Deep learning methods for nonlinearity mitigation in coherent fiber-optic communication links

Vladislav Neskorniuk

Doctor of Philosophy

Aston University
November 2022

Table of contents

List of abbreviations

Acronyms / Abbreviations

ADC	Analog-to-digital converter
ANN	Artificial neural network
APSK	Amplitude phase shift keying
ASE	Amplified spontateous emission
AWGN	Additive white Gaussian noise
BCH	Bose–Chaudhuri–Hocquenghem
BICM	Bit-interleaved coded modulation
BMI	Bit-wise mutual information
CDC	Chromatic dispersion compensation
CS	Constellation shaping
CW	Continuous waveform
DA	Data augmentation
DBP	Digital back-propagation
DM	Distribution matcher
DNN	Dense artificial neural network
DP	Dual-polarization
DPD	Digital pre-distortion

DSP	Digital signal processing
E2E	End-to-end
EA	Electrical amplifier
EGN	Enhanced Gaussian noise
FDE	Frequency-domain equalizer
FE	Feature extractor
FEC	Forward error correction
GAN	Graph-adversarial network
GCS	Geometrical constellation shaping
GD	Gradient descent
GMI	Generalized mutual information
GVD	Group velocity dispersion
HD-FEC	Hard-decision forward error correction codes
I	In-phase
i.i.d	Independent and identically distributed
ICR	Intergrated coherent receiver
JS	Joint constellation shaping
KDE	Kernel density estimator
LDBP	Learnt digital back-propagation
LDPC	Low-density parity check
LEAF	Large effective area fiber
MAE	Mean absolute error
MB	Maxwell-Boltzmann
MI	Mutual information

ML	Machine learning
MSCS	Multi-symbol constellation shaping
MSE	Mean squared error
MZM	Mach-Zehnder modulator
NLC	Nonlinearity compensation
NLE	Nonlinearity equaliser
NLIN	Nonliear interference noise
NN	Neural network
NPT	Nonlinear perturbation term
OA	Optical amplifier
PAS	Probabilistic amplitude shaping
PCS	Probabilistic constellation shaping
PD	Photdiode
PPD	Perturbation-based post-distortion
PRBS	Pseudo-random binary sequence
Q	Quadrature
QAM	Quadrature amplitude modulation
QPSK	Quadrature phase-shift keying
ReLU	Rectified linear unit
RNN	Recurrent neural network
RP	Regular perturbation
RRC	Root-raised cosine
RX	Receiver
SC	Signle-channel

SD-FEC	Soft-decision forward error correction code
SDR	Signal-to-distortion ratio
SeLU	Scaled exponential linear units
SGD	Stochastic gradient descent
SL-NLC	Supervised-learnt nonlinearity compensation
SMF	Single-mode fiber
SNR	Signal-to-noise ratio
SOI	Symbol of interest
SSFM	Split-step Fourier method
SSMF	Standard signle-mode fiber
TWC	TrueWave classic
TX	Transmitter

List of figures

Aston University

Deep Learning Methods for Nonlinearity Mitigation in Coherent Fiber-Optic Communication Links

Vladislav Neskorniuk

Doctor of Philosophy, November 2022

Nowadays, the demand for telecommunication services is rapidly growing. To meet this ever-increasing connectivity demand telecommunication industry needs to maintain the exponential growth of capacity supply. One of the central efforts in this initiative is directed towards coherent fiber-optic communication systems, the backbone of modern telecommunication infrastructure. Nonlinear distortions, i.e., the ones dependent on the transmitted signal, are widely considered to be one of the major limiting factors of these systems. When mitigating these distortions, we can't rely on the pre-recorded information about channel properties, which is often missing or incorrect, and, therefore, have to resort to adaptive mitigation techniques, learning the link properties by themselves. Unfortunately, the existing practical approaches are suboptimal: they assume weak nonlinear distortion and propose its compensation via a cascade of separately trained sub-optimal algorithms. Deep learning, a subclass of machine learning very popular nowadays, proposes a way to address these problems. First, deep learning solutions can approximate well an arbitrary nonlinear function without making any prior assumptions about it. Second, deep learning solutions can effectively optimize a cluster of single-purpose algorithms, which leads them to a global performance optimum.

In this thesis, two deep-learning solutions for nonlinearity mitigation in high-baudrate coherent fiber-optic communication links are proposed.

The first one is the data augmentation technique for improving the training of supervised-learned algorithms for the compensation of nonlinear distortion. Data augmentation encircles a set of approaches for enhancing the size and the quality of training datasets so that they can lead us to better supervised learned models. This thesis shows that specially designed data augmentation techniques can be a very efficient tool for the development of powerful supervised-learned nonlinearity compensation algorithms. In various testcases studied both numerically and experimentally, the suggested augmentation is shown to lead to the reduction of up to 6× in the size of the dataset required to achieve the desired performance and a nearly 2× reduction in the training complexity of a nonlinearity compensation algorithm. The proposed approach is generic and can be applied to enhance a multitude of supervised-learned nonlinearity compensation techniques.

The second one is the end-to-end learning procedure enabling optimization of the joint probabilistic and geometric shaping of symbol sequences. In a general end-to-end learning approach, the whole system is implemented as a single trainable NN from bits-in to bits-out. The novelty of the proposed approach is in using cost-effective channel model based on the perturbation theory and the refined symbol probabilities training procedure. The learned constellation shaping demonstrates a considerable mutual information gains in single-channel 64 GBd transmission through both single-span 170 km and multi-span 30x80 km single-mode fiber links. The suggested end-to-end learning procedure is applicable to an arbitrary coherent fiber-optic communication link.

Keywords: Telecommunications; nonlinear optics; coherent detection; digital signal processing; machine learning; deep learning; end-to-end learning; constellation shaping.

Acknowledgements

First of all, I would like to thank my main academic supervisor Prof. Sergei K. Turitsyn for providing the opportunity to start this PhD, the continuous support and the insightful discussions during its course. Another person, whom I highly thankful for the leadership and sharing his experience is Dr. Vahid Aref. The thorough and detailed discussions we had during my secondment to Nokia Bell Labs greatly shaped my PhD research and myself as a researcher. Next, my heartful thanks go to my academical co-advisor Dr. Jaroslaw E. Prilepsky for the guidance and the relentless encouragement shared, especially during the challenging first year of PhD. I am highly thankful to Christiane Doering-Saad for perfect project management which made my PhD journey as smooth as possible.

I gratefully acknowledge the support by the Marie Skłodowska-Curie project FONTE, a doctorate scheme which provided me with the opportunity to develop my skills in both industry and academia. I am thankful to Pedro J. Freire, Andrea Carnio, Dr. Fred Buchali, Dr. Domenico Marsella, Dr. Antonio Napoli, and Dr. Nelson Costa for fruitful collaboration and insightful discussions we had. Special thanks go to Vinod Bajaj and Dr. Stenio Magalhães Ranzini, with whom we shared the unique experience of this programme at Stuttgart.

I would also want to thank all the awesome friends and colleagues I met during this PhD for making this study enjoyable.

Last by not the least, I would like to thank my parents Victoria Okocha, Aleksandr Neskorniuk, and my grandmother Maya Okocha, to all of whom I dedicate this thesis, for their unconditional support, upbringing, and the love I enjoyed during my life.

Chapter 1

Introduction

1.1 Deep learning in communication systems

Nowadays, machine learning is an established tool for solving the tasks in the various domains, including social, natural sciences and engineering [4–6]. The recent significant advances in the computational power and data storage led this process to gain an unprecedented momentum. Deep learning - a class of machine learning methods focused on application of artificial neural network algorithms inspired by the architecture of brain [7], has become the leading approach in many areas, due to its huge flexibility [8, 9]. A lot of fields were revolutionized by a the creation of record-beating machine learning techniques, notably, computer vision [10], natural language processing [11], and speech processing [12]. Similarly, in the optical communications domain, albeit there was an early interest in using machine learning techniques in late 1990s [13], only in this decade we observed lots of applications of machine learning techniques to a huge range of tasks in optical communications [14, 15].

1.1.1 Prerequisites for deep learning application

Artificial neural networks (ANNs) can be seen as a computational algorithm made of a sequence of linear transformations with trainable parameters alternating with the point-wise nonlinear functions, referred to as the activation functions [9]. Using sophisiticated optimization techniques, the parameters of these algorithms can be optimized to approximate a complex nonlinear function in the process named *deep learning*. The quality of the resulting deep learnt algorithm relies on the presence of a huge body of data objects representative of searched pattern, making deep learning a *data-driven* approach. The state-of-the-art deep learning approach is usually comprised of advanced gradient descent based optimization tech-

niques [16] and the automatic gradient computation routines [17] based on back-propagation algorithm [18].

The data-driven deep learning approach to optical communications tasks can pose a vital alternative to the conventional solutions based on the domain knowledge. Nonetheless, the task at hand usually must fall into one of categories in order for deep learning to be able to surpass the classical methods of solving it [19, 20]. In the reminder of the section, we will review these use cases.

First, the absence of a mathematical model describing the properties of objects invlolved into the posed problem, can serve as the sign of a possible success of deep learning based solution. The reason is the inability to create a baseline domain-based algorithm in the absence of the underlying process model, which makes any operational deep learning solution a leading one. At the same time, the successful application of deep learning to the problem requires the availability of the representative dataset. In more detail, the dataset objects should correctly represent the space of the outcomes of the underlying process. Usually, the dataset should be also of a big enough size, to provide the precise enough representation of the underlying process.

Second, if there exists a settled mathematical model, describing the process underlying in the considered task, one has first to check the presence of the existing task solutions involving the algorithms based on this model. If there are no existing solutions, the deep learning might be a viable alternative. In case the dataset required for deep learning is not available, usually the mathematical model can be employed to generate the required synthetic dataset - a process called *data augmentation*.

Finally, if both the established model and the solutions based on it are available for the considered task, deep learning might produce a solution with better performance-to-complexity ratio or better performance overall, which might be beneficial given the limitations imposed by the task.

Despite optical communications engineering being a mature field which has both the established channel models and the effective solutions based on them, the applications of deep learning still have considerable promise there. First, lost of important tasks in this field are not solved yet or use sub-optimal solutions open for the performance boosts by the implementation of deep learning techniques. Second, for the tasks where an appropriate solution is available, the deep learning might still outperform this solution by incorporating the domain knowledge into the architecture of a learnt algorithm (notably in [21]), or by offering a less precise but more cost-effective alternative. Therefore, the detailed analysis should be done in any case to decide on the applicability of deep learning.

1.1.2 Deep learning applications to optical telecommunications

The demand for internet traffic grows nowadays being pushed forward by the steady exponential growth of the number of the interconnected devices and the amount of data, consumed by each device [22]. To match this demand surge, the modern communication systems constantly expand their complexity and scale. This unprecedented growth presents the strong challenges for the management, developement, and control of the optical communication systems serving as the backbone of the modern communication infrastructure [23]. Unfortunately, the conventional approaches are exausting their capabilities to keep up with the ramping requirements to the latency, quality of service, and flexibility required from the modern links [24]. Meanwhile, the booming opportunities for collecting the datasets describing the operation of communication systems at both networking and physical level generates unprecedented opportunities for developing cost-efficient scalable machine learning solutions addressing the problems emerging in these layers [14, 25].

For the physical layer of optical communication, which this thesis is focused on, deep learning seems to be a highly promising tool for improving the digital signal processing (DSP) algorithms. In state-of-the art links, the compensation of a huge range of transmission impairments, introduced both by the transmitter devices and the optical channel, is usually done by the digital signal processing algorithms, by thus making them an effective standard approach improving the quality of transmission and the reach of modern communication systems [26, 27]. The design of DSP is usually employed in state-of-the art telecommunications is the modular one: DSP is usually represented as a cascade of several algorithm each addressing its own flavor of distortion.

Let us briefly describe the digital signal processing stack typically applied to coherent fiber-optic communication links. At the transmitter (TX), the transmitted information, usually represented by a sequence of bits ('0's and '1's), is first expanded by forward error correction (FEC) encoder, and (optionally) distribution matcher, then sampler converts it to a sequence of complex valued numbers corresponding to the transmitted symbols, which are finally processed by a digital pre-distorter aiming at pre-compensating the transmitter device nonlinearities and, partially, channel nonlinear distortions. At the receiver (RX), the received symbol sequence is first equalized by a cascade of adaptive and static digital filters, second, timing recovery restoring the location of the boundaries of time-slots corresponding to each symbol is applied, third, carrier phase and frequency recovery, and, finally, the symbols are demodulated to bit sequences which are later post-processed by forward error correction decoder before being fed to customer.

The aforementioned modular architecture of transmitter and receiver is convenient from an engineering point of view, since it greatly simplifies the control and training of individual DSP blocks.

Nowadays, the application of deep learnt algorithms to execute the function of some of the blocks of a digital signal processing algorithm is a hot topic in both academical and industrial research. Usually, the deep learning is suggested for improving the performance-to-complexity ratio of the DSP algorithms or the overall improvement of the DSP performance. A common approach used in developing effective DSP algorithms is the usage of the domain knowledge about the underlying distortion approached by a particular algorithm.

For instance, the deep learned artificial neural networks were successfully applied to create a near-optimal symbol demodulator for the additive white Gaussian noise channel [28]. Deep learning of the forward error correction codes decoding was also studied in details [29–31]. Notably, in [31] it was shown that the deep-learnt recurrent-neural-network-based (RNN) algorithms can be employed as the optimal decoders of convolutional codes.

Another important application of deep learning to optical telecommunications are receiver-based digital signal processing algorithms aimed at equalization of nonlinear distortions injected into the transmitted signal during its propagation over the fiber-optic communication link. Typically, the proposed deep learnt solutions are designed using the *domain knowledge* about the structure and, if available, the mathematical model of the distortions addressed by an algorithm.

For example, one model-based approach on which deep learning algorithms could be based is digital back-propagation (DBP) [32]. This is an efficient approach for compensating the optical channel distortions, governed by Manakov equations which involves solving the equations with the negated parameters, by thus reversing the distortions introduced into the propagated signal. DBP is based on the split-step Fourier method of solving Manakov equations, where the equations are modeled as a repeating sequence of the so-called "linear step" and "nonlinear step" alternating operators aimed at compensating, respectively, the linear and nonlinear distortions emerging in the optical channel. A classical DBP-based approach has two problems. First, the required number of iterating steps and, therefore, the algorithm complexity increase with the rise in launch power, transmission distance, and bandwidth. Second, for effective distortion compensation DBP requires the precise information about the parameters of compensated link. These two problems are addressed by the deep learning approach referred to as the learned digital back-propagation (LDBP) [21, 33–35]. There, the neural network implements the "linear steps" of DBP as trainable general linear functions with the deep learned parameters, similar to the layers of a convolutional

neural network. This architecture leads to LDBP being able to learn the parameters of the compensated link and being more cost-efficient than the classical DBP.

Also, the deep learned equalization of optical channel nonlinearities could be done by "vanilla" deep learned neural network developed for other tasks. This approach was pioneered in early 1990s, when neural networks were suggested for the nonlinearity equalization in satellite communication links [36, 37]. A review and a comparative study of "vanilla" NN application for channel nonlinear compensation was done in [38] and in the papers cited in [38].

Another way of nonlinearity mitigation is digital pre-distortion (DPD), i.e., pre-compensation of the link nonlinear signal distortion at the transmitter by pre-processing in the digital domain the generated transmitted symbol sequence prior to sending it to the digital-to-analog convertor. Typically, the DPD-based solutions address nonlinear distortions generated not only by the optical channel, but also by the transmitter devices. The most popular classical approach to nonlinear DPDs employs Volterra series. Nonlinear digital pre-distortion by Volterra series was succesfully demonstated both in wireless [39–43] and coherent optical [44–49] links.

Deep learning application to nonlinear digital pre-distortion was pioneered in 1980s [50, 51]. Nowadays, on the wave of ever-increasing interest to neural networks (NNs), many deep learned NN-based architectures were proposed for digital pre-distortion [52–60].

In wireless communication links, first, a memory-neglecting deep learned DPD based on simple dense neural network was proposed in [54]. Next, more complex architectures including the memory effects were proposed: time-delay [52, 55, 58], convolutional [56] neural networks. Notably, residual neural networks [61], a record-breaking technique in computer vision, were also succesfully applied to DPD in wireless links [53]. In [57], the DPD compensating the nonlinear distortion of the whole transmitter was realized on the basis of recurrent neural networks.

At the same time, the deep learned digital predistortion for coherent optical links is not so developed yet as for wireless ones. There, deep learned dense neural networks were suggested for memoryless digital pre-distortion of a Mach-Zehnder modulator [59] and low-resolution digital-to-analog converter [62]. A model-based approach was taken in [63], where the deep learned DPD based on Wiener-Hammerstein model was proposed. Finally a practical dense-neural-network-based DPD solution was proposed [60]. The implementation of this DPD led to the the transmission records being set for single-channel [60, 64] and multi-channel dense-wavelength-division-multiplexed transmission [65] over 80 km single mode fiber coherent optical links.

All the aforementioned deep learning solutions consider the optimization of only some of the algorithms constituting the whole digital signal processing stack. Notably, the considered

solutions focus on the DSP located either on the transmitter or receiver. Nonetheless, the performance of an optical coherent communication link is defined by the joint operation of the whole DSP. Therefore, the approach of designing and optimizing the communication system on the basis of a separate consideration of each DSP block most likely leads to sub-optimal overall performance of the link.

Meanwhile, a more holistic approach to deep learning the DSP, referred to as *end-to-end learning* was recently proposed [66]. It utilizes the well-known wide nonlinear function approximation capabilities of neural networks [9]. In this approach, first, the whole optical communication link is modelled end-to-end from bits in to bits out as an artificial neural network. In the resulting ANN-based communication links the parameters of DSP blocks are left as the trainable blocks. Then, the parameters of all the link block are simultaneously deep learned to optimize the overall performance of the whole communication links. Such a comprehensive approach to the DSP design allows obtaining the link performance level unreachable by a separate optimization of DSP blocks. The applications of end-to-end learning are especially visible for the transmission schemes, where the set of optimal DSP modular is not known or too computationally expensive to be implemented. The end-to-end learning was first introduced for wireless communications [66–68]. Furthermore, there were a lot of applications of end-to-end learning to the coherent communication links. The detailed review of the end-to-end learning application to these links is given in Section 3.1 of this thesis.

To conclude, the ever growing demand for the connectivity poses new challenges in the design of the modern fiber-optic communication links. These challenges can be addressed by the novel approaches to the digital signal processing based on deep learning algorithms. The existing deep learning applications can be grouped around the three main usecases: equalization, pre-distortion, and end-to-end learning.

1.2 Thesis outline

In this section we outline the structure of the remainder of the thesis.

Section 1.3 outlines the main principles of the deep learning. It starts from the description of the tasks usually solved by machine learning, the class of algorithms to which deep learning belongs. Next, the section outlines the main defining components of deep learning: artificial neural networks and advanced calculation of gradients fed into the optimization.

Section 1.4 describes the performance metrics of a coherent optical communication and the upper bounds on them. The section starts from outlining the principal scheme of a digital communication system. Then, the section considers various estimations on the achievable

information rate of a communication system: Shannon capacity, constrained capacity, and generalized mutual information. The section continues with a more in-detail description of forward error correction codes and modulation formats affecting the choice of performance metrics and the overall performance of the link. The section concludes with the review of practical ways to measure the performance of the link.

In the following parts of thesis, we describe our original research.

Chapter 2 describes the data aumgentation technique developed for reducing the complexity of supervised learning the algorithms aimed at the equalization of channel nonlinearities in a coherent fiber optic communication link. Data augmentation describes the set of techniques aimed at enhancing the performance and reducing the training complexity of a supervised-learnt algorithm by synthetically expanding the training dataset via diluting it with the artificial data. The chapter starts from postulating Manakov equations, the model describing the evolution of signal during its propagation over the fiber-optic communication link. Then, the data augmentation technique is proposed, based on generating new synthetic solutions of Manakov equations out of existing ones. Finally, the effect of proposed data augmentation technique is demonstrated both numerically and in the experiment for several links and nonlinearity compensation algorithms. First, the proposed data augmentation technique singificantly improved the performance of the nonlinearity compensation algorithms when trained on deficient data. Second, for nonlinearity equalizers trained on big enough dataset, the data augmentation technique led to nearly twice reduction in the complexity of training.

Chapter 3 outlines the end-to-end learning algorithm for finding the optimal constellation shaping of an arbitrary coherent optical communication link. The proposed algorithm implements the simplified numerical model of the whole link from bits-in to bits-out. The chapter starts from the description of how each of the communication link blocks, transmitter, channel model, receiver, is implemented. The end-to-end algorithm description is concluded with the loss functions: symbol-wise and bit-wise (generalized) mutual information and the procedure for calculating the gradients used in their gradient-based optimization. The chapter is concluded with the numerical demonstration of the performance gains by the proposed end-to-end learning approach which managed to learn effective single- and multi-symbol constellation shapings in both state-of-the-art single-span and long-haul coherent optical communication links.

Finally, *Chapter 4* concludes the thesis.

1.3 Deep learning principles

In this section we describe in more detail the three main components of the deep learning: artificial neural network (ANN) algorithms and the formulation of the task of ANN parameters optimization, the gradient descent optimization methods used to train them, and the back-propagation algorithm used to calculate the gradients used to run the training.

1.3.1 Types of machine learning tasks

Deep learning is a type of machine learning. Therefore, before going deep into the deep learning specifics we first describe the problems which can be addressed by the machine learning.

Machine learning can be defined a process of building and operating the methods able to 'learn', in other words, to improve their performance on a set of tasks using some relevant data. Usually, the learning process involves the algorithm recognizing and quantifying some pattern in data [4]. The data used by the machine learning algorithm is referred to as the *training dataset*.

Machine learning approaches are ususally categorized into the three main groups depedning on the nature of the data used during training:

- Supervised learning.
- Unsupervised learning.
- Reinforcement learning.

In the *supervised learning* approach, the training dataset is comprised by the pre-known set of inputs and desired outputs (labels), referred to as the *labeled data*. Supervised learning is the most common flavor of machine learning. The biggest challenge of the supervised learning is to create the algorithm with high *generalizability* on the unseen data, i.e. in other words, which returns correct outputs when fed with the inputs from the data points not included. We consider this problem in Chapter 2.

In the *unsupervised learning* approach, the objects of a training dataset have no desired outputs, only inputs, by thus leaving the algorithm to find its own ways to organize the dataset objects. Usually, unsupervised learning is done as a goal in itself to find some obscure pattern in the training data, or a means of a *feature engineering*, i.e., preprocessing of the labeled data before using it in the supervised learning. The end-to-end learning of a communication link, a promising unsupervised learning method, is considered in Chapter 3.

Finally, in the *reinforcement learning* approach, the algorithm generates it's own training dataset by interacting with the environment. The problem considered there is to find the set of actions maximizing the reward in a particular context. The main question answered in the design of a reinforcement learning algorithm is how to share the available resources between *exploration* - trying out new types of actions in order to find how much gain they return, and *exploitation* - making use of actions about which the system already knows that they provide high returns. The applications of the reinforcement learning to optical communications are beyond the scope of this thesis. We advise the interested reader to check the review [15] on this topic.

1.3.2 Artificial neural networks

Originally, the artificial neural network concept was introduced in the 1940s as a simplified imitation of the biological neural networks consitituting the human brain [69]. A typical *artificial neural network* (ANN) is a sequence of interconnected layers - the groups of basic processing elements, referred to as the *artificial neurons*.

In more detail, an *artificial neural network* can be understood as a set of interconnected nodes named *artificial neurons* which roughly model the biological neurons from animal brain. The connections between nodes are named *edges*. Every connection, similar to synapses of a biological brain, transmits "signals" from one artificial neuron to another. Usually the "signal" is a real-valued number and the connection applies some linear transformation with the trainable parameters to the transmitted "signals". The parameters of these linear transformations are referred to as *weights*. Artificial neurons generate the sent "signals" out of the received ones by applying a point-wise nonlinear function to them. The nonlinear fucntion is referred to as an *activation function*. Typically, the activation function is chosen to make the artificial neuron behave as a gate, generating non-zero output "signals" only if the sum of the input "signals" overcomes a particular limit. The intuitive visualization of the ANN concept can be found in [70].

In the late 1980s it was mathematically proven that an artificial neural network, albeit of an extremely big size, can be optimized to approximate an arbitrary nonlinear transformation between the closed and bounded subsets of the of multi-dimensional real-valued vector space $\mathbb{R}^n$, i.e. an arbitrary practically important nonlinear function [71–74]. This statement is referred to as the *universal approximation theorem* in the machine learning community.

1.3.2.1 Artificial neuron

In this section we bring a more mathematically rigorous definition of an artificial neuron. It can be represented as a mapping of its real-valued inputs $x_i \in \mathbb{R}, i = 1 \dots n$ to a single

real-valued output $y \in \mathbb{R}$ done following the procedure

$$y = f_{\text{model}}(x) = f_\alpha \left(\sum_{i=1}^{n} w_i x_i + b \right), \tag{1.1}$$

where $w_i \in \mathbb{R}$ are weights, b is a free parameter referred to as *bias*, and f_α is the activation function. The choice of activation function is one of the most important in the artificial neural network design since it defines the range of nonlinear approximation capabilities of the resulting algorithm. There are several most popular activation function choices.

The first one is the *logistic function* which maps the real number range $\mathbb{R} = (-\infty, \infty)$ to the interval $(0, 1)$

$$f_\alpha(u) = \frac{1}{1 + \exp(-u)}. \tag{1.2}$$

Logistic function is popular in the classification tasks, because it can map an arbitrary model output to a probability range of an object belonging to a particular class. There exists a multiple-input and multiple-output generalization of logistic function referred to as the *softmax*

$$f_\alpha^j(u_1 \dots u_n) = \frac{\exp(u_j)}{\sum_{i=1}^{n} \exp(u_i)}. \tag{1.3}$$

where f_α^j, $j = 1 \dots n$ and u_j, $j = 1 \dots n$ are, respectively, the softmax inputs and outputs. Because of the notable properties of its outputs $f_j \in (0, 1)$, $\sum_{j=1}^{n} f_j = 1$ softmax outputs can be interpreted as a discrete probability distribution. For this reason, softmax is frequently used in neural network architectures for multi-class classification.

Another popular option is *hyperbolic tangent* which maps real-valued input to a $(-1, 1)$ range

$$f_\alpha(u) = \text{tahh(u)} = \frac{\exp(u) - \exp(-u)}{\exp(u) + \exp(-u)}. \tag{1.4}$$

Both hyperbolic tangent and logistic function belong to the same class of so-called *sigmoid* functions, the ones whose plot around $u = 0$ resembles "S"-curve.

Finally, the most popular option [75] is *rectified linear unit (ReLU)*, which nullifies negative inputs and keeps the positive ones

$$f_\alpha(u) = \max(0, u). \tag{1.5}$$

Compared to the aforementioned sigmoid functions, ReLU has an advantage of being a much simpler one. It is less computationally expensive to calculate and speed-up the training by avoiding the problem of *vanishing gradients*, characteristic to sigmoid functions.

Sometimes, a 'softer' version of ReLU, referred to as a *Leaky ReLU*, is used, which passes through negative values, albeit, with a small slope

$$f_\alpha(u) = \begin{cases} u, \text{ if } u \geq 0; \\ k \cdot u, \text{ if } u < 0, \end{cases} \tag{1.6}$$

where $k \ll 0$. Leaky ReLU is popular in contexts where one wants to combine the simplicity of ReLU with the ability to keep information about negative inputs too, for instance in generative adversarial networks [76] and object detection networks [77].

1.3.2.2 Dense artificial neural network

Dense artificial neural network is the simplest yet already powerful ANN architecture. It also known as a multilayer perceptron. It consists of interconnected layers of neurons. The neurons in each layer have no connection with each other, but have an edge connection to each of the layers of the preceding layer (or network inputs for the case of the first layer). In a more mathematically rigorous way, an N-layer dense artificial neural network maps the input real-valued vector $\mathbf{x}^0$ to the output vector $\mathbf{x}^L$ via the following iterative procedure

$$\mathbf{x}^l = f_\alpha^k \left(\mathbf{W}_l \mathbf{x}^{l-1} + \mathbf{b}_l \right), \quad l = 1, \ldots, N. \tag{1.7}$$

Here $x^l \in \mathbb{R}^{d_l}$ and $x^{l-1} \in \mathbb{R}^{d_{l-1}}$ are, respectively, outputs of l-th and $(l-1)$-th layers of the dense ANN, d_l is the dimensionality of l-th layer, $W_l \in \mathbb{R}^{d_l \times d_{l-1}}$ is l-th layer weight matrix, with $\mathbf{b}_l \in \mathbb{R}^{d_l}$ and f_α^k being its bias vector and activation function, correspondingly.

Nowadays, more sophisticated artificial neural network architectures were developed, notably, convolutional, recurrent, residual ANNs [9] and transformers [78]. However, they are not used in the following chapters, and, therefore, left beyond the scope of this thesis.

1.3.3 Learning procedure

Along with the aforementioned artificial neural networks, optimization techniques, tailoring the parameters of ANN to the training datasets, are the integral part of the deep learning. In this section, we will, first, describe the optimization objective; then, introduce the gradient descent optimization techniques used to reach this goal, and, will conclude the section with the description of the backpropagation algorithm used for the effective calculation of these gradients.

1.3.3.1 Optimisation goal. Loss function.

In machine learning the common practice is to formalize the goals of the training procedure via a loss function $\mathcal{L}$ - a single overall quantitative measure of the quality of training, usually a real-valued one $\mathcal{L} \in \mathbb{R}$. The loss function is defined in a way so that the desired solution should minimize its value.

Now we'll bring a more mathematically rigorous definition of loss function $\mathcal{L}$. In the following we focus on the case of training over a labeled dataset, considered in this thesis. Let the training data $\mathbf{D}$ consist of the set of input vectors $\mathbf{x}_i = [x_{i,1}, \ldots, x_{i,n}]$, $x_{i,j} \in \mathbb{R}$ and the desired output vectors $\mathbf{y}_i^{\text{data}} = [y_{i,1}, \ldots, y_{i,n}]$, $y_{i,j} \in \mathbb{R}$, corresponding to each input $\mathbf{x}_i$, i.e., $\mathbf{D} = \left[\{\mathbf{x}_1, \mathbf{y}_1^{\text{data}}\}, \ldots, \{\mathbf{x}_N, \mathbf{y}_N^{\text{data}}\}\right]$. Next, we define the output predicted by the trained algorithm for every dataset input as $\mathbf{y}_i^{\text{model}} = f_{\text{model}}(x_i, \theta)$. Obviously, the model predictions depend also on its parameters θ. For instance, for dense artificial neural network Eq. (1.7) the parameters are model weights and biases, i.e. $\theta_{\text{DNN}} = \{\mathbf{W}_1, \ldots, \mathbf{W}_L, \mathbf{b}_1, \ldots, \mathbf{b}_L\}$.

In terms of these notations, the loss function $\mathcal{L}$ can be defined as

$$\mathcal{L}(\theta, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} l\left(\mathbf{y}_i^{\text{data}}, f_{\text{model}}(\mathbf{x}_i, \theta)\right) \tag{1.8}$$

where $l(\mathbf{y}_i^{\text{data}}, \mathbf{y}_i^{\text{model}})$ is per-example loss function, and N is the training data cardinality. At the same time, the task of training a deep learning model can be expressed as

$$\theta_{\text{learned}} = \underset{\theta}{\operatorname{argmin}} \mathcal{L}(\theta, \mathcal{D}), \tag{1.9}$$

i.e. finding the set of model parameters θ_{learned} minimizing the loss function $\mathcal{L}$ given the training dataset $\mathbf{D}$.

There are several popular loss functions [4, 79]. The supervised learning task is usually subdivided into *regression* and *classification* tasks each having its own goals and, therefore, requiring, its own loss functions.

For the *regression* task, the model is required to predict the continuous value corresponding to the input. That mean that the desired outputs and hence model predictions are defined as real-valued variables $y_{i,j}^{\text{data}}, y_{i,j}^{\text{model}} \in \mathbb{R}$. The most popular loss functions for this case are mean squared error (MSE) and mean absolute error (MAE).

The per-example loss function $l_{\text{MSE}}(\mathbf{y}_i^{\text{data}}, \mathbf{y}_i^{\text{model}})$ of the mean-squared absolute error is defined as

$$l_{\text{MSE}}(\mathbf{y}_i^{\text{data}}, \mathbf{y}_i^{\text{model}}) = \left(\mathbf{y}_i^{\text{data}} - \mathbf{y}_i^{\text{model}}\right)^2 \tag{1.10}$$

At the same time the per-example loss function $l_{\text{MAE}}(\mathbf{y}_i^{\text{data}}, \mathbf{y}_i^{\text{model}})$ of the mean-squared absolute error is defined as

$$l_{\text{MAE}}(\mathbf{y}_i^{\text{data}}, \mathbf{y}_i^{\text{model}}) = \left|\mathbf{y}_i^{\text{data}} - \mathbf{y}_i^{\text{model}}\right|. \tag{1.11}$$

Mean squared error is computationally simpler to optimize. Nonetheless, it is highly sensitive to incorrect *outliers* present in the training data, i.e., the data points erroneously having a big incorrect label value. Therefore, MAE is the preferred option when working with noisy training dataset.

In the *classification* task the model predicts the probability of the object described by training data input belonging to a particular class, or a group of classes. As a result, the desired and predicted output values are defined as vectors of probabilities of the object belonging to any of the range of classes, i.e. $y_{i,j}^{\text{data}}, y_{i,j}^{\text{model}} \in (0, 1)$. Furthermore, if single-class classification is considered, the output should describe a discrete probability distribution and, therefore, an additional condition is applied to answers that $\sum_{j=1}^{n} y_{i,j} = 1, \forall i$.

