Machine Learning-Based Characterization of SNR in Digital Satellite Communication Links

Brecht Dhuyvetters*, Daniel Delaruelle†, Hendrik Rogier*, Tom Dhaene*, Dries Vande Ginste*, Domenico Spina*
*IDLab, Department of Information Technology, Ghent University - imec,
Technologiepark-Zwijnaarde 126, 9052 Ghent, Belgium
†ST Engineering iDirect (Europe) NV Cy, Laarstraat 5, 9100 Sint-Niklaas, Belgium
domenico.spina@ugent.be

***Abstract*—Signals traveling through a Satellite Communication (SatCom) channel are subject to noise and interference effects, impacting their Signal-to-Noise ratio (SNR). Furthermore, nonlinear distortion arising from the nonlinear characteristic of the amplifiers in the system also adversely impacts performance. Current state-of-the-art techniques estimate these effects by including a sequence of known pilot symbols in the transmitted signals. While robust, a downside of these approaches is that pilot symbols do not include useful information, thus introducing overhead. This paper presents a Machine Learning (ML) approach to characterize the SNR, using the received signal in the return link of SatCom systems, independent of the signal's distortion level and without relying on pilot symbols. The proposed technique is validated through a suitable application example: the characterization of SNR in a SatCom system using a 16-APSK modulation scheme.**

I. INTRODUCTION

Satellite communication (SatCom) has become ubiquitous in our current society. Satellites are used in almost all communication systems around us: broadcast TV, videoconferencing, aeronautical and maritime communications, to name but a few. One important field where SatCom is rapidly gaining momentum is the cellular communications sector. Historically, mobile service providers have been reluctant to integrate SatCom as part of their cell backhaul due to the delay-introducing characteristic and lower-bandwidth of SatCom systems compared to terrestrial systems. However, these limitations are now being overcome owing to the many technological advancements in the sector, such as High Throughput Satellites (HTS), data-prefetching and higher-order modulation schemes [1]. Mobile operators are therefore ready to embrace SatCom next to the existing terrestrial solutions.

However, using higher-order modulation schemes in modern SatCom technologies also brings a few new challenges. Due to their high Peak-To-Average Power Ratio (PAPR), they are very sensitive to nonlinear effects, induced, e.g., by the nonlinear characteristic of High-Power-Amplifiers (HPAs) onboard the transmitting user terminal. Furthermore, frequency interference and thermal noise also have a negative impact on the satellite-terrestrial radio system and need to be accounted for. A correction mechanism typically applied at the transmitter is called predistortion [2]. As the name suggests, the signal is predistorted using an approximation of the inverse amplifier input-output characteristic and channel non-idealities. Other techniques, which cancel out the non-idealities at the receiver side, such as symbol-based equalization, are able to provide significantly better performance and energy efficiency. However, these techniques rely on the knowledge of the channel's characteristics. In practice, acquiring this information is not only difficult, it also introduces overhead [3].

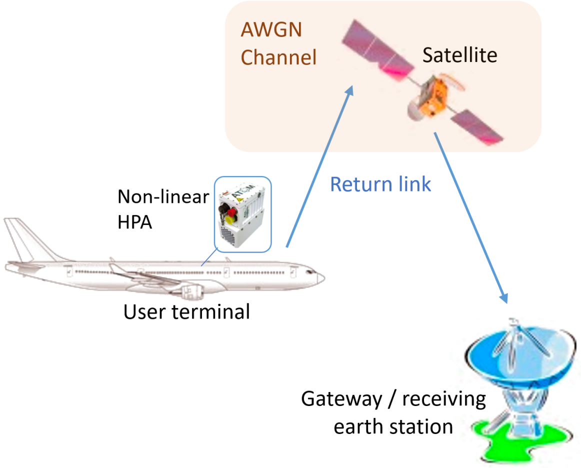

Fig. 1. Overview of the SatCom system return link.

