Learning from data: Applications of Machine Learning in optical network design and modeling

José Alberto Hernández
Dept. Ingeniería Telemática
Universidad Carlos III de Madrid, Spain
Email: josealberto.hernandez@uc3m.es

***Abstract*—This article overviews the uses and applications of classical Machine Learning techniques in a variety of network problems. We first overview the basics of statistical learning, including the main algorithms and methodologies involved in the process of designing good Machine Learning models. The second part addresses a number of network use cases where ML can be used to complement and extend existing network models and algorithms, including the classical Routing and Wavelength Assignment (RWA) problem and fiber access delay modelling.**

***Index Terms*—Machine Learning; Communication Networks; Simulated data; Passive Optical Networks; Routing and Wavelength Allocation.**

I. Introduction

We live in a digital interconnected society which generates tons of data continuously with our mobile phones and computers. According to [1], in 2016 it was estimated that 90% of the data has been generated in the previous two years only, at a rate of 2.5 quintillion bytes per day. Many companies have realised that raw data can be stored and processed at low cost, and that modern Big Data technologies can extract value from it.

Big Data comprises the ability to manipulate large volumes of data and extract the fifth V out of it: Value. In reality the challenges of Big Data comprises the four Vs: Volume, Velocity, Variety and Veracity. Big Data is often split into two main branches of research: Technologies and Analytics. The former is mostly focused on the technologies and mechanisms to efficiently store and process large datasets (the ”muscle”), while the latter is in charge of extracting the value and (often hidden) patterns from such data. Data analytics comprises a wide range of statistical and mathematical tools like Data Mining, Statistical Inference, Machine Learning, Artificial Intelligence and Knowledge-Discovery from data.

Recently, Machine Learning (ML) and Artificial Intelligence (AI) techniques have arrived at the optical communications arena, promising to reduce the complexity of network operations and management. AI/ML together with the rise of Software Defined Networking (SDN) open a wide range of possibilities towards the automation of network operations, including network performance monitoring and troubleshooting, even promising some sort of plug-and-play or zero-touch network operations [2], [3].

The number of applications of AI/ML in the context of computer networks and optical communications are starting to appear [4]. Examples of applications include network traffic prediction to later optimize resource allocation [5], quality of transmission (QoT) prediction of lightpaths [6] or solving the classical Routing and Wavelength Assignment (RWA) using a trained ML model from past RWA configuration data [7]. The reader is referred to [8]–[10] for a wide overview of applications of ML/AI techniques in the context of computer and communication networks.

In this article, we first overview the main theoretical foundations of supervised ML techniques, covering the whole process of training, validating and testing models, finding the right model for each problem and over viewing the main ML models available in the literature for experimentation with networking datasets. Finally, we show two use cases where classical Supervised Learning techniques can provide a performance increase with respect to other conventional approaches.

II. Overview of supervised ML techniques

A. Steps and methodology

Every data-science project starts with a question of interest or hypothesis which you are interested in quantitatively answer based and supported on data. After this, the next step is to collect the appropriate data that may have the answer your question.

Good quality data is critical, and it often occurs that a dataset cannot answer your initial hypothesis/question, even with the best ML algorithm available in the state of the art. If that is the case, you probably need to reformulate your question or collect new data, both long and wide (more samples and more features).

Once good quality data has been collected, the next step in the classical data science methodology process is to write some code to produce some visualisations on attempts to get some understanding on the data variables and their relationships.

The last stage before modelling is to prepare the dataset by cleaning missing values, normalising the data (using z-score or others) or create extra dummy variables from original ones.

Once the whole dataset is clean and ready for processing, one can start building models. The first stage requires to divide the whole dataset into a training set (typically 80% of the total data) and a test set (remaining 20%). The training set will be used to build a model while the test set will be used to evaluate its quality or goodness of fit on unseen data.