The most popular loss function for classification tasks is *cross-entropy*, also known as *log-loss*, with the per-example loss function l_{CE} defined as

$$l_{\text{CE}}(\mathbf{y}_i^{\text{data}}, \mathbf{y}_i^{\text{model}}) = -\sum_{j=1}^{n} y_{i,j}^{\text{data}} \log\left(y_{i,j}^{\text{model}}\right) + \left(1 - y_{i,j}^{\text{data}}\right) \log\left(1 - y_{i,j}^{\text{model}}\right). \tag{1.12}$$

1.3.3.2 Gradient descent optimization method

Gradient descent (GD) is an iterative optimization technique for minimizing a function using information about its gradients. Gradient descent is the de-facto stadard for training deep learning solutions [9]. The technique was suggested in 1847 by Augustin Cauchy for optimization of linear functions [80] and nearly a century later was extended for nonlinear optimization problems by Haskell Curry [81].

The gradient descent method is based on utilizing the famous gradient property of pointing in the direction opposite to the steepest descent of the function. The method assumes that the optimized function is defined and differentiable in a neighbourhood of the considered point. This condition is typically fulfilled during minimizing the loss function as part of the artificial neural network training, with the notable exception of optimization of the mean absolute error loss, which is non-differentiable at $\mathcal{L}_{\text{MAE}} = 0$ point.

In general, the gradient descent method proposes making small iterative steps in the direction opposite to the gradient in order to find the set of variables minimizing the studied

function. For a scalar function $f(\mathbf{x})$ of a vector variable $\mathbf{x} \in \mathbb{R}^n$ the method can be stated as

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \frac{\delta f(\mathbf{x}_t)}{\delta \mathbf{x}_t}, \tag{1.13}$$

where $\delta f(\mathbf{x}_t)/\delta \mathbf{x}_t$ is the gradient, $\eta \in \mathbb{R}$ is a learning rate - a hyper-parameter of a gradient descent method. We expect the function value to descrease after every algorithm iteration $f(\mathbf{x}_{t+1}) < f(\mathbf{x}_t)$, the behavior is referred to as the *convergence* of a gradient descent. Eventually the gradient descent is expected to stop at a function minimum, where the gradient is equal to zero. The choice of the learning rate values heavily affects the convergence - too big η values prevent the algorithm convergence to the minimum, while too small η values increase the number of iterations required to approach the function optimum.

In the context of deep learning, first, the optimized variable is a set of artificial network parameters θ. Second, the gradient of a loss function over the ANN parameters $\delta\mathcal{L}/\delta\theta$ is calculated over all the objects of the training dataset $\mathcal{D}$. As a result, the gradient descent algorithm for artificial neural network training takes shape

$$\theta_{t+1} = \theta_t - \eta \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}, \mathbf{y}^{\text{data}}) \in \mathcal{D}} \frac{\delta l(\theta_t, \mathbf{x}, \mathbf{y}^{\text{data}})}{\delta \theta_t}, \tag{1.14}$$

where $|\mathcal{D}|$ is the cardinality of the training dataset, and $l(\theta_t, \mathbf{x}, \mathbf{y}^{\text{data}})$ is the pre-sample loss function introduced in the loss function definition Eq. (1.8).

A notorious flaw of the gradient descent optimization technique is that it converges to the *local minimum* $\lim_{t\to\infty} \mathbf{x}_t \to \mathbf{x}_{\text{LM}}$, i.e. the point where the function takes the minimum value in some points neighbourhood $O(\mathbf{x}_{\text{LM}})$, i.e

$$\mathbf{x}_{\text{LM}} : f(\mathbf{x}_{\text{LM}}) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in O(\mathbf{x}_{\text{LM}}) \in \mathbb{R}^n, \tag{1.15}$$

instead of the desired *global minimum* point $\mathbf{x}_{\text{GM}}$, the smallest possible value reachable by the optimized function overall.

Furthermore, the gradient descent algorithm might even not reach the local minimum. The gradient value may reach zero and cause gradient descent to stop at a point where there is no local minimum, notably, in a *saddle point* [9]. In practical deep learning, several improvements to the gradient descent method were suggested in the context of deep learning to improve the chances of it reaching a reasonably small loss function value. The two most notable improvements of gradient descent - stochastic gradient descent and Adam optimization algorithm, are considered in the two following Sections 1.3.3.2.1 and 1.3.3.2.2,

respectively. For a more detailed description of the optimization methods used in deep learning an interested reader is referred to [16].

1.3.3.2.1 Stochastic gradient descent

As noted in the previous section, to execute the vanilla gradient descent iteration Eq. (1.14) gradient has to be averaged over the whole, typically large, training dataset $\mathcal{D}$. This averaging requires a lot of numerical resources, by thus, making the gradient descent algorithm computationally too expensive. A popular approach to reduce the complexity of a gradient descent algorithm is to use a stochastic approximation [82] of a gradient during an optimization algorithm iteration Eq. (1.8). The resulting algorithm is known as a *stochastic gradient descent (SGD)* [9, 16, 83]. In more detail, for every iteration SGD calulates the gradient using a small randomly selected sample of training data points $\mathcal{D}' \subset \mathcal{D}, |\mathcal{D}'| \ll |\mathcal{D}|$. This subset is referred to as a *mini-batch*. The resulting algorithm can be formulated as

$$\theta_{t+1} = \theta_t - \eta \frac{1}{|\mathcal{D}'_t|} \sum_{(\mathbf{x}, \mathbf{y}^{\text{data}}) \in \mathcal{D}'_t} \frac{\delta l(\theta_t, \mathbf{x}, \mathbf{y}^{\text{data}})}{\delta \theta_t}, \quad \mathcal{D}'_t \subset \mathcal{D} \tag{1.16}$$

Notably, since the gradient calculation is approximate in SGD, even the convergence of this algortihm to a local minimum of a loss function $\mathcal{L}$ is not guaranteed, compared to the vanilla gradient descent Eq. (1.14). Nonetheless, the stochastic gradient descent, typically, finds the set of ANN parameters θ resulting is a reasonably low value of a loss function [9].

The batch size $|\mathcal{D}'|$ is typically kept fixed during the algorithm execution and so constitutes a hyper-parameter of the stochastic gradient descent algorithm. It is typically chosen not too small to avoid introducing excessive noise in the stochastic gradient computation and not too big to keep the SGD algorithm numerical complexity at a reasonable level.

1.3.3.2.2 Adaptive learning rates. Adam optimization algorithm

The choice of a learning rate (η in Eqs. (1.14, 1.16)) heavily affects the convergence rate of a gradient descent optimization algorithm, and, therefore, constitutes another important consideration in its design.

The first approach is to introduce into the training procedure the scheduler changing the learning rate in course of the optimization procedure. These approaches are referred to as the *adaptive learning rate* ones. The common intuition in there is that:

- In the beginning of the optimization, it is better to have bigger learning rate, when we are far from the minimum and gradients are huge, to have higher speed of convergence.

- At later stages of the optimization, it is better to decrease the learning rate to avoid overshooting the minimum point.

Many particular heuristics varying the learning rate were proposed and are actively used nowadays in deep learning algorithm training. An interested reader can find the description of these heuristics in the documentation of the popular deep learning packages, namely, PyTorch [84].

Another popular way to fine-tune the learning rates, is to set unique learning rate per every ANN parameter. Since the number of parameters of a typical artificial neural network is extremely big, such a precise tuning of learning rate can be done only via an automatic algorithm.

Adam [85] (an acronym for adaptive moment estimation) is the most popular stochastic gradient descent algorithm. The algorithm is based on customizing the learning rate computation per parameter. The intuition here is: if we consider the gradient descent as a ball running down a slope, then Adam behaves like a heavy ball with friction, therefore, preferring 'flat' minima in error surface [16, 86].

Mathematically speaking, first, Adam introduces the exponentially moving average of mean μ_t and variance ν_t of gradient

$$\begin{aligned} \mu_{t+1} &= \beta_1 \cdot \mu_t + (1-\beta_1) \cdot \nabla_t \\ \nu_{t+1} &= \beta_2 \cdot \nu_t + (1-\beta_1) \cdot (\nabla_t)^2 \\ \nabla_t &= \frac{1}{|\mathcal{D}'_t|} \sum_{(\mathbf{x},\mathbf{y}^{\text{data}}) \in \mathcal{D}'_t} \frac{\delta l(\theta_t, \mathbf{x}, \mathbf{y}^{\text{data}})}{\delta \theta_t}, \quad \mathcal{D}'_t \subset \mathcal{D} \end{aligned} \tag{1.17}$$

where ∇_t is the stochastic estimation of gradient over mini-batch $\mathcal{D}'$ from Eq. (1.16), and $(\cdot)^2$ is point-wise squaring. After some bias corrections, μ_{t+1} and ν_{t+1} are used to calculate an update of the artificial neural network parameters

$$\theta_{t+1} = \theta_t - \eta \cdot \frac{\mu_{t+1}}{\sqrt{\nu_{t+1}} + \epsilon}, \tag{1.18}$$

where $\epsilon \ll 1$ is a regularization term added for the numerical stability.

Usually, practical deep learning solutions involve both ways of fine-tuning the learning rate mentioned in this section: adaptive learning rate scheduling routes and Adam stochastic gradient descent optimisation.

1.3.3.3 Backpropagation

The speed of gradient calculation is the bottleneck defining the overall exectuion speed of gradient descent optimization. In artificial neural network training this problem is addressed by *backpropagation* - a simple and cost-effective algorithm for gradient computation [9, 18]. In this section we explain the algorithm on the example of a dense artificial neural network. Nonetheless, generalizations of backpropagation algorithm exist for the other classes of artificial neural networks. A great video full of intuitive animations illustrating the backpropagation algorithm can be found there [87].

The backpropagation is based on the *chain rule* of calculating the derivatives of a function cascade. The chain rule states that the derivative of the composition of two differentiable functions $h(x) = f(g(x))$ can be calculated as

$$\frac{\delta h}{\delta x} = \frac{\delta f}{\delta g} \cdot \frac{\delta g}{\delta x}. \tag{1.19}$$

If the considered function can be represented as the recursive repeat of the same function block $h = f_N(x_N, f_{N-1}(x_{N-1} \dots f_1(x_1)))$, one can utilize the already calculated and stored gradients over the parameters of the later function iterations $\delta h/\delta x_i$, $\delta h/\delta f_{i-1}$ to calculate the gradients over the parameters of earlier iterations, e.g. $\delta h/\delta x_{i-1}$, $\delta h/\delta f_{i-2}$ via chain rule. The naive alternative - applying chain rule to calculate the gradients of each iteration separately is computationally much costlier.

Let us now notice that the dense artificial neural network (DNN) Eq. (1.7) can be represented as the aforementioned recursive application of the same functional block

$$\begin{aligned} \mathbf{x}^L &= f_{\text{DNN}}(\mathbf{x}^0), \\ \mathbf{x}^l = f_\alpha^l(\mathbf{s}_l), \quad \mathbf{s}_l &= \mathbf{W}_l \mathbf{x}^{l-1} + \mathbf{b}_l, \text{ for } l = 1, 2, \dots, L, \end{aligned} \tag{1.20}$$

where $\mathbf{s}_l$ is the sum of the 'signals' received by an artificial neuron plus its bias $\mathbf{b}_l$. To remind, point-wise nonlinear activation fucntion f_α^l is applied to the sum of 'signals' $\mathbf{s}_l$ to calculate the output of the ANN layer $\mathbf{x_l}$.

In these terms, the backpropagation can be described via the iterative algorithm 1. The algorithm description uses the following vector notation of the chain rule. Let $h(\mathbf{y})$ be the scalar function of a vector argument $\mathbf{y} \in \mathbb{R}^n$ which is itself mapped from another vector $\mathbf{x} \in \mathbb{R}^m$ via a differentiable mapping $\mathbf{y} = g(\mathbf{x})$. Then, the gradient of h over $\mathbf{x}$ can be expressed via

$$\frac{\delta h}{\delta \mathbf{x}} = \left(\frac{\delta \mathbf{y}}{\delta \mathbf{x}}\right)^T \frac{\delta h}{\delta \mathbf{y}} \tag{1.21}$$

Algorithm 1: Backpropagation algorithm for calculation of the loss gradients over parameters of dense artificial neural network

$\mathbf{g} \leftarrow \delta\mathcal{L}/\delta\mathbf{x}^L$ `// Initialize buffer with the gradient of loss w.r.t the final DNN output`

for $l = L, L-1, \ldots, 1$ **do**

`/* Iterate starting from the final DNN layer */`

$\delta\mathcal{L}/\delta\mathbf{s}_k = \left(\delta\mathbf{x}^k/\delta\mathbf{s}_k\right)^T\mathbf{g}$ `// Calculate gradients over input 'signal'`

$\mathbf{g} \leftarrow \delta\mathcal{L}/\delta\mathbf{s}_k$ `// Store these gradients`

$\delta\mathcal{L}/\delta\mathbf{b}_k = (\delta\mathbf{s}_k/\delta\mathbf{b}_k)^T\mathbf{g} = \mathbf{g}$ `// Calculate gradients over bias`

$\delta\mathcal{L}/\delta\mathbf{W}_k = (\delta\mathbf{s}_k/\delta\mathbf{W}_k)^T\mathbf{g}$ `// Calculate gradients over weights`

$\delta\mathcal{L}/\delta\mathbf{x}_{k-1} = \left(\delta\mathbf{s}_k/\delta\mathbf{x}^{k-1}\right)^T\mathbf{g}$ `// Calculate gradients over the previous layer output`

$\mathbf{g} \leftarrow \delta\mathcal{L}/\delta\mathbf{x}^{k-1}$ `// Store these gradients`

end

where

$$\frac{\delta\mathbf{y}}{\delta\mathbf{x}} = \begin{pmatrix} \frac{\delta y_1}{\delta x_1} & \cdots & \frac{\delta y_1}{\delta x_m} \\ \vdots & \ddots & \vdots \\ \frac{\delta y_n}{\delta x_1} & \cdots & \frac{\delta y_n}{\delta x_m} \end{pmatrix} \tag{1.22}$$

is the $n \times m$ Jacobian matrix of the mapping $g : \mathbf{x} \rightarrow \mathbf{y}$, and the gradient of a scalar function over a vector variable is defined as

$$\frac{\delta h}{\delta\mathbf{x}} = \left[\frac{\delta h}{\delta x_1}, \frac{\delta h}{\delta x_2}, \ldots, \frac{\delta h}{\delta x_n}\right]. \tag{1.23}$$

The gradients of the loss over the artificial neural network parameters $\delta\mathcal{L}/\delta\theta = [\delta\mathcal{L}/\delta\mathbf{W}_1, \ldots, \delta\mathcal{L}/\delta\mathbf{W}_L, \ldots, \delta\mathcal{L}/\delta\mathbf{b}_1, \ldots, \delta\mathcal{L}/\delta\mathbf{b}_L]$ obtained by the backpropagation algorithm are recorded and fed into a gradient descent optimization algortihm.

There exist the generalization of backpropagation for all the conventional artificial neural network algorithms [9]. In general, the backpropagation algorithm can be formulated for any differentiable function which can be represented as an unidirectional computational graph. This generalized backpropagation is implemented in modern deep learning computational packages, e.g., in PyTorch [17]).

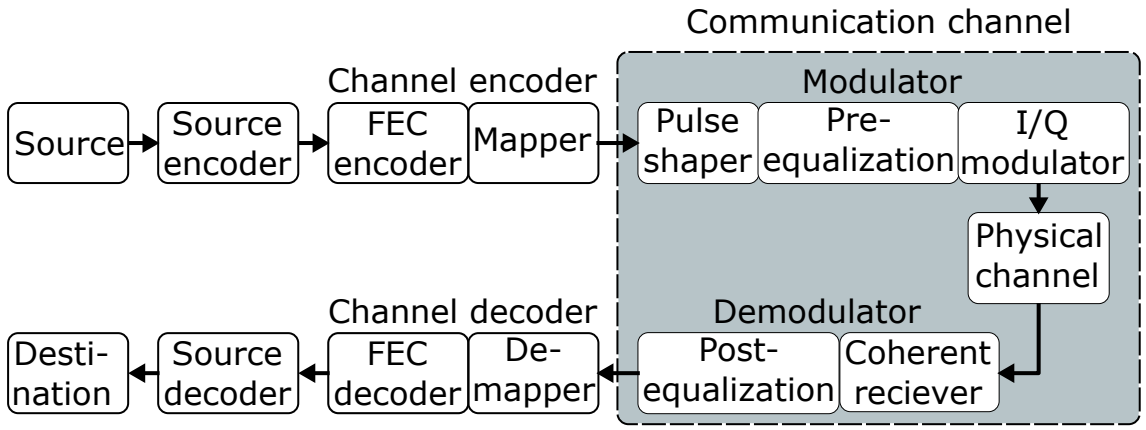


Fig. 1.1 Principal scheme of a digital communication system.

1.4 Digital telecommunication system

Having introduced in the previous Section 1.3 the main components of the deep learning, in this section we consider the basic digital telecommunication system and the key metrics of its performance.

1.4.1 The general scheme of a digital telecommunication system

The main goal of a *digital telecommunication system* is to ensure the nearly error-free information transmission via a given physical media undeterred by the signal distortions, both deterministic and stochastic, arising in the media. The main functional blocks constituting a digital communication link are given in Figure 1.1. Using this scheme, now we bring a general description of how it operates.

Telecommunication starts with a *digital source* generating the stream of information bits ('0's and '1's) describing the transmitted message. Then, the generated bit stream undergoes the so-called *source encoding* where the bit sequence is compressed to remove redundancy. Source encoding decreases the transmission cost by reducing the number of bits needed to be sent to transmit the required message. The bit sequence at the output of source encoder can be safely assumed to be a sequence of independent and identically distributed (i.i.d.) bits. Next, the compressed i.i.d. bit sequence is processed by a *channel encoder*, which increases the robustness of the message transmission to link distortions by introducing the controlled redundancy (i.e., more bits) into the transmitted bit sequence. The process of managing the errors by injecting controlled redundancy is referred to as *forward error correction (FEC)*.

After FEC the resulting pre-processed bits are mapped to *symbols* drawn from a fixed finite set referred to as a *constellation alphabet*. The constellation alphabet along with some additional rules on the symbol choice, e.g., the symbol occurrence probability, is referred to as the *modulation format*. The block converting the FEC-encoded bit sequence into the symbol sequence is referred to as *mapper*. For a constellation alphabet of size M, the number of the bits carried per symbol is $m \leq \log_2(M)$. Notably, $m = \log_2(M)$ only in the case of *equiprobable* symbols, i.e. when the occurence probabilitiy of each symbol in the transmitted sequence are equal.

For example, let's consider the constellation alphabet $\mathbf{S}$ of size $|\mathbf{S}| = M$ defining the set of symbols $\mathbf{S} = \{s_1, s_2, \ldots, s_M\}$ for the transmission using an equiprobable modulation format. This case, the mapper will map each m encoded bits into a single symbol $x_k \in \mathbf{S}$ resulting in the transmitted bit sequence $(x_1, x_2, \ldots, x_k, \ldots)$, $x_k \in \mathbf{S}$.

Nowadays, the information is usually transmitted over a physical medium via electromagnetic waves. Since electromagnetic waves are analogous, the discrete symbols produced by the mapper have to be transformed to a stream of analogous time-limited states in a procedure referred to as a *pulse shaping*. The choice of proper pulse shaping is similarly important as of the modulation format. While the latter defines how much information can a single symbol carry, the former one determines the time- and spectral- width of a pulse carrying the single symbol, and, therefore, the upper limit on the pulse rate.

In addition to pulse shaping, the various types of *pre-equalisation* techniques, aimed at mitigating the deterministic distortions arising in the link, are applied to the transmitted symbol and pulse sequences. The current availability of high-resolution digital-to-analog converters opens the way for implementing sophisticated pre-equalization algorithms aiming at the signal distortions injected by both transceiver devices and physical media.

After pulse shaping and pre-equalisation, the signal is modulated onto the carrier and transmitted from source to destination via a physical media, referred to as the *physical channel*. In coherent optical communications, considered in this thesis, the signal is first converted to an analogous electrical signal form by a *digital-to-analog converter* and then carved out of a laser-generated continuous waveform radiation via an *electro-optic I/Q-modulator*. Optical fiber is used as a physical channel there. At the destination, after the physical channel the received electromagnetic waveform is converted back to the digital domain. There, first, *post-equalization* digital signal processing algorithm stack is applied to the received signal to filter the deterministic distortions out of it and obtain back a sequence of discrete symbols $\mathbf{y} = (y_1, y_2, \ldots, y_k, \ldots)$.

The received filtered symbol sequence $\mathbf{y}$ is next processed by a *channel decoder*. There, first, the *demodulator* maps each received symbol to a most likely transmitted bit sequence

corresponding to it. Second, the *forward error correction (FEC) decoder* removes the redundancy introduced during the FEC encoding from the received bit sequences.

Finally, *source decoder* restores the transmitted message by decompressing the received bits.

Following the information theory notation, all the digital communication system blocks between channel decoder and channel encoder are named *communication channel*. In other words, these are all the communication link blocks working at the symbol level.

1.4.2 Estimations of achievable information rate

1.4.2.1 Shannon capacity

The *channel capacity C* is the maximum information rate achievable in a given digital communication system. It was formulated by Claude Shannon in his seminal paper [88] in 1948. In other words, the information rate R at which a reliable communication can be achieved, is smaller or equal to channel capacity, i.e., $R \leq C$.

Statistical properties of the channel define its capacity. To illustrate this mechanism, we further consider the simplest model of a communication channel - the additive white Gaussian noise (AWGN) one. In this approach, the input and output of the channel are connected as

$$Y = X + Z. \tag{1.24}$$

Here $X, Y \in \mathbb{C}$ are complex-valued variables representing, repspectively, the input and the output symbols of the channel, while $Z = \mathcal{N}\left(0, \sigma^2_{\text{AWGN}}\right) \in \mathbb{C}$ is the channel distortion represented by a complex-valued random Gaussian noise with zero mean and variance σ^2_{AWGN}. For this system, a Gaussian probabilistic distribution will define the conditional probability of receiving a particular realization of output $y \in \mathbb{C}$ given the particular input symbol $x \in \mathbb{C}$ being sent into the channel

$$p_{Y|X}(y|x) = \frac{1}{\pi\sigma^2_{\text{AWGN}}} \exp\left(\frac{-|y-x|^2}{\sigma^2_{\text{AWGN}}}\right). \tag{1.25}$$

This conditional distribution $p_{Y|X}(y|x)$ defines the channel statistic properties and can be used to estimate the capacity of the channel. For a given channel input and output, *mutual information (MI) I* estimates the maximum amount of information which can be sent over a

channel [89]. It can be expressed as

$$\begin{aligned} I(X,Y) &= \iint dxdy\, p_{XY}(x,y) \cdot \log_2\left(\frac{p_{Y|X}(y|x)}{p_Y(y)}\right) = \\ &= \iint dxdy\, p_{XY}(x,y) \cdot \log_2\left(\frac{p_{XY}(x,y)}{p_X(x) \cdot p_Y(y)}\right), \end{aligned} \tag{1.26}$$

where $p_{XY}(x,y)$ is the joint probability distribution and $p_X(x), p_Y(x)$ are marginal probability density functions of, respectively, channel input X and output Y.

Let's notice that the only variables in mutual information which the system designer is able to control are the channel conditional distribution $p_{Y|X}(y|x)$ and the probability density function of the channel input $p_X(x)$. The joint probability distribution $p_{XY}(x,y)$ and the marginal probability density function of the channel output $p_Y(y)$ can be expressed from them as

$$\begin{aligned} p_{XY}(x,y) &= p_{Y|X}(y|x) \cdot p(x), \\ p_Y(y) &= \int dx\, p_{Y|X}(y|x) \cdot p(x). \end{aligned} \tag{1.27}$$

In turn, the channel capacity can be introduced as the maximum of mutual information over all possible input distributions $p_X(x)$, provided the fixed channel conditional distribution $p_{Y|X}(y|x)$, i.e.,

$$C = \max_{p_X} \left[I(X,Y) \right]. \tag{1.28}$$

In [88] Shannon proved that for an AWGN channel the following complex-valued Gaussian input distribution maximizes the mutual information

$$p_X(x) = \frac{1}{\pi P} \exp\left(\frac{-|x|^2}{P}\right). \tag{1.29}$$

where P is the signal power. By substituting the optimal input probability distribution Eq. (1.29) into the mutual information $I(X,Y)$ definition Eq. (1.26) we obtain the following famous formula of the channel capacity of an AWGN channel, also known as the Shannon capacity

$$C = \log_2\left(1 + \mathrm{SNR}\right), \tag{1.30}$$

expressed in bits per channel used. Here SNR is the *signal-to-noise ratio (SNR)*

$$\mathrm{SNR} = \frac{P}{\sigma_2}, \tag{1.31}$$

in other words the ratio of average signal power P to the variance of the signal distortion σ^2. The channel capacity, therefore, expresses the maximum information rate achievable over the AWGN channel with a given SNR value.

1.4.2.2 Constrained capacity

The definitions of the previous Section 1.4.2.1 were obtained in the assumption of input X and output Y of the channel being defined as a continuous variable allowing the infinite variants of inputs and outputs. Meanwhile, in practice, as mentioned in Section 1.4.1, the input X is sampled from a discrete finite distribution referred to as a constellation alphabet. The resulting upper limit on the information rate, corresponding to a finite input alphabet, is referred to as the *modulation constrained capacity* and is lower than the Shannon one Eq. (1.30).

Furthermore, the mutual information definition Eq. (1.26) developed for continuous inputs has to be adapted to become applicable to the more realistic case of digital communication system with discrete inputs. For a finite communication alphabet we have to estimate the integral over all the possible input and output realizations in definition Eq. (1.26) via a Monte-Carlo time averaging over a long enough transmitted symbol sequence of length $N \to \infty$

$$\tilde{I}(X,Y) = \frac{1}{N}\sum_{k=1}^{N} \log_2\left(\frac{p(y_k|x_k)}{\sum_{s\in\mathbf{S}} p(y_k|x)p_X(x)}\right), \tag{1.32}$$

where x_k, y_k are, correspondingly the symbols transmitted and received at k-th timeslot, and $\mathbf{S}$ is the constellation alphabet. This formula estimates the maximum information rate achievable in a digital communication system with a given finite constellation $\mathbf{S}$ and set of symbol occurence probability $p_X(x)\, x \in \mathbf{S}$. Nonetheless, to reach this limit, the FEC coding scheme and mapping should be jointly designed, an approach referred to as the *coded modulation*.

1.4.2.3 Generalized mutual information. Source entropy

The coded modulation approach results in rather complex systems, therefore, in practical digital communication links a simpler alternative is usually employed - bit-interleaved coded modulation (BICM) scheme.

Bit-interleaved coded modulation (BICM) systems employ a more flexible design with a separate forward error correction (FEC) encoder and mapper. BICM scheme has several important details. First, the bits at the output of FEC encoder are interleaved to spread possible burst errors affecting a group of neighbouring bits over the whole bit sequence.

Second, at the receiver side, the information about the received bit is communicated from demapper to the FEC decoder via *soft bits* - a posteriori probabilities of bit values or their logits.

Usually, the implementation of bit-interleaved coded modulation scheme causes only minor capacity losses compared to the full coded modulation scheme, if mapping uses Gray code [90], the so-called *Gray mapping*. Gray mapping is designed in a way that confusing the received constellation symbol with a neighboring one results in a single bit error.

Thanks to bit-interleaving, one can consider a bit-interleaved coded modulation system as separated into m parallel independent memoryless binary channels. The maximum achievable information rate in BICM system is typically referred to as *generalized mutual information* (GMI) and defined as [91, 92]

$$\mathrm{GMI} \approx H(X) - \frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{m}\log_2\left(\frac{\sum_{x\in\mathbf{S}} q_{X|Y}(y_i|x)p_X(x)}{\sum_{x\in\mathbf{S}^+} q_{Y|X}(y_i|x)p_X(x)}\right), \tag{1.33}$$

Here $q_{Y|X}$ is the approximation of the actual channel conditional probability distribution $p_{Y|X}$ used by demapper to produce the soft bit estimations, $\mathbf{S}$ is the constellation alphabet, and $\mathbf{S}^+ \subset \mathbf{S}$ is the subset of all alphabet symbols, which have the same value of the i-th bit similar to the bit label of the k-th transmitted symbol, and $H(X)$ is the *source entropy*. It describes the amount of information carried by a single transmitted symbol in a given modulation format, in other words. By thus, source entropy $H(X)$ defines the upper limit on information rate achievable in the link implementing the given modulation format. Source entropy is expressed as

$$H(X) = -\sum_{x\in\mathbf{S}} p_X(x)\cdot\log_2\left(p_X(x)\right). \tag{1.34}$$

Notably, $H(X)$ reaches maximum when the modulation format is equiprobable, i.e., when $p_X(x) = 1/M, \forall x \in \mathbf{S}$ with $M = |\mathbf{S}|$ being the constellation alphabet size. This case, source entropy acquires its maximum value

$$\underset{X,p_X}{\operatorname{argmax}}\, H(X) = \log_2(M). \tag{1.35}$$

1.4.3 Forward error correction codes

To remind, the main purpose of forward error correction codes to make the transmitted bit sequence robust to distortions by adding redundancy to it. At the receiver, FEC decoder uses this redundancy to recover the original information-bearing bit sequence near error-free. The

important FEC parameter - *code rate* is defined by

$$r_c = \frac{K_c}{N_c}, \tag{1.36}$$

where K_c and N_c are the sizes of the processed block of bits before and after the FEC encoding, correspondingly. In other words, FEC encoder takes the blocks of K_c bits and injects $N_c - K_c$ into each of them. Also, we can define the *FEC overhead* by

$$OH = \frac{N_c - K_c}{K_c} = \frac{1 - r_c}{r_c}. \tag{1.37}$$

The Shannon capacity and the GMI Eq. (1.33) are defined in an assumption of an ideal forward error correction code used with the infinite block length $N_c \to \infty$. Application of a practical FEC code with finite block lengths causes the information rate in the system to go below the aforementioned achievable information rate estimations.

The performance of a digital telecommunication system is often quantified via the *bit-error-rate (BER)*, a ratio of the number of erroneously received bits to the number of all the transmitted ones. The efficacy of a forward error correction coding scheme is, therefore, often evaluated via a *coding gain* metric, a difference of signal-to-noise ratios required for the system to reach a desired bit-error-ratio level with and without forward error correction. Typically, optical communication systems, considered in this thesis, require bit-error-rates (BER) no more than 10^{-15}.

There exist a lot of different forward error correction coding schemes: Reed-Solomon codes, Hamming codes, Bose–Chaudhuri–Hocquenghem (BCH) codes, turbo codes, low-density parity check (LDPC) codes, et al. Nowadays, low density parity check codes with coding overhead $OH \approx 20\%$ are the common choice for the state-of-the-art coherent optical transmission systems with the transmission rate beyond 100 GB/s per channel. Spatially coupled low-density parity check codes, a new flavor of LDPC codes is becoming increasingly more popular nowadays with enables capacity reaching performance in the links at a reasonable complexity costs.

Although forward error correction is not the main topic of this thesis, we refer an interested reader to [93] for the comprehensive review of forward error correction applications to coherent optical communications.

1.4.4 Modulation format. Constellation shaping

In coherent optical communication systems the information is encoded onto the amplitude and phase of the propagated wave packets. As a result, *constellation alphabet* **S**, i.e., the set

of all possible transmitted symbols, for this type of systems can be represented as a set of complex-valued numbers $s_k \in \mathbf{S} \in \mathbb{C}$. The real and imaginary components of a constellation symbol are referred to as the *in-phase (I)* and *quadrature (Q)* components that are usually modulated using separate digital-to-analog converters. The *constellation diagrams*, i.e. the figures plotting the real $\mathrm{Re}(x)$ and imaginary parts $\mathrm{Im}(x)$ for all the symbols of the constellation alphabets $x \in \mathbf{S}$ are given in Figure 1.2.