This paper focuses on the characterization of the SNR in the satellite return link, which is of paramount importance to ensure the desired performance of modern SatCom systems. Figure 1 illustrates the system under study with the relevant nonideal components, where the satellite channel is modeled as an additive white Gaussian noise (AWGN) channel. In particular, we aim at characterizing the SNR via Machine Learning (ML) methods, without making use of pilot symbols and *independent* of the distortion level of the signal arising from the nonlinear characteristic of the HPA. Hence, only the raw and unknown received symbols at the gateway are to be used to quantify the signal's noise level. Reaching this goal independent from the distortion level of the signal is challenging, but of paramount importance, since, in practice, the distortion level introduced by HPAs is not known upfront.

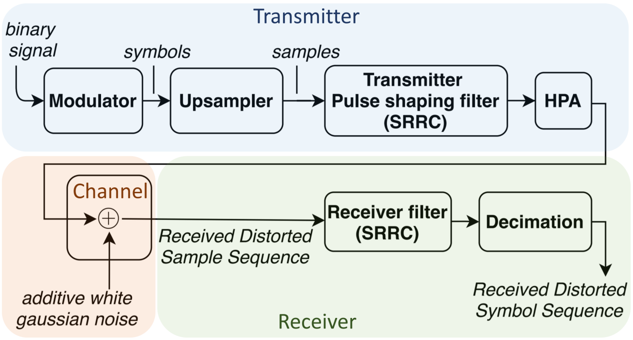
Transmitter
binary signal
symbols
samples
Modulator
Upsampler
Transmitter Pulse shaping filter (SRRC)
HPA
Channel
Received Distorted Sample Sequence
Receiver filter (SRRC)
Decimation
additive white gaussian noise
Receiver
Received Distorted Symbol Sequence

Fig. 2. Overview of the SatCom system's model in MATLAB.

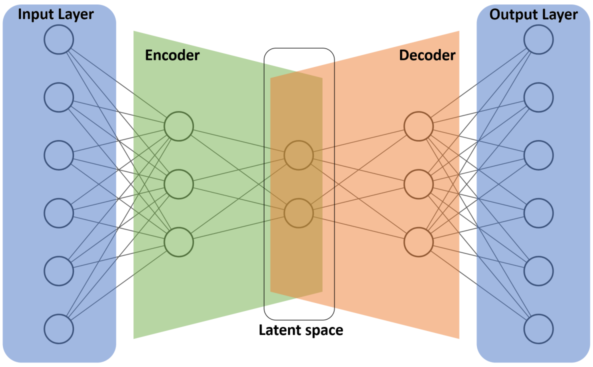

Fig. 3. General architecture of an autoencoder.

In this framework, we first build a MATLAB[1] model of the SatCom system in Fig. 1, which is described in detail in Section II. This model is used to generate the required signals for the training and testing of our ML-based technique. Next, Section III describes how the proposed ML approach can be employed to quantify the SNR, *independent* of the distortion level. A relevant application example, based on a 16-APSK modulation scheme, is presented in Section IV, while conclusions are drawn in Section V.

II. Modeling the SatCom System

In order to characterize the non-idealities of the system based on the received signals, a reliable model of the SatCom system is necessary. First we give a general overview of the system and its components, then we discuss in detail the characteristics of the HPA and AWGN channel.

A. *Overview of the Communication Framework*

The information from the user terminal, represented as a sequence of random bits, is first modulated using a 16-APSK modulation scheme. The resulting complex baseband signal, consisting of a sequence of complex-valued symbols, is then upsampled with an oversampling rate of 16 and pulse shaped, in this case using a Square-Root Raised Cosine (SRRC) filter. Another SRRC filter, which is matched to the transmitter filter (i.e., its complex conjugate), is used at the receiver. After pulse shaping, the last component in the transmitter is the HPA, which introduces distortion into the signal due to its non-linear input-output characteristic. The signal is consequently sent through the AWGN channel, corrupting the signal with Gaussian noise, to finally arrive at the receiver filter. Here it is again filtered and finally decimated, to obtain the received distorted symbol sequence. Figure 2 schematically shows the architecture of the SatCom system under study.

