Tyndall National Institute

Internship Report

Applying Machine Learning Techniques to extend the reach of IM/DD photonic communication channels

Thariq Shanavas
Senior Undergraduate,
Electrical Engineering (Major) and Physics (Minor)
Indian Institute of Technology, Bombay
Mumbai, Maharashtra, India

Internship duration: 14 May to 13 July 2018

Supervisor
Dr. Cleitus Antony
Photonic Systems Group
Tyndall National Institute
University College Cork
Ireland

Abstract

The project aims to develop machine learning techniques for nonlinear equalisation at the receiver end of an Intensity Modulation/Direct Detection fibre optic communication channel. The classification models discussed in this report are Feed forward neural networks, recurrent neural networks and Deep Convolutional networks. Techniques to train and visualise the network, effect of adding memory elements and overfitting in the context of communication networks are discussed. Experimental demonstration of 20Gbps NRZ signal over 30 Km is achieved using 10G-class optics. Performance of neural networks against varying hyperparameters is studied in detail. A few pitfalls of using neural networks that can lead to excellent yet misleading performance in laboratory conditions have been discussed in depth. The report concludes with the possible avenues for further research on this and related topics.

Acknowledgements

This project would not have been possible without the patient guidance of my supervisor, Dr. Cleitus Antony. I would like to thank Cleitus for the hours spent on patiently explaining, reviewing and critiquing the work contained in this document. I immensely enjoyed working under his guidance, and his advice and approach to research is will have a lasting impact on my future endeavors. Thanks again, Cleitus.

I'd also like to thank Dr. Paul Townsend, head of the Photonics Centre at Tyndall. Your review meetings have helped me stay on track, and your insightful remarks on my work gave me the direction I badly needed.

I'd also like to thank Stephen Murphy, member of the Photonic Systems group. Thanks, Stephen! Our discussions have influenced how this project turned out; your advice, tips and tricks on Machine Learning have been invaluable to my work.

I also extend my thanks to the Tyndall administrative staff and friendly HR for helping me navigate the legal nitty gritties, and helping me settle into this wonderful place smoothly.

Contents

Introduction

The project outlined in this report investigates whether machine learning can be used in the field of optical communication networks. The problem under investigation is one of classification: Can an equaliser at the receiving end of an optical communication network correctly classify the incoming bits into zeros and ones, taking into account the various nonlinearities introduced by dispersion, bandwidth limitation, thermal noise, etc.

Domain Knowledge

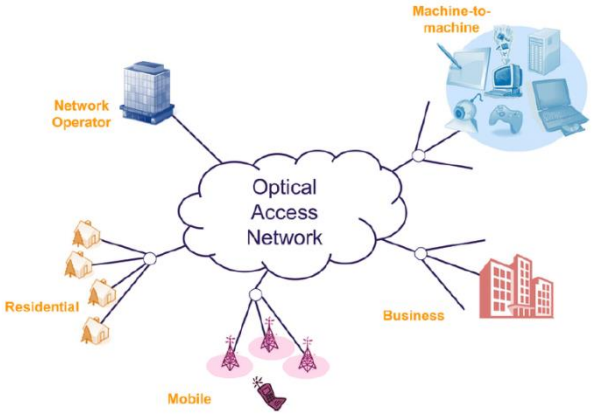
The primary aim of this project is to investigate the application of machine learning techniques to signal diagnostic tools in optical communications systems. It is important then to be made aware of which optical communications systems and what diagnostic tools are being examined. In the following sections these topics are elaborated on to give solid context to this project.

Optical Access Networks

The optical communications systems that this project concerns itself with are known as optical access networks.

Optical access networks are a fundamental component of the physical infrastructure of the modern internet (Kazovsky et al., 2007). Figure 1.1 gives an idea of what it is these networks do. In a nutshell they transfer data via optical fibers from the local optical line terminal which is connected to the wider internet, to individual homes and businesses. Optical access networks are the last leg of the physical layer of optical communication systems; the fiber optic cables and related components connecting homes and businesses to local optical line terminal units, and on to the wider Internet. Physical layer intelligence refers to the ability to actively monitor signal quality, and configure and extract information from various components of the access network. This is required to efficiently manage multiple services such as high bandwidth wireless 5G, optical residential and business broadband, as well as the Internet-of-Things (IoT) over a single optical access network.

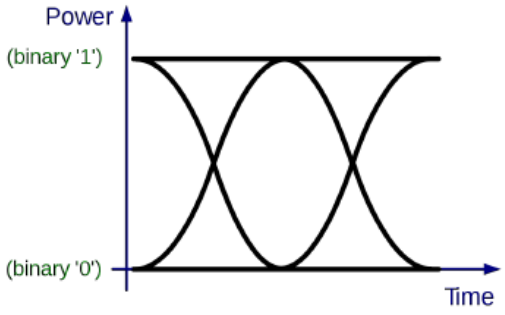
Next-generation optical access networks urgently need to be made smarter and more programmable to address dynamical network demands. Machine learning algorithms have the potential to be key building blocks in enabling physical layer intelligence in future optical communication networks. It is in this context that this current project, applying Deep Learning to optical signal diagnosis, is being carried out.



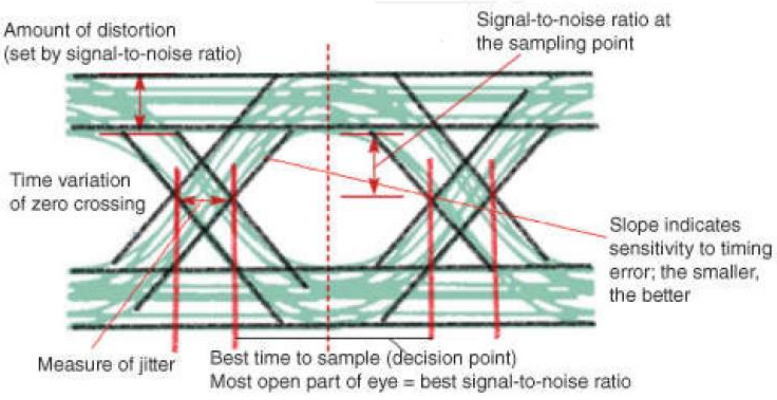
Distortions and Eye Diagrams

In order to examine the signal quality in a fibre optic cable component of an optical access network, a common diagnostic tool is the eye diagram. These are diagrams which contain a plethora of information about an optical signal, including impairments related to signal to noise ratio, dispersion, and bandwidth impairments. They get their names from the distinctive openings similar to an eye that occur in these diagrams.