Once a preliminar model is obtained, next step is about its fine tuning. Model fine tuning is expected to improve the

978-3-903176-21-8 © 2020 IFIP

model's quality by (1) optimally adjusting its hyperparameters (via cross-validation) to avoid overfitting and underfitting, and/or (2) reducing superfluous variables (feature selection, PCA, etc) to further enhance the model's performance metrics (accuracy, recall, F-score, AUC-ROC, etc).

B. Learning from data: Supervised vs Unsupervised ML

There are two main sets of Machine Learning techniques: Supervised and Unsupervised Learning.

The former are tools to build statistical models capable of estimating some output variable from an input dataset, i.e. learning from (labeled) example pairs: $(X_1, y_1), (X_2, y_2), \ldots, (X_N, y_N)$. When the output labels y_i are continuous, the supervised learning problem is often called regression; if y is discrete, the problem at hand is a classification task.

In unsupervised learning, there is no output y, only input unlabelled samples $X_1, X_2, \ldots, X_N$, and the goal is to learn relationships, properties, patterns and structure from the data samples. Examples of unsupervised algorithms include mainly clustering techniques, like k-means, hiearchical clustering, etc.

In the rest of the paper, we shall focus on supervising learning techniques.

C. Problem statement: Learning from data

In Supervised Machine Learning, the data scientist is often given a dataset with N data sample pairs $\{(X_i, y_i)\}_{i=1}^N$. Each sample pair (X_i, y_i) comprises:

- P features or predictors per sample: $(X_{i1}, X_{i2}, \ldots X_{iP})$
- One objective variable y (the label), either continuous or discrete.

In general, the first goal is to understand the relationships between the predictors $(X_1, X_2, \ldots, X_P)$ and the label y, for instance, answering questions like which are the most important/relevant features or the relationships between them. The second goal is to also build a model that captures such patterns and relationships, and can accurately predict the output label y from a given non-seen data observation X. In other words, the model should be able to generalize to unseen observations, that is the key idea of learning.

In mathematical terms, this idea of building a model results in a mathematical function f constructed by a particular ML technique making use of the training dataset only, as follows:

$$y = f(X_1, X_2, \ldots, X_P) + e \tag{1}$$

Here, f represents the information provided from X to y, and e represents the error which accounts for what X cannot explain about y.

D. Types of ML models

As previously stated, function f must capture the main patterns from the data, not necessarily has to be a perfect match of the data.

Each ML model constructs function f in a different way, but all of them seek to minimize some error e or penalty between the predicted value $f(X)$ and the real target value y for the training set.

The amount of ML models available is vast. A survey conducted in 2016 in KD Nuggets[1], revealed the top-10 most popular ML algorithms that every ML engineer need to know:

- Regression
- Clustering
- Decision Trees
- K-Nearest Neighbors
- Principal Component Analysis
- Random Forests
- Time-Series analysis
- Text Mining
- Support Vector Machines
- Ensemble Methods

Essentially, there is not a best model that suits every problem, this is often referred to as the no-free lunch (NFL) theorem of ML. Instead each model has its pros and cons, and it is often a good idea to try several models. In addition, some models are mostly intended for inference, i.e. seeking to understand the relationships between the predictors X and the label y (linear models and decision trees), while others are mostly focused on maximising accuracy and prediction (Support Vector Machines, Random Forests, Deep Neural Networks, etc).

The former set of models are often simple and interpretable parametric models that are especially well-suited when working with little data. The later type of algorithms use complex non-parametric models that act as a black-box, typically providing high prediction accuracy and recall but offering little interpretability; in addition such complex black-box models often require a lot more data to fit their parameters and produce successful results than simpler ones.

Is it good to have a complex model? In general, a model must be complex enough to capture the patterns (often non-linear) within the data, but not too complex since it may end up overfitting the data, i.e. the model matches the training data, but is not capable of generalising to new unobserved data samples.

In general, following the Occam's razor principle, under the assumption that two models have similar performance, a shorter explanation (simpler model) for observed data should be favored over a lengthier explanation.