B. *HPA & AWGN channel*

For the HPA, which is the last component in the transmitter, the Amplitude-to-Amplitude (AM-to-AM) and Amplitude-to-Phase (AM-to-PM) characteristics were sampled from a 3W solid state Ka-band amplifier, and saved into a tabular dataset in MATLAB. The power of the generated input signal is such that, for an input-backoff (IBO) of 0dB, the average signal power is equal to the input saturation power of the amplifier, as desired. Higher IBO levels result in more backoff from the saturation point, and thus less distortion.

The AWGN block is modeled by generating a sequence of Gaussian noise samples that are added to the signal, in such a way that the desired SNR is obtained. The noise power is equally divided over the real and complex parts of the samples. Now, the distorted and noisy sample sequence reaches the receiving gateway station. The ML model discussed in the sequel employs the received signal at this point, i.e., the received sequence of samples before sending it through the receiver filter. This signal better resembles an analog signal, due to the pulse shaping operation at the transmitter, and this is beneficial for the considered ML-based approach.

III. Methodology

A two-step procedure is proposed to quantify the SNR of the received signal. First, a suitable autoencoder (AE) is trained using the signals generated by the SatCom model. An overview on AEs and their application in our problem set-up is given in Section III-A, while the architecture of the specific AE used in this application is presented in Section IV. Second, in Section III-B we define three different metrics that relate the output of the AE to the signal's SNR, and investigate which one provides the best performance in Section IV.

A. *Autoencoders for SNR estimation*

AEs are Neural Networks with the same input and output dimensions, which are trained to compress the input into a lower-dimensional space (also called latent space), in such a way that the input can again be recovered from the latent space with minimal error [4]. In the literature, AEs have been used for several applications, including dimensionality reduction [5] and detection of errors and anomalies [6]–[8].

Figure 3 shows the structure of a generic AE, which is formed by two parts: the encoder and decoder. The first is given by the layers reducing the dimensionality of the input,

[1] The Mathworks Inc., Natick, MA, USA.

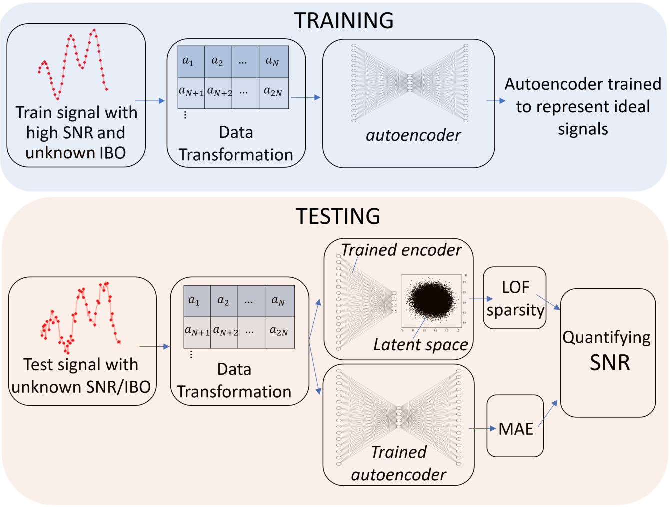
TRAINING
Train signal with high SNR and unknown IBO
a_1 a_2 ... a_N
a_{N+1} a_{N+2} ... a_{2N}
Data Transformation
autoencoder
Autoencoder trained to represent ideal signals
TESTING
Test signal with unknown SNR/IBO
a_1 a_2 ... a_N
a_{N+1} a_{N+2} ... a_{2N}
Data Transformation
Trained encoder
Latent space
LOF sparsity
Trained autoencoder
MAE
Quantifying SNR

Fig. 4. Schematic overview of the proposed approach for quantifying the SNR in received signals.

while the layers responsible for reconstructing the original input form the decoder. It is important to note that both encoder and decoder typically are nonlinear functions of the inputs, which is achieved by the use of nonlinear activation functions at the node outputs [4].