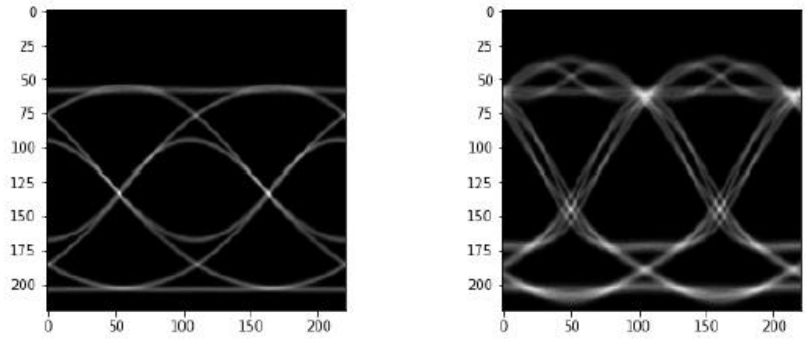
An eye diagram is created using an oscilloscope by repeatedly sampling an arbitrary data stream and then superimposing the positive and negative pulses over each other. Time lies along the x-axis, while amplitude/power lies along the y-axis which is used to encode the bit values 1 and 0. Every possible bit sequence should be captured by the eye diagram, such as 1 transitioning to 0, 0 to 1, 0 to 0 to 1, 1 to 0 to 1, etc. In this way the shape of the eye diagram and any distortions present can then be used to assess signal qualities and identify weaknesses in the system. A basic eye diagram shape is shown in the adjacent figure.



In a perfect world, eye diagrams would be rectangles, indicating instantaneous and perfect transitions between bit values. However in the real world it takes time for the system to relax from a bit value of 1 to a bit value of 0, and vice versa, resulting in distinctive eye patterns. Vertical eye opening shows the ability to distinguish between 1 and 0 value bits, while the horizontal opening gives the time period over which the signal can be sampled without errors.



Real world systems lead to different shapes in the eye diagram. Referring to the above figure, it's possible to see that distortion leads to a thicker upper line in the eye diagram, the slopes of the bit values transition indicate the systems tolerance of timing errors, etc. For example a system which is transmitting a long sequence of 1's followed by a single 0 bit value would be under stress, and how much of an effect this has on the overall signal quality can be assessed using an eye diagram, specifically looking at aberrations in the slope of the transition lines.



(a) Bandwidth impairment. (b) Dispersion impairment.

The above figure highlights the differences in the distortion induced by bandwidth limitation and fiber dispersion.

Pseudo Random Binary Sequences

A pseudorandom binary sequence (PRBS) is a binary sequence that can be deterministically generated and exhibits statistical behavior similar to a truly random sequence.

PRBS signals of interest in this document always have a fixed 'length'. For instance, a PRBS7 signal will be a sequence of seven bit numbers, serially unrolled. The rule for generating the next seven bits from the seven bits presently in memory is $x^7 + x^6 + 1$. (This is usually implementation dependent, but the 'update rule' is fixed for a given sequence).

The new seven bit number is stored into memory, and used to generate the next seven bits in the sequence. Similar update rules exist for PRBS15, PRBS31 and so on.

Linear shift registers are commonly used in hardware implementation of PRBS sequences, and the update rule is realized using logic gates. The architecture of linear shift registers allows the bits to be rolled out serially.

PRBS signals are commonly used in telecommunication systems as 'pilot signals' before the actual payload is transmitted. The pilot signal needs to be known at both the transmitter and receiver end, and needs to closely resemble the statistics of a truly random sequence.

However, the memory constrains prevent the use of any truly random sequence. Since PRBS sequences can be deterministically generated using limited memory and hardware constraints, they are preferred for pilot sequences. These pilot sequences may be used for receiver clock recovery or calibration of the ISI compensation systems (such as equalisers).

Reference: F. Jessie MacWilliams et al, IEEE vol. 64, No. 12, 1976

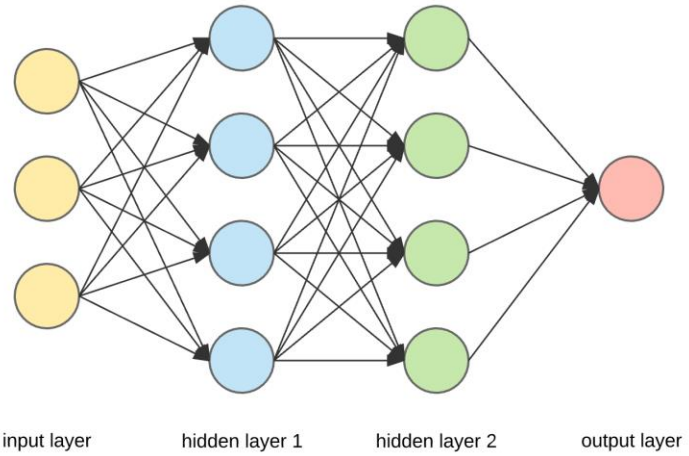
Neural Networks

Recently, Deep Learning has become a sort of 'buzz-word' across many industries due to its unrivalled success across fields such as computer vision, speech recognition, natural language processing, etc. The current project wishes to harness the power of Deep Learning for application in an optical communications setting.

Artificial Neural Networks

The architecture and central motivation of an artificial neural network is based on biological neuron activity, which can be read about in (Graupe, 2013).

An artificial neural network consists of layers of nodes, or neurons, with layers being connected sequentially, as in Figure. Each neuron receives input, and based on this calculates an output based on the type of artificial neuron that it is, which will be explained below. This output is then fed forward to the next layer, along with the output from every other neuron in the first layer and the process is repeated until the final layer is reached.



The first layer of an artificial neural network is known as the input layer, and quite naturally the final layer is known as the output layer. Layers of neurons which lie between the input and output layers are referred to as hidden layers, and are responsible for the majority of the networks computational power. Networks with many hidden layers are known as Deep Neural Networks, hence the term Deep Learning.