E. ML models and techniques

Linear Models are simple but very powerful, both suitable for classification and regression. They assume a linear relationship between the label y and the predictors $X_1, \ldots, X_P$:

$$y = \beta_0 + \sum_{j=1}^{P} \beta_j X_j$$

where the coefficients β_j are the model's parameters. The β_j parameters tell us the relationship between each predictor X_j

[1]See https://www.kdnuggets.com/2016/08/10-algorithms-machine-learning-engineers.html, last access Jan 2020

and the label y, thus providing a means to understand their impact on the model with the so-called p-values.

Decision/Regression Trees separate the input space X into J non-overlapped regions $R_1, \ldots, R_J$ which minimise some error function. There exist multiple algorithms to build Decision Trees, but the most popular ones are CART, C5.0 and ID3. These models are non-linear but offer interpretability, so they are a often a good comprimise between simple linear models and complex black box ones.

Random Forests are ensemble methods that combine multiple decision trees trained on different subsets of the training dataset. They are very powerful in terms of prediction accuracy and recall but highly black-box like.

Support Vector Machines (SVMs) build a hyperplane in a multi-dimensional space for separating the classes in a classification task. This hyperplane is constructed by maximising the separation margin between the classes. Thanks to the Kernels, a classification problem, which may not be linearly separable in the original space, is then transformed into a linearly separable problem in a higher-dimensional space. SVMs are also applicable to regression problems.

Finally, Artificial Neural Networks (ANNs) resemble biological neural networks by interconnecting several layers of nodes (neurons) interconnected with weights (synapses). Thanks to backpropagation, the weights at each neuron are adjusted to minimise some classification/regression error function. ANNs are well known to be universal function approximators.

Recently Deep Neural Networks (DNN) and its flavors (CNN, RNN, etc) have been proposed in the literature, showing outstanding performance in different areas like computer vision, speech recognition, natural language processing, etc. DNNs are essentially a variant of ANN with several hidden layers, and novel features like Rectifier Linear Unit (ReLU) activation functions (instead of classical sigmoid ones) and dropout strategies to mitigate overfitting.

With such an over-abundance of ML models, which one to choose? In conclusion, all types of models has pros and cons, and it is often a good idea to start with simple models to get a feeling of the relevant parameters and patterns before moving towards more sophisticated ones. Good reading texts regarding ML principles and models can be found in [11]–[13]. Finally, most popular software libraries with lots of ML models available to practitioners are Python's scikit-learn[2] and R's caret[3], along with Google's Tensorflow[4] and Microsoft Azure ML[5].

F. Model evaluation and fine-tuning

As previously stated, the ML model f constructed from the training dataset is not intended to be an exact match of the data, but instead it is expected to be capable of generalising the patterns beneath the data. Such a generalisation capability is measured by testing the model f on the test set and not only on the training set. In other words, the idea is to check whether or not the model is valid and performs well on unseen data.

There are a number of metrics to evaluate the goodness of fit of a given model; the most popular ones are the coefficient of determination R^2 for regression and Precision, Recall and F-score for classification. These metrics need to be computed for the train and test dataset separately and compared. In that case, three situations may occur:

- Underfitting: Both training and test accuracy are low. In this case, the model is either too simple or the data features/predictors $(X_1, \ldots, X_P)$ do not have a prediction power for modelling the label y (the dataset needs to be increased widewise).
- Overfitting: High accuracy in training but low performance in the test set. The model learns the training set but is not capable of generalising to new data samples. In this case, the model needs to be fine tuned and/or more data samples are need (increase the dataset lengthwise).
- Good fitting: High accuracy on both the training and test datasets. Typically, the training accuracy is slightly higher than the test accuracy one.

Finding a good fitting model often requires several steps of trial and error with different ML models and the fine tuning of their hyper-parameters. Essentially, fine tuning of hyper-parameters is conducted using cross-validation techniques.

Cross-Validation (CV) extends the idea of splitting the dataset into train/test sets, and is the best way to estimate the test error of a model, its generalisation capability, its most important predictors and the best hyper-parameters of a model.