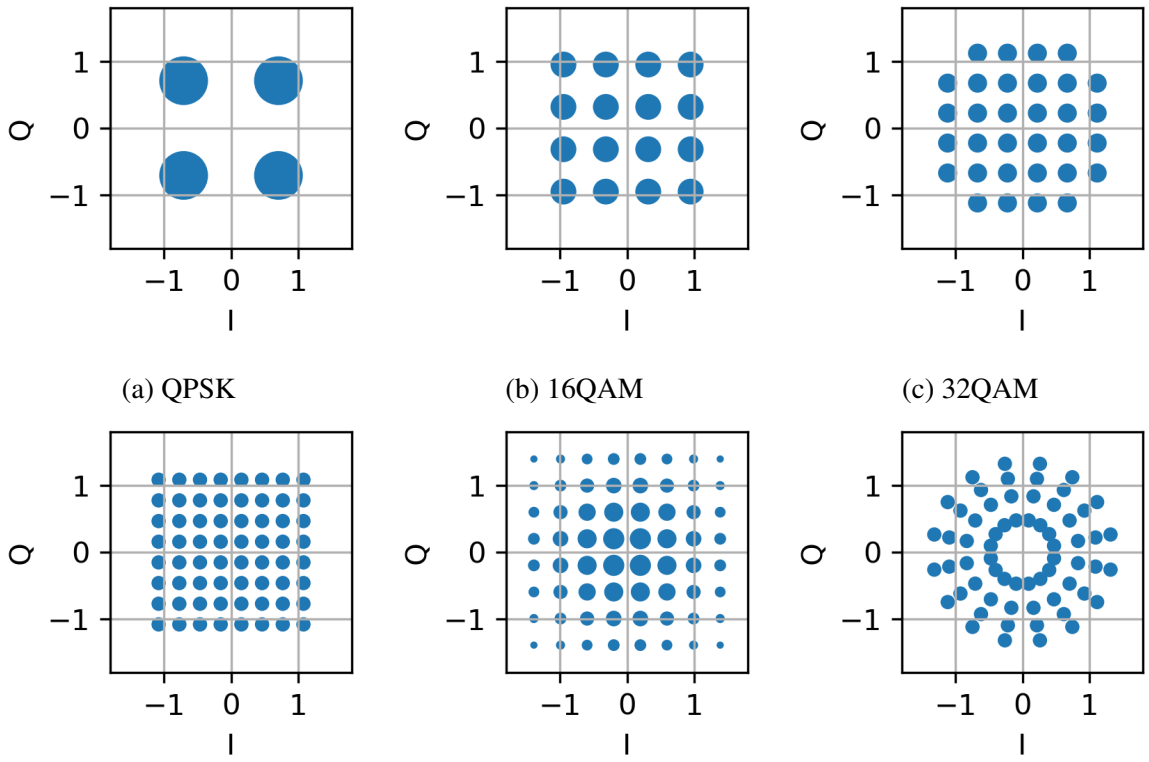
Remind ourselves that *modulation format* in addition to constellation alphabet includes the discrete probability distribution of each of the symbols being transmitted $p_X(x), \forall x \in \mathbf{S}$. Various modulation formats were suggested for coherent optical communications.

In coherent optical communication systems the baseline conventional family of modulation formats is equiprobable *quadrature amplitude modulation (QAM)*. Symbols in QAM constellations are defined on the square grid

$$s = a + b \cdot i;\ a, b \in [-1, 1, -3, 3, \ldots, \pm 2k + 1, \ldots],\ \forall s \in \mathbf{S}_{\mathrm{QAM}}, \tag{1.38}$$

with i being the imaginary unit, to simplify sampling of these symbols via low-resolution digital-to-analog converters. The particular QAM constellations differ in the number of symbols they include, with the constellation names explicitly including this number as M-QAM, where M is the number of symbols in the alphabet referred to as the constellation *order*, e.g., 16QAM. QAM constellation with $M = 4$, depicted on Figure 1.2a, is the notable exception being referred to as the *quadrature phase-shift keying (QPSK)*, since all the symbols there have the same amplitude. In practice, QAM constellations with symbol numbers equal to the natural powers of two $M = 2^m, m \in \mathbb{N}$ are only used, to simplify mapping by making each symbol to carry an integer number of bits $m = \log_2(M)$, e.g. 16QAM (Figure 1.2b), 32QAM (Figure 1.2c), 64QAM (Figure 1.2d), et al. Furthermore, the preference is given to the constellations carrying the number of bits divisible by two $m \bmod 2 = 0$, since they can be represented via a square grid which effectively uses the resolution of digital-to-analog converters.

The choice of the appropriate QAM constellation is about finding a balance. On the one hand, obviously, using QAM constellations with higher order is preferential, because, it can improve the information rate of the system by increasing the amount of information we pack in each symbol, $H(X)$ from the definition of generalized mutual information (GMI) Eq. (1.33). On the other hand, for the link with a fixed average power level and the transmitter resolution, the higher order constellations are more susceptible to the distortions arising in the link, because of more tightly packed constellation points. This affects the second term in GMI Eq. (1.33) via a channel probability distribution $q_{Y|X}$.



(a) QPSK (b) 16QAM (c) 32QAM

(d) 64QAM (e) PAS-64QAM (f) 64APSK

Fig. 1.2 Constellation diagrams illustrating the constellation alphabets **S** described in Section 1.4.4: quadrature phase shift keying (QPSK), 16-, 32-, and 64-symbol quadrature amplitude modulation (16QAM, 32QAM, 64QAM), 64-symbol probabilistic amplitude shaping (PAS-64QAM), and 64-symbol amplitude phase-shift keying. The size of each marker is proportional to the occurrence probability p_X of the corresponding symbol.

A detailed study of the dependence of the achievable information rate on signal-to-noise ratio in the link for various QAM constellation formats was done in [89]. Notably, this study highlighted that, counterintuitively, at low signal-to-noise ratio, lower order constellations can lead to better information rates overall.

Nonetheless, the conventional quadrature amplitude modulation formats, even with the fine-tuned constellation order, are suboptimal and lead to the information rates considerably lower than the Shannon capacity Eq. (1.30). *Gaussian-like* modulation formats were proposed for reaching the Shannon capacity. In this approach, the finite constellation alphabet is *shaped* to make the symbol distribution to approach the optimal continuous Gaussian distribution Eq. (1.29), via a technique named *constellation shaping (CS)*. First, the shaping can be done by rearranging the equiprobable alphabet symbols to make their spatial distribution to appear like Gaussian, the method referred to as the *geometrical constellation shaping*

(GCS). Second, the QAM constellation points can be drawn non-uniformly according to a discrete propability distribution approaching the Gaussian one - the technique known as the *probabilistic constellation shaping (PCS)*. Finally, probabilistic and geometric shapings can be combined to better approach the Gaussian - the method referred to as the *hybrid constellation shaping*. Although constellation shaping is well known in the general information theory [94, 95], it was introduced for coherent optical communication theory only recently, where it was shown to considerably outperform the conventional QAM formats [92, 96–102].

A notable example of practical geometric constellation shaping in coherent optical communications was proposed in [102]. It combined *amplitude phase shift keying (APSK)*, assembling the constellation alphabet of several rings with uniformly distributed symbols, an approach well known on satellite communication, with Gray coding. In [100], a 64-symbol amplitude phase-shift keying (64APSK), depicted on Figure 1.2f, was applied for long-haul 6000 km coherent fiber-optic transmission, where it outperformed the convetional 64QAM.

A practical implementation of probabilistic constellation shaping based on QAM constellation was recently proposed in [103]. This method was later referred to as *probabilistic amplitude shaping (PAS)*. In this approach, the locations of original QAM symbols remain intact, while the probability mass function is defined following a Maxwell-Boltzmann distribution with a parameter $\nu \leq 0$ as

$$p_X(x) = \frac{\exp(-\nu|x|^2)}{\sum_{x\in\mathbf{S}} \exp(-\nu|x|^2)}. \tag{1.39}$$

The parameter ν can be optimized to modify the source distribution and, by thus, to maximize the achievable information rate for a given signal-to-noise ratio present in the link. In general, the probabilistic shaping of a QAM constellation with the probablility distribution Eq. 1.39 is referred to as the *Maxwell-Boltzmann shaping*.

Another notable feature of the PAS approach is that imposing the probabilistic mass function onto the constellation symbols is done by a separate block referred to as the *distribution matcher (DM)*. At the transmitter, the distribution matcher is applied to a bit sequence before the FEC encoder. Accordingly, at the receiver, the inverse DM is applied after the forward error correction code decoder. The design of a distribution matcher is a complex topic and is beyond the scope of this thesis.

The probabilistic amplitude shaping approach to PCS have two notable advantages. First, it uses the conventional QAM constellations, which are desirable thanks to their overall simplicity leading to easiness of implementation via low-resolution transceivers, and for which a cost-effective bit-interleaved coded modulation scheme can be implemented with low cost via Gray mapping. Second, the separation of FEC encoder and distribution

matcher allows finer rate adaptation, leading to overall higher information rates [96]. Notably, in [101] it was shown that for 64-symbol transmission over 6600 km probabilistic amplitude shaping outperformed the amplitude phase shift keying one, while at the same time showing higher tolerance to phase noise and frequency offsets, leading to lower implementation penalties [104]. Because of these advantages, the probabilistic amplitude shaping has become the most important constellation shaping technique.

Unfortunately, the AWGN channel model considered in the previous discussions, is a significant oversimplification of the actual distortion introduced by the fiber-optic communication link. In reality, because of nonlinear distortions present in the link, the distortion injected into the transmitted signal depends on its properties. In general, this dependence is described as enhanced Gaussian noise (EGN) model [105, 106] which models the nonlinear distortion as a constellation dependent noise. Nonlinearity-aware shaping is an important research topic, with some solutions already being implemented in the recent coherent optical telecomunication products, e.g., by Ciena [107] and Infinera [108]. This topic is covered in Chapter 3 of this thesis.

1.4.5 Performance metrics of a digital communication system

Despite the bit-error rate of the bit sequence after forward error correction decoder (post-FEC BER) being one of the most important performance metrics of a digital communication system, it is almost impossible to estimate it directly in practice. With the required post-FEC BER $< 10^{-15}$, it takes too long to send enough bits over the link to measure it.

Thankfully, the two approaches were suggested to estimate the post-FEC BER indirectly via the other metrics. The first one is *FEC limit* - the bit-error rate before the FEC decoder required to ensure that post-FEC BER is lower than 10^{-15}. This approach works well with the hard-decision forward error correction codes (HD-FEC), operating on the demapped bit sequence of 0s and 1s, however, it is less accurate when applied to the soft-decision forward error correction code (SD-FEC) decoder, operating on soft bits, i.e. the estimated probabilities of '1's being transmitted in each of received bit slots [109]. When an SD-FEC is used, generalized mutual information (GMI) Eq. (1.33) and mutual information (MI) Eq. (1.32) provide better post-FEC BER estimations for, respectively, bit-interleaved coded modulation and coded modulation systems. It this thesis, we focus on the three aforementioned metrics - pre-FEC BER, MI, GMI.

While we already described the MI and GMI estimation process in the previous section, we now focus on pre-FEC BER calculation. The pre-FEC BER can be estimated from the seqences of transmitted $\mathbf{x} = [x_1, x_2, \ldots, x_k, \ldots]$ and the received symbols $\mathbf{y} = [y_1, y_2, \ldots, x_k, \ldots]$

as

$$\mathrm{BER} = \frac{1}{N \cdot m} \sum_{k=1}^{N} d_{\mathrm{Hamming}} \left(\mathcal{M}[x_k], \mathcal{M}[\tilde{x}_k] \right). \tag{1.40}$$

Here N is the symbol sequence length, m is the number of bits per symbol, $\mathcal{M}[\cdot]$ is mapping of the symbols to corresponding bit labels, d_{Hamming} is the Hamming distance between the two bit sequences, and $\tilde{x}_k$ is the k-th hard-decided symbol.

$$\tilde{x}_k = \underset{x \in \mathbf{S}}{\operatorname{argmin}} [|y_k - x|^2] \tag{1.41}$$

with $\mathbf{S}$ being the constellation alphabet.

Sometimes, bit-error rate is converted to the performance metric named Q^2-factor expressed in decibels, defined as

$$Q^2 = 20 \log_{10} \left(\sqrt{2} \cdot \mathrm{erfc}^{-1}(2 * \mathrm{BER}) \right) \tag{1.42}$$

where $\mathrm{erfc}^{-1}(\cdot)$ is the inverse complimentary error function.

Another popular metric of a digital communication system is signal-to-noise ratio (SNR) Eq. 1.31 - the ratio of the average signal power to the signal distortion variance. This metric directly quantifies the signal distortions, and, therefore, does not depend on the choice of modulation format and modulation scheme like the aforementioned pre-FEC BER, MI, GMI. In state-of-the-art long-haul high-baudrate fiber-optic communication links all the distortions arising in the channel can be well approximated via a Gaussian noise [105], which makes the signal-to-noise ratio, estimating the variance of the an effective metric for estimating the nonlinear distortions. SNR can be estimated from the sequences of received $\mathbf{y}$ and transmitted $\mathbf{x}$ symbols via

$$\widetilde{\mathrm{SNR}} = \left[\frac{(\mathbf{x}, \mathbf{x})^2 (\mathbf{y}, \mathbf{y})^2}{(\mathbf{x}, \mathbf{y})^2} - 1 \right] \tag{1.43}$$

where $(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{N} a_i b_i^*$, $\mathbf{a}, \mathbf{b} \in \mathbb{C}^N$ is a dot product of complex-valued vectors.

Chapter 2

Data augmentation for nonlinearity compensation algorithms in coherent optical communications

2.1 Introduction

The fiber Kerr nonlinearity is often considered as one of the major limiting factors affecting the performance of modern optical communication systems [89]. Nonlinear signal distortions restrict the possibility to mitigate the detrimental impact of random noise on the data transmission quality by merely increasing the optical signal power, in contrast to the linear channels. Several approaches have been proposed and studied for the digital Kerr nonlinearity compensation (NLC) at the receiver and transmitter [110–112]. An important group of those constitute the supervised-learnt nonlinearity compensation (SL-NLC) algorithms which estimate the system parameters required for the compensation by analysing the pre-collected training datasets formed by pairs of known transmitted patterns and the corresponding received signals [4]. The key features of SL-NLC algorithms are: (i) relaxing the requirement of a priori knowledge of the channel parameters, and (ii) their capability to automatically adapt themselves to the changes in the link by re-training on the newly collected dataset. Importantly, these algorithms include not only the emerging machine learning (ML) based techniques covered in reviews [15, 25, 113], notably, [34, 114–117], but also more conventional approaches like the adaptive perturbative post-distortion (PPD) [118].

The training of SL-NLC algorithms can consume considerable time and computational resources [34, 119, 120], which leads to inevitable challenges in practical implementation of such methods. As a matter of fact, currently, the training stage of the SL-NLC algorithms

is typically executed offline using the separately collected transmission data [34, 120], and the training complexity is proportional to the size of the dataset used [119]. Therefore, decreasing the size of this dataset, under the condition of keeping the same performance level, allows us to reduce the numerical complexity of the training stage.

Apart from the numerical complexity issues, the size of the required training dataset may impact the overall performance of a communication system. The transmission of known signal patterns during collection of large training datasets increases the redundancy in the system and reduces the amount of available information that it carried. Further, when flexible routing is used or if the parameters of communication system evolve fast, the frequent recollection of large datasets needed for SL-NLC re-training can effectively reduce the overall capacity of the link. Besides this, the required dataset may be too large to fit into the limited memory resources.

Nonetheless, the naive decimation of the training dataset may lead it to the loss of accuracy in representing the ensemble of real world objects (e.g., sequences of quadrature-amplitude modulated (QAM) symbols) [4, 119]. This typically manifests itself in overfitting, i.e. the trained model performing poorly on the real-world data while showing satisfactory results on the training one. Therefore, it is desirable to have the possibility to reduce the required dataset in a flexible manner without causing a substantial drop in the performance or, alternatively, to improve the performance by using the same dataset. One of the techniques used in ML to efficiently address the overfitting issue is *data augmentation* (DA) – the expansion of the dataset by adding the synthetically generated new training objects. The new objects can be generated in many ways depending on the dataset structure. For example, in computer vision, the simple transformations of the images (e.g., mirroring, rotation, cropping) are widely used to expand incomplete training datasets [121]. In optical communications, the DA has been recently considered in network scenario for predicting failures [122, 123] and traffic peculiarities [124, 125]. The aforementioned network applications suggested new object generation by graph-adversarial networks (GANs) [123–125] utilising the heuristics [122]. It is worth noting, that training supervised learnt algorithms for every particular task requires a unique dataset structure, and, hence, a unique data augmentation procedure. Therefore, aforementioned data augmentation techniques from the networking layer are not applicable to signal distortion mitigation in the physical layer of optical communications, considered in this article.

In this work, we propose the DA technique for improving the training of SL-NLC algorithms. To the best of our knowledge, this is the first time that a DA technique has been proposed for nonlinear distortions' compensation in fiber-optic communication systems. We show that the suggested DA technique may improve the system performance in two different

contexts. The first case is when we have an insufficient amount of training data leading to poor performance of SL-NLC. As mentioned before, the maximum size of the available dataset can be bounded by capacity and memory limits. In this case, the DA can enable the SL-NLC algorithm to reach the same level of performance as if it was trained with a much larger dataset. The second case is when a dataset is large enough to enable the optimal SL-NLC performance. Here, the DA can be used to shrink an available dataset, by thus, reducing also the overall numerical cost of algorithm training while preserving a similar performance level.

We have studied the effect and benefit of the proposed DA both numerically and experimentally. For generality, we considered different types of systems numerically and experimentally. In numerical study, we focused on the case of the more idealized link with the optical intra-channel nonlinearity being the only source of nonlinear distortion. In the numerical modeling, in order to demonstrate the generality of the approach, we have shown the effect of DA on two dissimilar SL-NLC algorithms (see Figure 2.2) when considering several testcases. For clarity, only the channel-induced distortions described by the Manakov equations (2.1) were enabled in the numerical simulations. For the considered testcases, we show that the DA leads to the same performance as when using $4\times$ – $6\times$ larger datasets (see Figure 2.4). Moreover, in the case of a large enough dataset, the DA enabled reducing the numerical complexity by $2\times$ while still leading to the same system performance (see Figure 2.5). Meanwhile, in the experimental study we applied the DA to a realistic link where the distortions caused by transceiver device nonlinearities were also present. In the experiment we considered a field trial of a metro coherent-detection system employing a low-cost transceiver (i.e., with considerable transceiver-induced impairments). In the experiment, the DA enabled reducing the training dataset by $4\times$ while keeping the same NLC performance of the case without it (see Figure 2.7).

This Chapter is based on the original published contribution [126]. Also, original unpublished material is included.

The remainder of the chapter is organized as follows. In Sec. 2.2 we introduce the data augmentation procedure and illustrate how it can be implemented in a Kerr nonlinearity equaliser (NLE) based on SL-NLC. Sec. 2.3 describes the SL-NLC algorithms considered in this work and the testcases evaluated in the numerical simulations. In Sec. 2.4 we describe the obtained numerical results, while Sec. 2.5 describes the experimental study. Finally, Sec. 2.6 outlines the main conclusions of the chapter.

2.2 Data augmentation mechanism

To demonstrate the proposed technique, we consider, without loss of generality, the Manakov equations describing the practical case of the evolution of a dual-polarization light envelope during propagation down a fiber-optic communication link. Within certain limits, we write these equations as [89]:

$$\frac{\partial u_{h/v}}{\partial z} = \frac{G(z)}{2} u_{h/v} - i\frac{\beta_2(z)}{2}\frac{\partial^2 u_{h/v}}{\partial t^2} + i\frac{8\gamma(z)}{9}(|u_h|^2+|u_v|^2)u_{h/v} + \xi(z,t). \tag{2.1}$$

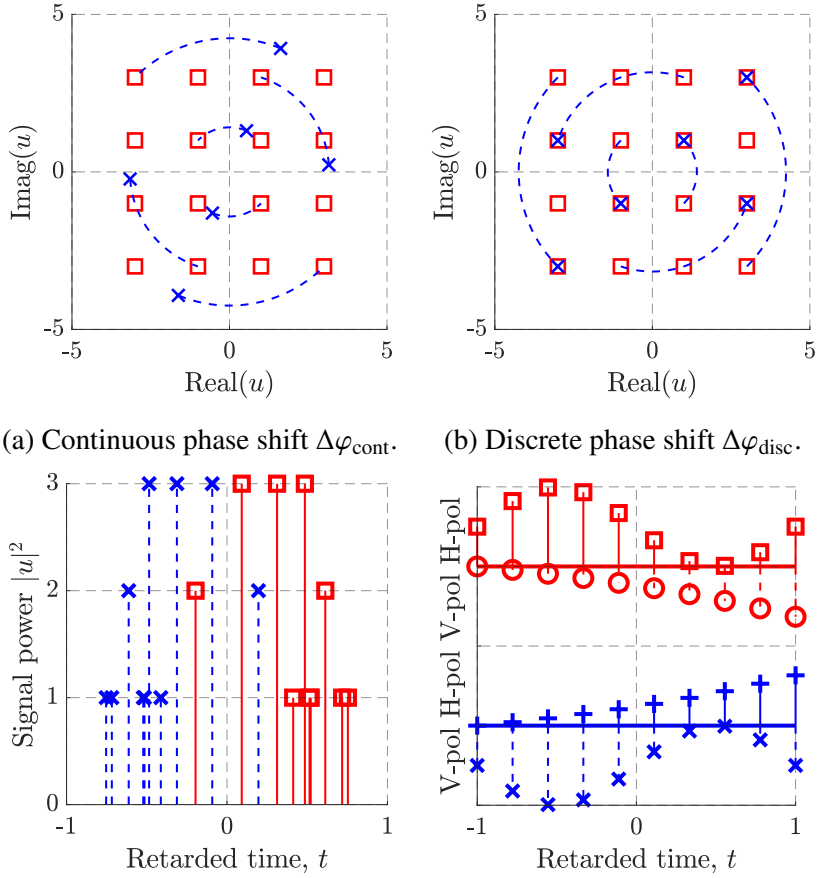
Here $u_h(z,t)$ and $u_v(z,t)$ are, respectively, the horizontal (h) and vertical (v) polarizations of the two-component optical signal waveform $u(z,t)$; $\beta_2(z)$ is the group velocity dispersion (GVD) coefficient; $\gamma(z)$ is the effective nonlinear coefficient; $G(z) = -\alpha(z) + \sum_{m=1}^{N_{\mathrm{OA}}} \Gamma_m \delta(z - L_m)$ stands for the optical power loss $\alpha(z)$ fully compensated by the lumped optical amplifiers (OAs) with gain Γ_m situated at L_m positions in the end of every fiber span (the summation runs over the span number m), and $\xi(z,t)$ is the amplified spontaneous emission (ASE) noise injected by OAs, which is modelled as additive white Gaussian noise (AWGN) added at each amplification point.

Let us now define a solution of Manakov equations Eq. (2.1) through the pair of functions corresponding to the channel input and output: $\{u_{h/v}(0,t), u_{h/v}(z,t)\}$. Several transformations can be used to construct new solutions of Eq. (2.1) starting from the existing ones without solving the equation again. The simplest solution-generating transformations, considered further in this paper, are listed in Table 2.1. One can check their validity by substituting the generated solutions into Eq. (2.1).

In essence, the NLC algorithms aim at predicting the transmitted signal $u_{h/v}(0,t)$ given the information about the received one $u_{h/v}(z,t)$ by reverting the deterministic nonlinear propagation effects introduced by the channel. Therefore, the training dataset of the supervised-learnt nonlinearity compensation (SL-NLC) algorithms is formed by the sampled input-output pairs $\{u_{h/v}(0,t), u_{h/v}(z,t)\}$ and/or the features derived from them. A larger and more diverse dataset usually leads to the better performance of a supervised-learnt algorithm. This is because the expanded dataset represents more accurately the multitude of all possible received signals [119] and, hence, enables the algorithm to learn more accurately the approximation of the inverse channel. Nonetheless, when the dataset is large and diverse, the further increase of its size leads to negligible or even no performance improvement.

We propose to use the transformations given in Table 2.1 and depicted in Figure 2.1 for synthetically expanding the training dataset of the SL-NLC algorithm. The synthetic expansion of the data is referred to as the data augmentation (DA) in the general ML context.

We implement the DA in a way that preserves memory, therefore eliminating the need to remember all the synthetic data. For every training epoch (i.e., single pass of the algorithm optimizer over the dataset during training) we generate the unique training dataset by taking the pre-collected original one and applying the transformations from Table 2.1 to a randomly picked part of its objects. As a result, every training epoch is done on a new dataset made up from original and artificial objects, unlike the conventional training approach which uses a fixed original dataset. The augmented dataset size is kept constant during all epochs.



(a) Continuous phase shift $\Delta\varphi_{\text{cont}}$. (b) Discrete phase shift $\Delta\varphi_{\text{disc}}$.

(c) Time-inversion t_{inv}. (d) Polarization swapping H/V_{swap}.

Fig. 2.1 The scheme of the transformations generating the new solutions of Manakov Eq. (2.1) out of the existing ones. The more rigorous description of the transformations is given in Table 2.1.

The procedure describing in more details the training of SL-NLC algorithm with the augmented dataset is presented in pseudocode Algorithm 2. Furthermore, we have published the Python code implementing it on [127]. We assume that every dataset object consists of the desired transmitted symbol of interest (SOI) in both polarizations H_{TX}, V_{TX} and the input vector containing features like the received symbol sequence $\bar{H}$, $\bar{V}$ centered around

Transformation name	Symbol	Mathematical formulation
Continuous phase shift	$\Delta\varphi_{\text{cont}}$	$\{\bar{u}_{h/v}(0,t), \bar{u}_{h/v}(z,t)\} \rightarrow \{\bar{u}_{h/v}(0,t)\cdot\mathbf{exp}^{i\varphi}, \bar{u}_{h/v}(z,t)\cdot\mathbf{exp}^{i\varphi}\}$
Discrete phase shift	$\Delta\varphi_{\text{disc}}$	$\{u_{h/v}(0,t), u_{h/v}(z,t)\} \rightarrow \{\bar{u}_{h/v}(0,t)\cdot\mathbf{exp}^{i\varphi}, \bar{u}_{h/v}(z,t)\cdot\mathbf{exp}^{i\varphi}\}$
Time-inversion	t_{inv}	$\{\bar{u}_{h/v}(0,t), u_{h/v}(z,t)\} \rightarrow \{\bar{u}_{h/v}(0,\boldsymbol{-t}), u_{h/v}(z,\boldsymbol{-t})\}$
Polarization swapping	H/V_{swap}	$\{u_{h/v}(0,t), u_{h/v}(z,t)\} \rightarrow \{u_{\boldsymbol{v\leftrightarrow h}}(0,t), u_{\boldsymbol{v\leftrightarrow h}}(z,t)\}$

Table 2.1 Transformations generating the new solutions of Manakov Eq. (2.1) out of the existing ones. These transformations can be used to synthetically expand the training dataset in data augmentation.

SOI H_0, V_0, i.e., $\bar{H} = \{H_{-n}, ..., H_{-1}, H_0, H_1, ..., H_n\}$. For continuous ($\Delta\varphi_{\text{cont}}$) and discrete ($\Delta\varphi_{\text{disc}}$) phase shift, the unique phase rotation randomly chosen from, respectively, $[0, 2\pi)$ or $\{0, \pi/2, \pi, 3\pi/2\}$ is applied to each object in the dataset. In the case of time-inversion (t_{inv}) and polarization swapping (H/V_{swap}), the transformation is applied to a randomly chosen half of training objects to get a higher variability in data. The time-inversion t_{inv} is obtained by reversing the order of elements in the input feature vector $H_j, V_j \rightarrow H_{-j}, V_{-j}\, \forall j$. The polarisation swapping is done by exchanging h- and v- polarisation components in the received sequences $\bar{H} \leftrightarrow \bar{V}$ and the transmitted SOIs $H_{\text{TX}} \leftrightarrow V_{\text{TX}}$. To apply DA to the other features derived from the received signal $\bar{H}, \bar{V}$, like nonlinear perturbation terms (NPTs) (see Eq. (2.4)), one can, first, apply transformations to the received signal $\bar{H}, \bar{V}$ and, then, generate the needed features out of the augmented signal, or find analytically the effect of transformations on the features and directly apply it for feature augmentation.

Moreover, DA can combine several transformations from Table 2.1, leading to the better NLC performance caused by stronger data versatility. For instance, in Sec. 2.4 we consider the augmentation based on the combination of discrete phase shift, time-inversion and polarization swapping: ($\Delta\varphi_{\text{disc}} + t_{\text{inv}} + H/V_{\text{swap}}$) and show that it significantly outperforms the augmentations based on just a single transformation (Figure 2.4).

Algorithm 2: Training of an SL-NLC algorithm implementing the suggested data augmentation.

Input: A randomly initialised SL-NLC algorithm (*model*); a training dataset (*dataset*) formed by the objects (*object*) each consisting of: a transmitted symbol of interest (SOI) (H_{TX}, V_{TX}) and received soft-symbol sequences centered around SOI ($\bar{H}, \bar{V}$)

Output: The trained SL-NLC algoritm *model*

for *every epoch* **do**
  for *object* ***in*** *dataset* **do**
    if $\Delta\varphi_{cont}$*-based augmentation is applied* **then**
      choose random $\Delta\varphi$ on interval $[0, 2\pi)$
      object $\leftarrow$ *object* $\times \exp(i\Delta\varphi)$
    end
    if $\Delta\varphi_{disc}$*-based augmentation is applied* **then**
      choose $\Delta\varphi$ randomly from set $\{0, \pi/2, \pi, 3\pi/2\}$
      object $\leftarrow$ *object* $\times \exp(i\Delta\varphi)$
    end
    $\bar{H}, \bar{V}, H_{\mathrm{TX}}, V_{\mathrm{TX}} \leftarrow$ *object*
    if t_{inv}*-based augmentation is applied* **then**
      choose $\mathcal{T}$ randomly from set {True, False}
      if $\mathcal{T}$ **then**
        `// Invert the time order of received signal`
        $\bar{H} \leftarrow \bar{H}$ flipped around SOI H_0
        $\bar{V} \leftarrow \bar{V}$ flipped around SOI V_0
      end
    end
    if H/V_{swap}*-based augmentation is applied* **then**
      choose $\mathcal{P}$ randomly from set {True, False}
      if $\mathcal{P}$ **then**
        `// swap h and v polarizations`
        $\bar{H} \leftrightarrow \bar{V}$ `// in received signal`
        $H_{\mathrm{TX}} \leftrightarrow V_{\mathrm{TX}}$ `// in desired signal`
      end
    end
    object $\leftarrow \bar{H}, \bar{V}, H_{\mathrm{TX}}, V_{\mathrm{TX}}$
    add to *object* the other features generated from $\bar{H}$, $\bar{V}$
  end
  update *model* parameters using augmented *dataset*
end
return *model*

On the other side, only the solution-generating transformations of Eq. (2.1) can be used to effectively augment the dataset. Some general symmetries, including the ones of the signal constellation, namely of quadrature-amplitude modulation (QAM), cannot be used for data augmentation and would lead to the poorer performance of NLC. To illustrate this effect, we also considered the augmentation based on the complex conjugation of a randomly chosen half of training objects in the dataset:

$$X^*: \quad \{\bar{u}_{h/v}(0,t), \bar{u}_{h/v}(z,t)\} \to \{u^*_{h/v}(0,t), \bar{u}^*_{h/v}(z,t)\}, \tag{2.2}$$

where $()^*$ stands for the complex conjugation. Eq. (2.2) constitutes the symmetry of QAM constellation but not of the channel model Eq. (2.1), and was shown to decrease the performance of SL-NLC, see Figure 2.4.

It is worth noting that the transformations shown in Table 2.1 are still valid if the received signal $u_{h/v}(z,t)$ is multiplied by an arbitrary complex-valued coefficient $\mathcal{K} \in \mathbb{C}$. This means that the data augmentation can be used for improving the SL-NLC methods operating on the data already processed by linear digital signal processing (DSP) algorithms [26], notably, adaptive filters and phase recovery. The reason is that the collateral effect of DSP on the ideally restored received signal can be viewed as a multiplication by a constant $\mathcal{K}_{\text{DSP}} \in \mathbb{C}$ minimising the mean squared error between the transmitted $u_{h/v}(0,t)$ and the received $u_{h/v}(z,t)$ signals:

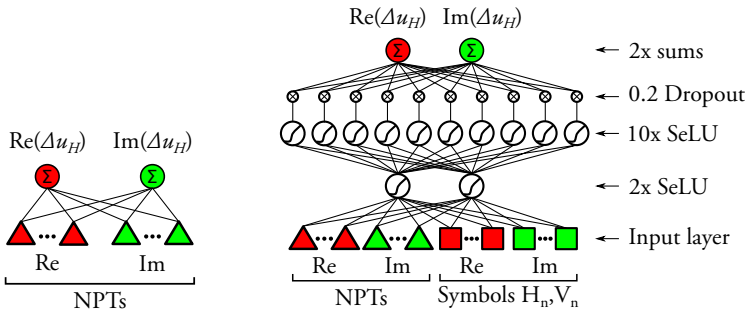
$$\mathcal{K}_{\text{DSP}} = \min_{\mathcal{K}} \left\| \mathcal{K} \cdot u_{h/v}(z,t) - u_{h/v}(0,t) \right\|, \tag{2.3}$$

where $\|\cdot\|$ is the Euclidean norm. To illustrate the robustness of data augmentation gains for the SL-NLC combined with other DSP algorithms, we included the normalization Eq. (2.3) in the numerical simulations, which are described in detail in Secs. 2.3 and 2.4. Furthermore, in Sec. 2.5 we show (see Figure 2.7) the considerable performance improvement resulting from augmentation in the experiment where the full receiver-based DSP was applied.

2.3 Setup of the numerical study

2.3.1 Considered SL-NLC algorithms

To demonstrate the generality of the proposed augmentation technique, we have considered several different transmission scenarios where we applied it to the two distinct supervised-learnt nonlinearity compensation (SL-NLC) algorithms presented in the literature. We want to concentrate on studying the effect of data augmentation, and, therefore, we have brought



(a) PPD scheme. (b) DNN scheme.

Fig. 2.2 Schemes of the published supervised-learnt nonlinearity compensation (SL-NLC) algorithms considered in this chapter: the perturbation-based post-distortion (PPD) and the deep neural network (DNN).

the architecture and hyper-parameters of the algorithms considered in the article as close as possible to their published versions.