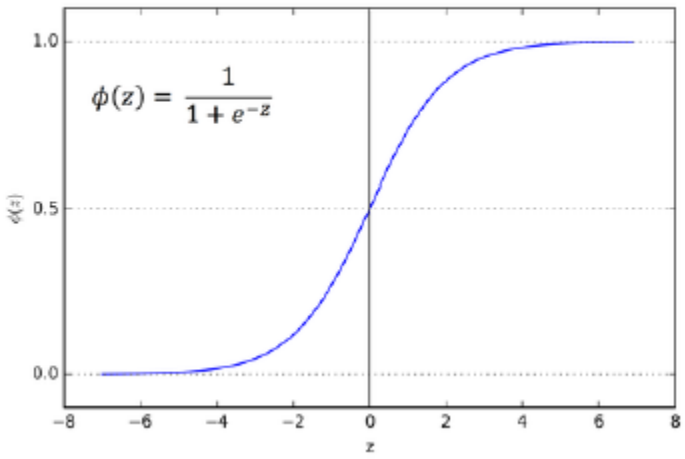
One of the most basic forms of neuron in an artificial neural network is the perceptron. Zooming in on a single neuron in the first hidden layer in the network shown in the figure, it is possible to see that this neuron will receive an input from each of the input nodes in the input layer, let's label these inputs as x1; x2; x3. If our selected neuron is a perceptron, then each input is multiplied by a real number known as a weight, which is intended to mirror the importance of the input to the desired output of the neuron. So for three inputs there will be three weights: w1, w2 and w3. Perceptrons have only two possible output values, 1 and 0, and are therefore binary. A threshold value is decided on such that the weighted sum of inputs must exceed this for the perceptron to output a 1. This is better phrased as

$$Perceptron\ Output = u(w.x + b), where\ u\ is\ the\ unit\ step\ function.$$

This function, however, is non-differentiable and leads to difficulties in the mathematical treatment. More complicated activation functions like the sigmoid makes the implementation of neural networks more elegant.

$$\sigma = \frac{1}{1 + e^{-z}}$$

$$z = w.x + b$$



The sigmoid activation function

Other popular alternatives are the tanh function, the Rectified Linear Unit (ReLU) or the Leaky ReLU. ReLU and leaky ReLU functions are popular for deep networks because of its versatility against the vanishing gradient problem. See X. Glorot et al for a thorough comparison.

None of this however allows an artificial neural network to learn. What is required is a way to set the weights and biases for each neuron in a network in such a way as to solve specific problems. This is achieved using learning algorithms, which take a set of labeled observations and allow a network to map the observations to their respective labels through iteratively decreasing a cost function associated with the output of the network and the target observation labels. Training an artificial neural network can be reduced then to an optimisation problem where the parameters being optimised are the weights and biases of every neuron in the network.

A cost function in a very basic sense takes the output of a network, and calculates how far from the intended value (in classification this would be the target class) this output is. By minimising the cost function over all of a training set, an artificial neural network can be made to learn to map patterns in input data to specific target classes.

Using gradient descent in conjunction with backpropagation allows the cost function to influence the weights of every neuron in an artificial neural network. Leaving $w_{ij}(t)$ represent the weight term between neurons i and j at discrete time t, C be the cost function, then the updated weights $w_{ij}(t + 1)$

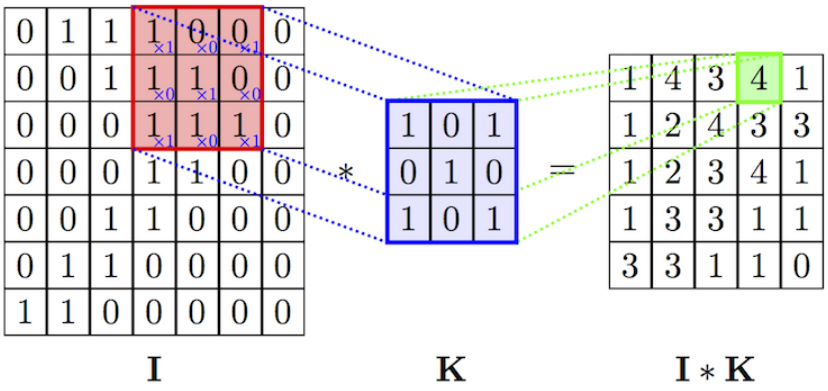
can be found using the gradient descent equation. The analytical derivative of the cost function over the network weight is computed using backpropagation algorithm.

$$w_{ij}(t+1) = w_{ij(t)} + \eta \frac{\partial C}{\partial w_{ij}}$$

Convolutional Layers

Although artificial neural networks as described above found major success in regression and classification applications, certain problems still presented a major obstacle to even these advanced networks. For example to feed the raw image data of a 220x220 pixel black and white picture, a neural network would need an input layer consisting of 48,400 neurons which requires ludicrous amounts of processing power. The challenges of image and audio recognition were largely overcome with the inception of the convolutional layer, a new class of network layer to be incorporated into the architecture of a neural network, see (Abdel-Hamid et al., 2014).

Convolutional layers are placed at the beginning of an artificial neural network, and are designed to extract feature information from an image such as edges. They do this using a convolution kernel, which it is assumed here encodes a method for feature extraction.



Convolutional kernel acting on an image

Mathematically expressed as,

$$(I \star K)_{xy} = \sum_{i=1}^{h} \sum_{j=1}^{w} K_{ij} \cdot I_{x+i-1,y+j-1}$$

Where h is the height of the kernel and w is the width. There are multiple kernels in a convolutional layer to extract different features from the input.

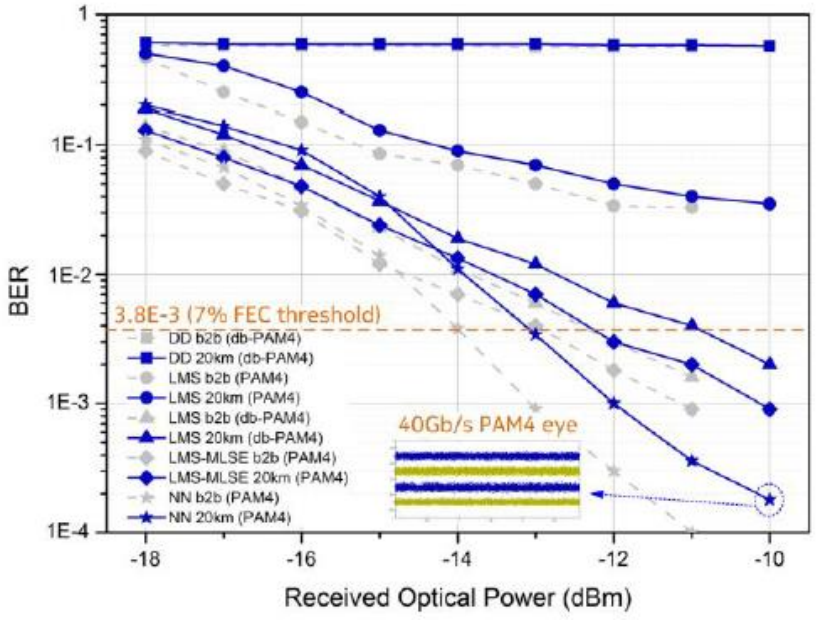
Literature Review

In recent years, the methods and techniques of machine learning and artificial intelligence have been gaining traction in the field of optical communications, that is communication systems based on light signals travelling through fibre optic cables. Machine learning techniques allow for the extraction of information from complex and highly non-linear processes, and provide means for reasoning about such systems and automating decision making. Optical communications systems stand to gain much in terms of performance, efficiency, and increased system capacity through adoption of the latest techniques in machine learning.