In our problem set-up, first a suitable AE is trained to reconstruct "ideal" received signals, i.e., signals with high SNR. For this application, a SNR of 40dB is used during the training phase. Then, during the testing phase, feeding unknown signals with different ranges of SNR to the AE will yield different latent space representations and higher reconstruction errors than during the training phase, as the AE was not trained to reconstruct them. By defining suitable metrics based on the density of the latent space or on the reconstruction error, as shown in Section III-B, it is possible to accurately estimate the SNR of unknown received symbol sequences. This methodology is summarized in Fig. 4, where the data transformation step refers to a suitable preprocessing of the received signal, which is described in Section IV.

However, noise is not the only non-ideal characteristic of received signals, the distortion caused by the HPA must be taken into account as well. Hence, the training set must include signals with a wide range of different distortion levels. For this application, the IBO varies between -20dB and 15dB, while the SNR is kept fixed at 40dB. In this way, the AE will learn to reconstruct signals with different distortion levels. Therefore, after training, a degradation of the reconstruction performance will depend mainly on the SNR's variations and not on the different distortion levels. Since the training set includes a broad range of possible signals, which the AE must be able to reconstruct with sufficient and comparable precision, we decided to use convolutional layers instead of fully connected layers for the AE. This particular architecture is called convolutional autoencoder (CAE). The convolutional layers allow for extracting time-shift invariant temporal characteristics of the signal under study (also called features in a ML framework), and can learn more complex functions of the input without significantly increasing the model's complexity, due to the weight sharing nature of convolutional kernels [9]. This means that one set of weights (forming a kernel) is used (slides) over the entire input, allowing the CAE to better handle the broader training dataset.

B. Metrics

We now discuss the three different algorithms, or metrics, to relate the CAE's output to the SNR value.

The first algorithm is the Local Outlier Factor (LOF), which is applied to the latent space representation. The LOF compares the local density of each point in the latent space to the local density of its k nearest neighbors [10] in order to individuate outliers. Typically, this algorithm is adopted in combination with AEs in the framework of anomaly detection problems [7], where the goal is to identify infrequent deviant events, which do not conform to an expected behavior. The main assumption of the LOF is that deviant events lie in areas of lower density of the latent space compared to normal samples. As a result, the more relevant the difference from the "normal" behaviour, the higher is the corresponding score given by the LOF algorithm. However, in our problem set-up the noise constantly corrupts the received signal, rather than occasionally causing an unexpected behaviour, as in anomaly detection problems. Therefore, rather than using directly the score provided by the LOF, the standard deviation of the LOF's score over an entire symbol sequence is used to estimate the SNR of the received signal.

The second metric is the sparsity of the latent space. Contrary to the LOF algorithm, which compares local densities, the sparsity metric here defined only considers the absolute distances of each point in the latent space to its k nearest neighbors, and averages out these distances over all points. In particular, the sparsity of the latent space can be expressed as

$$\text{sparsity} = \frac{\sum\limits_{X=1}^{N} \left. \frac{\sum_Y |X-Y|}{|\mathbb{H}_Y|} \right|_{Y \in \mathbb{H}_Y}}{N}, \tag{1}$$

where X and Y are two points in the latent space, $\mathbb{H}_Y$ the set of k nearest neighbors of Y and N is the total number of points in the latent space. Hence, it is a measure of the absolute sparsity of the latent space: the more distance between the points, on average, the greater the sparsity value.

The last metric adopted is the mean absolute error (MAE) between the input and output of the CAE. Therefore, this metric does not rely on the latent space representation, but only considers the reconstruction error.

Applying the metrics here defined to the latent space or the output of the CAE, as appropriate, constitutes the final step of the proposed methodology, as shown in Fig. 4.