The first is the perturbation-based post-distortion (PPD) algorithm [118, 128], the block diagram of which is depicted in Figure 2.2a. It is one of the most popular algorithms for Kerr nonlinearity compensation [111]. This algorithm is based on the assumption that the input of the channel $u_{h/v}(0,t)$ can be expressed through a sum of the received signal $u_{h/v}(z,t)$ with the weighted nonlinear perturbation triplets (NPTs) generated from it [129]:

$$u_h(0,t_0) = u_h(z,t_0) + \sum_{j,k} C_{j,k}(H_j H^*_{j+k} H_k + V_j V^*_{j+k} H_k), \tag{2.4}$$

where $H_j = u_h(z, t_0 + j\Delta t)$, $V_j = u_v(z, t_0 + j\Delta t)$ are, respectively, the elements of the received symbol sequences in horizontal (h) and vertical (v) polarizations $\bar{H}, \bar{V}$; Δt is the symbol period; $C_{j,k} \in \mathbb{C}$ are learnt coefficients; j and k are the symbol indices with respect to the symbol of interest (SOI) $u_h(z,t_0) = H_0$, $u_v(z,t_0) = V_0$. Since the coefficients $C_{j,k}$ are assumed to be independent of the received signal, they are learnt from the labeled training data via the optimisation procedure. The algorithm is trained to predict the nonlinear distortion in a single polarization: $\Delta u_{h/v} = u_{h/v}(z,t_0) - u_{h/v}(0,t_0)$.

The second algorithm is the deep neural network (DNN) proposed in [115, see Figure 4]. The scheme of this algorithm is given in Figure 2.2b. The DNN input consumes the received soft symbol sequences $\bar{H}$ and $\bar{V}$, each centered around the SOI, and the aforementioned NPTs are generated from these symbol sequences. The considered DNN is real-valued and, therefore, the real and imaginary parts of symbols and NPTs are separately fed to the DNN input. The DNN has two hidden layers with 2 and 10 neurons, respectively. They are followed by the output layer with two outputs, each one predicting the real or imaginary part

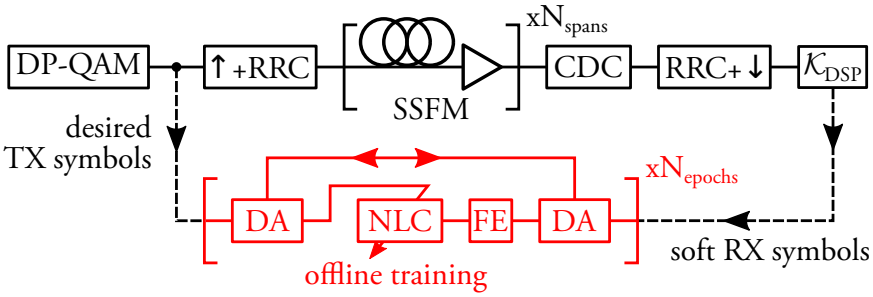


Fig. 2.3 Scheme of the numerically studied system implementing the offline training of an SL-NLC algorithm on the augmented dataset. DA stands for data augmentation and FE stands for feature extraction.

of nonlinear distortion of the evaluated signal polarisation $\Delta u_{h/v}$. The scaled exponential linear units (SeLU) [130] are used as the activation functions of the hidden neurons. Since we consider the regression task, there is no activation function on the output of the DNN. The second hidden layer is followed by a dropout layer [131], which omits randomly selected 20% of neurons during every training instance. Dropout is a regularization technique applied to prevent the neural network neurons from relying too much on specific set of input features and forces them to learn more robust features that generalize better to new data.

For PPD a training dataset object consists of the transmitted SOI in the studied polarization H_0, being the desired output of PPD, and a set of NPTs $(H_j H^*_{j+k} H_k + V_j V^*_{j+k} H_k)$ corresponding to SOI, which form the PPD input. For DNN the dataset object also includes another DNN input - a sequence of symbols transmitted in both polarizations H_j, V_j centered around the symbol of interest.

2.3.2 Numerical testcase

In the numerical study we consider an idealized case of the single-channel long-haul optical transmission link where channel Kerr nonlinearities, introduced via Manakov equations Eq. (2.1), are the only source of nonlinear distortions. Provided that, for long-haul links, fiber media response is, indeed, the leading source of nonlinear distortion [132, Sec 2.3], we assume this numerical model to be a reasonable approximation of this class of optical systems.

The scheme of the numerically considered communication system is given in Figure 2.3. To illustrate the effect of the application of the proposed data augmentation to the SL-NLC algorithms, we numerically simulated the transmission of a single-channel signal at 64 GBaud pre-shaped by a root-raised cosine (RRC) filter with 0.06 roll-off at 512 GSamples s^{-1}. We considered the following three testcases: (i) dual-polarisation (DP)-16QAM transmission over

a system consisting of 25×80 km large effective area fiber (LEAF) spans; (ii) DP-16QAM over 25×80 km standard single-mode fiber (SSMF) spans; (iii) DP-64QAM over 13×80 km SSMF spans. Optical signal evolution during fiber propagation was simulated by solving the Manakov equations Eq. (2.1) via split-step Fourier method (SSFM) [132]. In addition to SSMF, by far the most widely deployed fiber, we also decided to test our data augmentation in a different type of fiber with larger nonlinearities. We chose LEAF optical fiber as the one of the most widely deployed [133] non-zero dispersion shifted fibers in the world, i.e., the fibers having low chromatic dispersion in the studied wavelength region. The suppressed dispersive broadening leads to stronger Kerr nonlinearities in LEAF fibers. The considered parameters of LEAF fiber are: $\alpha = 0.225$ dB km^{-1} is the attenuation parameter, $D = 4.2$ ps nm^{-1} km^{-1} is the dispersion coefficient, and $\gamma = 1.3$ W^{-1} km^{-1} is the effective nonlinearity coefficient; while the SSMF ones are: $\alpha = 0.21$ dB km^{-1}, $D = 16.8$ ps nm^{-1} km^{-1}, and $\gamma = 1.14$ W^{-1} km^{-1}. Every span was followed by an optical amplifier (OA) with the noise figure NF = 4.5 dB, which fully compensated fiber losses and added the amplified spontaneous emission (ASE) noise. At the receiver, after full electronic chromatic dispersion compensation (CDC) by the frequency-domain equaliser and downsampling to the symbol rate, the received symbols were normalised by Eq. (2.3) to the transmitted ones. No other transceiver distortion was considered.

The algorithms were applied off-line to the pre-collected received and transmitted soft symbol sequences. Since both considered algorithms aim at predicting the nonlinear distortion $\Delta u_{h/v}$, the considered datasets consisted of: 1) the inputs formed by the sequences of the received symbols H_n, V_n centered around SOI H_0, V_0 and the NPTs generated from these sequences; 2) the nonlinear distortion $\Delta u_{h/v}$ induced into the SOI being the desired NLC response. To produce the training and testing datasets, two random symbol sequences were separately generated by a Mercenne twister pseudo-random number generator [134] and, later, propagated over the link. The size of the testing dataset was kept at 2^{17} objects in all simulations, while the size of training data varied. To correctly represent the memory of the nonlinear distortion, we considered symbol sequences of length $2 * 75 + 1$ centered around the SOI, i.e. $|j| \leq 75$. The symbol sequence length is taken from [120]. Indeed, we found that this length is enough to get the noticeable performance gain by nonlinearity compensation, which is further used to illustrate the data augmentation functioning. Following the conclusions of Ref. [120], we considered only the NPTs fulfilling the condition: $|k| \leq \min\{\lceil 75/|j| \rceil, 75\}$, where $\lceil \cdot \rceil$ stands for ceil function, i.e., rounding toward positive infinity. This procedure produced 301 complex-valued input symbols and 1929 NPTs generated from both polarizations.

Both PPD and DNN algorithms were trained using the same loss function and optimiser. The following mean-squared error (MSE) loss function was implemented:

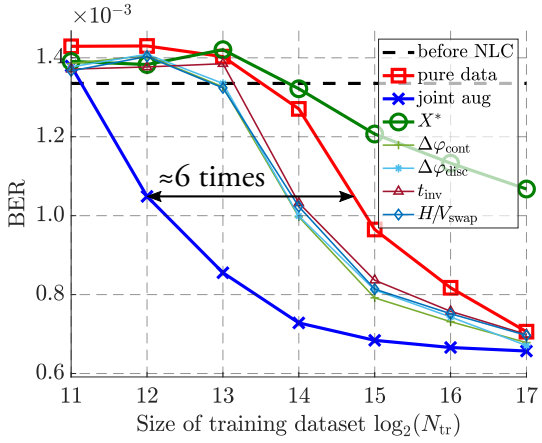
$$\mathrm{MSE} = \|H_{\mathrm{TX}} - (H_0 - \Delta u_h)\|, \tag{2.5}$$

where H_{TX}, V_{TX} are the SOIs transmitted in h and v polarizations, H_0, V_0 are the received SOIs and $\Delta u_{h/v}$ is the nonlinear distortion in a single polarisation predicted by the NLC algorithm. The MSE loss function given by (2.5) was optimised using the adaptive moments (Adam) optimiser [85] and the automatic differentiation was realised using the PyTorch package [135]. The training was carried out for 200 epochs with the batch size of 100. The training dataset was shuffled at the beginning of every epoch to avoid overfitting caused by learning the connections between the neighboring training pairs [136]. For every studied testcase, we used several learning rates and chose the one leading to the best bit-error rate on the testing data. The grid of the considered learning rates was a geometrical progression. For the PPD algorithm, the grid contained 10 values in the range $\eta_{\mathrm{PPD}} = \{10^{-3.5}, ..., 10^{-6.5}\}$; while for DNN, it contained 10 values in the range $\eta_{\mathrm{DNN}} = \{10^{-1}, ..., 10^{-7}\}$.

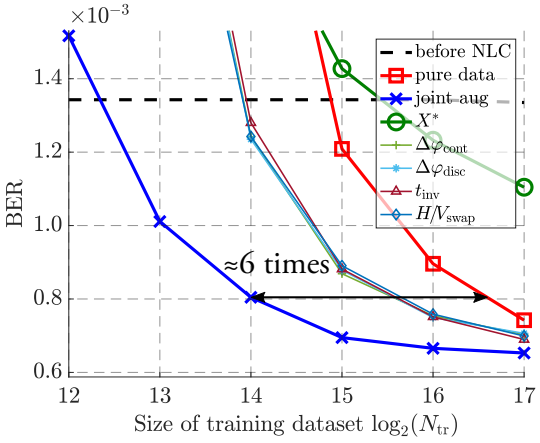
2.4 Numerical results

2.4.1 Performance improvement on deficient datasets

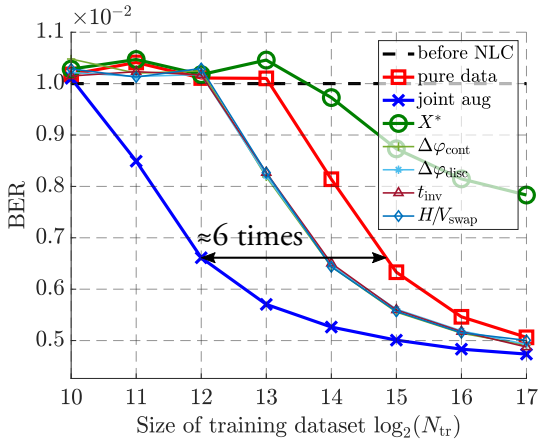
Firstly, we examined the impact of the data augmentation (DA) on the relation between the size of the training dataset N_{tr} and the bit-error rate (BER) achieved by the considered supervised-learnt nonlinearity compensation (SL-NLC) algorithms. With this objective, we compared the BER achieved by the perturbation-based pre-distortion (PPD) (Figure 2.2a) and the deep neural network (DNN) (Figure 2.2b) algorithms trained with datasets of different sizes N_{tr}. We define the size of the dataset N_{tr} as the number of objects in it. It is worth recalling from Section 2.2 that, since we implement DA via replacing a random part of the objects of the original dataset with the synthetic ones at the beginning of each epoch, both augmented and the original training datasets have the same number of objects N_{tr}. For every considered size N_{tr} and augmentation type, we separately optimised the power level on a grid with 1 dB step size and the learning rate on grids η_{PPD} or η_{DNN} to reach the best BER value. To remove local performance fluctuations, the BER was averaged over 20 consequent epochs before its minimal value was recorded. We compared BERs obtained on the collected dataset: (i) before SL-NLC; (ii) after SL-NLC trained with the collected non-augmented (pure) data; (iii) after SL-NLC trained with the data augmented by a single transformation from Table 2.1 ($\Delta\varphi_{\mathrm{cont}}$, $\Delta\varphi_{\mathrm{disc}}$, t_{inv}, H/V_{swap}); or (iv) by the combination of



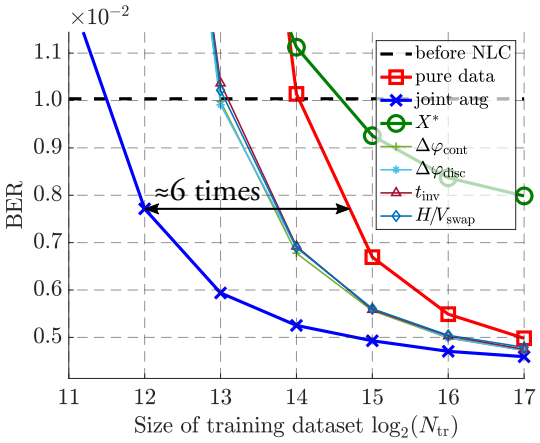
(a) DNN at SSMF 25×80 km DP-16QAM.



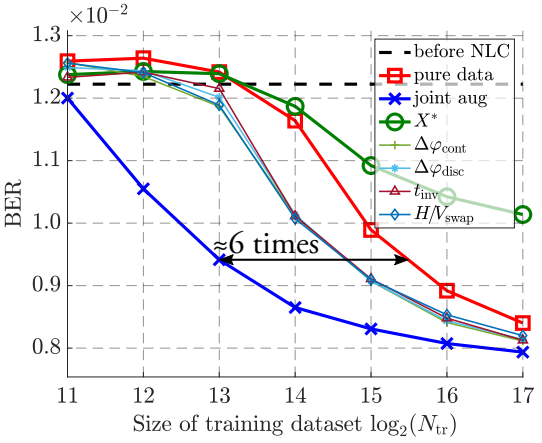
(b) PPD at SSMF 25×80 km DP-16QAM.



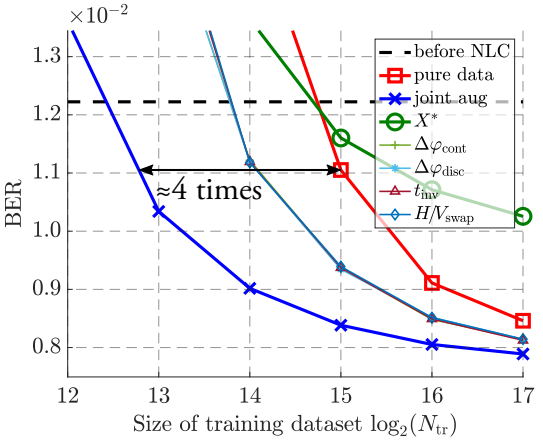
(c) DNN at LEAF 25×80 km DP-16QAM.



(d) PPD at LEAF 25×80 km DP-16QAM.



(e) DNN at SSMF 13×80 km DP-64QAM.



(f) PPD at SSMF 13×80 km DP-64QAM.

Fig. 2.4 Bit-error rate (BER) obtained by the perturbation-based post-distortion (PPD) and the deep neural network (DNN) SL-NLC algorithms trained with datasets of various sizes N_{tr}. The datasets are: non-augmented (pure data), augmented by the application of a signle transformation from Table 2.1 and Eq. (2.2), or with the joint augmentation (joint aug) combining the transformations $\Delta\varphi_{\text{disc}} + t_{\text{inv}} + H/V_{\text{swap}}$. The horizontal dashed line at each plane depicts the BER value before the application of NLC.

several transformations ($\Delta\varphi_{\text{disc}} + t_{\text{inv}} + H/V_{\text{swap}}$). We chose to omit the continuous phase shift $\Delta\varphi_{\text{cont}}$ out of joint augmentation because it works similarly to the discrete one $\Delta\varphi_{\text{disc}}$ and provided the same BER improvement but at the cost of higher complexity.

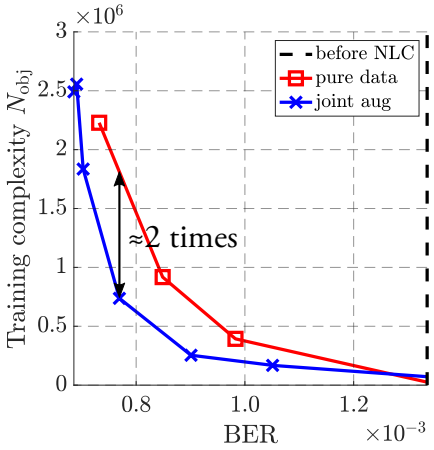
The dependence of the achieved BERs on the size of training dataset N_{tr}, is presented in Figure 2.4. The solid dashed black line in Figure 2.4 is the BER measured before SL-NLC. In all studied testcases, the application of a single transformation-based data augmentation ($\Delta\varphi_{\text{cont}}$, $\Delta\varphi_{\text{disc}}$, t_{inv}, H/V_{swap}) led to same BER as when using $2\times$ larger pure training dataset, while the augmentation based on the combination of three transformations ($\Delta\varphi_{\text{disc}} + t_{\text{inv}} + H/V_{\text{swap}}$; marked as "joint aug" in the Figure 2.4) enabled achieving the same BER with an approximately $4\times$ to $6\times$ smaller dataset. Figure 2.4 also shows that the BER improvement obtained via the data augmentation decreases, as expected, when increasing the size of the training dataset, and it becomes negligible when the dataset is very large ($\gtrsim 10^5$ objects). This saturation effect supports our claim that the data augmentation leads to performance gains mainly by adding variability to the training dataset and, therefore, can be applicable to a huge range of SL-NLC algorithms. The data augmentation based on complex conjugation X^* Eq. (2.2) (wide green line in Figure 2.4) led to the smallest performance increase among considered testcases, thus supporting the claim that only the transformations generating the true new solutions of channel model Eq. (2.1) are effective for data augmentation.

2.4.2 Reduction of training cost on full datasets

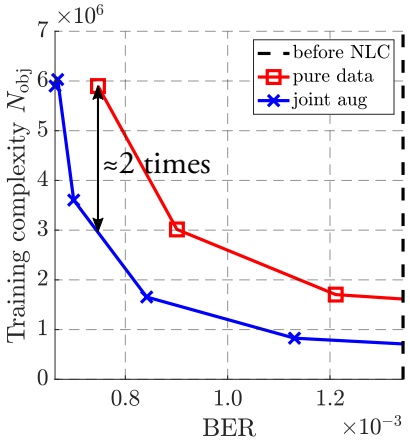
In this section we analyze the potential of data augmentation to reduce the numerical cost of training an SL-NLC algorithm. We start by assuming that our dataset is large enough to reach the desired SL-NLC performance level without augmentation. Since the training complexity is proportional to the size of the dataset N_{tr}, it is interesting to decrease the cost of SL-NLC training by performing it on a smaller dataset. The results reported in Figure 2.4 already indicate that, if data augmentation is used, the same BER value can be reached using a dataset several times smaller than the original one.

When training an SL algorithm we repeatedly optimize it for each element of the dataset using the same routine [4]. Therefore, the numerical cost of training the same algorithm on different datasets is directly proportional to the number of the dataset objects N_{obj}. Typically, during training the algorithm passes several times over the whole dataset, with every pass referred to as an epoch. Hence, the numerical cost of the training can be estimated as

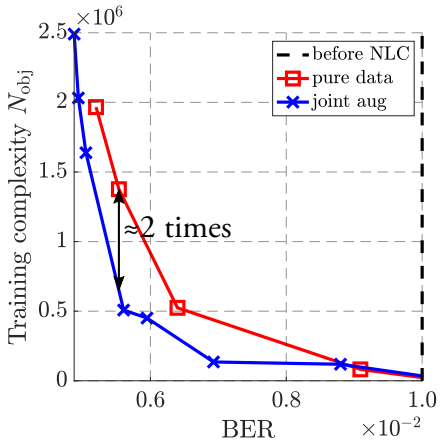
$$\text{cost} \approx N_{\text{obj}} = \text{number of epochs} \times N_{\text{tr}}, \tag{2.6}$$



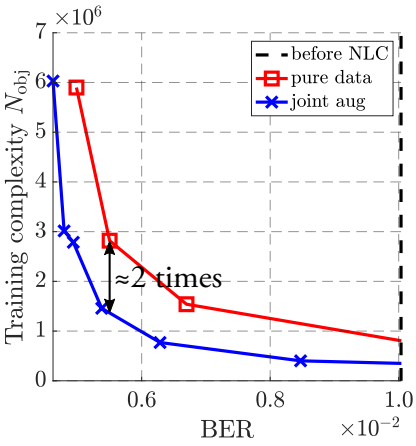
(a) DNN at SSMF 25×80 km DP-16QAM.



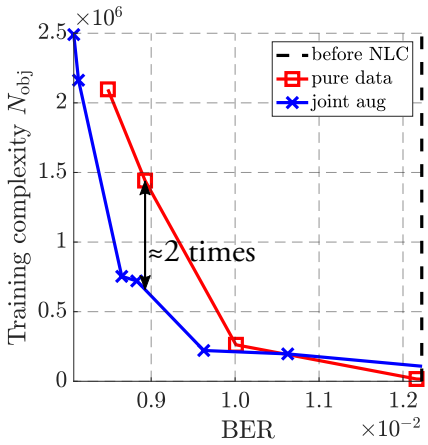
(b) PPD at SSMF 25×80 km DP-16QAM.



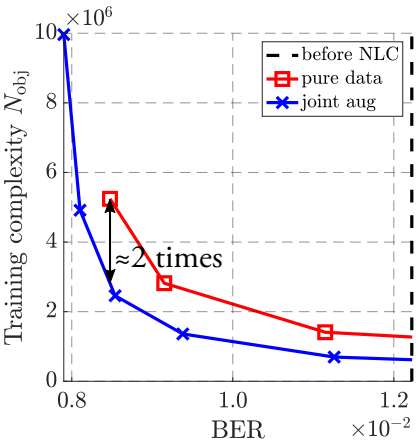
(c) DNN at LEAF 25×80 km DP-16QAM.



(d) PPD at LEAF 25×80 km DP-16QAM.



(e) DNN at SSMF 13×80 km DP-64QAM.



(f) PPD at SSMF 13×80 km DP-64QAM.

Fig. 2.5 Dependence of the training complexity estimation Eq. (2.6) on the achieved BER for the considered SL-NLC algorithms: the perturbation-based post-distortion (PPD) and the deep neural network (DNN). The complexity is compared for the NLC algorithms trained with the datasets: non-augmented (pure) or augmented by the triple of transformations listed in Table 2.1 $\Delta\varphi_{\text{disc}} + t_{\text{inv}} + H/V_{\text{swap}}$ (joint aug). The vertical dashed line at the right border each plane depicts the BER value before the application of NLC.

where N_{tr} is the size of the training dataset fully processed each epoch. Since the training procedure does not depend on the dataset content, we used Eq. (2.6) to compare the cost of training the SL-NLC algorithm on both the augmented and pure data. We assumed that the complexity of augmenting the dataset by the simple analytical operations listed in Table 2.1 is negligible compared to the training costs.

The numerical complexity of applying augmentation is much smaller than one of the model optimisation and feature extraction for the considered algorithms. Numerical cost can be expressed as a number of required multiplications, since the complexity of all the other operations is much smaller [32, 137]. Polarisation H/V_{inv} and time t_{inv} inversion are done by a mere re-indexing of the input elements not requiring any multiplications. Discrete phase shift $\Delta\varphi_{\text{disc}}$ by angles proportional to $\pi/2$ can be done multiplier-free by swapping and changing signs of the real and imaginary parts of the processed complex variable. Finally, the continuous phase shift $\Delta\varphi_{\text{disc}}$ can be implemented in hardware at low numerical cost by CORDIC algorithm [138]. On the contrary, even the preparation of qubic nonlinear perturbation terms (NPTs) used in the input of studied DNN and PPD algorithms requires 4 complex multiplications per NPT Eq. (2.4), which leads to $1929 \times 4 = 7116$ multiplications.

Figure 2.5 shows the estimated numerical cost – following Eq. (2.6) – required to achieve the desired BER level by the PPD and DNN SL-NLC algorithms when trained on the pure data (red line) and on the augmented one (blue line). Here we considered only the joint augmentation: $\Delta\varphi_{\text{disc}} + t_{\text{inv}} + H/V_{\text{swap}}$. For every N_{tr}, we optimised the power level and the learning rate (as in Figure 2.4). However, we noticed that the number of epochs required to reach the minimum BER at the optimal learning rate was $2-3\times$ higher than for that for the slightly bigger learning rate. Thus, in this subsection, we estimated the cost according to Eq. (2.6) with a slightly bigger learning rate than the optimal value. The complexity reduction brought by the introduction of a sub-optimal learning rate greatly outweighted the slight reduction in BERs. As mentioned before, this choice provided nearly $2-3\times$ drop in the required number of training epochs at the cost of no more than 5% BER increase. We define the number of epochs in Eq. (2.6) as the one with which we reach the optimal BER with 3% accuracy.

Figure 2.5 shows that, in all studied testcases, the dataset reduction enabled by joint augmentation allows the PPD algorithm to reach the target BER at nearly $2\times$ lower training complexity. For the DNN, the proposed dataset reduction technique also reduced the required training complexity by $\sim 2\times$ for the BERs situated in the region of interest (near the smallest achieved BER).

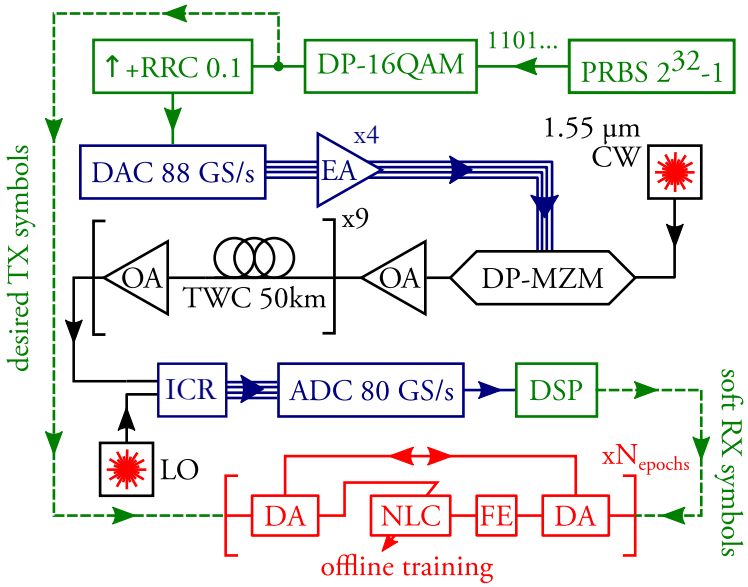


Fig. 2.6 Scheme of the experimental setup. Green solid lines: digital electrical signals; blue lines: analogous electrical signals; black lines: optical signals; green dashed lines: data collected for the offline processing by the DNN (Figure 2.2b). The acronyms used are introduced in Section 2.5.

2.5 Experimental study

In Section 2.4, we numerically illustrated the effect of data augmentation on the supervised-learnt nonlinearity compensation (SL-NLC) algorithms acting on the pure Kerr nonlinear channel, governed by the Manakov equations described by Eq. (2.1). Nonetheless, in practical fiber-optic communication systems, not only Kerr nonlinearities but also transceiver impairments play a significant role in generating signal distortion [139]. Therefore, to evaluate the applicability of data augmentation, in this section we demonstrate how the transceiver impairments may affect augmentation performance gains. With that objective, we considered an experimental model of a metro communication link where both Kerr nonlinearity and the transceiver impairments make a noticeable contribution to the signal distortion. For this reason, the link, considered in the experiment, significantly differs from the link, considered numerically in the previous Section 2.4. We found that the transformations from Table 2.1, except for time-inversion t_{inv}, can be effectively used for data augmentation even in the presence of considerable components distortion.

2.5.1 Experimental setup

Figure 2.6 illustrates the experimental setup. We considered a single-channel transmission. At the transmitter (TX), the dual-polarisation (DP)-16QAM 34.4 GBd symbol sequence was mapped out of data bits generated by a $2^{32}-1$ order pseudo random binary sequence (PRBS), which has a periodicity of 2^{32} bit or $2^{30} \approx 10^9$ 4-bit 16QAM symbols. The length of the considered dataset ($\leq 2^{17} \sim 10^5$) was much less than the periodicity of the used PRBS. Therefore, the PRBS-generated dataset didn't repeat itself, and so, SL-NLC algorithms weren't been able to memorize the repeating sequence, which could led to inadequate performance estimations, as [136] warns. On the other hand, the considered PRBS sequence was long enough to neglect the effects caused by possible imbalance between "0" and "1" bits representation in the sequence. Digital root-raised-cosine (RRC) filter with roll-off 0.1 was applied to the symbol sequence to limit the channel bandwidth to 37.5 GHz. The filtered digital samples were uploaded to a digital-to-analog converter (DAC) operating at 88 Gsamples s^{-1}. The analog electrical outputs of DAC were amplified by an electrical amplifier (EA) and drove a dual-polarization in-phase/quadrature (IQ) Mach–Zehnder modulator (DP-MZM), modulating the continuous waveform (CW) carrier produced by an external cavity laser (ECL) at $\lambda = 1.55$ μm. We transmitted the signal at different power levels in order to be able later to choose the optimal one. The investigated power levels formed a grid with 1 dBm stepsize.

The optical path consisted of 9×50 km spans of TrueWave Classic (TWC) optical fiber. Each span was followed by an EDFA-type optical amplifier (OA), compensating for the losses on the fiber span. The parameters of the used TWC fiber spans at $\lambda = 1.55$ μm are: $\alpha = 0.23$ dB km^{-1} attenuation coefficient, $D = 2.8$ ps nm^{-1} km^{-1} dispersion coefficient, and $\gamma = 2.1$ W^{-1} km^{-1} effective nonlinear coefficient. The OA noise figure was in the 4.5 to 5 dB range.

At the receiver (RX), the optical signal was firstly detected using an integrated coherent receiver (ICR). The resulting electrical signal was sampled at 80 Gsamples s^{-1} by an analog-to-digital converter (ADC), and processed using a linear receiver-based digital signal processing (DSP). In the DSP, the bulk accumulated chromatic dispersion was firstly compensated by a frequency domain equalizer (FDE), which was followed by the removal of the carrier frequency offset. Next, the constant-amplitude zero-autocorrelation-based training sequence was located in the received frames. This training sequence was then extracted and used to estimate the equalizer transfer function. After the equalization the following algorithms were applied: polarization demultiplexing, time correction, and pilot-aided carrier phase recovery. Finally, the soft symbols at the output of the DSP were collected for an offline processing by the studied NLC algorithm.

The feature extractor (FE) was then employed to prepare the sequences of input symbols $\bar{H}, \bar{V}$ and nonlinear perturbation terms (NPTs) Eq. (2.4) fed into the SL-NLC algorithms as input. The transmitted soft symbols were used as the desired response during the training of the SL-NLC algorithm. For every studied power level, the two separately generated random symbol sequences were transmitted through the system and used, respectively, to prepare the training and testing datasets. The testing datasets considered in the experiment had 2^{17} objects.

2.5.2 Experimental results

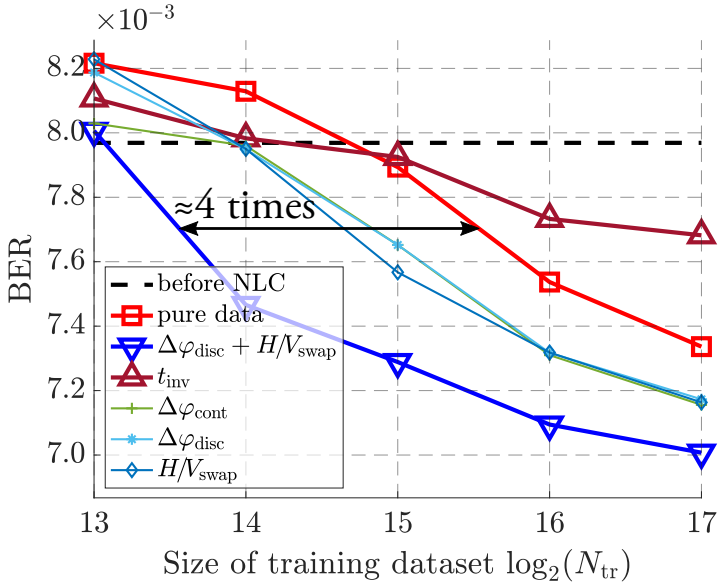


Fig. 2.7 BER before and after the DNN (Figure 2.2b) SL-NLC algorithm trained with the experimentally measured datasets of various sizes N_{tr} augmented in different ways.

Similar to the numerical study in Sec. 2.4, we considered the perturbation-based post-distortion (PPD) and the deep neural network (DNN) SL-NLC algorithms of Sec. 2.3. Nonetheless, we found that the PPD led to only marginal performance improvement in terms of BER for the considered testcase, which demonstrates that it is not designed to operate in the considered metro systems with considerable transceiver-induced distortions [139, 140]. Specifically, PPD relies on the overall nonlinear distortion being representable as the linear combination of qubic triplets, which is correct only in the case when the optical channel is the dominant source of nonlinearity. Therefore, we limit our further discussions to the effect of data augmentation on the DNN training.

