using Design Compiler [7] for logic synthesis using 45nm process technology. According to Fig. 3(b), the area of the multipliers can be reduced significantly. In contrast, VNLEs and NN-NLEs trained with individual wavelengths still require full multipliers for hardware implementation and therefore cannot profit from any area reduction.



(a) Multiplier simplification.



(b) Area ratio for different 8-bit quantized weight after multiplier simplification.

Fig. 3. Illustration of multiplier simplification and area ratio for different weights.

4. Experimental Setup

Fig. 4 illustrates the IM/DD measurement setup and offline DSP for 10 km PAM4 transmission with 112 GBd per lane. The 3 dB bandwidths are written below each electrical/optical component. At the transmitter (Tx), pseudo-random binary sequences (PRBS) are Gray-mapped to PAM-4 symbols after duobinary precoding [3]. A raised cosine (RC) filter implements pulse shaping with roll-off factor of 0.14. After resampling, a 120 GS/s arbitrary waveform generator (AWG) converts the digital samples to analog signals which are amplified by a 60 GHz driver amplifier (DA) towards an O-band Mach Zehnder modulator (MZM). While the focus are standard O-band CWDM4 wavelengths 1270nm, 1290m, 1310nm and 1330nm, further captures at in-between wavelengths allow for better insights of the performance/wavelength relationship. After 10 km standard single mode fiber (SSMF) transmission, a variable optical attenuator (VOA) controls the received optical power (ROP) at the input of a praseodymium-doped fiber amplifier (PDFA) at 7 dBm. Thereafter, the broadband noise of the PDFA is suppressed by an optical filter. The filtered optical signal is fed to a photodiode (PD) whose electrical output is digitized by a 256 GS/s digital oscilloscope. At the receiver (Rx), timing recovery first operates at 2 samples per symbol (sps), and then the signals are downsampled to 1 sps for equalization. Modulo 4 [3] and PAM4 decisions are performed before BER estimation.

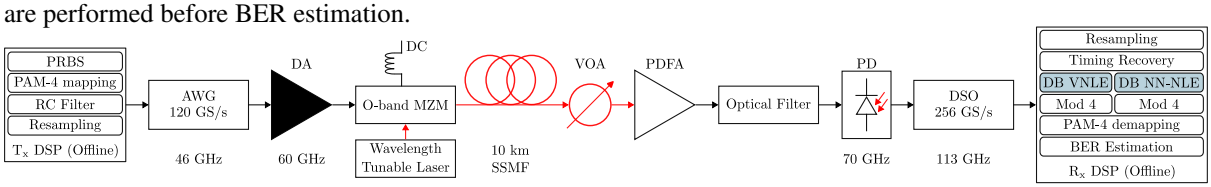


Fig. 4. Experimental 10km setup with offline DSP.

The design results for VNLE and NN-NLEs are shown in Table 1. VNLE and NN-NLEs applied in this paper are both trained on a duobinary (DB) target [3]. DB NN-NLEs are trained using MTL with 500 epochs to reach a low mean squared error. We use 182k PAM4 symbols for training and other 45k symbols for inference.

All multipliers have 8-bit input and 16-bit output. Inputs with larger bit-width are truncated to the 8 most significant bits (MSB). The bit-width of adders increases level by level to keep the carry bit. Uniform quantization is adopted. Lower bit-width also can be applied during hardware implementation, without impact on the proportional area savings.

5. Results and Discussions

Table 1, Fig. 5(a) and Fig. 5(b) show the hardware area and BER comparison between DB VNLE and DB NN-NLEs. The MTL-trained small DB NN-NLE achieves similar BER but has about 71% area reduction compared with DB VNLE. The MTL-trained large DB NN-NLE requires similar area as DB VNLE but achieves lower BER.

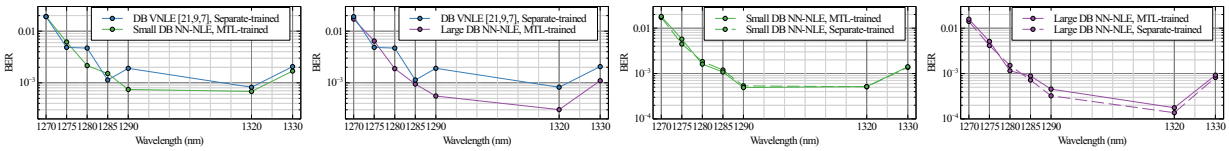
It is notable that the MTL-trained DB NN-NLEs achieves similar BER compared with the DB NN-NLEs trained for each wavelength individually, as shown in Fig. 5(c) and Fig. 5(d). However, the MTL-trained small and large DB NN-NLE achieves 63% and 67% reduction in hardware area respectively compared with separate-trained counterparts according to Table 1, because shared weights are used to simplify multipliers as described in Sec. 3.