In CV, the dataset is split into K chunks of equal size (K = 5 or 10 typically, i.e. 5-fold or 10-fold cross-validation). Then, the model is trained with $K - 1$ folds or chunks and tested on the remaining one. This process is repeated K times, holding out one chunk at a time. Train and test errors are studied for all K train/test data splits to evaluate if the model is suffering from overfitting/underfitting, or to select the best input predictors or model's hyper-parameters.

III. Two optical networking use cases of ML with simulated data

When real datasets are not available or hard to collect, simulated data con provide a good source to build accurate models. The next sections provide a number of examples in this direction.

A. Classification use case: Learning RWA configurations

In our previous work in [7], [14], we used ML models to solve the Routing and Wavelength Allocation (RWA) problem in a faster way than the ILP or a heuristic algorithm.

Essentially, the RWA problem is transformed into an ML-based multi-class classification problem, where the RWA solution is provided by a classifier in response to a given input traffic matrix. Essentially, an ML algorithm is trained with a

[2]Scikit-learn, https://scikit-learn.org/stable, last access Jan 2020.

[3]The caret package, http://topepo.github.io/caret/index.html, last access Jan 2020.

[4]TensorFlow website, https://tensorflow.org, last access Jan 2020.

[5]Microsoft Azure ML, https://studio.azureml.net/, last access Jan 2020.

dataset optimally labelled after solving the ILP for a number of input traffic matrices and needs to decide for an optimal RWC upon a traffic matrix input.

The idea is the following: the network simulator Net2Plan [15], [16] is fed with a 48,096 different real traffic demand matrices and executed to solve the optimal RWA problem using the well-known ILP formulation. These traffic matrices comprise real traffic measurements collected during six months with a 5 minute granularity in 2004 on the Abilene network topology of Fig. 1. These data are publicaly available[6]. In such scenario, the ILP formulation takes approximately 1-2 seconds to produce the optimal RWA configuration. On this network topology and input traffic matrices, the ILP generates approximately 1,500 different RWC classes, thus providing the correct labels to be used in a supervised ML algorithm.

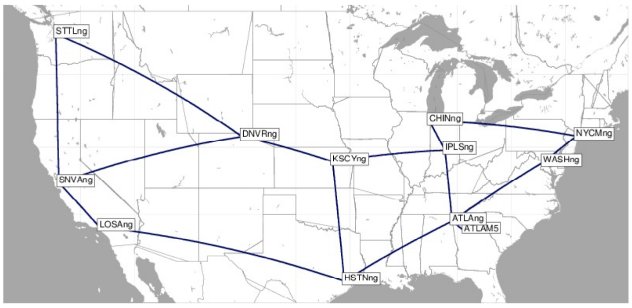

Fig. 1. Abilene network topology, figure from SNDLib.

A Deep Neural Network (DNN) model has been selected as candidate ML model. In particular, our DNN configuration comprises six fully connected layers, with dropout in the first and fourth layers and l2 regularization in the third and fifth layers. Activation is performed using the well-known rectified linear unit (ReLU) and hyperbolic tangent (tanh) functions. The optimization process for the training phase has been performed using Tensorflow's Stochastic Gradient Descent (SGD) aiming at minimizing softmax cross-entropy as the classification loss function. Specifically, the network performs 16,000 training steps (each step uses 400 training samples as batch size) with a learning rate of 0.02. This DNN architecture has been trained and tested with the ABILENE dataset, leading to the results of Table I.

The results show that the trained DNN provides a feasible class in about 95% of the cases or above (i.e. a feasible RWC is that which satisfies the RWA constraints, although it may not optimize the fitness function of the ILP), and further provides a balance precision/recall tradeoff (i.e. F-score) of about 0.7 and above. In addition, the resulting network metrics after solving the RWA problem using our trained DNN shows network related metrics (link load, hop count and wavelength-link resources) very similar to those provided by the ILP solutions.