The state of the art in neural network based equalisation as of July 2018 has been demonstrated through 50Gbps IM/DD PAM4 transmission over 20Km with 10G-class optics using feedforward networks by C. Ye et al at ECOC2017 and 56 Gbps IM/DD PAM4 transmission over 25Km with 10G-class optics using convolutional neural networks by P. Li et al at OFC2018. The same group by P. Li demonstrated 100Gbps IM/DD transmission using PAM8 and PAM16 transmission over 25Km with 20G-class optics using convolutional networks.

In (C. Ye, 2017), a feed forward neural network with 8 (presumably symbol spaced, not explicitly mentioned anywhere) input taps and one hidden layer with 10 nodes has been compared against a feed forward equaliser with similar number of input taps. Frequency chirp is expected to be minimal since an external Mach-Zehnder modulator (MZM) was used in place of a directly modulated laser (DML).

The paper used PRBS15 sequence of length 2^{15}-1, with the first 500 symbols reserved for training. Notably, the NNE was not compared against the decision feedback equaliser.

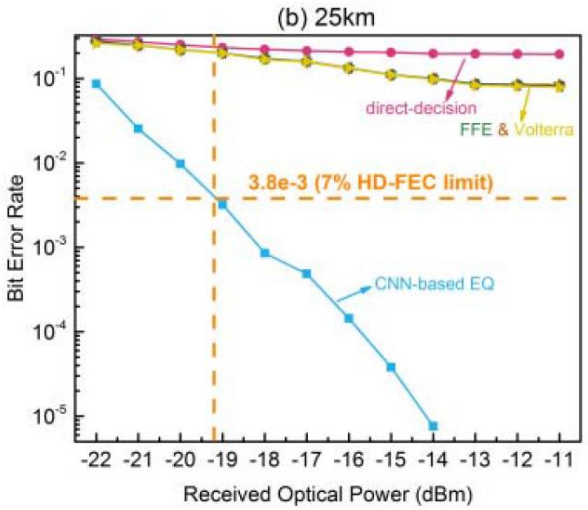


The paper also reports some additional improvement in both FFE and NNE when using MLSE, but the basic premise of NNE outperforming FFE holds.

In (P. Li et al, OFC2018), a convolutional network with two convolutional layers followed by three fully connected layers has been used.

The input layer takes 101 symbol spaced taps. The first two layers are 1D convolutional layers with one input channel to six output and six input to 16 output each, both of which have a sliding window size of five and are followed by a nonlinear activation layer of ReLU. After the convolutional layers, the output is unrolled and fed into three consecutive linear layers with 1688 input to 120 output, 120 input to 84 output and 84 input to four output respectively. The first two linear layers are followed by a ReLU layer while the third is followed by Softmax layer, whose four output units denote the four PAM4 symbols. The network is trained by mini-batch gradient descent. PRBS15 is used for training and validation.

A 10G-class O-band DML is used to generate the PAM4 signals at 28Gbaud or 56Gbps. The signal is converted back to electrical domain using a 10G-class photodiode. Very significant improvement over FFE and Volterra is reported.



Notably, the paper uses 101 input taps, which is rather unrealistic in practice.

The paper (T. Eriksson et al) describes the possible pitfalls of using neural networks in optical systems. The paper describes how neural networks are prone to predicting the sequence used for training rather than forming a nonlinear decision boundary.

For instance, if the training and payload sequences are PRBS15, then an equaliser with information on the previous 15 or more samples can use the nature of PRBS to predict the current symbol. In such a scenario, the equaliser reaches artificially low BER but performs worse than simple threshold based detection if the training and payload sequences are different. One way to prevent this is to limit the number of input taps keeping in mind the training sequence. For example, if the training sequence is PRBS15, the input should never be more than 31 symbol-spaced taps long. (The reference is the 16th tap in case of 31 input samples. This contains the entire previous 15 bit segment. The network then learns the relation between the last 15 bits and the present bit)

Methodology

In this section, the experimental setup and equaliser structure is described.

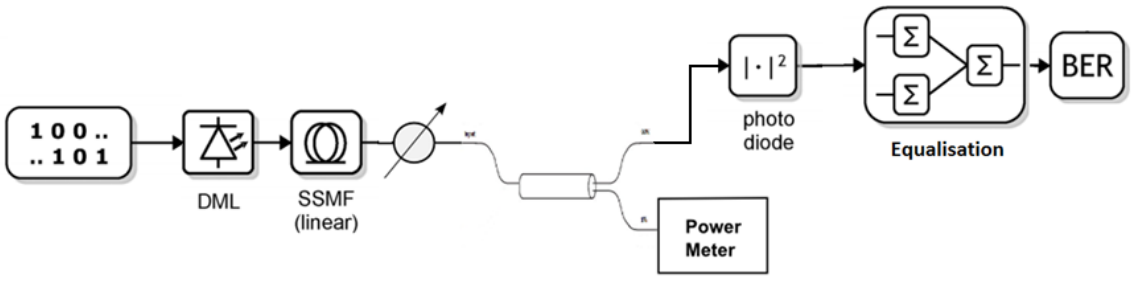
Experimental Setup

An Anritsu MP1800A Pattern Generator used to generate two coherent NRZ signals at 10G each, which is then multiplexed to 20G using Anritsu MP1821A MUX. This then drives an 18G-class DML (EX-DY7002) through a Bias-T after suitable amplification to match recommended power levels. The DML is biased at 1.9V.

The optical signal at 1550nm is sent through a 30.5Km spool of fibre, and suffers 0.2 dB/Km attenuation and 18.2ps/nm-Km dispersion. At the receiving end, the fibre is connected to a 10G-class photodiode (DSC-R402) through an Agilent 81571A optical attenuator module. The photodiode is coupled to a power meter (Agilent 81635A) via a 99:1 optical coupler.

The RF output of the photodiode is captured by a LeCroy Real Time scope (50GHz Bandwidth). The traces are then processed offline. To cross-check the Bit Error Rate measurements, one sweep was taken by the Agilent N4901B Serial BERT.