We analyzed how does the training dataset size N_{tr} and the application of different augmentation types affect the BER obtained by the DNN. The training procedure and the dataset structure were the same as the ones described in Sec. 2.3 and 2.4. Similarly to the numerical study, for every studied dataset size N_{tr} and augmentation type we recorded the best BER obtained at the optimal learning rate and signal power level.

Figure 2.7 shows that the data augmentation based on one of the following transformations from Table 2.1: discrete phase shift $\Delta\varphi_{\text{disc}}$, continuous phase shift $\Delta\varphi_{\text{cont}}$, and polarisation swapping H/V_{swap} – enables the DNN to obtain the same BER using a dataset with nearly half the size compared to the training with the non-augmented (pure) dataset. Moreover, just like the numerical results of Sec. 2.4, the data augmentation by a joint application of the several transformations $\Delta\varphi_{\text{disc}} + H/V_{\text{swap}}$ performed better than any single-transformation-based augmentation and allowed $\approx$4$\times$ reduction of the dataset size while keeping the same BER. Unlike the numerical simulations (Figure 2.4), the time-inversion-based augmentation t_{inv} led to an increase of the BER after DNN with respect to training it with the pure dataset. We associate that with the asymmetrical memory introduced by the transceiver impairments violating the time-inversion symmetry, which was not considered within the numerical study. Thus, we conclude that some augmentation methods can be affected by the specific transceiver properties. Nevertheless, the overall performance improvement and complexity reduction demonstrate the viability of the augmentation strategy for the realistic applications.

2.6 Summary

We proposed the data augmentation technique for improving the training of supervised-learnt algorithms for the compensation of nonlinear distortion (SL-NLC) in fiber-optic communication systems. The technique is based on the generation of new training data objects out of the pre-measured ones using special transformations such as the ones given in Table 2.1. We showed the validity of our approach by using both numerical and experimental data. In the numerical study, we focused on systems where the Kerr nonlinearity was the dominant distortion. For these testcases, the data augmentation enabled the NLC to achieve comparable performance when using up to 6$\times$ less data with respect to the conventional training, see Figure 2.4. In the case of having a large enough dataset, we showed that the data augmentation enables reducing the size of the dataset leading to the equivalent SL-NLC performance at $\sim$ 2$\times$ lower training complexity, see Figure 2.5. The experimental study was aimed at the illustration of data augmentation applicability for the practical case of the links distorted not only by Kerr nonlinearity, but also by the transceiver-induced distortions. In this case, the data augmentation enabled reducing the training dataset by a factor of $\approx$4$\times$ without

reducing the NLC performance, see Figure 2.7. We anticipate that the proposed generic approach can find applications in a number of problems where underlying propagation equation symmetries can be used for augmentation of the relevant datasets used for signal processing.

2.6.1 Contribution statement

I have developed the method proposed in this article, obtained the presented numerical results, wrote its text, and prepared all the illustrations by myself. The computer code used in the presented research was written by me and Pedro Freire. The experimentally measured traces were kindly provided by Antonio Napoli, Bernhard Spinnler, and Wolfgang Schairer. I have applied nonlinear compensation and data augmentation to the experimental data by myself. The research presented here was done under supervision of Jaroslaw E. Prilepsky, Antonio Napoli, and Sergei K. Turitsyn.

Chapter 3

End-to-end learning in the coherent fiber-optic communications

3.1 Introduction

When dealing with optical communication systems, there is still the lack of the general communication theory describing nonlinear fiber channels (where the dispersive effects intertwine with the fiber nonlinearity), in contrast to the classical additive white Gaussian noise (AWGN) communication channel. Thus, many fundamental and practically important questions related to the optimal coding, modulation, pulse-shaping, and channel equalization for the nonlinear optical transmission systems, remain either partially solved or even yet to be answered [23, 141]. In particular, the optimal signal statistics (the Gaussian distribution for AWGN [88]) is not known for the transmission affected by the simultaneous action of fiber dispersion and nonlinearity. Addressing this problem numerically, in general, is not practically feasible due to the high computational cost of modelling high-speed transmission via dispersive nonlinear channels. Thus, there remains the research challenge related to developing practical transmitters and receivers with signal format and modulation inherently adjusted to nonlinear transmission. We note that currently-used transceivers are suboptimal, which bring about a cap on the achievable data rates and transmission distances. Recently, machine learning (ML) methods and, in particular, artificial neural networks (NNs) have been applied to the design of optical communication systems, see, e.g., recent Refs. [14, 15, 113, 142–145] and numerous literature sources therein. Albeit it is rather difficult (if possible at all) to obtain the general conclusions on the optimal signal shaping and modulation in the realistic nonlinear fiber systems, it is possible to obtain some sub-optimal results by using specific ML techniques tailored to deal with complex nonlinear problems.

It is important to point out that the properties of optical fiber-channels strongly depend on signal parameters: the power is linked to the nonlinear effects, and the carrier pulse width defines the dispersive signal broadening and respective memory effects. Then, the transmission distance defines how essential the noise-induced corruptions are, the strength of nonlinear signal distortions, and the dispersive effects. Therefore, different nonlinear fibre channels can have rather distinct optimal signal modulations and coding.

The end-to-end (E2E) learning approach, proposed in [66], is a machine learning technique offering a way to automatically tailor signal modulation and coding for an arbitrary communication link. The basic idea of E2E learning lies in representing the link from messages-in to messages-out as a single NN, differentiable with respect to the link parameters, to simultaneously optimize these parameters by using, e.g., the efficient gradient-descent-based optimization. Many applications of the E2E learning in different communication links then followed [146, 147].

The first applications of E2E learning to optical communication systems were done for intensity modulation / direct detection (IM/DD) systems. In Refs. [114, 148–150], the E2E learning of geometric constellation shaping (GS), i.e., optimal symbol locations, for IM/DD optical communication systems was proposed. further, in [151] the E2E learning of waveforms was considered for the specific case of a link based on nonlinear frequency division multiplexing.

The following works are dealt with the E2E learning of the optimal constellation points locations, i.e. on the GS, and the pre-distortion techniques in coherent systems. In [152–156], the E2E learning of single-symbol GS was considered for a coherent communication system. While [152–155] considered the distortions generated only by the nonlinear channel, in Ref. [156] a more realistic link model that included the laser noise, was studied. In Refs. [60, 64, 65, 147, 157–159], the E2E learning of GS, signal waveform, and nonlinear pre-distortion resistant to transmitter distortions was considered. However, the true complexity of nonlinear fiber-optic channel distortions was neglected in these works, namely, the distortions were either neglected completely, or modelled via a simplified Gaussian noise model. Conversely, in [160] the authors considered the joint E2E learning of GS and linear pre-distorter mitigating the fiber channel distortion. Nonetheless, the pre-distorter learnt in that work was rather trivial: it combined the Nyquist pulse shaper and chromatic dispersion compensator, and, hence, it actually did not contribute to the nonlinearity mitigation.

In this chapter, we propose a machine learning algorithm for E2E learning of the constellation shaping that takes into account the nonlinearities and memory present in optical channel distortions. With this algorithm, we jointly optimized the symbol locations in the constellation diagram, the symbol probabilities, and the nonlinear pre-distortion. The learnt

transmitted signal distribution chooses the transmitted symbol based not only on the message sent in the corresponding time slot, as in the conventional constellation shaping, but also on the messages sent in the neighbouring time slots. Therefore, we refer to the resulting signal distribution as the multi-symbol constellation shaping (MSCS).

In the proposed E2E learning algorithm, we implement two key new features. The first one is the utilization of the auxiliary channel model based on perturbation theory [2], which allows us to reduce the computational efforts/resources needed to model the complex mixture of nonlinear and linear distortions taking place in fiber-optic links. Second, we implemented the procedure for the simultaneous learning of symbol probabilities and locations, proposed first in Ref. [161] for the AWGN channel.

We applied the proposed algorithm to a single-channel 64 GBd transmission over a pair of completely different state-of-the-art links to indicate that the proposed method is rather generic and is applicable to improving the quality of transmission in an arbitrary coherent fiber-optic communication link.

First, we considered the single-span 64 GBd transmission of a 256-symbol constellation over 170 km standard single mode fiber (SMF) link, where the expected gains by nonlinear shaping are particularly high [162]. In this test, we compared the learnt constellations with the conventional Maxwell-Boltzmann (MB) probabilisitc shaping, where the latter is optimal for the AWGN channel [163]. The learnt multi-symbol constellation shaping led to the bit-wise mutual information (BMI) gain of 0.48 bits/2D-symbol over the conventional MB shaping. Furthermore, we show that the proposed E2E learning is applicable for the cases when, because of hardware- or complexity-related limitations, we cannot use multi-symbol constellation shaping. For the same test case, we learnt a single-symbol joint probabilistic and geometric shaping showing 0.074 bits/2D-symbol BMI gain over the reference MB shaping. On top of it, for the case when geometric shaping is not an option, we learnt the single-symbol probabilistic shaping which outperforms the MB shaping by 0.043 bits/2D-symbol in terms of BMI.

Second, we successfully applied the E2E learning technique to a 64 GBd transmission of 64-symbol signal over long-haul 30x80 km (2400 km) SMF link. For this testcase, the E2E-learnt multi-symbol constellation shaping led to symbol-wise mutual information (MI) gain of ≈ 0.20 bits/2D-symbol over the reference constellation. Similarly to the aforementioned single-span case, the E2E learning managed to learn the effective single-symbol constellation shaping resulting in reasonable performance gain even for the case when complexity limitations prevent us from using the multi-symbol constellation shaping. Particularly, the E2E-learnt single-symbol constellation shaping resulted in MI gain of ≈ 0.14 bits/2D-symbol over the reference one.

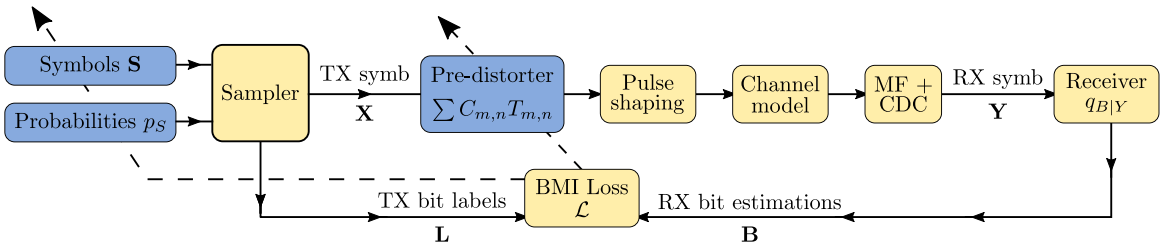


Fig. 3.1 Principal scheme of the end-to-end learning algorithm implemented in this paper. Blue denotes trainable blocks, dashed lines denote feedback from loss function. MF stands for matched filtering, CDC stands for chromatic dispersion compensation.

This Chapter is based on the original published contributions [1, 3, 164]. Also, original unpublished material is included.

The remainder of the chapter is organized as follows. Section 3.2 describes in the proposed E2E learning algorithm. Section 3.3 describes the considered testcase and the results achieved by the proposed E2E learning. Section 3.4 concludes the chapter.

3.2 Models and the algorithm description

3.2.1 End-to-end learning

In the end-to-end learning approach, the whole system from bits-in to bits-out, including transmitter, channel, receiver, is implemented as a single NN, thus enabling the joint training of transmitter and receiver. The idea of the approach was first suggested in Ref. [66]. The scheme of end-to-end platform used in this paper is given in Fig. 3.1.

3.2.2 Transmitter design, constellation shaping, and pre-distortion

We consider the two-stage transmitter, consisting of a sampler followed by a nonlinear pre-distorter. Note that the transmitter's separability into autonomous stages, with each stage having its well-defined purpose, improves its interpretability and cost-efficiency [147].

In the sampler, the input data is mapped to some complex-valued constellation points $s_m \in \mathbf{S} \in \mathbb{C}, |\mathbf{S}| = M$ and then transmitted over a channel. A fixed bit label $\mathbf{l}_m = [l_{m,1}, \ldots, l_{m,K}]$, $l_{m,k} \in \{0,1\}$, $K = \log_2(M)$ is set for each symbol in the alphabet $\mathbf{l}_m \Longleftrightarrow s_m, \forall s_m \in \mathbf{S}$. The input data is generated in a way that symbols s_m are sampled according to the discrete probabilistic distribution $p_S(s_m) \forall s_m \in \mathbf{S}$. We assume here that for an arbitrary distribution p_S there exists the mapper which maps the original input data stream in such a manner that the resulting symbols are distributed according to p_S. Because of the imperfections of the

communication channel, the information rate of the system depends on the locations of constellation points **S** and their probability distribution p_S. Therefore, it is possible to maximize the system information rate by optimizing symbol locations **S** and occurrence probabilities p_S. Optimizing **S** is called a geometric constellation shaping (GS), and optimizing p_S is called a probabilistic constellation shaping (PS) [92]. Notably, for linear channels, the family of Maxwell-Boltzmann (MB) distributions leads to the optimal PS [163]

$$p_S(s_m) \approx \exp\left(-\theta|s_m|^2\right), \tag{3.1}$$

where θ is a distribution hyper-parameter referred to as the MB shaping parameter. Nonetheless, finding the optimal shaping for nonlinear-dispersive optical fiber channels is a subtle problem [162].

In this work, we consider a separate optimization of PS, GS, along with the joint optimization of both **S** and p_S; the latter method is referred to as the joint shaping (JS). The sampler generated the dual-polarized signal $X = \{X_h, X_v\}$ with both polarization components X_h, X_v sampled from the same alphabet $x_{i,h/v} \in \mathbf{S}$ according to the same distribution p_S.

The sampler was followed by a trainable nonlinear pre-distorter. The goal of the latter is to pre-compensate the nonlinear channel distortions through pre-processing the symbol sequence $X_{h/v}$ generated by the sampler, before the sequence is sent into the optical channel. The pre-compensation is made in such a way that the symbol sequence at the channel output, Y, approximates the sampled input symbol sequence. We consider a perturbation-based pre-distortion (PPD) [129], which adds to each transmitted symbol $x_{i,h/v}$ an additive correction $\Delta_{\text{PPD}}x_{i,h/v}$, depending on the characters transmitted in the neighboring slots in both polarizations $x_{i,h}, x_{i,v}$. The pre-distortion Δ_{PPD} is defined as a linear combination of cubic polynomials $T_{m,n}$ calculated from the symbols co-propagated with the pre-processed symbol at the neighboring time slots. In more detail, for any symbol $x_{i,h/v}$ transmitted at the i-th time slot in the H- or V-polarization, the pre-distortion takes the form:

$$\Delta_{\text{PPD}}(x_{i,h/v}) = \sum_{m,n} C_{m,n} \cdot T_{m,n};\ T_{n,m} = x_{i+n,h/v} \cdot \left(x^*_{i+n+m,h/v} x_{i+m,h/v} + x^*_{i+n+m,v/h} x_{i+m,v/h}\right), \tag{3.2}$$

where $x_{i+m,h/v}$ is the symbol in H-/V-polarization shifted by m time slots from the target symbol $x_{i,h/v}$, and $C_{m,n}$ are the trainable weights.

The performance-to-complexity ratio of PPD is determined by the range of polynomials $T_{m,n}$ taken into consideration in a particular algorithm. Since the PPD is a linear regression over $T_{m,n}$ terms, the importance of each term in the trained PPD can be assessed by the absolute value of the coefficient $|C_{m,n}|$ corresponding to it. Hence, one can reach a cost-

effective PPD by training an excessively complex one and then pruning its coefficients, i.e. zeroing ones with the absolute value smaller than the chosen cut-off value $C_{m,n} := 0$, for $|C_{m,n}| < C_{\text{cutoff}}$. This makes the cut-off value C_{cutoff} an important hyper-parameter of PPD learning, defining the performance-to-complexity ratio of the resulting algorithm. We consider its optimization as part of E2E learning in Sec. 3.3.1.4.

3.2.3 RP-based channel model

The second autoencoder block, the *auxiliary channel model*, maps the signal generated by the transmitter X to a sequence of symbols Y collected by the receiver. The model includes both deterministic and stochastic distortions, expressed via the probability of Y given X, $p_{Y|X}$. The approximate channel model must be computationally simple and easy to differentiate, to allow for fast transmitter learning; but, simultaneously, it has to describe the distortion introduced by the channel accurately enough, so that the learned transmitter and receiver could emulate well a real-world communication link.

The nonlinear dispersive channel is typically modelled by the Manakov equations, see Eq. (3.3) below, which are simulated by a serial cascade of alternating convolutional and pointwise nonlinear operators; the solution scheme is referred to as the split-step Fourier method (SSFM) [165]. The SSFM can be represented by the convolutional NN consisting of many layers [35, 160], and the complexity of such a convolutional NN makes the calculation of gradients of the model outputs over its inputs (in the back-propagation learning [9]) very slow and rather challenging. First, the learning process implies that all the intermediate states are stored, and it makes the process memory hungry. Second, the back-propagation through many layers often results in numerical errors, leading to the infamous uncontrolled growth or vanishing of gradients [61, 144]. One way to bypass these problems is to consider a channel approximation with simplified models. For instance, for this purpose the following approaches were proposed: the E2E learning using a dispersion-free nonlinear channel model [166]; a nonlinear interference noise (NLIN) model [153, 164, 167], which considers nonlinear distortion as a costellation-dependent additive Gaussian noise [105, 106]; or neglecting the optical channel nonlinearity entirely by modelling its distortion as an additive white Gaussian noise [147]. Nonetheless, these models neglect the channel memory. On top of it, NLIN and AWGN-based channel models erase the information about the determinism of nonlinear distortions, replacing it with stochastic noise. These two factors prevent any memory-aware constellation shaping or pre-distorters from learning about the inter-symbol behavior of nonlinear distortion.

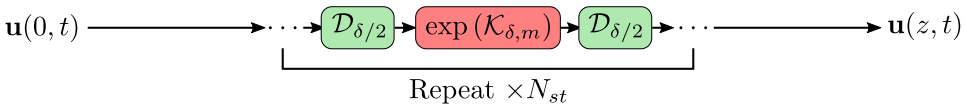


Fig. 3.2 Principal scheme of the split-step Fourier method, Eq. (3.5). The scheme is given to illustrate the derivation of RP model, Eq. (3.4). This figure is licensed under the CC BY 4.0 license, taken from [1] created by V. Neskorniuk, et al.

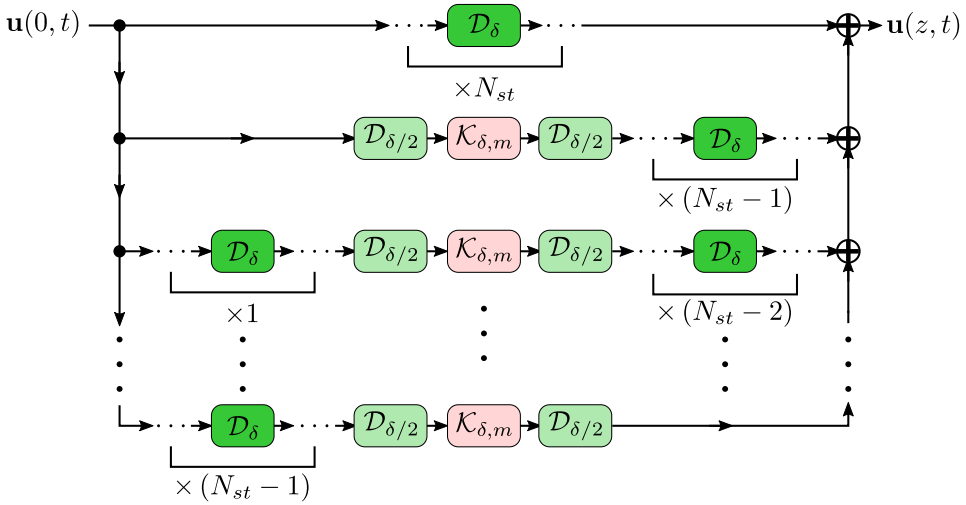


Fig. 3.3 Principal scheme of the first-order regular perturbation (RP) model [2] introduced in Eq. (3.4). This figure is taken from [1].

Following our previous work [3], in this paper, we propose the application of the first-order regular perturbation model (RP-model) as an auxiliary channel model. In this section, we describe the model and the benefits of its application in more detail.

Consider the Manakov equations describing the evolution of the waveform of a dual-polarized optical signal $\mathbf{E}(z,t) = \mathbf{u}(z,t)\sqrt{f(z)}$ during its propagation over a fiber-optic link with lumped optical amplifiers (OAs) [168]:

$$\frac{\partial \mathbf{u}}{\partial z} = -i\frac{\beta_2}{2}\frac{\partial^2 \mathbf{u}}{\partial t^2} + i\frac{8}{9}\gamma f(z)\|\mathbf{u}\|^2\mathbf{u} + \eta(z,t), \tag{3.3}$$

where $f(z) = \exp\left(-\alpha z + \alpha L_{\text{sp}}\lfloor z/L_{\text{sp}}\rfloor\right)$ models the optical losses (with α being the attenuation coefficient) and amplification, L_{sp} denotes the fiber span length, β_2 and γ are the chromatic dispersion and Kerr nonlinearity coefficients; $\eta(z,t)$ denotes the amplified spontaneous emission noise (ASE) injected by OAs.

The first-order regular perturbation (RP) [2, 168, 169] is an elaborate method to approximate $\mathbf{u}(z,t)$ in a weakly nonlinear regime. The principal scheme of RP model is given in Figure 3.3. The channel output $\mathbf{u}(z,t)$ is approximated using the perturbations according to

expressions:

$$\begin{gathered}\mathbf{u}(z,t) = \mathbf{u}_{\mathrm{L}}(z,t) + \mathbf{u}_{\mathrm{NL}}(z,t) + O(\gamma^2),\\ \mathbf{u}_{\mathrm{L}}(z,t) = \mathcal{D}_z\left[\mathbf{u}(0,t) + \eta(z,t)\right],\\ \mathbf{u}_{\mathrm{NL}}(z,t) = \sum_{m=1}^{N_{\mathrm{br}}-1} \mathcal{D}_{z-(m-0.5)\delta}\left[\mathcal{K}_{\delta,m}[\mathbf{u}_{\mathrm{L}}((m-0.5)\delta,t)]\right],\end{gathered} \tag{3.4}$$

with

$$\begin{gathered}\mathcal{D}_z[\cdot] = \mathcal{F}^{-1}\left[\exp(i\beta_2 z\omega^2/2)\mathcal{F}[\cdot]\right],\\ \mathcal{K}_{\delta,m}[\mathbf{u}(t)] = i\frac{8}{9}\gamma\frac{1-e^{-\alpha\delta}}{\alpha} f\left((m-\frac{1}{2})\delta\right)\|\mathbf{u}(t)\|^2\mathbf{u}(t).\end{gathered}$$

Here $\delta = z/(N_{br}-1)$ is the algorithm's spatial step size, $\|\cdot\|$ is the Euclidean vector norm, and $\mathcal{F}$ denotes the Fourier transform. $\mathcal{D}_z[\cdot]$ is the operator introducing the chromatic dispersion accumulated by our signal over the distance z, $\mathcal{K}_{\delta,m}[\cdot]$ introduces the Kerr nonlinear phase shift accumulated over the fiber span of length δ centered around the point $(m-0.5)\delta$. Since the linear distortion $\mathbf{u}_{\mathrm{L}}$ and every term of sum in $\mathbf{u}_{\mathrm{NL}}$ can be calculated independently, we refer to their calculation routines as to "branches". The number of branches N_{br} is a main RP-model hyper-parameter, defining both its precision of approximation and its complexity. Typically, N_{br} is chosen between 2 and 10 resulting in step size being in the range $\delta \approx z/10 \ldots z$.

The RP model can be better understood through its comparison with the split-step Fourier method (SSFM) applied for solving the Manakov equations [2], see Figs. 3.2, 3.3. SSFM is formulated as a sequence of alternating steps: the linear steps introducing the dispersive broadening $D_\delta[\cdot]$ and accounting for linear losses, and the full nonlinear one, $\exp(K_{\delta,m})$, introducing Kerr nonlinear phase shift:

$$\begin{gathered}\mathbf{u}(z,t) = \underbrace{\mathcal{D}_{\delta/2}\exp\left(\mathcal{K}_{\delta,m}\right)\mathcal{D}_{\delta/2}}_{\text{repeat } z/\delta \text{ times}}[\mathbf{u}(0,t)],\\ \exp\left(\mathcal{K}_{\delta,m}\right)[\mathbf{u}(t)] = \exp\left(i\frac{8}{9}\gamma\frac{1-e^{-\alpha\delta}}{\alpha} f\left((m-\frac{1}{2})\delta\right)\|\mathbf{u}(t)\|^2\right)\mathbf{u}(t).\end{gathered} \tag{3.5}$$

So we can think of the RP model as of the simplified SSFM model, Eq. (3.5), where at every particular nonlinear step we neglect all the nonlinear steps done before or occurring after it. The RP model assumes the first order approximation in the nonlinear parameter γ, meaning that we disregard the effect of all nonlinear steps on each other, and reduce the exponent in the nonlinear step of SSFM in Eq. (3.5), to a linear approximation $\exp\left(\mathcal{K}_{\delta,m}\right) \approx \mathcal{K}_{\delta,m}$, i.e. use the expansion $\exp(x) \approx 1 + x + O(x^2)$.

Compared to the “conventional” channel modelling by applying the full SSFM, the RP-model application for channel modelling has two main advantages in the context of end-to-end learning. First, the key advantage of RP is its parallel structure, i.e. its separability into the branches, where each particular branch requires approximately the same routine for its contribution’s calculation. This RP property allows speeding up the computation of signal evolution and the computation of gradients via the "back-propagation" stage. Hence, the overall E2E learning is parallelizable in a straightforward way, which obviously benefits our using modern multi-thread computational hardware, e.g. multicore processors and graphics processing units (GPUs). Second, the parallel structure of the RP model can increase the numerical stability of end-to-end learning. As noted in [21], the SSFM structure mimics the one of a typical convolutional NN made up of interchanging convolutions and point-wise nonlinearities. These complex multi-layered structures are prone to numerical errors in gradient estimation, referred to an infamous exploding/vanishing problem [61, 144]. The simpler structure of RP model allow us to circumvent the gradient estimation problems by having a single nonlinear step per branch.

Notably, in addition to all the aforementioned benefits, the RP model offers a rather good approximation for the precise SSFM model. We compare RP and SSFM channel models later for state-of-the-art short-range and long-haul links, correspondingly, in Sections 3.3.1.2 and 3.3.2.2.

3.2.4 Receiver

For each received symbol $y_i \in \mathbf{Y}$, the receiver can estimate the posterior probabilities of either each symbol $s_m \in \mathbf{S}$, $M = |\mathbf{S}|$ or the bit label $\mathbf{b}_i = [b_{i,1}, \ldots, b_{i,K}], K = \log_2(M)$ being transmitted in the corresponding i-th time slot. Particularly, in the first case, the receiver estimates the vector of posterior probabilities $q_{S|Y}(x_i = s_m|y_i)$ of each alphabet symbol $s_m \in \mathbf{S}$ being transmitted in the time slot x_i which corresponds to the received one y_i. In the second case, the receiver estimates the posterior probability $q_{B|Y}(b_{i,j} = b|y_i)$ of bit values $b = \{0, 1\}$ sent in every j-th element of the transmitted bit vector $\mathbf{b}_i$.

We applied a commonly used mismatched Gaussian receiver to estimate the posterior probabilities [170, 171]. In this approach, the conditional probability linking the channel input and output, $q_{Y|X}$, is assumed to obey a Gaussian distribution:

$$q_{Y|X}(y_i|x_i) = \frac{1}{\pi\sigma_G^2} \exp\left(-\frac{\|y_i - x_i\|^2}{\sigma_G^2}\right), \tag{3.6}$$

where σ_G is estimated as the mean squared error between the received symbol sequence $\mathbf{Y}$ and the transmitted symbol sequence $\mathbf{X}$ recorded before the pre-distorter: $\sigma_G^2 = \mathbb{E}\left[\|\mathbf{Y}-\mathbf{X}\|^2\right]$.

First, the estimated distribution $q_{Y|X}$ is used to estimate the symbol posterior probabilities $q_{S|X}$ in a closed form via the Bayesian rule from the symbol occurrence probabilities p_S and the assumed conditional channel probability distribution $q_{Y|X}$. This results in the following expression

$$q_{S|Y}(s_m|y_i) = \frac{q_{Y|X}(y_i|s_m)p_S(s_m)}{\sum_{\tilde{m}=1}^{M} q_{Y|X}(y_i|s_{\tilde{m}})p_S(s_{\tilde{m}})}. \tag{3.7}$$

The bit posteriors $q_{B|Y}$ can be estimated from the symbol ones $q_{S|Y}$ if we have the information about bit labels corresponding to each transmitted symbol $\mathbf{l_m} \Longleftrightarrow s_m, \forall s_m \in \mathbf{S}$.

Let us introduce $\mathbf{S}_{b_j=b} = \{s_m \in \mathbf{S} : l_{m,j} = 1\}$ as the subset of transmission alphabet for which the j-th bit in the corresponding bit labels is set as "1". The bit posterior probability is then estimated as

$$q_{B|Y}(b_{i,j}=1|y_i) = \sum_{s_m \in \mathbf{S}} p_{B|S}(b_{i,j}=1|s_m)\, q_{X|Y}(s_m|y_i) = \sum_{s_m \in \mathbf{S}_{b_j=1}} q_{X|Y}(s_m|y_i), \tag{3.8}$$

since $p_{B|S}(b_{i,j}=1|s_m) = \{1, \text{ if } s_m \in \mathbf{S}_{b_j=1};\ 0, \text{ if } s_m \notin \mathbf{S}_{b_j=1}\}$. Similarly, defining $\mathbf{S}_{b_j=0} = \{s_m \in \mathbf{S} : b_{i,j} = 0\}$, we get:

$$q_{B|Y}(b_{i,j}=0|y) = \sum_{s_i \in \mathbf{S}_{b_j=0}} q_{X|Y}(s_i|y). \tag{3.9}$$

Note that $q_{B|Y}(b_{i,j}=0|y) + q_{B|Y}(b_{i,j}=1|y) = 1,\ \forall y$.

3.2.5 Loss and the training procedure

Having described the E2E learning principle of operation and the blocks of the E2E learning platform, we now bring in a more rigorous description of the learning process. We followed the training procedure proposed in Ref. [161]. During the training, we optimize the symbol locations, probabilities, and pre-distorter parameters via a batch gradient descent procedure, i.e., by a repeated generation of the training symbols' batches of fixed size and updating the trainable parameters using the loss gradients (averaged over the batches).

Let us first describe how the training batch of size N is generated. Recall that the considered transmitter consists of two stages: sampler and pre-distorter. The sampler randomly draws with replacement N indices from the discrete symbol probability distribution $P_S = \{p_S(s_1), p_S(s_2), \ldots, p_S(s_M)\}$, $s_m \in \mathbf{S}$. Each drawn index is mapped to a corresponding transmitted normalized symbol $x_i \in \{s_m / \sqrt{\sum_{m=1}^{M} p_S(s_m) s_m^2}\}$ and a bit label $\mathbf{b}_i \in \mathbf{L}$. The sam-

pled normalized symbols x_i and labels $\mathbf{b_i}$ are stacked, respectively, in the input symbol vector $\mathbf{X} = [x_1, x_2, \ldots, x_N]$ and the bit vector $\mathbf{B} = [\mathbf{b}_1, \ldots \mathbf{b}_N]$. The symbol power normalization is required to keep the batch power constant during the training. Thereby, we prevent the algorithm from optimizing the information rate by trivial power level shifts.

The generated training batch is processed by a cascade of pre-distorter, followed by the channel model. The cascade maps input symbols $\mathbf{X}$ to the vector of channel output symbols $\mathbf{Y} = [y_1, y_2, \ldots, y_N]$ according to the cascade's joint probability distribution $P_{Y|X}$. The differentiability of both pre-distorter and channel model allows calculating the gradients of output symbols over the input symbols and over the pre-distorter parameters. Finally, the receiver estimates the probabilities of "1" and "0" transmitted in every bit slot corresponding to each of the received symbols: $q_{B|Y}(b_{i,j} = b|y_i)$, $y_i \in Y, b \in \{0, 1\}$.

Having considered the dataset generation, let us describe the losses optimized in the tasks of end-to-end learning. In [109] it was recommended that the performance metrics based on information theory offer the most precise predictions of the resulting link performance. In line with this recommendation, as performance metrics we considered the so called "mismatched" information rates, which define the performance of communication system taking into account the non-ideality of decoder. In more detail, the optimal performance in the link is reached if the decoder produces its decisions based on the real channel conditional probability distribution $p_{Y|X}$. At the same time, the real-world decoders always have to approximate the distribution $p_{Y|X}$ with a simpler one $q_{Y|X}$ since $p_{Y|X}$ is usually extremely complex. The "mismatched" performance metrics, therefore, estimate an upper bound on the channel performance for the link using the imprecise decoder $q_{Y|X}$, in comparison to the classical metrics assuming ideal $p_{Y|X}$-based decoding.

Particularly, we consider the two "mismatched" metrics introduced in [92]. The first one is the *mismatched bit-wise mutual information (BMI)*, also referred to as the generalized mutual information (GMI) [153, 172]. BMI defines the information rate achievable in a system with a bit-metric decoding, a decoding principle where every bit is considered separately. The second one is *mismatched symbol-wise mutual information (MI)*, usually referred to just as the mutual information, assuming the per-symbol decoding. The principal difference between BMI and MI is that the former neglects the information present in the inter-bit dependencies $p(b_{i,j_1}|b_{i,j_2}) \forall j_1 \neq j_2$ and, hence, produces the lower bounds on information rate.