[6]SNDLib public repository, available at http://sndlib.zib.de/, last access March 2020.

Alg	Fscore train	Fscore test	link load	hop count	Feasibility test
Dataset 1: 20 lambdas at 400Gb/s; 1470 RWC classes					
ILP	-	-	0.025	2.64	100%
DNN	0.886	0.884	0.025	2.64	97.77%
Dataset 2: 40 lambdas at 100Gb/s; 1775 RWC classes					
ILP	-	-	0.06	3.06	100%
DNN	0.748	0.712	0.06	3.05	94.65%

TABLE I
RWA AS A MULTI-CLASS CLASSIFICATION TASK: RESULTS

In addition, once the DNN is trained, which takes around 15 minutes, the time taken to query the DNN for the optical RWA config is about 10 ms (solving the ILP from scratch requires 1-2 seconds). In this sense, a DNN can be trained offline with real data and then can solve the RWA problem in two orders of magnitude smaller than the ILP. In fact, running a heuristic to solve the same problem requires 200 ms, one order of magnitude larger than a well-trained DNN.

B. Regression use case: Learning PON delay models

Previous work [17]–[20] have found closed-form equations for the average delay experienced by packets in the upstream channel of a Passive Optical Network (PON) employing the classical IPACT Dynamic Bandwidth Allocation (DBA) algorithm [21]. Although the equation for the average delay is well known and quite accurate to the simulations, there is no closed-form solution for other interesting delay-based performance metrics like delay percentiles, i.e. the delay experienced by a percentage of packets. Delay percentiles are of paramount importance in evaluating the suitability of PONs for certain types of traffic, like real-time (i.e. video, etc) or C-RAN based fronthaul traffic like CPRI [22]–[27].

In this light, finding models that can accurately characterise, for instance, the 99th delay percentile (i.e. the maximum delay threshold such that 99% of the packets suffer a delay below that threshold) are of interest in the design of certain scenarios [28]. In this light, we have simulated a 1 Gb/s EPON with 16 ONUs, 20 km distance between ONU-OLT and different traffic loads, and further collected the resulting simulated packet delay data. From this dataset, we can obtain different packet delay percentiles and use a non-linear regression model to accurately characterise packet delay percentiles with a closed-form equation derived from data.

As shown in Fig. 2, the ML model derived from the simulated 90th percentile packet delay data is visually very accurate (with coefficient of determination $R^2 = 0.99$, 1 is the maximum). The ML model finds the best values of the $beta_j$ parameters from the data in the following equation:

$$Delay\_perc = \frac{\beta_1 + \beta_2 \rho}{\beta_3 + \rho} + \beta_4 \qquad (2)$$

as a function of network load ρ.

The different β_j parameters have been fit from the simulated dataset using library *nls* from open-source R programming language. Table II shows the β_j coefficients for the 90% and 99% delay percentiles, along with the R^2 metric.

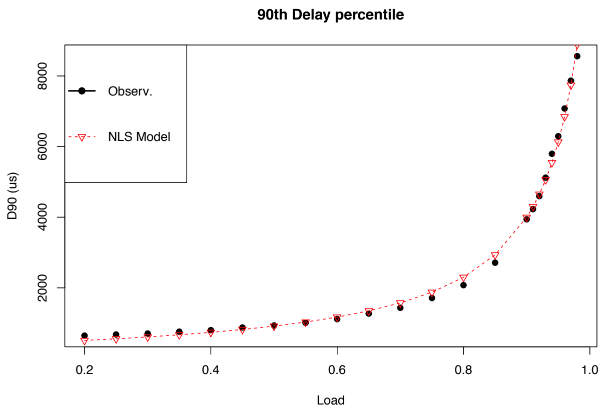

Fig. 2. 90th delay percentile: simulation and ML regression model.