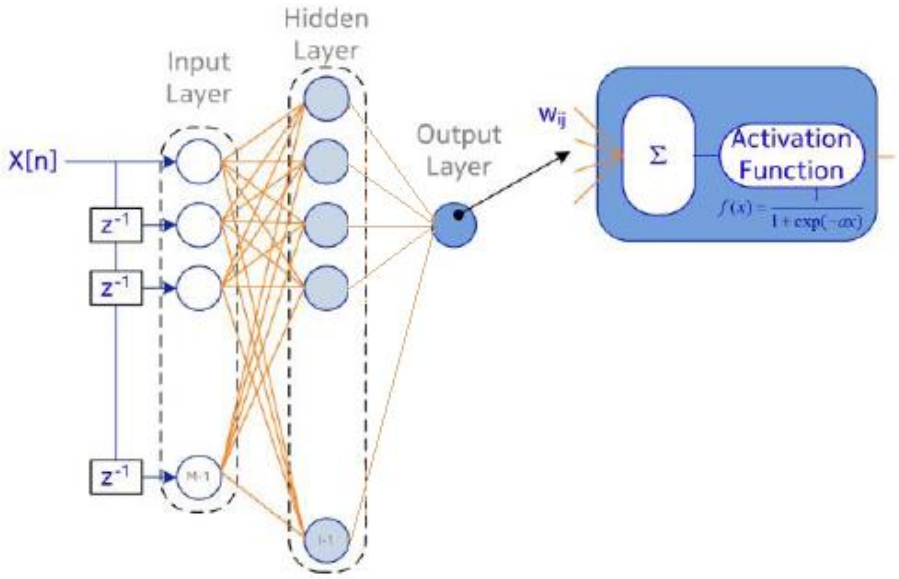
The captured traces are processed using a Feed Forward Equaliser, Decision Feedback Equaliser, Feed Forward Neural Network and Convolutional Neural Network. The bit error rate achieved by the various equalisers against bandwidth and dispersion limited cases have been studied.



Equaliser Structure

The incoming signal is sampled before sending it to the equaliser. There may be one, two or more samples per symbol. For instance, if the incoming bitstream is at 10Gbaud, if the sampling happens at 10GHz (with suitable clock recovery, so that the sampling does not happen at the transition edge), the signal is said to be sampled at 1 sample per symbol. Likewise, if the sampling happens at 20GHz, the signal is said to have two samples per symbol.

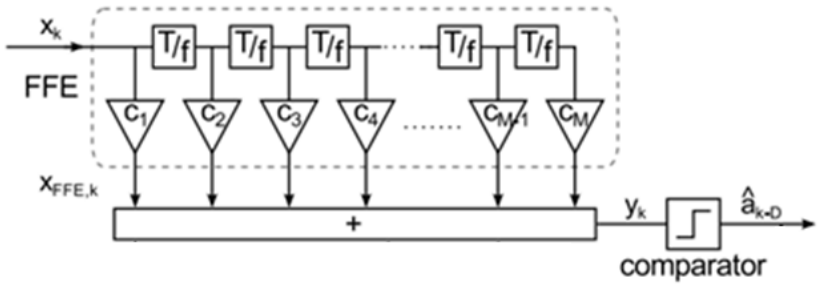
The input size of the equaliser is taken to be 20 samples irrespective of the number of samples per symbol in this report, unless mentioned otherwise.



In the above figure, x[n] represents the incoming bitstream. The current value x[n] forms the first element in the input vector for the equaliser. x[n] is delayed by one sample to get the second input, i.e. x[n-1].

The input is likewise delayed by 19 samples to form the input vector for the equaliser. Here, x[n-10] is taken to be the reference tap. i.e. the desired output of the equaliser is the bit value the x[n-10]th sample corresponds to. This is to account for the fact that due to pulse spreading, the future bits might have some impact on the current bit: A delay of 10 samples provides sufficient room for the equaliser to adjust for ISI.

Among the most commonly used equalisers are the Feed Forward equalisers.



Feed Forward Equaliser (FFE)

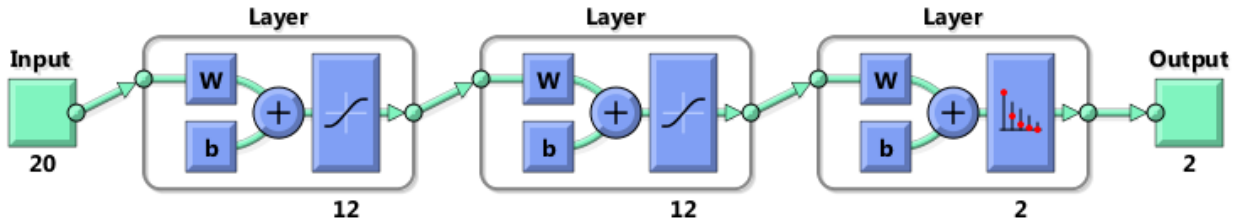
The FFE essentially adjusts the weights c_n and adds it up before comparing it with a hard decision boundary. The FFE is essentially linear, and at the optimal state, attempts to create a finite impulse response (FIR) filter that approximately inverts the channel transfer function.

This poses several issues – for instance, the channel can be modeled by a transfer function only when the channel response is linear and time invariant (LTI system). This is often not the case, as distortions introduced by dispersion in the fibre and chirp from the DML are inherently nonlinear. FFE can have limited performance in this case.

Since machine learning allows us to easily learn complex nonlinear decision boundaries, neural networks are a suitable alternative to linear equalisers.

Neural Network Equaliser

For the Feed Forward Equaliser, we have used a network with two hidden layers, activated by the tanh function. There were 12 nodes in each hidden layer. The output layer had two nodes and was activated by the softmax function. For simulations with PAM4, the Output layer had 4 nodes, activated by softmax.



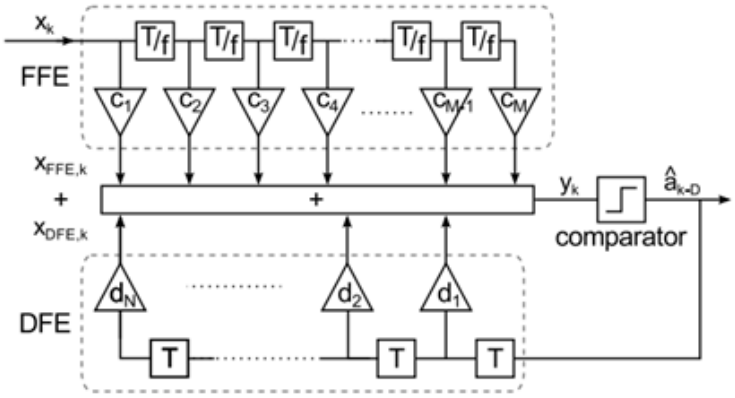
Feed Forward network structure. w indicates the weight matrix, b indicates the bias value. Numbers below each layer indicate layer size.