3.2.5.1 Mismatched bit-wise mutual information loss

Mismatched bit-wise mutual information (BMI) loss $\mathcal{L}_{\text{BMI}}$ is defined as [92]

$$\begin{aligned}
\mathcal{L}_{\text{BMI}} &:= \max\left[0;\, H(\mathbf{S}) - \sum_{j=1}^{K} H(\mathbf{B}_j|\mathbf{Y})\right], \\
H(\mathbf{B}_j|\mathbf{Y}) &:= -\sum_{b\in\{0,1\}} p(b_{*,j}=b)\cdot\sum_{i=1}^{N} p(y_i|b_{i,j}=b)\log_2\big(q_{B|Y}(b_{i,j}=b|y_i)\big), \\
H(\mathbf{S}) &:= -\sum_{m=1}^{M} p_S(s_m)\log_2(p_S(s_m)), \quad p(b_{*,j}=b) = \sum_{m=1}^{M} p_S(s_m)\{l_{m,j}=b\},
\end{aligned} \tag{3.10}$$

where N is dataset size, M is a transmission alphabet cardinality, $K = \log_2(M)$ is a bit label length. $p(b_{*,j}=b)$ is a marginal probability of a particular bit value $b \in \{0,1\}$ being transmitted in a j-th position of the bit label corresponding to an arbitrary time-slot, with $*$ being a placeholder for arbitrary index. $H(\mathbf{S})$ is the source entropy quantifying the amount of information carried per a transmitted symbol, $H(\mathbf{B}_j|\mathbf{Y})$ is the j-th bit entropy conditioned on the channel output, which quantifies the amount of uncertainty left about the j-th bit in the transmitted bit label $b_{i,j}$ after processing the corresponding received symbol y_i. In turn, the BMI itself (i.e. our loss $\mathcal{L}$) is interpreted as the amount of information one can extract about the transmitted bit sequences $\mathbf{B}$ from the bit posterior probabilities $q_{B|Y}(b_{i,j}=b|y_i)$ estimated by the receiver via Eqs. (3.8), (3.9).

All the parts of E2E learning algorithm: transmitter, channel model, receiver, and loss, were implemented via PyTorch deep learning package [135]. The package implements the *autograd* algorithm [17], which calculates the gradient of loss over the trainable parameters, and the gradient-based optimization routines using the calculated gradients to find the optimal values of symbols. We now describe the calculation of the gradients in more detail.

We start from calculating the loss gradients over symbol's locations and pre-distorter parameters:

$$\begin{aligned}
\frac{\delta\mathcal{L}_{\text{BMI}}}{\delta s_m} &= -\sum_{j=1}^{K}\frac{\delta H(\mathbf{B}_j|\mathbf{Y})}{\delta s_m}, \\
\frac{\delta H(\mathbf{B}_j|\mathbf{Y})}{\delta s_m} &= -\sum_{b\in\{0,1\}} p(b_{*,j}=b)\cdot\sum_{i=1}^{N} p(y_i|b_{i,j}=b)\frac{\delta\log_2\big(q_{B|Y}(b_{i,j}=b|y_i)\big)}{\delta s_m},
\end{aligned} \tag{3.11}$$

where the second line expression is simplified since $q_{B|Y}(b_{i,j}=b|y_i)$ is the only term in $\mathcal{L}_{\text{BMI}}$, which depends on symbols and pre-distorter parameters.

As far as the discrete dataset is used, $p(y_i|b_{i,j}=b)$ can be estimated as

$$p(y_i|b_{i,j}=b)=\begin{cases}\dfrac{1}{\text{number of batch points with } b_{i,j}=b} & \text{for } (y_i,\mathbf{b}_i) \text{ pairs in the batch;}\\ 0 & \text{for all other } (y,\mathbf{b}_i) \text{ pairs.}\end{cases} \tag{3.12}$$

Furthermore, according to the law of large numbers, for big enough batches ($N\to\infty$) we have:

$$p(b_{*,j}=b)\approx\frac{\text{number of batch points with } b_{i,j}=b}{N}. \tag{3.13}$$

We can simplify Eq. (3.11) by substituting Eqs. (3.12), (3.13) into it:

$$\frac{\delta\mathcal{L}_{\text{BMI}}}{\delta s_m}=\frac{1}{N}\sum_{j=1}^{K}\sum_{i=1}^{N}\frac{\delta\log_2(q_{B|Y}(b_{i,j}|y_i))}{\delta s_m}. \tag{3.14}$$

Similarly, for the loss gradient over the pre-distorter parameters $C_{m,n}$, we have:

$$\frac{\delta\mathcal{L}_{\text{BMI}}}{\delta C_{m,n}}=\frac{1}{N}\sum_{j=1}^{K}\sum_{i=1}^{N}\frac{\delta\log_2\left(q_{B|Y}(b_{i,j}|y_i)\right)}{\delta C_{m,n}}. \tag{3.15}$$

Finally, the gradients $\delta\log_2\left(q_{B|Y}(b_{i,j}|y_i)\right)/\delta s_m$ and $\delta\log_2\left(q_{B|Y}(b_{i,j}|y_i)\right)/\delta C_{m,n}$ can be calculated by *autograd* algorithm.

In contrast to the gradients above, it is more difficult to calculate the gradient of BMI loss over the symbol probability distribution $\delta\mathcal{L}_{\text{BMI}}/\delta p_S(s_m)$. First, we cannot use the approximation Eq. (3.13), since we have to take into account the dependence of $p(b_{*,j}=b)$ on symbol probabilities $p_S(s_m)$. Second, the source entropy $H(\mathbf{S})$ cannot be ignored here, since it also depends on symbol probability distribution $p_S(s_m)$. Therefore, the gradient $\delta\mathcal{L}_{\text{BMI}}/\delta p_S(s_m)$ has a more complex form:

$$\begin{gathered}\frac{\delta\mathcal{L}_{\text{BMI}}}{\delta p_S(s_m)}=\frac{\delta H(\mathbf{S})}{\delta p_S(s_m)}-\sum_{j=1}^{K}\frac{\delta H(\mathbf{B}_j|\mathbf{Y})}{\delta p_S(s_m)},\\ \frac{\delta H(\mathbf{B}_j|\mathbf{Y})}{\delta p_S(s_m)}=-\sum_{b\in\{0,1\}}p(b_{*,j}=b)\cdot\sum_{i=1}^{N}p(y_i|b_{i,j}=b)\frac{\delta\log_2\left(q_{B|Y}(b_j=b|y_i)\right)}{\delta p_S(s_m)}\\ -\sum_{b\in\{0,1\}}\frac{\delta p(b_{*,j}=b)}{\delta p_S(s_m)}\cdot\sum_{i=1}^{N}p(y_i|b_{i,j}=b)\log_2\left(q_{B|Y}(b_j=b|y_i)\right).\end{gathered} \tag{3.16}$$

The first term of $\delta H(\mathbf{B}_j|\mathbf{Y})/\delta p_S(s_m)$ is simplified in the same way as Eq. (3.14) by substituting Eqs. (3.12), (3.13) into it. The second term can be simplified by the substitution of the actual value of $\delta p(b_{*,j} = b)/\delta p_S(s_m) = \{l_{m,j} = b\}$. After that, we arrive at the expression:

$$\begin{aligned}\frac{\delta \mathcal{L}_{\text{BMI}}}{\delta p_S(s_m)} = \frac{\delta H(\mathbf{S})}{\delta p_S(s_m)} + \frac{1}{N}\sum_{j=1}^{K}\sum_{i=1}^{N}\frac{\delta \log_2\left(q_{B|Y}(b_{i,j}|y_i)\right)}{\delta p_S(s_m)} \\ + \sum_{j=1}^{K}\sum_{n=1}^{N} p(y_i|b_{i,j} = l_{m,j})\log_2(q_{B|Y}(b_j = l_{m,j}|y_i)).\end{aligned} \tag{3.17}$$

The straightforward application[1] of autograd to calculating the gradient $\delta \mathcal{L}_{\text{BMI}}/\delta p_S(s_m)$ results in the third term in Eq. (3.17) being neglected. The reason is that the autograd assumes that the proportion of objects with different classes in the dataset does not depend on the parameters of trained algorithms, which is the case when an ordinary machine learning algorithms is trained. In other words, the autograd incorrectly assumes that $\delta p(b_{*,j} = b)/\delta p_S(s_m) = 0$. Therefore, to obtain the correct value of $\delta \mathcal{L}_{\text{BMI}}/\delta p_S(s_m)$ we calculated the first two terms of Eq. (3.17) via autograd and, then, added to them the separately calculated third term.

The computed gradients: $\delta \mathcal{L}_{\text{BMI}}/\delta s_m$, $\delta \mathcal{L}_{\text{BMI}}/\delta C_{m,n}$, and $\delta \mathcal{L}_{\text{BMI}}/\delta p_S(s_m)$, were then used by the Adam optimizer [85] to train the respective communication system parameters.

3.2.5.2 Mismatched symbol-wise mutual information loss

Mismatched bit-wise mutual information (MI) loss $\mathcal{L}_{\text{MI}}$ is defined as

$$\begin{aligned}\mathcal{L}_{\text{MI}} &:= \max\left[0;\, H(\mathbf{X}) - H(\mathbf{X}|\mathbf{Y})\right], \\ H(\mathbf{X}) &:= -\sum_{m=1}^{M} p_S(s_m)\log_2(p_S(s_m)), \\ H(\mathbf{X}|\mathbf{Y}) &:= -\sum_{m=1}^{M} p_S(s_m)\cdot\sum_{i=1}^{N} p(y_i|x_i = s_m)\log_2\left(q_{X|Y}(x_i = s_m|y_i)\right)\end{aligned} \tag{3.18}$$

where N is a dataset size, M is a transmission alphabet cardinality. $p(y_i|x_i = s_m)$ is a conditional probability of a particular symbol y_i being received given that $s_m \in S$ symbol was transmitted in the i-th time-slot x_i corresponding to the received one. $H(\mathbf{X})$ is the source entropy quantifying the amount of information carried per a transmitted symbol, $H(\mathbf{X}|\mathbf{Y})$ is the source entropy conditioned on the channel output, which quantifies the amount of

[1] By the "straightforward" autograd application we imply applying it to the calculation of gradients over the whole BMI loss $\mathcal{L}_{\text{BMI}}$, Eq. (3.10), instead of the separate processing of loss terms from Eq. (3.17).

uncertainty left about which symbol $s_m \in S$ was transmitted in slot x_i after processing the corresponding received symbol y_i. In turn, the MI $\mathcal{L}_{\text{MI}}$ itself can be interpreted as the amount of information one can extract about the transmitted symbol sequence $\mathbf{X}$ from the symbol posterior probabilities $q_{S|Y}(s_m|y_i)$ estimated by the receiver via Eq. (3.7).

As with BMI optimization, described in Section 3.2.5.1, we utilize the PyTorch deep learning package [135] to automatically calculate the gradient of loss $\mathcal{L}_{\text{MI}}$ over the parameters of the learnt shaping via *autograd* algorithm [17]. To do this, following the E2E learning paradigm, we implement the transmitter, channel model, receiver, and loss, as trainable blocks described via PyTorch's constructions. Unfortunately, similar to BMI optimization, the straight-forward application of *autograd* to the calculation of loss gradients would result in wrong results. In the following we describe the proper gradient calculation procedure.

We start from calculating the loss gradients over symbol's locations $s_m \in S$ and pre-distorter parameters $C_{m,n}$. First, let's focus on the gradient over symbol locations s_m. Let's note that $q_{S|Y}(s_m|y_i)$ is the only term in $\mathcal{L}_{\text{BMI}}$, which depends on symbols and pre-distorter parameters. Therefore,

$$\begin{gathered}\frac{\delta \mathcal{L}_{\text{MI}}}{\delta s_l} = -\frac{\delta H(\mathbf{X}|\mathbf{Y})}{\delta s_l}, \\ \frac{\delta H(\mathbf{X}|\mathbf{Y})}{\delta s_l} = -\sum_{m=1}^{M} p_S(s_m) \cdot \sum_{i=1}^{N} p(y_i|x_i = s_m) \frac{\delta \log_2\big(q_{S|Y}(s_m|y_i)\big)}{\delta s_l},\end{gathered} \tag{3.19}$$

where $l \in \{0, 1, \ldots, M\}$ is an arbitrary symbol index. Since we operate on the discrete-sampled finite-batch dataset, $p(y_i|x_i = s_m)$ can be estimated from this limited sample as

$$p(y_i|x_i = s_m) \approx \begin{cases} \dfrac{1}{\text{number of batch points with } x_i = s_m} & \text{for } (y_i,\, s_m) \text{ pairs belonging to the batch;} \\ 0 & \text{for all the other } (y,\, s_m) \text{ pairs.} \end{cases} \tag{3.20}$$

Furthermore, according to the law of large numbers, for the batch of the size big enough $N \to \infty$ we have:

$$p_S(s_m) \approx \frac{\text{number of batch points where } x_i = s_m}{N}. \tag{3.21}$$

We can simplify Eq. (3.19) by substituting Eqs. (3.20), (3.21) into it:

$$\frac{\delta \mathcal{L}_{\text{MI}}}{\delta s_l} = \frac{1}{N} \sum_{i=1}^{N} \frac{\delta \log_2\big(q_{S|Y}(x_i|y_i)\big)}{\delta s_l}, \tag{3.22}$$

where x_i denotes the symbols actually transmitted in the i-th timeslot corresponding to y_i one. We consider here the receiver decisions $q_{S|Y}$ only for the pairs of the transmitted and received symbols (x_i, y_i) belonging to the considered batch, not for an arbitrary alphabet symbol s_m. This results from the simplification introduced by Eq. (3.20) that $p(y_i|s_m) = 0$, if $s_m \neq x_i$.

Similarly, for the loss gradient over the pre-distorter parameters $C_{m,n}$:

$$\frac{\delta \mathcal{L}_{\text{MI}}}{\delta C_{m,n}} = \frac{1}{N} \sum_{i=1}^{N} \frac{\delta \log_2\left(q_{S|Y}(x_i|y_i)\right)}{\delta C_{m,n}}. \tag{3.23}$$

Autograd algorithm can be further applied to calculate the gradients $\delta \log_2\left(q_{S|Y}(x_i|y_i)\right)/(\delta s_l)$ and $\delta \log_2\left(q_{S|Y}(s_l|y_i)\right)/\delta C_{m,n}$.

The main difficulties arise when calculating the gradients of MI loss $\mathcal{L}_{\text{MI}}$ over the symbol probability distribution $\delta \mathcal{L}_{\text{MI}}/\delta p_S(s_l)$. First, we can no longer use the approximation Eq. (3.21) to replace the symbol probabilities p_S with a constant, since p_S is the differentiated variable now. Second, the source entropy $H(\mathbf{X})$ has to be taken into account, since it also depends on the variable symbol probability distribution p_S. Therefore, the gradient $\delta \mathcal{L}_{\text{MI}}/\delta p_S(s_l)$ is more complex than the previously considered ones:

$$\begin{aligned} \frac{\delta \mathcal{L}_{\text{MI}}}{\delta p_S(s_l)} &= \frac{\delta H(\mathbf{X})}{\delta p_S(s_l)} - \frac{\delta H(\mathbf{X}|\mathbf{Y})}{\delta p_S(s_l)}, \\ \frac{\delta H(\mathbf{X}|\mathbf{Y})}{\delta p_S(s_l)} &= -\sum_{m=1}^{M} p_S(s_m) \cdot \sum_{i=1}^{N} p(y_i|x_i = s_m) \frac{\delta \log_2\left(q_{S|Y}(s_m|y_i)\right)}{\delta p_S(s_l)} \\ &\quad - \sum_{i=1}^{N} p(y_i|x_i = s_l) \log_2\left(q_{S|Y}(s_l|y_i)\right). \end{aligned} \tag{3.24}$$

We simplify the first term of $\delta H(\mathbf{X}|\mathbf{Y})/\delta p_S(s_m)$ in the same way as Eq. (3.22) by substituting Eqs. (3.20), (3.21) into it. Also, we can simplify the second term of $\delta H(\mathbf{X}|\mathbf{Y})/\delta p_S(s_m)$ by substituting Eq. (3.20) there. After that, we arrive at the expression:

$$\begin{aligned} \frac{\delta \mathcal{L}_{\text{MI}}}{\delta p_S(s_l)} &= \frac{\delta H(\mathbf{S})}{\delta p_S(s_l)} + \frac{1}{N} \sum_{i=1}^{N} \frac{\delta \log_2\left(q_{B|Y}(x_i|y_i)\right)}{\delta p_S(s_m)} \\ &\quad + \sum_{n=1}^{N} p(y_i|x_i = s_l) \log_2\left(q_{S|Y}(x_i|y_i)\right). \end{aligned} \tag{3.25}$$

In view of the approximation defined by Eq. (3.20) the third term can be understood as a mathematical expectation of the receiver decision over all the batch data pairs (x_i, y_i) where

the symbol of interest s_l was transmitted $y_i : x_i = s_l$, i.e

$$\sum_{n=1}^{N} p(y_i|x_i = s_m) \log_2 \left(q_{S|Y}(x_i|y_i)\right) = \mathop{\mathbb{E}}_{i:x_i=s_l} \left[\log_2 \left(q_{S|Y}(s_l|y_i)\right)\right] \tag{3.26}$$

Similar to BMI, the straightforward application of *autograd* to calculating the gradient $\delta\mathcal{L}_{\mathrm{MI}}/\delta p_S(s_l)$ results in the neglection of the third term in Eq. (3.25). The reason is that autograd assumes that the distribution of transmitted characters in batch does not depend on the training parameters, while this is not true for the symbol occurence probabilities $p_S(s_l)$. Hence, to obtain the correct gradient value $\delta\mathcal{L}_{\mathrm{MI}}/\delta p_S(s_l)$ we calculated the first two terms of Eq. (3.25) via autograd and, then added to them the third term.

The transmitter parameters s_l, $p(s_l)$, and $C_{m,n}$ were trained via Adam optimizer [85] consuming the respective gradients $\delta\mathcal{L}_{\mathrm{MI}}/\delta s_l$, $\delta\mathcal{L}_{\mathrm{MI}}/\delta C_{m,n}$, and $\delta\mathcal{L}_{\mathrm{MI}}/\delta p_S(s_l)$.

3.3 Results

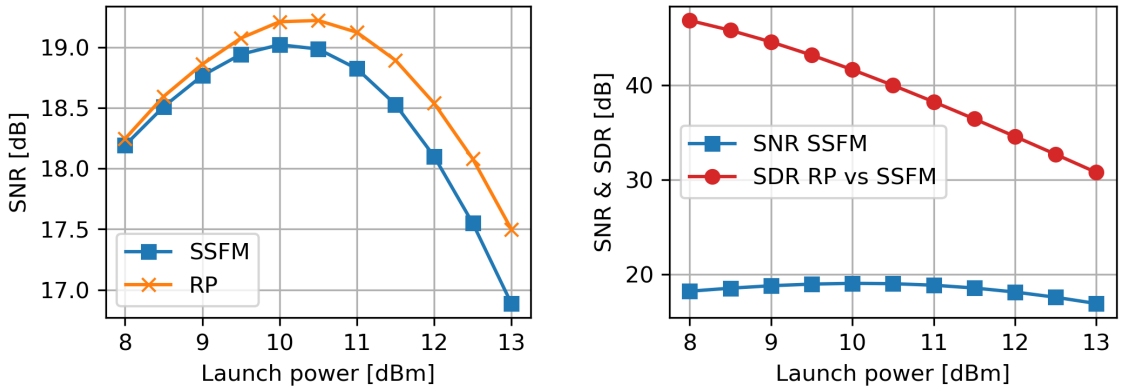
3.3.1 End-to-end learning the single-span link

3.3.1.1 Testcase

Having proposed the end-to-end learning algorithm for the general coherent fiber-optic communication link, we illustrate its benefits on the particular case of a single-span link. More specifically, we numerically consider the dual-polarized (DP) 64 GBd transmission of 256-symbol constellations over the 1x170 km single mode fiber (SMF) link. A root-raised-cosine with roll-off factor of 0.1 is used for pulse shaping. The SMF parameters were taken as: chromatic dispersion coefficient $D = 16.8$ ps/(nm*km), effective nonlinear coefficient $\gamma = 1.14$ (W*km)$^{-1}$, loss coefficient $\alpha = 0.21$ dB/km. The span is followed by a lumped optical amplifier (OA) with the noise figure 4.5 dB.

During the E2E system training, we used the RP model, Eq. (3.5), with $N_{br} = 100$ branches, as an auxiliary channel model. RP model was applied at 2x upsampling to grasp the spectrum distortion caused by nonlinearity. At the same time, the performance of the learnt constellations was estimated using the "precise" SSFM, i.e. via the channel model shown in Eq. (3.4).

In our current work we focus on a single-span link, because in such a case the nonlinearity-aware constellation shaping is expected to produce a considerable gain [162]. However, we emphasize that our method is equally applicable for other fiber-optic communication systems, where, of course, the ultimate gain figures can be different. For instance, in [3] we



(a) The received signal-to-noise ratio (SNR) for SSFM and RP models.

(b) Signal-to-distortion ratio (SDR) comparing the mismatch in the deterministic distortion injected by SSFM and RP with the SNR for SSFM model.

Fig. 3.4 Comparison between channel models based on first-order regular perturbation (RP) and split-step Fourier method (SSFM) approximating the 64 GBd single channel dual-polarised transmission of unshaped 256QAM signal over 1x170 km SMF link. This figure is taken from [1].

successfully applied the similar E2E learning technique to train the geometric constellation shaping and the nonlinear pre-distorter for the 64 GBd transmission over the long-haul 30x80 km (2400 km) SMF link.

3.3.1.2 RP model channel approximation precision

We start from showing that RP model, despite being a simplified one, offers a decent approximation of the precise SSFM model. For the testcase, considered in this paper, we compared how do RP and SSFM model the propagation of an unshaped 256-QAM signal. In Fig. 3.4, first, we plot the signal-to-noise ratio (SNR) of the received signals after the chromatic dispersion compensation (CDC) was applied. One can see that the SNR is nearly the same for both the SSFM and RP models in the weakly nonlinear regime (up to 10 dBm), and, most notably, the correspondence is still good around the optimal launch power level of $P_{\text{opt}} \approx 10$ dBm. To demonstrate that this SNR-value-similarity comes directly from RP model correctly approximating the determenistic nonlinear distortions introduced by SSFM, we compared the outputs of the models in the noiseless scenario: we fed the same input signal to both models, while they generated no ASE noise, i.e. them were modelling only the deterministic distortions introduced into the propagated signal - chromatic dispersive broadening and Kerr nonlinearity. We quantified the difference between the outputs of nois

eless RP $\mathbf{Y}_{\text{RP}}$ and SSFM $\mathbf{Y}_{\text{SSFM}}$ in terms of signal-to-distortion ratio (SDR), defined as $-20\log_{10}(\|\mathbf{Y}_{\text{SSFM}}\|/\|\mathbf{Y}_{\text{RP}}-\mathbf{Y}_{\text{SSFM}}\|)$, also added to Fig. 3.4b. We see there that up to 10.5 dBm, the SDR is at least 20 dB larger than the received SNR, modelled by SSFM, implying that the approximation error of the RP model is much smaller than the total distortion in the link. As from Fig. 3.4 we readily see that the RP model renders a very good channel approximation for the case considered, and the deterministic mismatch between the models cannot noticeably affect the E2E system training.

3.3.1.3 Learning the constellation shaping without pre-distorter

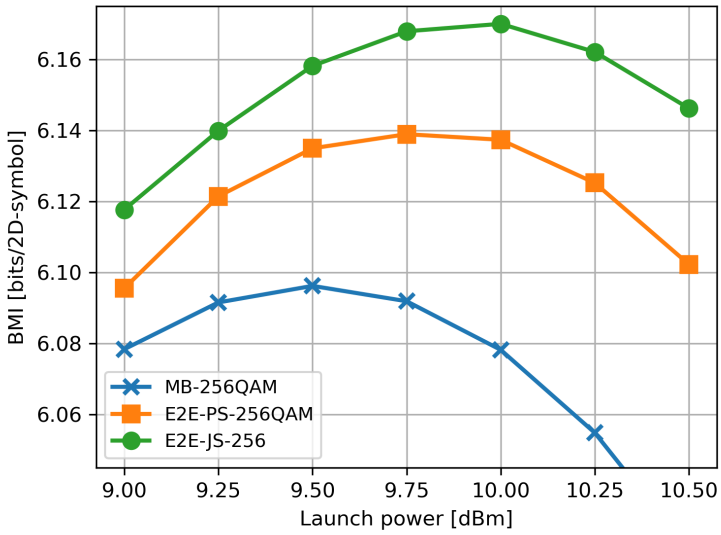
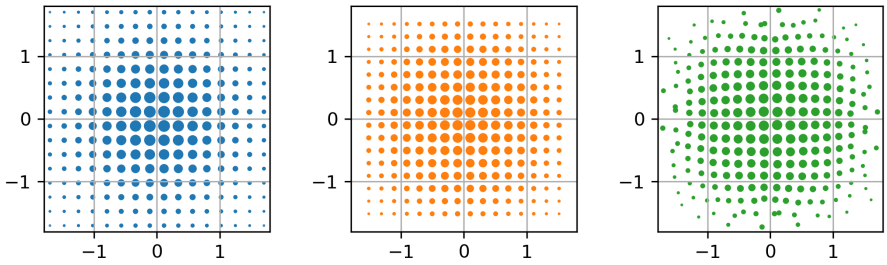


Fig. 3.5 The performance of the constellation shaping without pre-distortion: the reference Maxwell-Boltzmann (MB-256QAM) shaping, learnt probabilistic shaping (E2E-PS-256QAM), and the learnt joint probabilistic and geometric constellation shaping (E2E-JS-256). This figure is taken from [1].

Once we have tested the accuracy of the RP model, we turn to the results for the E2E learning the constellation shaping.

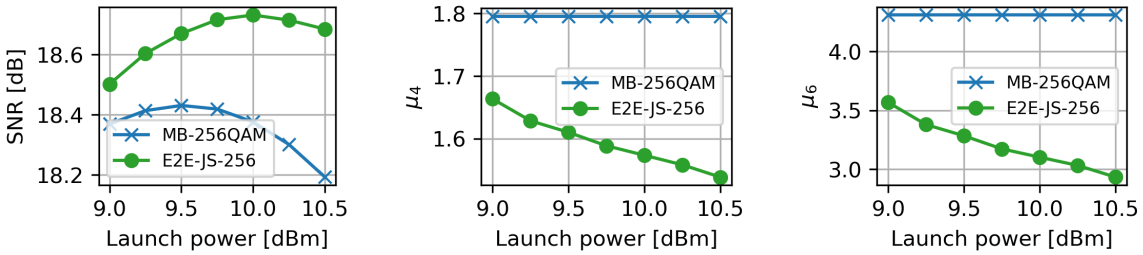
In the first case, we consider learning the constellation shaping in a link without a pre-distorter, i.e. it was disabled and we learnt only PS and GS of separate symbols. This option is well suited for the use cases when there is no opportunity to implement a separate pre-distorter at the transmitter, typically, because of the limited complexity and/or power budget.

Before training, we initialized the encoder with a MB-shaped 256QAM constellation, defined in Eq. (3.1). To find the MB shaping parameter θ we, first, initialized the constellation



(a) MB-256QAM (b) E2E-PS-256QAM (c) E2E-JS-256

Fig. 3.6 Constellations applied at the optimal power level for the case of the link where no nonlinear pre-distortion has been applied. This figure is taken from [1].



(a) Effective SNR. (b) 4th constellation moment μ_4. (c) 6th constellation moment μ_6.

Fig. 3.7 The comparison between the metrics of reference Maxwell-Boltzmann 256QAM (MB-256QAM) and the E2E learnt JS (E2E-JS-256) constellations. The effective SNR was measured in the 1x170 km SMF link modelled by precise SSFM. This figure is taken from [1].

as an unshaped 256QAM one and recorded the SNR produced by this constellation between the input and output of the auxiliary channel model. The MB shaping parameter θ was chosen as to have the optimal value for the AWGN channel producing the same SNR level. The MB constellation with the θ parameter value found in this way was used as a starting seed for all the following learning of constellation shapings and its performance was used as a reference one.

We did the two types of constellation shaping learning. In the first experiment, we learnt just the PS. This is a preferable case for the links, where introducing the GS is not favored. The benefit of PS is that, while providing the performance gain, it keeps the square grid of QAM constellation intact (see Fig. 3.6b). Having the square grid of the transmitted constellation allows one to use the cost-effective blind DSP algorithms to recover the signal at the receiver [26] and lower the required precision of a digital-to-analog converter at the transmitter by limiting the range of signal powers along I- and Q- components present in the transmitted signal. In the second experiment, we jointly learnt the PS and GS to highlight the

full single-symbol joint shaping (JS) gains reachable by the E2E learning strategy. For the PS, the training started from the reference MB shaped constellation. Then, the PS, learnt at the first stage, was used as an initial seed for the JS E2E learning.

The whole training procedure: constellation initialization as the MB-shaped one, learning the PS, and learning the JS, was done separately for a range of power levels. Fig. 3.5 showcases the performance achieved by these shaped constellations in a validation SSFM simulation. The learnt PS (E2E-PS-256QAM) resulted in the 0.043 bits/2D-symbol BMI gain on top of MB-shaping, and led to ≈ 0.25 dBm increase in optimal power level. The learnt JS (E2E-JS-256) including both PS and GS, led to nearly doubling the performance gain: it resulted in 0.074 bits/2D-symbol BMI gain on top of MB-shaping.

The better performance of learned constellations can be partly explained by the fact that they produce a higher effective SNR in the link. Figure 3.7a shows that the effective SNR of the learned JS is higher than that of the reference MB shaping. Furthermore, we can quantify the difference between the constellations, leading to different SNR values. The extended Gaussian noise (EGN) model [105] suggests that the SNR of a signal propagated over a non-linear channel depends on the constellation and is inversely proportional to its 4th μ_4 and 6th μ_6 standard moments, where the k-th standard moment μ_k of the input symbol sequence $\mathbf{X}$ is defined as

$$\mu_k[\mathbf{X}] = \frac{\mathbb{E}\left[|\mathbf{X} - \mathbb{E}[\mathbf{X}]|^k\right]}{\left(\mathbb{E}\left[|\mathbf{X} - \mathbb{E}[\mathbf{X}]|^2\right]\right)^{k/2}} \tag{3.27}$$

and $\mathbb{E}[\cdot]$ stands for the expectation value. Indeed, Figs. 3.7b, 3.7c show that μ_4, μ_6 moments of the E2E learnt JS are lower than the ones for the MB constellation. Furthermore, the difference in SNR, and μ_4, μ_6 moments, increases with the rise of the launch power, when the nonlinear distortions strengthen.

3.3.1.4 Learning the cost-effective pre-distorter via end-to-end learning

Once we have considered the single-symbol conventional constellation shaping, we move to the more advanced case of memory-aware shaping, i.e., the shaping when the transmitted symbol depends not only on the message transmitted in the corresponding time slot, but also on the messages transmitted in the neighbouring slots. To be effective, a constellation shaping should take into account the symbols co-propagating in the neighbouring slots, since they contribute to the deterministic nonlinear distortions introduced into the transmitted symbol of interest. We name this approach as the *multi-symbol constellation shaping* (MSCS).

Thankfully, the RP-based auxiliary channel model, Eq. (3.4), implemented into the E2E learning algorithm proposed in this work, allows the E2E learning of MSCS, since the RP introduces inter-symbol deterministic nonlinear distortions. Conversely, the EGN-based

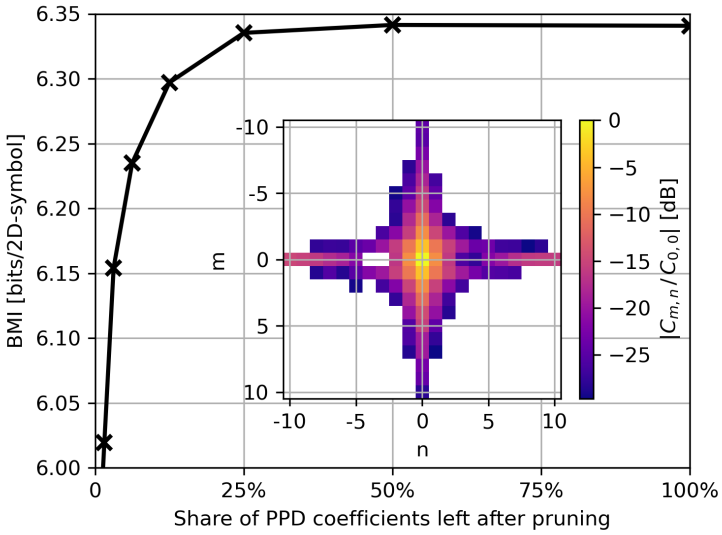


Fig. 3.8 The dependence of the performance of pruned indirectly learnt perturbation-based pre-distorter (PPD) with $|m|, |n| \leq 10$ on the margin of the most significant $C_{m,n}$ coefficients non-zeroed during the pruning procedure. The PPD training was done on the RP model, while the performance was measured in precise SSFM simulation. Inset: The distribution of PPD coefficients $C_{m,n}$ learnt and pruned with the cut-off leaving 25% of the coefficients. PPD coefficients zeroed by pruning are denoted as white squares. This figure is taken from [1].

auxiliary channel model [105], suggested in previous works for end-to-end learning the coherent optical communications [152, 167], does not support the MSCS learning: the EGN replaces the actual inter-symbol nonlinear distortion with the symbol-independent Gaussian noise and, therefore, prevents the E2E learning of dependencies between the neighbouring symbols.