Biases in our MTL training are still CD-specific. Fig. 6 shows the BER degradation when the biases in the DB NN-NLE are also fixed, which confirms the importance of the flexible biases.

Table 1. Design and hardware area evaluation results of DB VNLE and DB NN-NLEs.

Notation*	Design†	Training	Area (μm^2)‡			
			Multipliers	Adders	Activation	Total
DB VNLE	[21, 9, 7]	Separate	1.34E+05	1.42E+04	0.00E+00	1.48E+05
Small DB NN-NLE	21|11|7|1	MTL	1.48E+04	2.31E+04	4.98E+03	4.29E+04
Small DB NN-NLE	21|11|7|1	Separate	8.71E+04	2.31E+04	4.98E+03	1.15E+05
Large DB NN-NLE	41|23|11|1	MTL	4.61E+04	8.88E+04	9.41E+03	1.44E+05
Large DB NN-NLE	41|23|11|1	Separate	3.34E+05	8.88E+04	9.41E+03	4.32E+05

* VNLE and NN-NLEs applied in this paper are both trained on a DB target [3].
† The notation for DB VNLE indicates the number of memory taps of [1^{st}, 2^{nd}, 3^{rd}] order.
† The notation for DB NN-NLEs indicates the number of neurons in each layer. Each layer is fully-connected to the next one layer. The activation function of 1^{st}, 2^{nd}, 3^{rd} and 4^{th} layer is none, tanh, tanh and linear, respectively (1^{st} layer is the input).
‡ Area evaluation is conducted using Design Compiler [7] for logic synthesis using 45nm process technology.
‡ Activation functions are implemented using H-tanh, a low-cost variant of tanh [3].



(a) (b) (c) (d)

Fig. 5. BER comparison. (a) and (b) compare BER between DB VNLE and DB NN-NLEs. (c) and (d) compare BER between MTL and separate learning for DB NN-NLEs.

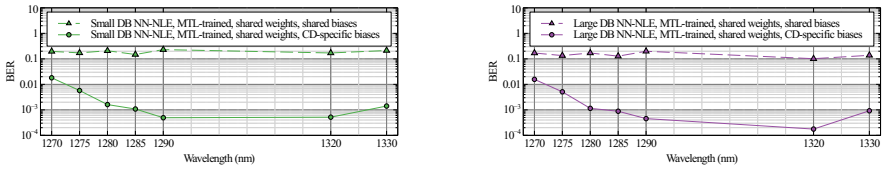


Fig. 6. BER comparison between shared biases and CD-specific biases.

6. Conclusions

We demonstrate an area-efficient neural network equalizer compensating CD at multiple wavelengths effectively. Thanks to the use of fixed weights, the multipliers implementing the neural network are simplified significantly, leading to a 71% reduction of hardware area in a representative VLSI implementation, while achieving a similar BER compared with a Volterra equalizer.

7. Acknowledgment

The work by Ulf Schlichtmann and Bing Li was partially supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 497488621.

References

[1] J. C. Cartledge, "Volterra Equalization for Nonlinearities in Optical Fiber Communications," in *Advanced Photonics*, 2017.
[2] C. Bluemm, *et al.*, "Equalizing Nonlinearities with Memory Effects: Volterra Series vs. Deep Neural Networks," in *ECOC*, 2019.
[3] C. Bluemm, *et al.*, "800Gb/s PAM4 Transmission over 10km SSMF Enabled by Low-complex Duobinary Neural Network Equalization," in *ECOC*, 2022.
[4] P. J. Freire, *et al.*, "Performance versus complexity study of neural network equalizers in coherent optical systems," in *Journal of Lightwave Technology*, 2021.
[5] S. Vandenhende, *et al.*, "Multi-task Learning for Dense Prediction Tasks: A Survey," *PAMI*, 2021.
[6] Z. Chen, *et al.*, "GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multi-task Networks," in *ICML*, 2018.
[7] P. Kurup, *et al.*, *Logic Synthesis Using Synopsys®*. Springer Science & Business Media, 2012.