The network was trained in batch mode with Conjugate Gradient descent using backpropagation. Tanh function was preferred in the hidden layer for its fast training characteristic for shallow networks. Softmax function in the output layer was chosen for its probabilistic interpretation of activation when trained using Cross-Entropy loss function.

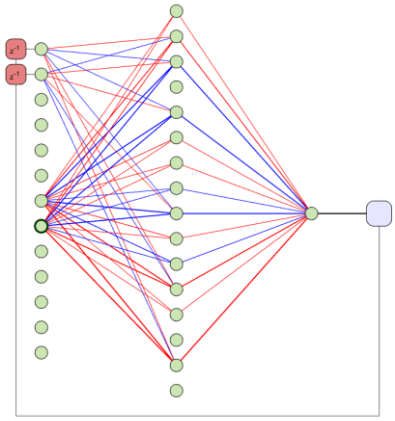
The dataset available for training is divided into training set (70%), validation set (15%) and test set (15%). The network is trained iteratively on the training set, until the gradient of the cost function falls below a threshold (1e-6), number of iterations exceeds a preset number (1000) or the error in the validation set increases steadily for the past 6 iterations. The rationale behind 'early stopping' by cross validation error is that if the network performs increasingly worse on a separate dataset, it is strongly indicative that the network is overfitting the training set. It has been observed that under noisy conditions, the training ususally exits by increasing validation error.

Decision Feedback Neural Networks

To further improve the performance of the neural network, past decisions were fed back into the input vector. This led to considerable performance improvement (to be discussed in a later section).



To further understand the nature of the network, especially in the presence of decision feedback, we developed a simple tool to visualise the weights of a feed forward network. The left most layer is the input layer, with the sixth from the bottom being the reference tap. (This is just another way of saying the signal is intentionally delayed by six samples). The top two values in the input vector correspond to the decision feedback.



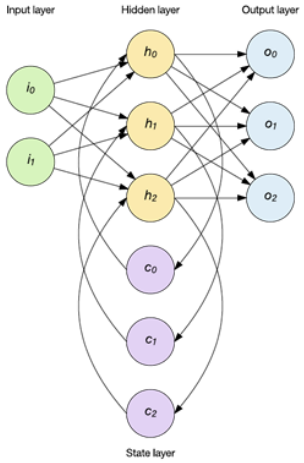
DF-NNE

Only the strong connections are shown by the tool. Also note that since this is a binary classification problem (for NRZ) we could use one logistic sigmoid activated unit at the output to simplify the diagram, without sacrificing performance. The past decisions have been observed to be strongly propagated into the next layers.

Comparison with Recurrent Network

In the machine learning community, a decision feedback element is usually introduced in neural networks by the use of recurrent networks.

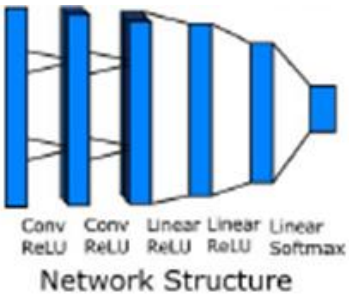
Recurrent networks are popular in applications that require long dynamic response – for instance speech recognition or text summarization.



We have implemented recurrent neural networks for equalisation, but it was not found to give significant performance boost over feed forward equalisers – presumably because of the vanishing gradient problem and the other difficulties in training a recurrent network. Moreover, it is much harder to implement true recurrent networks in an ASIC chip because the memory units need to store floating points, compared to bits for decision feedback. Owing to the practical reasons, we did not further investigate recurrent networks.

Convolutional Neural Network

The feedforward network was compared against a six layer convolutional neural network. The layers are described in more detail below.



Network Structure

Input Layer: The input layer of the convolutional network is a 3D matrix of size $1 * (Input\ Size) * 1$ where $Input\ Size$ is the number of input taps.

Convolutional Layer: The input layer is followed by the first convolutional layer with 20 learnable filters, each of size 1*5*1. Each of these filters is convolved with the input layer to create an output volume of size $1 * (Input\ Size - 5 + 1) * 20$. Note that $(Input\ Size - 5 + 1)$ is the length of a convolution between two vectors of size (Input Size) and 5.

Batch Normalisation Layer: The convolutional Layer is followed by a Batch Normalisation Layer. This layer normalizes the output of the first convolutional layer before applying the nonlinear activation function by subtracting the mean and dividing by standard deviation. This significantly increases training speed and reduces dependence on random initialization. See (Ioffe et al for more details)

ReLU activation layer: The output of the Batch Normalisation layer is sent through a ReLU function. ReLU is preferred for deep neural networks due to its resilience against the Vanishing Gradient problem.

Convolutional Layer: The ReLU layer is followed by the second convolutional layer with 10 learnable filters, each of size 1*5*20. Each of these filters is convolved with the output of the previous layer to create an output volume of size $1 * (Input\ Size - 8) * 10$.

Batch Normalisation Layer

ReLU activation layer

Fully Connected Layer: The fully connected layer 'rolls out' the output of the preceding layer. It is then fully connected to the next layer of size 10.

Batch Normalisation Layer

ReLU activation layer

Fully Connected Layer: This layer fully connects the output from the previous layer (size 10) to another layer of size 10.