We propose the MSCS implementation as a combination of single-symbol JS, described in the previous subsection, and the nonlinear PPD-based pre-distorter, Eq. (3.2), jointly trained in a single run of E2E learning. Before considering the MSCS training as a whole, we focus on the E2E learning for the cost-effective pre-distorter. We note that it is important to reduce the complexity of pre-distorter, inasmuch as its complexity defines the additional costs of MSCS implementation over the single-symbol JS, and, therefore, denominates the feasibility of the whole MSCS implementation.

As mentioned in Sec. 3.2.2, the PPD-based pre-distorter performance-to-complexity ratio is defined by the range of nonlinear perturbation terms $T_{m,n}$. An effective way to set it is by keeping in the algorithm only the terms $T_{m,n}$ for which the corresponding coefficient $|C_{m,n}|$ has the absolute value above the fixed cut-off threshold: $|C_{m,n}| > C_{\text{cutoff}}$. We refer to this

approach as the pre-distorter pruning, in analogy with the similar technique from neural networks optimization [173].

We seek to find the optimal C_{cutoff} value by a grid search. First, we loaded the unshaped 256QAM constellation to the sampler, and excluded it from E2E learning. Then we trained the PPD with $|m|, |n| \leq 10$, and pruned it with the range of various cut-off values C_{cutoff}. We measured the performance of the generated family of E2E learnt pre-distorters on the precise SSFM channel model. The dependence of the measured performances of the link with the pruned PPD, on the share of pruned coefficients $C_{m,n}$, is given in Fig. 3.8. The figure shows that when we keep aroung 25% of the most significant PPD terms $T_{m,n}$, adding the new ones leads to negligible improvement in the resulting algorithm's performance. This pruning scheme will be used in the following MSCS learning.

Notably, the distribution of the learnt PPD coefficients $|C_{m,n}|$ agrees with the approximation suggested in [120], implying that $|C_{m,n}| \sim 1/|mn|$. The inset of Fig. 3.8 shows the distribution of the absolute value of coefficients $|C_{m,n}|$ in the aforementioned indirectly trained PPD with 25% coefficients left. One can see that, the cut-off boundary, where $|C_{m,n}| \approx C_{\text{cutoff}}$, indeed has the hyperbolic shape $m \sim 1/n$, as the theory suggests[120].

3.3.1.5 Learning the memory-aware constellation shaping

We have defined the cost-effective perturbation-based pre-distorter (PPD) in the previous subsection, and now we consider the E2E learning of the MSCS enabled by the pre-distorter. The MSCS estimates the performance gains reachable by the proposed E2E learning algorithm, when no complexity-limiting constraints are put on it. The set of non-pruned triplets $T_{m,n}$, defined during the PPD initialization, was kept fixed during the following E2E learning. After the initialization, we simultaneously optimized the PPD with joint probabilistic and geometric single-symbol constellation shaping, thereby arriving at the MSCS shaping.

The pre-distorter initialization followed by the E2E MSCS learning was done separately for a range of launch powers. The performance of the resulting learnt constellation measured on the precise SSFM channel model is given in Fig. 3.9. For comparison, we also plotted the performance of the single-symbol constellation shapings learnt in Sec. 3.3.1.3: MB-256QAM and E2E-JS-256. Compared to the reference MB constellation shaping, the MSCS led to the considerable improvement in system's GMI, giving 0.48 bits/2D-symbol, and the optimal power level moved up by ≈ 1.25 dBm.

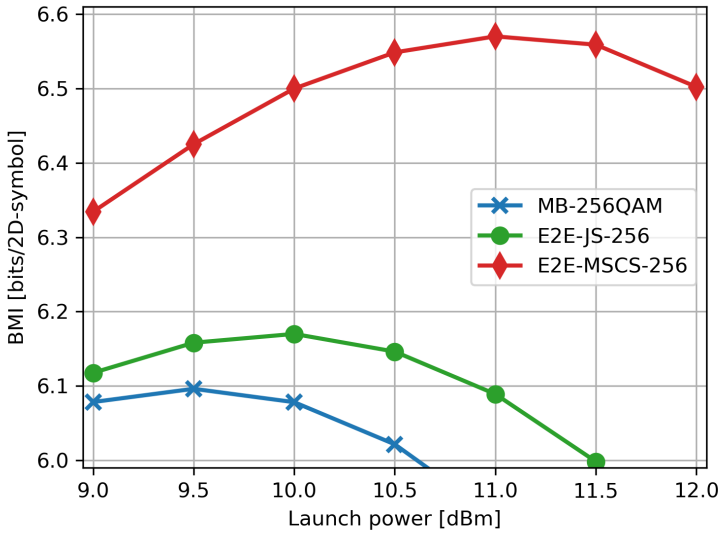


Fig. 3.9 The performance of the end-to-end learnt multi-symbol constellation shaping (E2E-MSCS-256). For reference, we added to the figure the performance of Maxwell-Boltzmann (MB-256QAM), and the learnt JS (E2E-JS-256). This figure is taken from [1].

3.3.2 End-to-end learning the long-haul link

3.3.2.1 Testcase

In the previous section, we brought the results achieved by the end-to-end (E2E) learning of the constellation shaping for a single-span link. In this section, we apply E2E learning to a state-of-the-art coherent link. By considering this testcase we intend to show that the proposed E2E learning algorithm is applicable to a broad range of fiber-optic communication links.

In more detail, in this section as a testcase we numerically consider single-channel (SC) dual-polarized (DP) 64-symbol transmission at 64 Gbaud over a long-haul link formed by 30 spans of 80 km SSMF. The pulse shaping and fiber parameters were taken the same as in the previous Section 3.3.1.1. Every span was followed by an ideal lumped OA with noise figure NF = 4 dB.

Another difference from the previous testcase is that here we considered a multi-stage RP model as an auxiliary channel model during E2E learning. The compared RP-based channel model was made by a sequence of the three same RP algorithms each covering one third of the link and covering every 80 km fiber span with 10 branches. This configuration of RP model gave the optimal accuracy to complexity ratio for the considered testcase.

Furthermore, the receivers considered in this testcase differed in two main ways from the ones considered in the previous testcase. First, they were designed to map every received

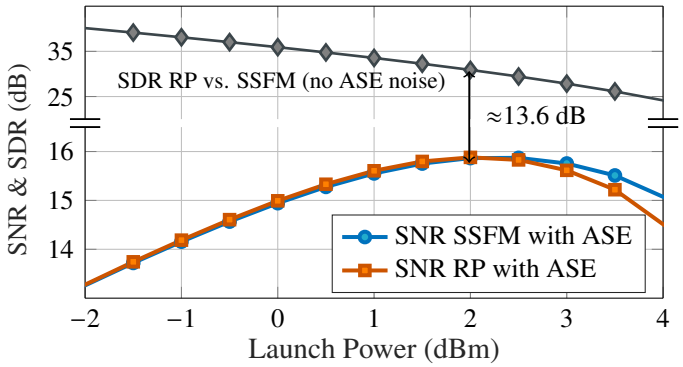


Fig. 3.10 Comparison of 3-stages RP model with the SSMF simulation. The approximation error of the RP model is much smaller than the total distortion. This figure is taken from article [3]. ©IEEE 2022.

symbol $y_i \in \mathbb{Y}$ to a set of posterior probabilities $P(s_m|y_i)$ of each constellation point $s_m \in S$ being transmitted, not bit posteriors $P(b_{i,j}|y_i)$ considered in previous sections. The second difference was that the decoders considered in this section used more complex trainable algorithms to estimate the posteriors.

First, we applied a neural-network (NN)-based trainable receiver. It was implemented as a dense NN with trainable weights. The network started with the input layer having the two neurons accepting separately the real $\mathrm{Re}[y_i]$ and imaginary $\mathrm{Im}[y_i]$ parts of the received symbol y_i. The input layer is followed by two hidden layers, with 32 neurons each, and a 64-neuron output layer. The hidden layers and the output layer, correspondingly, used rectified linear unit (ReLU) [174] and softmax [9] as activation functions.

Second, to illustrate the quality of the learnt NN-based receiver we compared it with the optimal decoder implementation estimating the symbol posteriors $q_{X|Y}(s_m|y_i)$ via formula Eq. (3.7) with the channel conditional probability distribution $q_{Y|X}$ being estimated via kernel density estimator (KDE) instead of analytical expression Eq. (3.6). In more detail, the channel conditional probability linking the channel input and output, $q_{Y|X}$ is estimated from the set of transmitted $\mathbf{X}$ and received symbols $\mathbf{Y}$ via kernel density estimator (KDE) implementation from [175].

Meanwhile, similar to the previous test, while end-to-end learning the constellation shaping was done using RP as an auxiliarry channel model, the performance of learnt constellations was estimated using the "precise" SSFM solver of Manakov equations, Eq. (3.4).

3.3.2.2 RP model precision

In the beginning, we checked whether the RP model is accurate enough by comparing it with precise SSFM in terms of signal-to-noise ratio $\mathrm{SNR}(\mathbf{X},\mathbf{Y}) = 20\log_{10}(||\mathbf{X}||/||\mathbf{Y}-\mathbf{X}||)$, where $\mathbf{X}$ is the baseband signal at the input of the channel and $\mathbf{Y}$ is the signal on the output of the channel post-processed by an ideal chromatic dispersion compensation (CDC). The comparison procedure was similar to the one outlined in Section 3.3.1.2. In more detail, we fed RP and SSFM with the same DP-64QAM symbol sequence $\mathbf{X}$ and compared SNRs between the input $\mathbf{X}$ and the output post-CDC $\mathbf{Y}$ baseband symbol sequences. The obtained SNRs for RP and SSFM models are given in Fig. 3.10. One can see that the difference between SNRs for RP and SSFM is at negligibly small level < 0.06 dB up until the practically important optimal power level $P = 2.5$ dBm, where the links typically operate. Nevertheless, at higher powers, the SNRs mismatch between RP and precise SSFM model increases, since the assumption of weak nonlinearity laid in the foundation of RP model is no longer applicable here.

Similar to Section 3.3.1.2, we illustrated the strength of RP model approximation error by comparing the deterministic distortion introduced by RP and SSFM models. To do this, we considered RP and SSFM models with the ASE noise being turned off. We fed both noiseless models with the same unshaped DP-64QAM signal and compared their outputs. As in previous section, the difference between the RP and SSFM mode outputs, $\mathbf{Y}_{\mathrm{RP}}$ and $\mathbf{Y}_{\mathrm{SSFM}}$, respectively, was quantified in terms of the signal-to-distortion ratio defined as $\mathrm{SDR} = 20\log_{10}(\|\mathbf{Y}_{\mathrm{SSFM}}\|/\|\mathbf{Y}_{\mathrm{RP}}-\mathbf{Y}_{\mathrm{SSFM}}\|)$. The results are given in Figure 3.10. Notably, up to the optimal power level 2.5 dBm, SNRs of RP and SSFM models agree. Furtermore, up to this level SDR is at least 13 dB larger than the received SNR, implying that the approximation error of the RP model is much smaller than the total distortion in the link. As expected, at higher power levels in highly nonlinear regime the disagreement between RP and SSFM models strengthens.

3.3.2.3 Performance gains of end-to-end learning

Following Section 3.3.2, for the aforementioned testcase of a long-haul link, we considered end-to-end learning of single-symbol and multi-symbol constellation shapings. First, we learnt the single-symbol geometrical shaping of a 64-letter constellation (GS-64). Second, we learnt the multi-symbol constellation shaping made up by the aforementioned geometrical constellation shaping and nonlinear pre-distortion (GS-64 + PPD), in a way similar to the one considered in Section 3.3.1.5. Particularly, in an end-to-end learning process we simultaneously optimized the constellation shaping and the weights of perturbation-based

pre-distorter. We considered a perturbation-based pre-distorter Eq. (3.2) with the range of considered nonlinear perturbation triplets $T_{m,n} : |m| \leq 10, |n| \leq 10$. No pruning of nonlinear perturbation triplets $T_{m,n}$ was considered in this experiment.

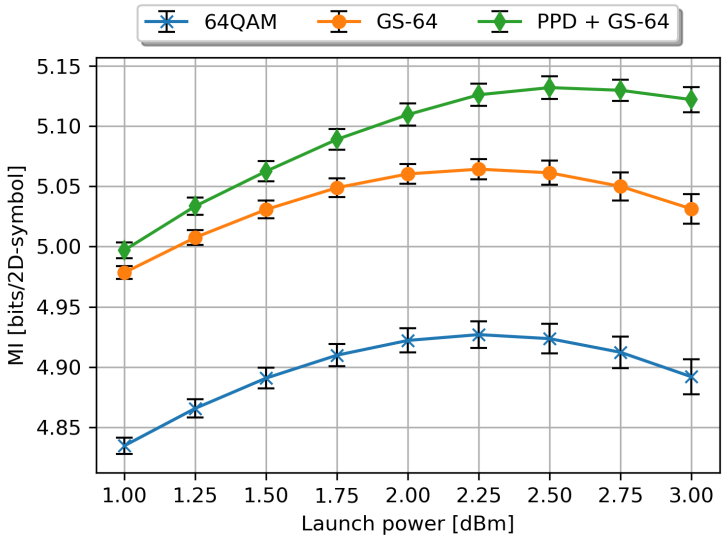
As performance metrics we optimized the symbol-wise mutual information (MI). The shapings were separately trained for every considered launch power level over RP-based auxiliary channel model. Later, we validated the performance of learnt constellations on the precise SSFM-modelled link.

Fig. 3.11a shows the mutual information reached by the validation SSFM-based link implementing the E2E-learnt single-symbol and multi-symbol geometric constellation shaping. As reference, we compare the performance of the learnt constellations with the unshaped 64QAM constellation (referred as 64QAM). Since, the autoencoder training results are dependent on the initial random generator seed, we run the training 10 times with 10 different starting seeds for every power level by redefining in every training run the starting NN weights distribution and the distribution of ASE noises injected by channel model. For each point on the Figure 3.11a we estimated the actual value of MI and its error via, correspondingly, the mean and standard deviation of the range of obtained MI values.

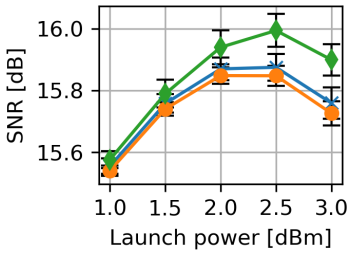
We observe that implementation of the E2E learnt constellations results in a considerable mutual information gain. The learnt single-symbol geometric constellation shaping produced MI gain of ≈ 0.14 bits/sym./pol. while the MI gain of a multi-symbol constellation shaping rose further to ≈ 0.20 bits/sym./pol. The optimal launch power also increased by ≈ 0.5 dB.

Next, we quantified the differences between the considered signal shapings. First, in Figure 3.11b we compared the effective signal-to-noise ratios produced by the considered shapings. Notably, the pure GS-64 shaped signal produced SNR worse than the unshaped 64QAM signal. At the same time, multi-symbol GS-64 + PPD shaping, managed to reverse this trend and produced effective SNR values higher than both 64QAM and GS-64 shapings.

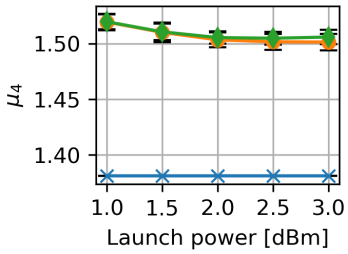
To illustrate the possible reasons for the reported SNR difference, we consider the statistics of the learnt constellations. As mentioned in Section 3.3.1.3, the SNR generated by constellation is known to be inversely proportional to its 4th μ_4 and 6th μ_6 order standard moments described by Eq. 3.27. On Figures 3.11c and 3.11d we compared, respectively, the 4th and 6th standard moments for the 64QAM, GS-64 constellation shaping, and the geometrical shaping component of the multi-symbol GS-64 + PPD shaping. Similar to our findings reported in [164], we found there that the lower SNR value of E2E learnt GS-64 shaping compared to the reference unshaped 64QAM constellation corresponds to the GS-64 signal having higher values of the standard moments μ_4 and μ_6. Notably, the moments' value for GS-64 and the GS part of multi-symbol shaping is nearly the same, which might



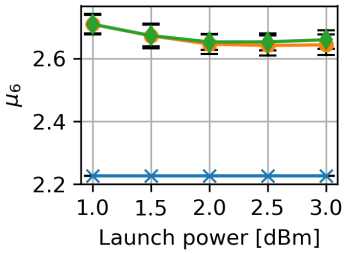
(a) Symbol-wise mutual information (MI).



(b) Effective signal-to-noise ratio (SNR).



(c) 4th order standard constellation moment.



(d) 6th order standard constellation moment.

Fig. 3.11 The results of end-to-end learning the geometric constellation shaping in a realistic case of a 64GBd 30x80 km SSMF link, described in Section 3.3.2.1. In the figures we compare the performance and metrics for unshaped 64QAM signal (64QAM), single-symbol geometrically shaped 64-letter constellation (GS-64), and the multi-symbol geometrical constellation shaping implemented as a combination of simultaneously learnt GS-64 shaping and the perturbation-based pre-distorter (PPD + GS-64). For GS-64 and PPD + GS-64 cases the stadard moments are calculated only for the single-symbol geometrically shaped constellation via formula Eq. 3.27.

indicate that higher SNR and MI values corresponing to GS-64 + PPD shaping compared to single-symbol GS-64 shaping are mostly caused by an addition of the trainable pre-distorter.

3.4 Summary

The end-to-end learning of the optical coherent detection communication system offers a possibility to specialize the characteristics of the transmitted signal to the properties of the nonlinear channel. However, we underline that the two main problems are typically associated with the E2E learning implementation.

The first problem relates to the overall complexity of modelling the channel distortions, where the latter constitute an intricate mix of instantaneous nonlinear fiber responses intertwining with a dispersive pulse broadening. The modelling of modern high-baudrate links is especially difficult because of a huge dispersive memory, implying a considerable complexity and time consumption of the split step simulations. In our work, we address this issue by proposing a parallelizable simplified channel model based on the first-order regular perturbtaion [2]. The model is described in Section 3.2.3.

The second challenge is the difficulty of introducing the concept of trainable discrete probability distribution into the machine learning algorithm. In our paper, this problem is addressed by adopting a novel training procedure proposed first in Ref. [161] for the AWGN channel. The training procedure combines the conventional batch gradient descent with the custom gradient calculation procedure. The training procedure is described in Section 3.2.5.

The resulting composite solution, proposed and demonstrated in this work, made possible learning the joint probabilistic and geometric shaping of symbol sequences. Even though the considered approach is still sub-optimal, the computed multi-symbol constellation shapings have shown the considerable performance improvement in the both considered testcases of short-haul and long-haul communication links. Particularly, E2E learnt joint probabilistic and geometric shaping has shown a considerable bit-wise mutual information (BMI) improvement (of 0.48 bits/2D-symbol) over the conventional Maxwell-Boltzmann shaping for a single-channel 64 GBd 256-symbol transmission over the 170 km SMF link. Furhtermore, for the case of a state-of-the-art long-haul transmission link - 64 GBd 64-symbol transmission over 30x80km SMF spans - the E2E learnt geometric shaping has shown a significant symbol-wise mutual information (MI) gain of 0.2 bits/2D-symbol over the reference unshaped constellation.

Moreover, we observed that the proposed end-to-end learning is applicable in situations when, because of hardware or complexity limitations, we cannot use the multi-symbol shaping. For the aforementioned single-span transmission testcase, we found that a single-symbol joint probabilistic and geometric shaping gives 0.074 bits/2D-symbol BMI gain over the reference MB shaping. Similarly, for the considered long-haul communication link, the single-symbol geometrical shaping outperformed the reference constellation by 0.14 bits/2D-symbol in terms of MI.

We believe that the end-to-end learning approach, proposed in this chapter, can lay a path to finding the optimal signal distribution for a variety of nonlinear fiber-optic channels.

3.4.1 Contribution statement and attribution

Chapters 3.1, 3.2, 3.3.1, and Figures 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9 are taken from article [1] which I co-authored as a leading and corresponding author. Figure 3.10 is taken from conference paper [3] which I also co-authored as a leading and corresponding author.

I have obtained all the results presented in this chapter, drafted its text, and prepared all illustrations, besides Figure 3.10, by myself. The method proposed in this chapter was developed by me together with the co-authors of these articles [1, 3, 164]. The computer code used in the presented research was written by myself, except for the RP model, written jointly with Andrea Carnio. This research was done under the supervision of Dr. Vahid Aref, Prof. Sergei K. Turitsyn, and Dr. Jaroslaw E. Prilepsky.

Chapter 4

Conclusion

4.1 Results summary

Meeting the capacity demands required nowadays from the telecommunication service providers necessitates the development and implementation of new approaches to coherent fiber-optic communication, which forms the backbone of the modern telecommunication infrastructure. Simultaneously, in the last two decades, a novel class of algorithms, referred to as *machine learning*, was shown to reach record-breaking performance in a huge scope of tasks. These algorithms can improve their performance on a given task by analyzing the set of processed objects, referred to as the training dataset. A subfield of machine learning, namely deep learning, has recently enjoyed considerable attention by the research community. In this thesis, we present applications of the solutions developed in general deep learning domain to the important problem of mitigating the nonlinear distortions arising in the physical layer of coherent fiber-optic links.

Chapter 1 lays the theoretical foundation for the key results presented in the following chapters of this thesis. It introduces the background and main concepts in both deep learning and digital communication systems, reviews the recent applications of machine learning and deep learning to mitigating the nonlinear distortions in fiber-optic communication links.

Chapter 2 proposes the method of using the pre-collected datasets in a more effective way for training the machine learning algorithms aimed at receiver-based nonlinear distortion compensation. The technique proposes expanding the existing dataset with synthetic points, generated from the naturally collected ones, an approach known as *data augmentation* in a broader machine learning domain. Particularly, the technique suggests employing the symmetries of the nonlinear Schrödinger equation, the numerical model linking the inputs and outputs of an optical channel. These symmetries are used to generate the synthetic data points, which are used along with the originally collected ones in the algorithm training.

It is shown both numerically and experimentally that the suggested data augmentation technique reaps considerable benefits when applied to the two dissimilar cases. First, it is shown to improve the performance of the nonlinearity compensation algorithm training on insufficient datasets. For the considered testcases, the nonlinearity compensator training on the augmented dataset led to the same performance, as if it was done on an at least 4 times bigger natural dataset. Second, when a big enough dataset is used, the data augmentation can reduce the numerical complexity of training the nonlinearity compensation algorithm. In more detail, data augmentation allows reducing the size of the dataset used in the algorithm training, while keeping the same level of the algorithm performance, which leads to the reduction in the training complexity. For the systems studied in this work, ≈ 2 times training complexity reduction was observed.

Chapter 3 proposes a novel approach for the nonlinearity mitigation by simultaneous learning of pre-distorter, equalizer, and constellation shaping robust to the nonlinear distortions present in a coherent fiber-optic link. These methods are referred to as end-to-end learning, since they are based on implementing via a single trainable neural network all the blocks of a fiber-optic communication link end-to-end. The parameters of the link are then taken from the neural network. The novelty of the proposed approach for end-to-end learning the coherent fiber-optic communications comes from the two innovations brought into it. First is the refined training procedure enabling the simultaneous optimization of both locations and the occurrence probabilities of symbols in the constellation alphabet. Second is the auxiliary channel model based on the first order perturbation theory allowing a cost-effective pre-distorter training. Since the resulting combination of the joint single-symbol shaping and nonlinear pre-distorter is effectively the constellation shaping involving several neighboring symbols, we refer to it as multi-symbol constellation shaping. The proposed end-to-end learning technique was successfully applied to learning the multi-symbol shaping in a single-channel transmission over both a state-of-the-art short-haul single-span link and a long-haul multi-span link.

Finally, this Chapter concludes the thesis by providing the overall summary of the results presented in thesis and the directions for future research.

4.2 Possible research directions

As mentioned in the previous section, the results presented in this thesis can be grouped around two topics - the data augmentation for nonlinearity compensation algorithms, and the end-to-end learning of multi-symbol shaping. In the following, we outline the possible directions for the research continuation in each of these fields.

Regarding the data augmentation, presented in Chapter 2 of this thesis, we believe that the following research should concentrate on bringing it close to the industrial applications:

- First, the channel model used to derive the transformations, present in the link, should be refined to include the transceiver distortions present in the link. The currently considered model Eq. (2.1) describes only the nonlinearities arising in the optical channel, and therefore, not all the transformations derived from it are applicable to the real-world links, where the overall distortion is more complex. Notably, in the field trial, presented in Section 2.5.2, the time-inversion-based data augmentation stopped providing performance gains. Therefore, a more systematic study of the real-world links is needed to make a decision on the limits of the applicability of data augmentation.

- Second, the application of data augmentation to a broader class of nonlinearity compensation algorithms should be considered to verify that gains of data augmentation have general applicability, i.e, that its performance gains do not depend on the particular choice of the compensator. Particularly, we suggest to consider various nonlinearity compensation algorithms based on artificial-neural-network solutions, reviewed in Section 1.1.2.

End-to-end learning of multi-symbol constellation shaping, proposed in Chapter 3 of this work, could be also improved in several ways:

- First, one can consider a more general task formulation for a multi-symbol constellation shaping. Particularly, one can remove the assumption, taken in this work, of multi-symbol shaping being a cascade of single-symbol conventional constellation shaping algorithms followed by a nonlinear pre-distorter, and consider the constellation shaping directly mapping several consecutive bit strings into a sequence of blocks. This should allow the end-to-end learning algorithm to find more optimal constellation shaping by allowing it to explore a higher number of degrees of freedom.

- Second, one can improve the applicability of the end-to-end learned constellation shaping by implementing a more precise auxiliary channel model as part of it. Particularly, employing an auxiliary channel model describing both transceiver and optical channel nonlinearities should be of great practical and scientific interest. In the results, presented in the thesis, we focused on the nonlinear distortion generated by the channel. Nonetheless, the optical channel is not a single source of nonlinear distortions in the fiber-optic communication link, the transceiver devices could also introduce considerable nonlinear distortions [176], which have to be taken into account.

- Third, one can improve the overall performance of the presented end-to-end learning solution by introducing a trainable receiver into it, able to do some nonlinearity equalization. In the presented work, we considered a fixed mismatched Gaussian receiver which performed no nonlinearity compensation, since we focused on learning the constellation shaping. Nonetheless, end-to-end learning the nonlinearity compensation on top of the constellation shaping should have a better performance than learning the nonlinearity compensation only.

4.3 List of publications

1. Neskorniuk, V., Carnio, A., Marsella, D., Turitsyn, S. K., Prilepsky, J. E., & Aref, V., "Memory-aware end-to-end learning of channel distortions in optical coherent communications," *Optics Express*, Under review.

2. Neskorniuk, V., Carnio, A., Marsella, D., Turitsyn, S. K., Prilepsky, J. E., & Aref, V., "Model-Based Deep Learning of Joint Probabilistic and Geometric Shaping for Optical Communication," *2022 Conference on Lasers and Electro-Optics (CLEO)*

3. Neskorniuk, V., Carnio, A., Bajaj, V., Marsella, D., Turitsyn, S. K., Prilepsky, J. E., & Aref, V., "End-to-End Deep Learning of Long-Haul Coherent Optical Fiber Communications via Regular Perturbation Model," *2021 European Conference on Optical Communication (ECOC).*

4. Neskorniuk, V. "Machine learning methods for nonlinearity mitigation in the physical layer of fiber-optic communication links," *2021 European Conference on Optical Communication (ECOC).*

5. Neskorniuk V., Buchali F.,Bajaj V., Turitsyn S. K., Prilepsky J. E. & Aref V., "Neural-Network-Based Nonlinearity Equalizer for 128 GBaud Coherent Transcievers," *2021 Optical Fiber Communications Conference and Exhibition (OFC).*

6. Neskorniuk, V., Freire, P.J., Napoli, A., Spinnler, B., Schairer, W., Prilepsky, J.E., Costa, N. & Turitsyn, S.K., "Simplifying the Supervised Learning of Kerr Nonlinearity Compensation Algorithms by Data Augmentation," *2020 European Conference on Optical Communications (ECOC).*

7. Freire, P.J., Neskornuik, V., Napoli, A., Spinnler, B., Costa, N., Prilepsky, J.E., Riccardi, E. and Turitsyn, S.K., "Experimental Verification of Complex-Valued Artificial Neural

Network for Nonlinear Equalization in Coherent Optical Communication Systems," *2020 European Conference on Optical Communications (ECOC)*.

8. Freire, P.J., Neskornuik, V., Napoli, A., Spinnler, B., Costa, N., Khanna, G., Riccardi, E., Prilepsky, J.E. and Turitsyn, S.K., "Complex-Valued Neural Network Design for Mitigation of Signal Distortions in Optical Links," in *Journal of Lightwave Technology*, vol. 39, no. 6, pp. 1696-1705.

References

[1] Vladislav Neskorniuk, Andrea Carnio, Domenico Marsella, Sergei K Turitsyn, Jaroslaw E Prilepsky, and Vahid Aref. Memory-aware end-to-end learning of channel distortions in optical coherent communications. *Optics Express*, 31(1):1–20, 2023. https://doi.org/10.1364/OE.470154.

[2] Armando Vannucci, Paolo Serena, and Alberto Bononi. The rp method: A new tool for the iterative solution of the nonlinear schrödinger equation. *Journal of Lightwave Technology*, 20(7):1102, 2002.

[3] Vladislav Neskorniuk, Andrea Carnio, Vinod Bajaj, Domenico Marsella, Sergei K Turitsyn, Jaroslaw E Prilepsky, and Vahid Aref. End-to-end deep learning of long-haul coherent optical fiber communications via regular perturbation model. In *2021 European Conference on Optical Communication (ECOC)*, pages 1–4. IEEE, 2021.

[4] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[5] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[6] Andriy Burkov. *The hundred-page machine learning book*, volume 1. Andriy Burkov Quebec City, QC, Canada, 2019.

[7] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[8] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436–444, 2015.

[9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[10] Fei-Fei Li, Jiajun Wu, and Ruohan Gao. Cs231n: Deep learning for computer vision, 2022. URL http://cs231n.stanford.edu/.

[11] Chris Manning. Cs224n: Natural language processing with deep learning, 2022. URL https://web.stanford.edu/class/cs224n/.

[12] Andrew Maas. Cs224s: Spoken language processing, 2022. URL https://web.stanford.edu/class/cs224s/.

[13] Mohamed Ibnkahla. Applications of neural networks to digital communications–a survey. *Signal processing*, 80(7):1185–1215, 2000.

[14] Polina Bayvel, Matthew Brand, Francesco Da Ros, Camille Delezoide, Qirui Fan, Marija Furdek, Christian Häger, Takeshi Hoshida, Luyao Huang, Memedhe Ibrahimi, Ognjen Jovanovic, Boris Karanov, Faisal Nadeem Khan, Toshiaki Koike-Akino, Keisuke Kojima, Alan Pak Tao Lau, Patricia Layec, Zhengxuan Li, Chao Lu, Carlos Natalino, Petros Ramantanis, Cristina Rottondi, Marc Ruiz, Laurent Schmalen, Behnam Shariati, Yingheng Tang, Takahito Tanimura, Massimo Tornatore, Alba P. Vela, Luis Velasco, Ye Wang, Wanting Xu, Yongxin Xu, Metodi Yankov, Lilin Yi, Shaoliang Zhang, and Darko Zibar. Machine learning for future fiber-optic communication systems. In Alan Pak Tao Lau and Faisal Nadeem Khan, editors, *Machine Learning for Future Fiber-Optic Communication Systems*. Academic Press, 2022. ISBN 978-0-323-85227-2. doi: https://doi.org/10.1016/B978-0-32-385227-2.00006-1. URL https://www.sciencedirect.com/science/article/pii/B9780323852272000061.

[15] Faisal Nadeem Khan, Qirui Fan, Chao Lu, and Alan Pak Tao Lau. An optical communication's perspective on machine learning and its applications. *Journal of Lightwave Technology*, 37(2):493–516, 2019.

[16] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[17] A gentle introduction to torch.autograd. https://pytorch.org/docs/stable/autograd.html. Accessed: Jule 28, 2022.

[18] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[19] Osvaldo Simeone et al. A brief introduction to machine learning for engineers. *Foundations and Trends® in Signal Processing*, 12(3-4):200–431, 2018.

[20] Erik Brynjolfsson and Tom Mitchell. What can machine learning do? workforce implications. *Science*, 358(6370):1530–1534, 2017.

[21] Christian Häger and Henry D Pfister. Nonlinear interference mitigation via deep neural networks. In *2018 Optical Fiber Communications Conference and Exposition (OFC)*, pages 1–3. IEEE, 2018.

[22] Cisco. Cisco annual internet report (2018–2023) white paper, 2018. URL https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf.

[23] Peter J Winzer, David T Neilson, and Andrew R Chraplyvy. Fiber-optic transmission and networking: the previous 20 and the next 20 years. *Optics Express*, 26(18): 24190–24239, 2018.

[24] AD Ellis, N Mac Suibhne, D Saad, and DN Payne. Communication networks beyond the capacity crunch, 2016.

[25] Francesco Musumeci, Cristina Rottondi, Avishek Nag, Irene Macaluso, Darko Zibar, Marco Ruffini, and Massimo Tornatore. An overview on application of machine learning techniques in optical networks. *IEEE Communications Surveys & Tutorials*, 21(2):1383–1408, 2018.