Metric	Model	β_1	β_2	β_3	β_4	R^2
90th delay	ML model	374	243	1.05	12	99.8%
99th delay	ML model	386	349	1.05	12	99.8%

TABLE II
PARAMETER FIT: AVERAGE DELAY AND PERCENTILES (β_j VALUES IN μs)

IV. SUMMARY AND CONCLUSIONS

This article has overviewed the fundamentals of supervised Machine Learning and their applications in building models in different scenarios of optical networks. Indeed ML models provide powerful methods to build models from data, either real or simulated when this is not possible. In particular, two classical applications in Routing and Wavelength Assignment in core networks (classification task) and delay modelling for Passive Optical Networks (regression task) have been shown.

V. ACKNOWLEDGMENTS

The author would like to acknowledge the support of the Spanish projects TEXEO (TEC2016-80339-R) and TAPIR-CM (grant no. P2018/TCS4496), and the EU-H2020 Metro-haul project (grant no. 761727) to the development of this work.

In addition, the author would also like to acknowledge the help of Dr. Amin Ebrahimzadeh for his datasets in the PON simulation scenario of section III-B.

REFERENCES

[1] I. M. C. technical report, "10 key marketing trends for 2017 and ideas for exceeding marketing expectations," IBM, Tech. Rep., 2016. [Online]. Available: https://paulwriter.com/wp-content/uploads/2017/10/10-Key-Marketing-Trends-for-2017.pdf

[2] D. Rafique and L. Velasco, "Machine learning for network automation: overview, architecture, and applications [invited tutorial]," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 10, no. 10, pp. D126–D143, Oct 2018.

[3] T. Tanaka, A. Hirano, S. Kobayashi, T. Oda, S. Kuwabara, A. Lord, P. Gunning, O. Gonzalez de Dios, V. Lopez, A. M. Lopez de Lerma, and A. Manzalini, "Autonomous network diagnosis from the carrier perspective [invited]," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 12, no. 1, pp. A9–A17, January 2020.

[4] D. Côté, "Using machine learning in communication networks (invited)," *J. Opt. Commun. Netw.*, vol. 10, no. 10, pp. D100–D109, Oct 2018. [Online]. Available: http://jocn.osa.org/abstract.cfm?URI=jocn-10-10-D100

[5] A. Azzouni and G. Pujolle, "A long short-term memory recurrent neural network framework for network traffic matrix prediction," *CoRR*, vol. abs/1705.05690, 2017. [Online]. Available: http://arxiv.org/abs/1705.05690

[6] R. M. Morais and J. ao Pedro, "Machine learning models for estimating quality of transmission in dwdm networks," *J. Opt. Commun. Netw.*, vol. 10, no. 10, pp. D84–D99, Oct 2018. [Online]. Available: http://jocn.osa.org/abstract.cfm?URI=jocn-10-10-D84

[7] I. Martín, S. Troia, J. A. Hernández, A. Rodríguez, F. Musumeci, G. Maier, R. Alvizu, and González de Dios, "Machine learning-based routing and wavelength assignment in software-defined optical networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 871–883, Sep. 2019.

[8] F. Musumeci, C. Rottondi, A. Nag, I. Macaluso, D. Zibar, M. Ruffini, and M. Tornatore, "A survey on application of machine learning techniques in optical networks," *arXiv preprint arXiv:1803.07976*, pp. 1–21, 2018.

[9] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shariar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 16, pp. 1–99, 2018.

[10] J. Mata, I. de Miguel, R. J. Duran, N. Merayo, S. K. Singh, A. Jukan, and M. Chamania, "Artificial intelligence (ai) methods in optical networks: A comprehensive survey," *Optical Switching and Networking*, vol. 28, no. 1, pp. 43–57, 2018.

[11] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[12] M. Kuhn and K. Johnson, *Applied predictive modeling*. Springer, 2013.

[13] R. Battiti and M. Brunato, *The LION way. Machine Learning* plus *Intelligent Optimization*. LIONlab, University of Trento, Italy, 2017. [Online]. Available: http://intelligent-optimization.org/LIONbook/

[14] I. Martín, J. A. Hernández, S. Troia, F. Musumeci, G. A. Maier, and O. G. de Dios, "Is machine learning suitable for solving rwa problems in optical networks?" in *European Conference on Optical Communication (ECOC)*, 2018, pp. 1–3.