ReLU activation layer

Fully Connected Layer

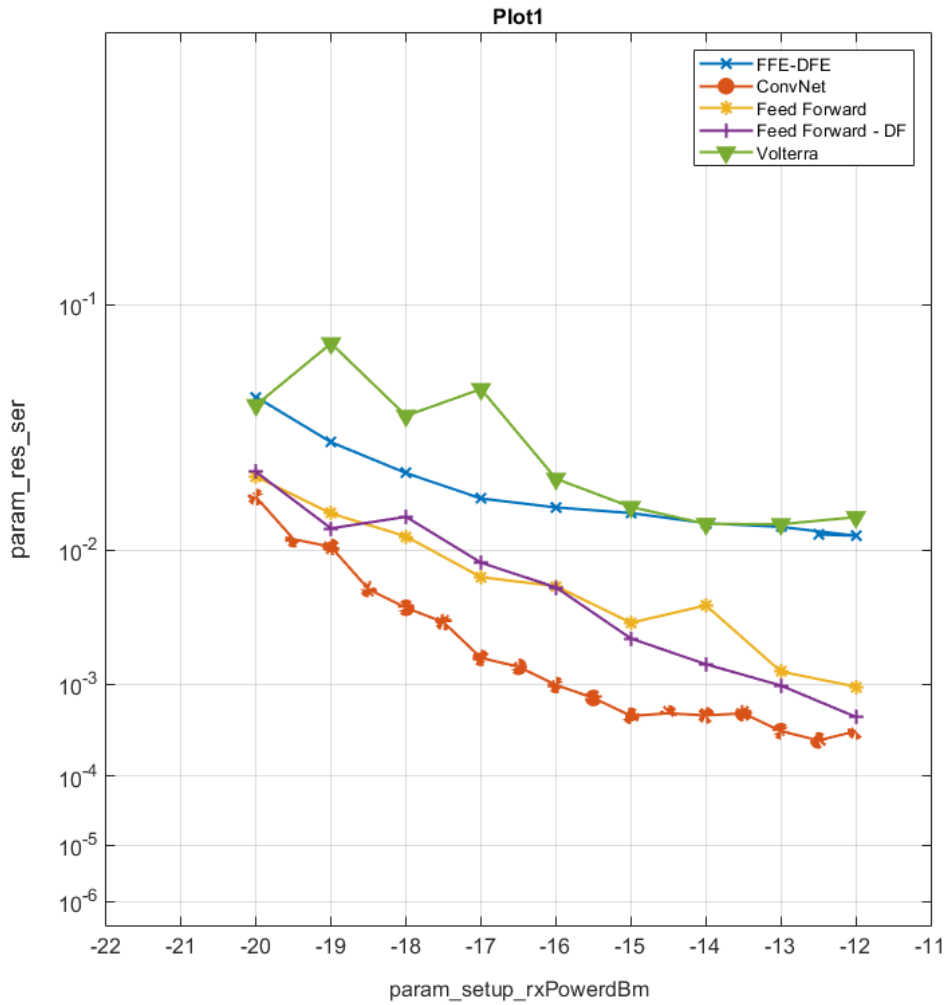
Softmax Layer: The final layer is activated with the softmax function. There are two nodes in the final layer for NRZ and four for PAM4.

The network is trained using **stochastic gradient descent with mini batches**. The convolutional Neural Network had sufficient learning capacity that providing past decisions provided no practical improvement.

Results and Discussion

This Section presents the results and discussion on the observations.

Results



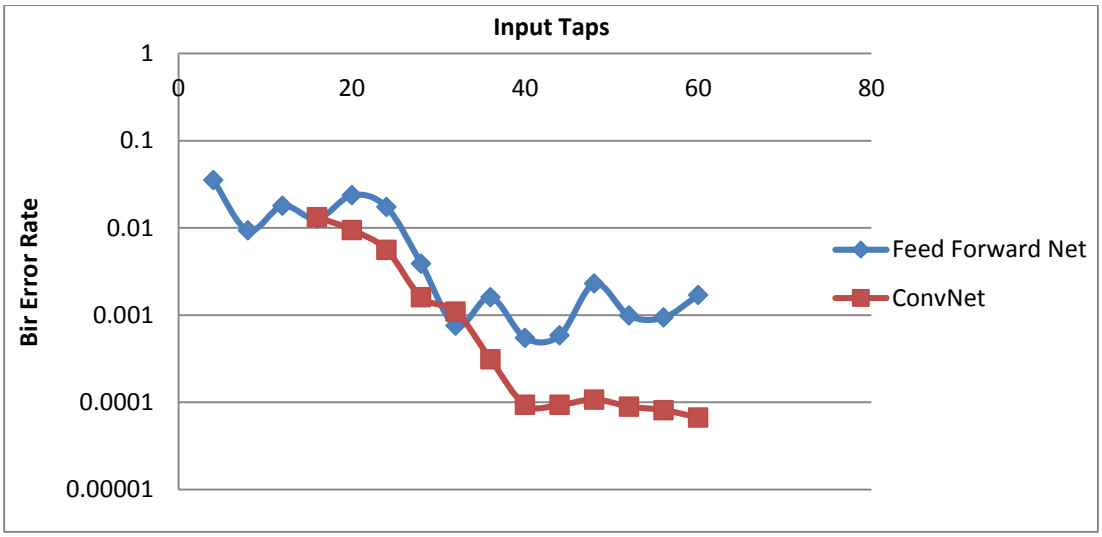
This data has been taken for 20Gbps signal sent over 10G-class optics with 30Km fibre. The convolutional network was found to outperform all other equalisers and with 20 input taps, and two samples per symbol. The FEC limit was reached at around -16dB received power. The training sequence was PRBS31 and payload was PRBS15.

Discussion

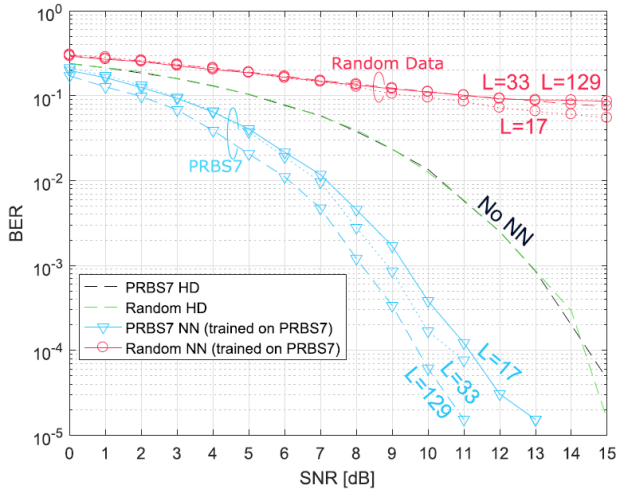
The primary goal of this project was to explore the applications of neural networks in equalisation. To achieve this, a feed forward neural network, feed forward neural network with decision feedback and convolutional network has been tested against those already well documented in the literature. The following topics cover some of the intricacies of the network.

Overfitting to the Training sequence

As highlighted in (T. Eriksson et al), if the input length is above approx. 33 symbol spaced taps is the training sequence is PRBS15, the network starts to learn the nature of the sequence rather than form a nonlinear decision boundary. To validate this claim, we have observed the BER as a function of input size for a 20Gbps NRZ sequence with one sample per symbol, trained and validated on PRBS15.



The sudden decrease in BER beyond 30 input taps can be attributed to the network learning the PRBS sequence. Similarly, if the network is trained on PRBS 15 and validated on a different sequence, the error rate increases dramatically beyond input size around 30.