[26] Md Saifuddin Faruk and Seb J Savory. Digital signal processing for coherent transceivers employing multilevel formats. *Journal of Lightwave Technology*, 35 (5):1125–1141, 2017.

[27] John G Proakis. *Digital signal processing: principles algorithms and applications*. Pearson Education India, 2001.

[28] Ori Shental and Jakob Hoydis. " machine llrning": Learning to softly demodulate. In *2019 IEEE Globecom Workshops (GC Wkshps)*, pages 1–7. IEEE, 2019.

[29] Tobias Gruber, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. On deep learning-based channel decoding. In *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE, 2017.

[30] Sebastian Cammerer, Tobias Gruber, Jakob Hoydis, and Stephan Ten Brink. Scaling deep learning-based decoding of polar codes via partitioning. In *GLOBECOM 2017-2017 IEEE global communications conference*, pages 1–6. IEEE, 2017.

[31] Daniel Tandler, Sebastian Dörner, Sebastian Cammerer, and Stephan ten Brink. On recurrent neural networks for sequence-based processing in communications. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pages 537–543. IEEE, 2019.

[32] Ezra Ip and Joseph M Kahn. Compensation of dispersion and nonlinear impairments using digital backpropagation. *Journal of Lightwave Technology*, 26(20):3416–3425, 2008.

[33] Christoffer Fougstedt, Christian Häger, Lars Svensson, Henry D Pfister, and Per Larsson-Edefors. Asic implementation of time-domain digital backpropagation with deep-learned chromatic dispersion filters. In *2018 European Conference on Optical Communication (ECOC)*, pages 1–3. IEEE, 2018.

[34] Vinicius Oliari, Sebastiaan Goossens, Christian Hager, Gabriele Liga, Rick M Butler, Menno van den Hout, Sjoerd van der Heide, Henry D Pfister, Chigo M Okonkwo, and Alex Alvarado. Revisiting efficient multi-step nonlinearity compensation with machine learning: An experimental demonstration. *Journal of Lightwave Technology*, pages pp. 3114–3124, 2020.

[35] Christian Häger and Henry D Pfister. Physics-based deep learning for fiber-optic communication systems. *IEEE Journal on Selected Areas in Communications*, 39(1): 280–294, 2020.

[36] S Chen, GJ Gibson, CFN Cowan, and PM Grant. Adaptive equalization of finite non-linear channels using multilayer perceptrons. *Signal processing*, 20(2):107–119, 1990.

[37] Nevio Benvenuto, Michele Marchesi, Francesco Piazza, and Aurelio Uncini. Non linear satellite radio links equalized using blind neural networks. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, pages 1521–1524. IEEE Computer Society, 1991.

[38] Pedro J Freire, Yevhenii Osadchuk, Bernhard Spinnler, Antonio Napoli, Wolfgang Schairer, Nelson Costa, Jaroslaw E Prilepsky, and Sergei K Turitsyn. Performance versus complexity study of neural network equalizers in coherent optical systems. *Journal of Lightwave Technology*, 39(19):6085–6096, 2021.

[39] Changsoo Eun and Edward J Powers. A new volterra predistorter based on the indirect learning architecture. *IEEE transactions on signal processing*, 45(1):223–227, 1997.

[40] Yong Hoon Lim, Yong Soo Cho, Il Whan Cha, and Dae Hee Youn. An adaptive nonlinear prefilter for compensation of distortion in nonlinear systems. *IEEE Transactions on Signal Processing*, 46(6):1726–1730, 1998.

[41] Jaehyeong Kim and K Konstantinou. Digital predistortion of wideband signals based on power amplifier model with memory. *Electronics Letters*, 37(23):1, 2001.

[42] Dennis R Morgan, Zhengxiang Ma, Jaehyeong Kim, Michael G Zierdt, and John Pastalan. A generalized memory polynomial model for digital predistortion of rf power amplifiers. *IEEE Transactions on signal processing*, 54(10):3852–3860, 2006.

[43] Dayong Zhou and Victor E DeBrunner. Novel adaptive nonlinear predistorters based on the direct learning algorithm. *IEEE transactions on signal processing*, 55(1): 120–133, 2006.

[44] Pablo Wilke Berenguer, Markus Nölle, Lutz Molle, Talha Raman, Antonio Napoli, Colja Schubert, and Johannes Karl Fischer. Nonlinear digital pre-distortion of transmitter components. *Journal of lightwave technology*, 34(8):1739–1745, 2015.

[45] Hananel Faig, Yaron Yoffe, Eyal Wohlgemuth, and Dan Sadot. Dimensions-reduced volterra digital pre-distortion based on orthogonal basis for band-limited nonlinear opto-electronic components. *IEEE Photonics Journal*, 11(1):1–13, 2019.

[46] Robert Elschner, Robert Emmerich, Carsten Schmidt-Langhorst, Felix Frey, Pablo Wilke Berenguer, Johannes K Fischer, Helmut Grießer, Danish Rafique, Jörg-Peter Elbers, and Colja Schubert. Improving achievable information rates of 64-gbd pdm-64qam by nonlinear transmitter predistortion. In *Optical Fiber Communication Conference*, pages M1C–2. Optica Publishing Group, 2018.

[47] Ginni Khanna, Bernhard Spinnler, Stefano Calabrò, Erik De Man, and Norbert Hanik. A robust adaptive pre-distortion method for optical communication transmitters. *IEEE Photonics Technology Letters*, 28(7):752–755, 2015.

[48] Yaron Yoffe, Ginni Khanna, Eyal Wohlgemuth, Erik De Man, Bernhard Spinnler, Norbert Hanik, Antonio Napoli, and Dan Sadot. Low-resolution digital pre-compensation enabled by digital resolution enhancer. *Journal of Lightwave Technology*, 37(6): 1543–1551, 2018.

[49] Antonio Napoli, Pablo Wilke Berenguer, Talha Rahman, Ginni Khanna, Mahdi M Mezghanni, Lennart Gardian, Emilio Riccardi, Anna Chiadò Piat, Stefano Calabrò, Stefanos Dris, et al. Digital pre-compensation techniques enabling high-capacity bandwidth variable transponders. *Optics Communications*, 409:52–65, 2018.

[50] Demetri Psaltis, Athanasios Sideris, and Alan A Yamamura. A multilayered neural network controller. *IEEE control systems magazine*, 8(2):17–21, 1988.

[51] A Bernardini, M Carrarini, and S De Fina. The use of a neural net for copeing with nonlinear distortions. In *1990 20th European Microwave Conference*, volume 2, pages 1718–1723. IEEE, 1990.

[52] Tomas Gotthans, Genevieve Baudoin, and Amadou Mbaye. Digital predistortion with advance/delay neural network and comparison with volterra derived models. In *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, pages 811–815. IEEE, 2014.

[53] Yibo Wu, Ulf Gustavsson, Alexandre Graell I Amat, and Henk Wymeersch. Low complexity joint impairment mitigation of i/q modulator and pa using neural networks. *IEEE Journal on Selected Areas in Communications*, 40(1):54–64, 2021.

[54] Chance Tarver, Alexios Balatsoukas-Stimming, and Joseph R Cavallaro. Design and implementation of a neural network based predistorter for enhanced mobile broadband. In *2019 IEEE international workshop on signal processing systems (SiPS)*, pages 296–301. IEEE, 2019.

[55] Reina Hongyo, Yoshimasa Egashira, Thomas M Hone, and Keiichi Yamaguchi. Deep neural network-based digital predistorter for doherty power amplifiers. *IEEE Microwave and Wireless Components Letters*, 29(2):146–148, 2019.

[56] Xin Hu, Zhijun Liu, Xiaofei Yu, Yulong Zhao, Wenhua Chen, Biao Hu, Xuekun Du, Xiang Li, Mohamed Helaoui, Weidong Wang, et al. Convolutional neural network for behavioral modeling and predistortion of wideband power amplifiers. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[57] Jinlong Sun, Wenjuan Shi, Zhutian Yang, Jie Yang, and Guan Gui. Behavioral modeling and linearization of wideband rf power amplifiers using bilstm networks for 5g wireless systems. *IEEE Transactions on Vehicular Technology*, 68(11):10348–10356, 2019.

[58] Slim Boumaiza and Farouk Mkadem. Wideband rf power amplifier predistortion using real-valued time-delay neural networks. In *2009 European Microwave Conference (EuMC)*, pages 1449–1452. IEEE, 2009.

[59] Maximilian Schaedler, Maxim Kuschnerov, Stefano Calabrò, Fabio Pittalà, Christian Bluemm, and Stephan Pachnicke. Ai-based digital predistortion for iq mach-zehnder modulators. In *2019 Asia Communications and Photonics Conference (ACP)*, pages 1–3. IEEE, 2019.

[60] Vinod Bajaj, Fred Buchali, Mathieu Chagnon, Sander Wahls, and Vahid Aref. Deep neural network-based digital pre-distortion for high baudrate optical coherent transmission. *Journal of Lightwave Technology*, 40(3):597–606, 2022.

[61] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[62] Mahmood Abu-Romoh, Stylianos Sygletos, Ian D Phillips, and Wladek Forysiak. Neural-network-based pre-distortion method to compensate for low resolution dac nonlinearity. In *45th European Conference on Optical Communication (ECOC 2019)*, pages 1–4. IET, 2019.

[63] Takeo Sasai, Masanori Nakamura, Etsushi Yamazaki, Asuka Matsushita, Seiji Okamoto, Kengo Horikoshi, and Yoshiaki Kisaka. Wiener-hammerstein model and its learning for nonlinear digital pre-distortion of optical transmitters. *Optics Express*, 28 (21):30952–30963, 2020.

[64] Vinod Bajaj, Fred Buchali, Mathieu Chagnon, Sander Wahls, and Vahid Aref. Single-channel 1.61 tb/s optical coherent transmission enabled by neural network-based digital pre-distortion. In *2020 European Conference on Optical Communications (ECOC)*, pages 1–4. IEEE, 2020.

[65] Vinod Bajaj, Fred Buchali, Mathieu Chagnon, Sander Wahls, and Vahid Aref. 54.5 tb/s wdm transmission over field deployed fiber enabled by neural network-based digital pre-distortion. In *Optical Fiber Communication Conference*, pages M5F–2. Optica Publishing Group, 2021.

[66] Timothy O'Shea and Jakob Hoydis. An introduction to deep learning for the physical layer. *IEEE Transactions on Cognitive Communications and Networking*, 3(4):563–575, 2017.

[67] Timothy J O'Shea, Kiran Karra, and T Charles Clancy. Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention. In *2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 223–228. IEEE, 2016.

[68] Sebastian Dörner, Sebastian Cammerer, Jakob Hoydis, and Stephan Ten Brink. Deep learning based communication over the air. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):132–143, 2017.

[69] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[70] But what is a neural network? https://www.youtube.com/watch?v=aircAruvnKk. Accessed: September 24, 2022.

[71] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[72] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[73] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[74] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993.

[75] Papers with code: Activation functions. https://paperswithcode.com/methods/category/activation-functions. Accessed: September 26, 2022.

[76] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[77] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[78] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[79] Machile learning handbook by yandex school of data analysis (in russian). https://ml-handbook.ru/, 2022. Accessed: September 28, 2022.

[80] Augustin Cauchy et al. Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.

[81] Haskell B Curry. The method of steepest descent for non-linear minimization problems. *Quarterly of Applied Mathematics*, 2(3):258–261, 1944.

[82] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[83] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[84] Pytorch: How to adjust learning rate? https://pytorch.org/docs/stable/optim.html#how-to-adjust-learning-rate, 2022. Accessed: September 29, 2022.

[85] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[86] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[87] Youtube: What is backpropagation really doing? by 3blue1brown. https://www.youtube.com/watch?v=Ilg3gGewQ5U, 2022. Accessed: September 29, 2022.

[88] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[89] René-Jean Essiambre, Gerhard Kramer, Peter J Winzer, Gerard J Foschini, and Bernhard Goebel. Capacity limits of optical fiber networks. *Journal of Lightwave Technology*, 28(4):662–701, 2010.

[90] Frank Gray. Pulse code communication. *United States Patent Number 2632058*, 1953.

[91] Junho Cho, Laurent Schmalen, and Peter J Winzer. Normalized generalized mutual information as a forward error correction threshold for probabilistically shaped qam. In *2017 European Conference on Optical Communication (ECOC)*, pages 1–3. IEEE, 2017.

[92] Georg Böcherer. Achievable rates for probabilistic shaping. *arXiv preprint arXiv:1707.01134*, 2017.

[93] Leszek Szczecinski and Alex Alvarado. *Bit-interleaved coded modulation: fundamentals, analysis and design*. John Wiley & Sons, 2015.

[94] Thomas M Cover and Joy A Thomas. Elements of information theory. *Wiley Series in Telecommunications*, 1991.

[95] G David Forney and L-F Wei. Multidimensional constellations. i. introduction, figures of merit, and generalized cross constellations. *IEEE journal on selected areas in communications*, 7(6):877–892, 1989.

[96] Fred Buchali, Fabian Steiner, Georg Böcherer, Laurent Schmalen, Patrick Schulte, and Wilfried Idler. Rate adaptation and reach increase by probabilistically shaped 64-qam: An experimental demonstration. *Journal of Lightwave Technology*, 34(7):1599–1609, 2016.

[97] J-X Cai, HG Batshon, Matthew V Mazurczyk, Oleg V Sinkin, D Wang, Milen Paskov, W Patterson, Carl R Davidson, P Corbett, G Wolter, et al. 70.4 tb/s capacity over 7,600 km in c+ l band using coded modulation with hybrid constellation shaping and nonlinearity compensation. In *Optical Fiber Communication Conference*, pages Th5B–2. Optical Society of America, 2017.

[98] Jin-Xing Cai, Hussam G Batshon, Matthew V Mazurczyk, Oleg V Sinkin, Ding Wang, Milen Paskov, Carl R Davidson, William W Patterson, Alexey Turukhin, Maxim A Bolshtyansky, et al. 51.5 tb/s capacity over 17,107 km in c+ l bandwidth using single-mode fibers and nonlinearity compensation. *Journal of Lightwave Technology*, 36(11): 2135–2141, 2018.

[99] Ivan Fernandez de Jauregui Ruiz, Amirhossein Ghazisaeidi, Omar Ait Sab, Philippe Plantady, Alain Calsat, Suwimol Dubost, Laurent Schmalen, Vincent Letellier, and Jeremie Renaudier. 25.4-tb/s transmission over transpacific distances using truncated probabilistically shaped pdm-64qam. *Journal of Lightwave Technology*, 36(6):1354–1361, 2018.

[100] Shaoliang Zhang, Fatih Yaman, Yue-Kai Huang, John D Downie, Ding Zou, William A Wood, Aramais Zakharian, Rostislav Khrapko, Snigdharaj Mishra, Vladimir Nazarov, et al. Capacity-approaching transmission over 6375 km at spectral efficiency of 8.3 bit/s/hz. In *Optical Fiber Communication Conference*, pages Th5C–2. Optical Society of America, 2016.

[101] Amirhossein Ghazisaeidi, Ivan Fernandez de Jauregui Ruiz, Rafael Rios-Müller, Laurent Schmalen, Patrice Tran, Patrick Brindel, Alexis Carbo Meseguer, Qian Hu, Fred Buchali, Gabriel Charlet, et al. Advanced c+ l-band transoceanic transmission systems based on probabilistically shaped pdm-64qam. *Journal of Lightwave Technology*, 35 (7):1291–1299, 2017.

[102] Zaishuang Liu, Qiuliang Xie, Kewu Peng, and Zhixing Yang. Apsk constellation with gray mapping. *IEEE communications letters*, 15(12):1271–1273, 2011.

[103] Georg Böcherer, Fabian Steiner, and Patrick Schulte. Bandwidth efficient and rate-matched low-density parity-check coded modulation. *IEEE Transactions on communications*, 63(12):4651–4665, 2015.

[104] Ivan Fernandez de Jauregui Ruiz, Amirhossein Ghazisaeidi, Rafael Rios-Muller, and Patrice Tran. Performance comparison of advanced modulation formats for transoceanic coherent systems. In *2017 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2017.

[105] Andrea Carena, Gabriella Bosco, Vittorio Curri, Yanchao Jiang, Pierluigi Poggiolini, and Fabrizio Forghieri. Egn model of non-linear fiber propagation. *Optics express*, 22 (13):16335–16362, 2014.

[106] Ronen Dar, Meir Feder, Antonio Mecozzi, and Mark Shtaif. Accumulation of nonlinear interference noise in fiber-optic systems. *Optics express*, 22(12):14199–14211, 2014.

[107] Ciena chalk talk - nonlinear probabilistic constellation shaping (pcs) in wavelogic 5 extreme. https://www.ciena.com/insights/videos/nonlinear-probabilistic-constellation-shaping-in-wavelogic-5-extreme.html, 2022. Accessed: October 6, 2022.

[108] Faster, further, smoother: The case for probabilistic constellation shaping. infinera white paper. https://www.infinera.com/white-paper/the-case-for-probabilistic-constellation-shaping/, 2022. Accessed: October 6, 2022.

[109] Alex Alvarado, Erik Agrell, Domaniç Lavery, Robert Maher, and Polina Bayvel. Replacing the soft-decision fec limit paradigm in the design of optical communication systems. *Journal of Lightwave Technology*, 33(20):4338–4352, 2015.

[110] Ronen Dar and Peter J Winzer. Nonlinear interference mitigation: Methods and potential gain. *Journal of Lightwave Technology*, 35(4):903–930, 2017.

[111] John C Cartledge, Fernando P Guiomar, Frank R Kschischang, Gabriele Liga, and Metodi P Yankov. Digital signal processing for fiber nonlinearities. *Optics Express*, 25(3):1916–1936, 2017.

[112] Olga Vassilieva, Inwoong Kim, and Tadashi Ikeuchi. Enabling technologies for fiber nonlinearity mitigation in high capacity transmission systems. *Journal of Lightwave Technology*, 37(1):50–60, 2018.

[113] Darko Zibar, Molly Piels, Rasmus Jones, and Christian G. Schäeffer. Machine learning techniques in optical communication. *Journal of Lightwave Technology*, 34(6):1442–1452, 2016. doi: 10.1109/JLT.2015.2508502.

[114] Boris Karanov, Mathieu Chagnon, Félix Thouin, Tobias A Eriksson, Henning Bülow, Domaniç Lavery, Polina Bayvel, and Laurent Schmalen. End-to-end deep learning of optical fiber communications. *Journal of Lightwave Technology*, 36(20):4843–4855, 2018.

[115] Valey Kamalov, Ljupcho Jovanovski, Vijay Vusirikala, Shaoliang Zhang, Fatih Yaman, Kohei Nakamura, Takanori Inoue, Eduardo Mateo, and Yoshihisa Inada. Evolution from 8qam live traffic to ps 64-qam with neural-network based nonlinearity compensation on 11000 km open subsea cable. In *Optical Fiber Communication Conference*, pages Th4D–5. Optical Society of America, 2018.

[116] Toshiaki Koike-Akino, Ye Wang, David S Millar, Keisuke Kojima, and Kieran Parsons. Neural turbo equalization: Deep learning for fiber-optic nonlinearity compensation. *Journal of Lightwave Technology*, 38(11):3059–3066, 2020.

[117] Qirui Fan, Gai Zhou, Tao Gui, Chao Lu, and Alan Pak Tao Lau. Advancing theoretical understanding and practical performance of signal processing for nonlinear optical communications through machine learning. *Nature Communications*, 11(3694):3694, 2020.

[118] Mahdi Malekiha, Igor Tselniker, and David V Plant. Efficient nonlinear equalizer for intra-channel nonlinearity compensation for next generation agile and dynamically reconfigurable optical networks. *Optics express*, 24(4):4097–4108, 2016.

[119] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[120] Shaoliang Zhang, Fatih Yaman, Kohei Nakamura, Takanori Inoue, Valey Kamalov, Ljupcho Jovanovski, Vijay Vusirikala, Eduardo Mateo, Yoshihisa Inada, and Ting Wang. Field and lab experimental demonstration of nonlinear impairment compensation using neural networks. *Nature communications*, 10(1):1–8, 2019.

[121] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[122] Lihua Cui, Yongli Zhao, Boyuan Yan, Dongmei Liu, and Jie Zhang. Deep-learning-based failure prediction with data augmentation in optical transport networks. In *17th International Conference on Optical Communications and Networks (ICOCN 2018)*, volume 11048, page 110482I. International Society for Optics and Photonics, 2019.

[123] Haotao Zhuang, Yongli Zhao, Xiaosong Yu, Yajie Li, Ying Wang, and Jie Zhang. Machine-learning-based alarm prediction with gans-based self-optimizing data augmentation in large-scale optical transport networks. In *2020 International Conference on Computing, Networking and Communications (ICNC)*, pages 294–298. IEEE, 2020.

[124] Shuai Li, Jin Li, Min Zhang, Danshi Wang, Chuang Song, and Xinghua Zhen. Adaptive traffic data augmentation using generative adversarial networks for optical networks. In *2019 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2019.

[125] Jin Li, Danshi Wang, Shuai Li, Min Zhang, Chuang Song, and Xue Chen. Deep learning based adaptive sequential data augmentation technique for the optical network traffic synthesis. *Optics Express*, 27(13):18831–18847, 2019.

[126] Vladislav Neskorniuk, Pedro J Freire, Antonio Napoli, Bernhard Spinnler, Wolfgang Schairer, Jaroslaw E Prilepsky, Nelson Costa, and Sergei K Turitsyn. Simplifying the supervised learning of kerr nonlinearity compensation algorithms by data augmentation. In *2020 European Conference on Optical Communications (ECOC)*, pages 1–4. IEEE, 2020.

[127] Vladislav Neskorniuk, Pedro J. Freire, Antonio Napoli, Bernhard Spinnler, Wolfgang Schairer, Jaroslaw E. Prilepsky, Nelson Costa, and Sergei K. Turitsyn. The example codes and numerical traces supplemented for article "data augmentation for the supervised learning of nonlinearity compensation algorithms in optical communications". https://doi.org/10.17036/researchdata.aston.ac.uk.00000481, September 2020. URL https://doi.org/10.17036/researchdata.aston.ac.uk.00000481.

[128] A. Redyuk, E. Averyanov, O. Sidelnikov, M. Fedoruk, and S. Turitsyn. Compensation of nonlinear impairments using inverse perturbation theory with reduced complexity. *Journal of Lightwave Technology*, 38(6):1250–1257, 2020.

[129] Zhenning Tao, Liang Dou, Weizhen Yan, Lei Li, Takeshi Hoshida, and Jens C Rasmussen. Multiplier-free intrachannel nonlinearity compensating algorithm operating at symbol rate. *Journal of Lightwave Technology*, 29(17):2570–2576, 2011.

[130] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, pages 971–980, 2017.

[131] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[132] Govind P Agrawal. *Nonlinear Fiber Optics*. Academic Press, Boston, fifth edition, 2013. doi: https://doi.org/10.1016/B978-0-12-397023-7.00002-4. URL http://www.sciencedirect.com/science/article/pii/B9780123970237000024.

[133] Corning Inc. Interoperability of corning leaf and corning metrocor fiber in metropolitan networks, 2019. URL https://www.corning.com/content/dam/corning/media/worldwide/coc/documents/Fiber/application-notes/WP8205.pdf.

[134] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

[135] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[136] Tobias A Eriksson, Henning Bülow, and Andreas Leven. Applying neural networks in optical communication systems: possible pitfalls. *IEEE Photonics Technology Letters*, 29(23):2091–2094, 2017.

[137] Tomofumi Oyama, Takeshi Hoshida, Hisao Nakashima, Takahito Tanimura, Zhenning Tao, and Jens C Rasmussen. Proposal of improved 16qam symbol degeneration method for simplified perturbation-based nonlinear equalizer. In *2014 OptoElectronics and Communication Conference and Australian Conference on Optical Fibre Technology*, pages 941–943. IEEE, 2014.

[138] Jack E Volder. The cordic trigonometric computing technique. *IRE Transactions on electronic computers*, 3:330–334, 1959.

[139] Lidia Galdino, Daniel Semrau, Domaniç Lavery, Gabriel Saavedra, Cristian B Czegledi, Erik Agrell, Robert I Killey, and Polina Bayvel. On the limits of digital back-propagation in the presence of transceiver noise. *Optics Express*, 25(4): 4564–4578, 2017.

[140] Vinícius Oliari, Erik Agrell, and Alex Alvarado. Regular perturbation on the group-velocity dispersion parameter for nonlinear fibre-optical communications. *Nature Communications*, 11(933):933, 2020.

[141] Erik Agrell, Alex Alvarado, and Frank R. Kschischang. Implications of information theory in optical fibre communications. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2062):20140438, 2016. doi: 10.1098/rsta.2014.0438. URL https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2014.0438.

[142] Francesco Musumeci, Cristina Rottondi, Avishek Nag, Irene Macaluso, Darko Zibar, Marco Ruffini, and Massimo Tornatore. An overview on application of machine learning techniques in optical networks. *IEEE Communications Surveys & Tutorials*, 21(2):1383–1408, 2019. doi: 10.1109/COMST.2018.2880039.

[143] Josh W. Nevin, Sam Nallaperuma, Nikita A. Shevchenko, Xiang Li, Md. Saifuddin Faruk, and Seb J. Savory. Machine learning for optical fiber communication systems: An introduction and overview. *APL Photonics*, 6(12):121101, 2021. doi: 10.1063/5.0070838.

[144] Pedro J. Freire, Antonio Napoli, Bernhard Spinnler, Nelson Costa, Sergei K. Turitsyn, and Jaroslaw E. Prilepsky. Neural networks-based equalizers for coherent optical transmission: Caveats and pitfalls. *IEEE Journal of Selected Topics in Quantum Electronics*, 28(4):1–23, 2022. doi: 10.1109/JSTQE.2022.3174268.

[145] Pedro J Freire, Daniel Abode, Jaroslaw E Prilepsky, and Sergei K Turitsyn. Power and modulation format transfer learning for neural network equalizers in coherent optical transmission systems. In *Signal Processing in Photonic Communications*, pages SpM5C–6. Optical Society of America, 2021.

[146] Xiao Chen, Julian Cheng, Zaichen Zhang, Liang Wu, Jian Dang, and Jiangzhou Wang. Data-rate driven transmission strategies for deep learning-based communication systems. *IEEE Transactions on Communications*, 68(4):2129–2142, 2020. doi: 10.1109/TCOMM.2020.2968314.

[147] Jinxiang Song, Christian Häger, Jochen Schröder, Alexandre Graell I Amat, and Henk Wymeersch. Model-based end-to-end learning for wdm systems with transceiver hardware impairments. *IEEE Journal of Selected Topics in Quantum Electronics*, 28 (4):1–14, 2022.

[148] Boris Karanov, Domaniç Lavery, Polina Bayvel, and Laurent Schmalen. End-to-end optimized transmission over dispersive intensity-modulated channels using bidirectional recurrent neural networks. *Optics express*, 27(14):19650–19663, 2019.

[149] Boris Karanov, Mathieu Chagnon, Vahid Aref, Domaniç Lavery, Polina Bayvel, and Laurent Schmalen. Concept and experimental demonstration of optical im/dd end-to-end system optimization using a generative model. In *Optical Fiber Communication Conference*, pages Th2A–48. Optica Publishing Group, 2020.

[150] Boris Karanov, Vinícius Oliari, Mathieu Chagnon, Gabriele Liga, Alex Alvarado, Vahid Aref, Domaniç Lavery, Polina Bayvel, and Laurent Schmalen. End-to-end learning in optical fiber communications: Experimental demonstration and future trends. In *2020 European Conference on Optical Communications (ECOC)*, pages 1–4. IEEE, 2020.

[151] Simone Gaiarin, Francesco Da Ros, Rasmus T Jones, and Darko Zibar. End-to-end optimization of coherent optical communications over the split-step fourier method guided by the nonlinear fourier transform theory. *Journal of Lightwave Technology*, 39(2):418–428, 2020.

[152] Rasmus T Jones, Tobias A Eriksson, Metodi P Yankov, and Darko Zibar. Deep learning of geometric constellation shaping including fiber nonlinearities. In *2018 European Conference on Optical Communication (ECOC)*, pages 1–3. IEEE, 2018.

[153] Rasmus T Jones, Metodi P Yankov, and Darko Zibar. End-to-end learning for gmi optimized geometric constellation shape. In *2019 European Conference on Optical Communication (ECOC)*, pages 1–3. IET, 2019.

[154] Kadir Gümüş, Alex Alvarado, Bin Chen, Christian Häger, and Erik Agrell. End-to-end learning of geometrical shaping maximizing generalized mutual information. In *2020 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2020.

[155] Vinícius Oliari, Boris Karanov, Sebastiaan Goossens, Gabriele Liga, Olga Vassilieva, Inwoong Kim, Paparao Palacharla, Chigo Okonkwo, and Alex Alvarado. High-cardinality hybrid shaping for 4d modulation formats in optical communications optimized via end-to-end learning. *arXiv preprint arXiv:2112.10471*, 2021.

[156] Ognjen Jovanovic, Metodi P Yankov, Francesco Da Ros, and Darko Zibar. End-to-end learning of a constellation shape robust to variations in snr and laser linewidth. In *2021 European Conference on Optical Communication (ECOC)*, pages 1–4. IEEE, 2021.

[157] Jinxiang Song, Christian Häger, Jochen Schröder, Alexandre Graell i Amat, and Henk Wymeersch. End-to-end autoencoder for superchannel transceivers with hardware impairment. In *Optical Fiber Communication Conference*, pages F4D–6. Optica Publishing Group, 2021.

[158] Zonglong He, Jinxiang Song, Christian Häger, Alexandre Graell i Amat, Henk Wymeersch, Peter A Andrekson, Magnus Karlsson, and Jochen Schröder. Experimental demonstration of learned pulse shaping filter for superchannels. In *Optical Fiber Communication Conference*, pages W2A–33. Optica Publishing Group, 2022.

[159] Vinod Bajaj, Mathieu Chagnon, Sander Wahls, and Vahid Aref. Efficient training of volterra series-based pre-distortion filter using neural networks. In *2022 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2022.

[160] Tim Uhlemann, Sebastian Cammerer, Alexander Span, Sebastian Dörner, and Stephan ten Brink. Deep-learning autoencoder for coherent and nonlinear optical communication. In *Photonic Networks; 21th ITG-Symposium*, pages 1–8. VDE, 2020.

[161] Vahid Aref and Mathieu Chagnon. End-to-end learning of joint geometric and probabilistic constellation shaping. In *2022 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2022.

[162] Tobias Fehenberger, Alex Alvarado, Georg Böcherer, and Norbert Hanik. On probabilistic shaping of quadrature amplitude modulation for the nonlinear fiber channel. *Journal of Lightwave Technology*, 34(21):5063–5073, 2016.

[163] Frank R Kschischang and Subbarayan Pasupathy. Optimal nonuniform signaling for gaussian channels. *IEEE Trans. Inf. Theory*, 39(3):913–929, 1993.

[164] Vladislav Neskorniuk, Andrea Carnio, Domenico Marsella, Sergei K Turitsyn, Jaroslaw E Prilepsky, and Vahid Aref. Model-based deep learning of joint probabilistic and geometric shaping for optical communication. *arXiv preprint arXiv:2204.07457*, 2022.

[165] Govind P Agrawal. Nonlinear fiber optics. In *Nonlinear Science at the Dawn of the 21st Century*, pages 195–211. Springer, 2000.

[166] Shen Li, Christian Häger, Nil Garcia, and Henk Wymeersch. Achievable information rates for nonlinear fiber communication via end-to-end autoencoder learning. In *2018 European Conference on Optical Communication (ECOC)*, pages 1–3. IEEE, 2018.

[167] Rasmus T Jones, Tobias A Eriksson, Metodi P Yankov, Benjamin J Puttnam, Georg Rademacher, Ruben S Luis, and Darko Zibar. Geometric constellation shaping for fiber optic communication systems via end-to-end learning. *arXiv preprint arXiv:1810.00774*, 2018.

[168] Antonio Mecozzi and René-Jean Essiambre. Nonlinear shannon limit in pseudolinear coherent systems. *Journal of Lightwave Technology*, 30(12):2011–2024, 2012.

[169] Francisco Javier Garcia Gomez and Gerhard Kramer. Mismatched models to lower bound the capacity of dual-polarization optical fiber channels. *Journal of Lightwave Technology*, 2021.

[170] Neri Merhav, Gideon Kaplan, Amos Lapidoth, and S Shamai Shitz. On information rates for mismatched decoders. *IEEE Transactions on Information Theory*, 40(6): 1953–1967, 1994.

[171] Metodi P Yankov, Francesco Da Ros, Edson P da Silva, Søren Forchhammer, Knud J Larsen, Leif K Oxenløwe, Michael Galili, and Darko Zibar. Constellation shaping for wdm systems using 256qam/1024qam with probabilistic optimization. *Journal of Lightwave Technology*, 34(22):5146–5156, 2016.

[172] Shaoliang Zhang and Fatih Yaman. Design and comparison of advanced modulation formats based on generalized mutual information. *Journal of Lightwave Technology*, 36(2):416–423, 2017.

[173] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(Mar):1157–1182, 2003.

[174] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[175] A fast and accurate state-of-the-art bivariate kernel density estimator with diagonal bandwidth matrix. https://uk.mathworks.com/matlabcentral/fileexchange/17204-kernel-density-estimation. Accessed: August 10, 2022.

[176] Vladislav Neskorniuk, Fred Buchali, Vinod Bajaj, Sergei K Turitsyn, Jaroslaw E Prilepsky, and Vahid Aref. Neural-network-based nonlinearity equalizer for 128 gbaud coherent transcievers. In *2021 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2021.