IV. Application Example

In order to feed data to a CAE, the signals are first preprocessed by standardizing them, such that their mean and

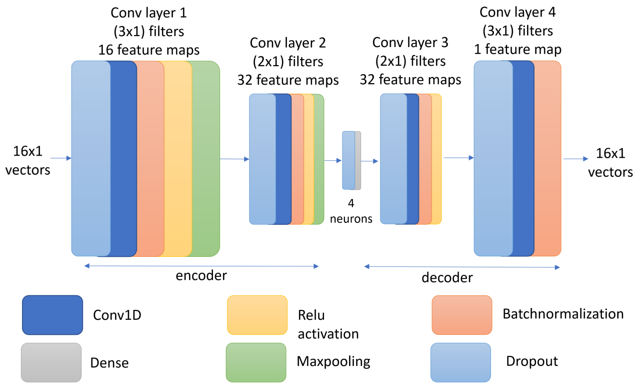

Fig. 5. Architecture of the CAE for quantifying the SNR.

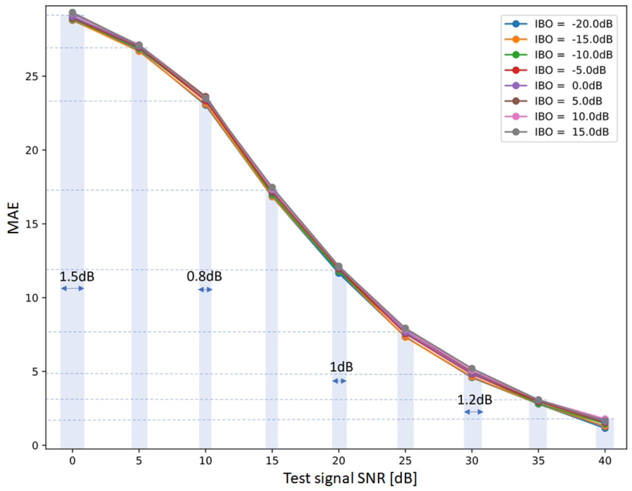

Fig. 6. MAE as a function of the test signal's SNR for varying IBOs, generated using seed = 1. Each discrete point represents a test signal with specific SNR and IBO.

SNR region	Deviation in MAE due to varying seeds	Resulting error in SNR prediction due to varying seeds	Final SNR estimating accuracy
0dB – 10dB	0.34	1.0dB	- Accurate to 2.5dB
10dB – 20dB	0.17	0.3dB	- Accurate to 1.1dB
20dB – 30dB	0.22	0.6dB	- Accurate to 1.6dB
30dB – 40dB	0.30	1.5dB	- Accurate to 2.7dB

TABLE I
IMPACT OF VARYING SEEDS ON THE SNR ESTIMATION

standard deviation are equal to zero and one, respectively, and then are split into sub-sequences. This corresponds to the data transformation step in Fig. 4. In this application, each sub-sequence is a vector of size 16, which is equal to the oversampling rate of the adopted modulation scheme, as described in Section II-A. In particular, each signal used for training, generated using a specific combination of SNR and IBO values, contains 2^{20} samples, and is thus split up into 2^{16} input vectors. It is important to remark that only the real part of the signal was fed to the CAE. Since the noise equally affects both real and imaginary parts of the received signal, as described in Section II-B, by adopting only the real (or the imaginary) part to detect the SNR it is possible to reduce the complexity of the CAE, without sacrificing the accuracy in the model's prediction.

The architecture of the final, tuned CAE is shown in Fig. 5. It consists of four convolutional layers and one intermediate dense layer of four neurons. Dropout layers were added for regularization. The input and output consist of 16 neurons, corresponding to the length of each input sub-sequence. Note that each input represents one symbol duration.

During the training phase, the Gradient Descent optimization algorithm [11] is to used to modify the model parameters in order to minimize a suitable loss function. The contractive loss is adopted in this contribution, which is a variation of the Mean Squared Error (MSE) loss function, adding a penalty term which causes the latent space to contract [8]. This allows the CAE to better handle small perturbations in the data during training. The CAE is trained for 120 epochs, using a learning rate of $8 \cdot 10^{-4}$ and batch size of 64 for the Gradient Descent algorithm.