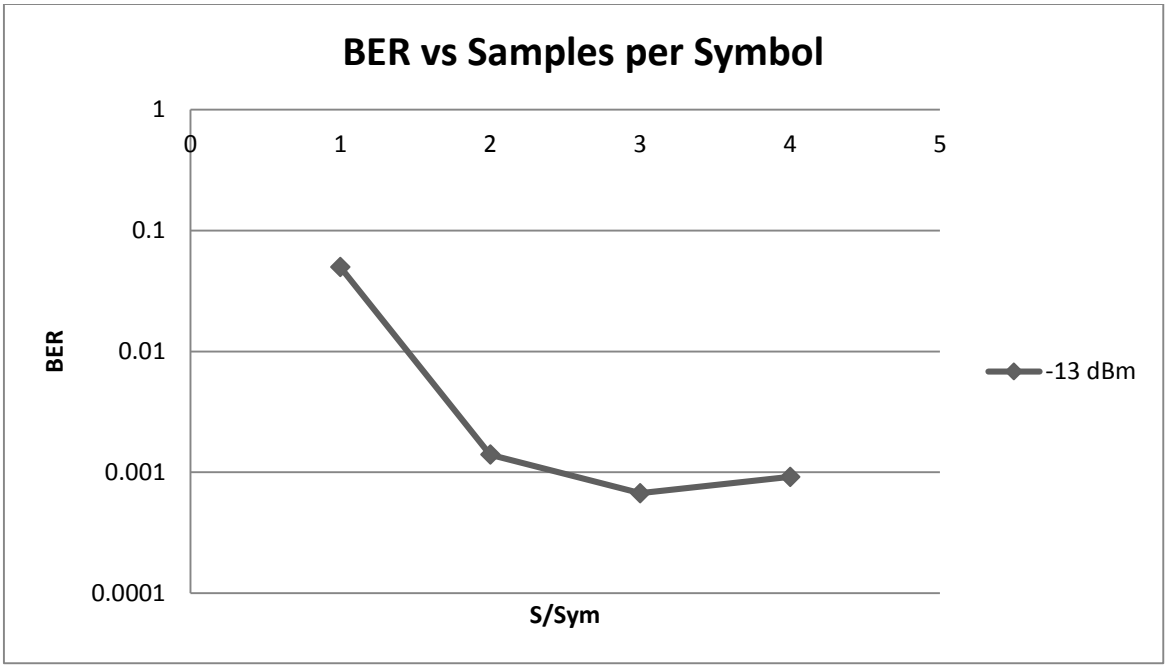


BER as a function SNR for hard decision on a PRBS7 and a random pattern, and for a neural network trained on a PRBS7 with either PRBS7 or random data as input. The input length to the neural network is 17, 33 or 129 bits symmetrical around the center estimated bit.

In practice, care must be taken to ensure that the input size is less than the training sequence PRBS length.

Effect of Samples per Symbol

It was observed that increasing the samples per symbol gave considerable improvement in performance. The following data was taken using a Convolutional Network equaliser, 20Gbps NRZ over 30 Km fibre. The number of samples per symbol was varied, keeping the total number of input samples at 20.



Scope for future work

To realize neural networks in hardware, it is necessary to vastly reduce the number of taps. This can be done by pruning, and is currently an area of research. The visualization tool developed during the course of this project can assist in understanding and provide insights into the pruning process.

The output of the convolutional layers in the CNN is definitely worthy of exploration. I suspect that the first convolutional layer would pick up rising and falling edges, and that the second convolutional layer might pick up longer meta-symbols (eg, 101 transition). It might be possible to vastly simplify the network by applying some basic filters to the data before applying the convolutional layers. The nature of these filters can draw inspiration from the ones that the currently implemented deep CNN learns in its first few convolutional layers.

Decision feedback has been observed to give a 1dB or so improvement in dispersion limited cases for neural networks - this is another avenue worthy of exploration. The recurrent network which I have implemented did not quite perform better than a simple feed forward network. It was also understandably more computationally difficult to train. I suspect that my implementation of RNN was getting stuck in local minima, or possibly ran into vanishing gradient problems. The RNN should ideally work at least as well as the DF-NNE.

At the systems level, the DML chirp is strongly dependent on the extinction ratio – which was currently set to very high (approx. 8dB). There might be some value in decreasing the extinction ratio and performing these tests again at reduced DML chirp.

Extending these results top PAM4 could effectively double the data rate. PAM4 has so far only been tested in simulation, and experimental demonstration could lead to interesting results.

References

1. Ye, C., Zhang, D., Huang, X., Feng, H., & Zhang, K. (2017). Demonstration of 50Gbps IM / DD PAM4 PON over 10GHz Class Optics Using Neural Network Based Nonlinear Equalization, 4–6. https://doi.org/10.1109/ECOC.2017.8346196
2. Eriksson, T. A., Bulow, H., & Leven, A. (2017). Applying Neural Networks in Optical Communication Systems: Possible Pitfalls. *IEEE Photonics Technology Letters*, *29*(23), 2091–2094. https://doi.org/10.1109/LPT.2017.2755663
3. Warm, S., Bunge, C., Wuth, T., & Petermann, K. (2009). Electronic Dispersion Precompensation With a 10-Gb / s Directly Modulated Laser. *IEEE Photonics Technology Letters*, *21*(15), 1090–1092. https://doi.org/10.1109/LPT.2009.2022957
4. Li, P., Yi, L., Xue, L., & Hu, W. (2018). 56 Gbps IM / DD PON based on 10G-Class Optical Devices with 29 dB Loss Budget Enabled by Machine Learning. *2018 Optical Fiber Communications Conference and Exposition (OFC)*, *1*, 3–5. https://doi.org/10.1364/OFC.2018.M2B.2
5. Li, P., Yi, L., Xue, L., & Hu, W. (2018). 100 Gbps IM/DD Transmission over 20Km SSMF using 20G-class DML and PIN enabled by machine learning, *2018 Optical Fiber Communications Conference and Exposition (OFC)*
6. Kazovsky, L. G., Shaw, W. T., Gutierrez, D., Cheng, N., & Wong, S. W. (2007). Next-generation optical access networks. *Journal of Lightwave Technology*, *25*(11), 3428–3442. https://doi.org/10.1109/JLT.2007.907748
7. Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. https://doi.org/10.1007/s13398-014-0173-7.2
8. MacWilliams, F. J., & Sloane, N. J. A. (1976). Pseudo-random sequences and arrays. *Proceedings of the IEEE*, *64*(12), 1715–1729. https://doi.org/10.1109/PROC.1976.10411
9. Graupe, D., 2013. Principles of artificial neural networks, 3rd edition. ed, Advanced series on circuits and systems. World Scientific, New Jersey
10. X. Glorot et al, Deep Sparse Rectifier Neural Networks https://www.utc.fr/~bordesan/dokuwiki/_media/en/glorot10nipsworkshop.pdf
11. Ossama Abdel-Hamid Abdel-rahman Mohamed, H. J. L. D. G. P. D. Y. (2014). Convolutional Neural Networks for Speech Recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *22*(10), 1533–1545. https://doi.org/10.1109/TASLP.2014.2339736