[15] P. Pavon-Marino and J. Izquierdo-Zaragoza, "Net2plan: an open source network planning tool for bridging the gap between academia and industry," *IEEE Network*, vol. 29, no. 5, pp. 90–96, Sep. 2015.

[16] I. Martín, J. A. Hernández, and Ó. G. de Dios, "Netgen: A fast and scalable tool for the generation and labeling of networking datasets," in *2019 21st International Conference on Transparent Optical Networks (ICTON)*. IEEE, 2019, pp. 1–4.

[17] F. Aurzada, M. Lévesque, M. Maier, and M. Reisslein, "Fiwi access networks based on next-generation pon and gigabit-class wlan technologies: A capacity and delay analysis," *IEEE/ACM Transactions on Networking*, vol. 22, no. 4, pp. 1176–1189, Aug 2014.

[18] F. Aurzada, M. Scheutzow, M. Reisslein, N. Ghazisaidi, and M. Maier, "Capacity and delay analysis of next-generation passive optical networks (ng-pons)," *IEEE Transactions on Communications*, vol. 59, no. 5, pp. 1378–1388, May 2011.

[19] I. Seoane, J. A. Hernández, R. Romeral, and D. Larrabeiti, "Analysis and simulation of a delay-based service differentiation algorithm for ipact-based pons," *Photonic Network Communications*, vol. 24, no. 3, pp. 228–236, 2012.

[20] J. García-Reinoso, J. A. Hernández, I. Seoane, and I. Vidal, "On the effect of sudden data bursts in the upstream channel of ethernet pons employing ipact under the gated-service discipline," *Optical Switching and Networking*, vol. 13, pp. 94–102, 2014.

[21] G. Kramer, B. Mukherjee, and G. Pesavento, "Ipact a dynamic protocol for an ethernet pon (epon)," *IEEE Communications Magazine*, vol. 40, no. 2, pp. 74–80, Feb 2002.

[22] A. de la Oliva, J. A. Hernández, D. Larrabeiti, and A. Azcorra, "An overview of the cpri specification and its application to c-ran based lte scenarios," *IEEE Communications Magazine*, vol. 54, no. 2, pp. 152–159, 2016.

[23] I. Ucar, J. A. Hernández, P. Serrano, and A. Azcorra, "Design and analysis of 5g scenarios with 'simmer': An r package for fast des

prototyping," *IEEE Communications Magazine*, vol. 56, no. 11, pp. 145–151, 2018.

[24] G. Otero Pérez, J. A. Hernández, and D. Larrabeiti, "Fronthaul network modeling and dimensioning meeting ultra-low latency requirements for 5g," *IEEE/OSA Journal Optical Communications and Networking*, vol. 10, no. 6, pp. 573–581, 2018.

[25] G. O. Pérez, J. A. Hernández, and D. L. López, "Delay analysis of fronthaul traffic in 5g transport networks," in *2017 IEEE 17th International Conference on Ubiquitous Wireless Broadband (ICUWB)*. IEEE, 2017, pp. 1–5.

[26] D. Eugui and J. A. Hernández, "Low-latency transmission of fronthaul traffic over xg (s)-pon with fixed-elastic bandwidth reservations," in *Optical Fiber Communication Conference*. Optical Society of America, 2019, pp. W2A–28.

[27] ——, "Analysis of a hybrid fixed-elastic dba with guaranteed fronthaul delay in xg (s)-pons," *Computer Networks*, vol. 164, p. 106907, 2019.

[28] G. Otero Pérez, D. Larrabeiti, and J. A. Hernández, "5g new radio fronthaul network design for ecpri - ieee 802.1cm and extreme latency percentiles," *IEEE Access*, vol. 7, pp. 82 218–82 230, 2019.