New and unknown signals (indicated as test signals in the following) can now be fed to the trained CAE, and the metrics discussed in the previous section can be used to quantify the SNR. Specifically, sequences are generated with IBOs between -20dB (high distortion) and 15dB (no distortion), and SNRs varying between 0dB and 40dB (in steps of 5dB). In order to verify the robustness of our approach to the different symbol sequence transmitted, 10 different signals are generated with a different seed of the random number generator for each combination of SNR and IBO. This results in a total of 720 sample sequences (9 SNR levels, 8 IBO levels, 10 seeds per level). The outcome of the metrics for each sequence is consequently plotted as a function of the SNR, as shown in Fig. 6 for the MAE metric. In particular, the results in Fig. 6 are obtained for a seed equal to one and varying IBO levels. It can be seen that the MAE varies almost linearly with respect to the SNR for a large range of values, and the test signal's SNR can be accurately predicted. The variability in the outcome due to varying IBO levels, although small, causes some loss in precision when predicting the SNR of a sequence. This is indicated with blue bars in the figure. By interpolating between the discrete points, it is possible to predict the SNR over a continuous range of values between 0 and 40dB.

Using other seeds for the random number generator yielded very similar results, indicating that the proposed method is robust to the varying information content in the received signal. After taking into account the impact of the varying seeds, the final prediction accuracy is obtained: the results for the MAE are summarized in Table I. Note that the last column in the table is obtained by adding up the loss in precision due to both the variability in sequences and unknown distortion levels. Four main regions can be distinguished, each

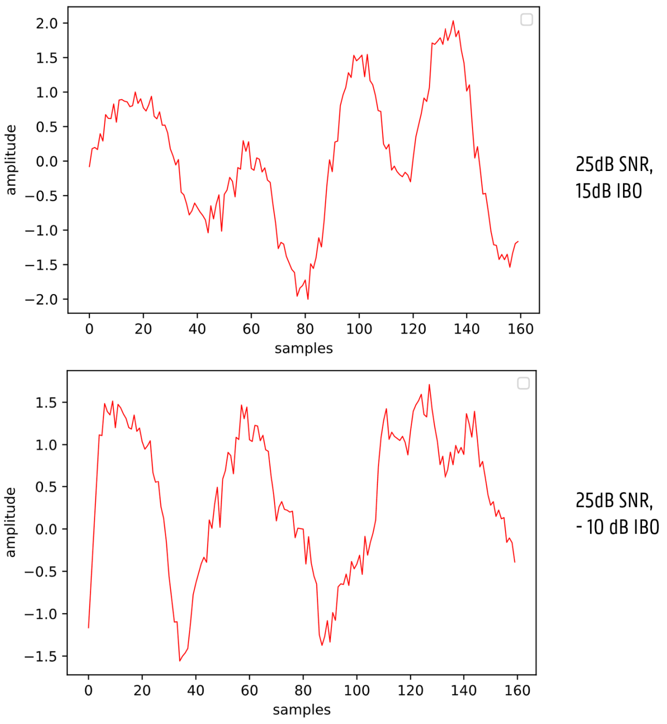

Fig. 7. Real parts of two signals containing the same information content, but with different distortion levels.

Metric	Robustness to signal randomness	IBO quantifying accuracy per SNR region			
		0-10dB	10-20dB	20-30dB	30-40dB
LOF (stdev)	Average	2.6dB	2.3dB	2.3dB	Not usable
Sparsity	Good	2.4dB	1.8dB	1.8dB	3.1dB
MAE	Good	2.5dB	1.1dB	1.6dB	2.7dB

TABLE II
ACCURACY OF SNR ESTIMATION FOR THE CONSIDERED METRICS

represented by a row of the Table I, and the accuracy is the highest for an SNR between 10dB and 30dB. Indeed, for an SNR above 30dB the corresponding reconstruction error is low (the test signal is similar to the training signal), reducing the precision in estimating the SNR; while for an SNR close to 0dB the signal is extremely corrupted by noise, thus reducing the accuracy of the model's predictions.

As an example, Fig. 7 shows the real parts of two signals representing the same symbol sequence with identical SNR, but with very different distortion levels. The SNR and IBO are considered unknown, and we want to estimate the SNR of the signals with the proposed technique. Feeding both signals to the CAE results in MAEs of 7.43 and 7.61 for the -10dB and 15dB IBO signals, respectively. By looking at Fig. 6, this indeed corresponds to an SNR of about 25dB, and according to Table I this estimate is accurate within a margin of 1.6dB.

The results for all metrics are summarized in Table II. The number of k nearest neighbours for the LOF and sparsity metrics providing the best results is not known upfront: a heuristic approach has been used, leading to $k = 800$ for the LOF and $k = 200$ for the sparsity. Note that the standard deviation of the LOF does not show significant changes for an SNR between 30dB and 40dB, making it impossible to estimate the SNR in that region. It can be seen that the MAE yields the best results among all metrics, with the exception of an SNR between 0dB and 10dB where the sparsity metric is marginally more accurate.

V. Conclusions

A ML-based characterization of the SNR in a Satellite Communication System is described in this contribution. The proposed method represents an alternative to current state-of-the-art approaches, which mainly rely on expert systems and dedicated pilot symbols in the waveform. It is robust to the nonlinear characteristic of the HPA onboard the transmitting user terminal. The proposed method is based on training a suitable CAE that, in combination with the MAE metric, is able to accurately estimate the SNR in receiving sequences. Many possibilities for future research exist. The current system can be further extended to take into account multiple amplifier characteristics and different constellation schemes. Furthermore, in order to deploy such a system in the field, other variables also need to be considered, such as fading or interference effects.

References

[1] E. Zeydan and Y. Turk, "On the Impact of Satellite Communications over Mobile Networks: An Experimental Analysis", IEEE Transactions on Vehicular Technology, 68(11): 11146–11157, Nov. 2019.

[2] T. Deleu, M. Dervin, K. Kasai and F. Horlin, "Iterative Predistortion of the Nonlinear Satellite Channel", IEEE Transactions on Communications, 62(8): 2916–2926, Aug. 2014.

[3] S. Dimitrov, "Non-linear distortion noise cancellation for satellite return links", Proceedings of IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 22-27 May 2016.

[4] Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives", IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8): 1798-–1828, Mar. 2013.

[5] N. Janakiramaiah, G. Kalyani, S. Narayana, T. Bala Murali Krishna, "Reducing Dimensionality of Data Using Autoencoders", Smart Intelligent Computing and Applications, Springer, Singapore, 2020.

[6] R. Medico, D. Spina, D. Vande Ginste, D. Deschrijver and T. Dhaene, "Autoencoding Density-Based Anomaly Detection for Signal Integrity Applications", Proceedings of IEEE 27th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS), San Jose, California, USA, 14-17 Oct. 2018.

[7] R. Medico, D. Spina, D. Vande Ginste, D. Deschrijver and T. Dhaene, "Machine-Learning-Based Error Detection and Design Optimization in Signal Integrity Applications", IEEE Transactions on Components, Packaging and Manufacturing Technology, 9(9): 1712–1720, Sept. 2019.

[8] B. R. Kiran, D. M. Thomas, and R. Parakkal, "An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos", Journal of Imaging, 4(2): 36, Feb. 2018.

[9] T. J. O'Shea, J. Corgan and T. C. Clancy, "Unsupervised representation learning of structured radio communication signals", Proceedings of First International Workshop on Sensing, Processing and Learning for Intelligent Machines (SPLINE), Aalborg, Denmark, 6-8 Jul. 2016.

[10] M. M. Breunig, H. Kriegel, R. T. Ng, and J. Sander, "LOF: identifying density-based local outliers", Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA, 16 - 18 May 2000.

[11] S. Ruder, "An overview of gradient descent optimization algorithms", arXiv preprint:1609.04747, 2016.