An Introduction to Machine Learning in Optical Transport networks

Massimo Tornatore

Department of Electronics, Information and Bioengineering
Politecnico di Milano, Italy
The presentation is organized into two main parts:

- **Part 1:** overview on Machine Learning
  - Basic concepts (supervised/unsupervised learning, bias/variance trade-off, etc.)
  - Some algorithms
    - Linear regression
    - Neural Networks
    - K-nearest neighbours

- **Part 2:** applications of ML to optical-network problems
  - Part 2a): QoT estimation and RSA
  - Part 2b): Failure management
  - Part 2c): Other application at physical and network layer
    - Traffic prediction, virtual topology design,...

**Note:** this is NOT a “pure” Machine Learning tutorial. The objective is to show how we applied ML to our research problems.
• “Field of study that gives computers the ability to learn without being explicitly programmed” (A. Samuel, 1959)
• “Teaching a computer to automatically learn concepts through data observation”
• …

• For our purposes: An math/statistical instrument to make decisions by inferring statistical properties of monitored data
  …in the context of optical networks

• Sometimes confused with other terms: AI, Deep Learning, Data Analytics, Data Mining, etc.
Many definitions with blurred borders
• At data plane - Complexity increase
  - Coherent Transmission System
    - Several system parameters to choose from: modulation techniques and formats, coding rates, symbol rate..
    - DSP: Huge availability of data
  - Elastic Optical Networks
    - Customizable channel width, BV-ROADM

• At control plane - New Enablers
  - Software Defined Networking
  - Intelligence (computing capabilities) everywhere (e.g., MEC)
  - Monitors

Why only now in optical networks?
• **Supervised-learning algorithms**
  - We are given “labeled” data (i.e., “ground truth”)
  - **Main objective**: given a set of “historical” input(s) predict an output
    - Regression: output value is continuous
    - Classification: output value is discrete or “categorical”

• **An example: Traffic forecasts**
  - Given traffic during last week/month/year
    - Predict traffic for the next period (regression)
    - Predict if available resources will be sufficient (classification)

• **Other examples**
  - Speech/image recognition
  - Spam classifier
  - House prices prediction/estimation
Supervised learning: an «optical» example

Training Phase

\[ \lambda = 1550, \text{ path} = \text{nodes A-C-D-G-F, } Mod = \text{QPSK, } \Rightarrow \text{BER}=10^{-5} \]

\[ \lambda = 1553, \text{ path} = \text{nodes B-G-D-F-E, } Mod = \text{QPSK, } \Rightarrow \text{BER}=10^{-2} \]

...

Active/Test Phase

Create path: \( \lambda = 1553, \text{ nodes A-C-D-G-F, } Mod \text{ QPSK } \Rightarrow \text{BER}=? \)

Supervised Learning: the algorithm is trained on dataset that consists of paths, wavelengths, modulation, and the corresponding BER. Then it extrapolates the BER in correspondence to new inputs.
Main categories of ML algorithms (2)

• Unsupervised-learning algorithms
  ▪ Available data is not “labeled”
  ▪ **Main objective**: derive structures (patterns) from available data
    o Clustering finding “groups” of similar data
    o Anomaly detection

• An example: cell-t+raffic classification
  ▪ Given traffic traces i
  ▪ understand if some cells provide similar patterns
    o Residential, business, close to theatre, cinema, stadium…
    o This information can be used to make network resources planning

• Other example
  ▪ Group people according to their interests to improve advertisement
Unsupervised learning: some examples

Data:
\( \lambda = 1550 \), path= nodes A-B-D-E, Mod = QPSK, BER=10^{-6}

\( \lambda = 1553 \), path= nodes A-C-D-G-F, Mod = BPSK, BER=10^{-2}

\( \lambda = 1544 \), path= nodes C-D-E-F, Mod = DPQPSK, BER=10^{-2}

\( \lambda = 1545 \), path= nodes B-D-G-F-E, Mod = 16-QAM, BER=10^{-7}

Unsupervised Learning: the algorithm identifies unusual patterns in the data, consisting of wavelengths, paths, BER, and modulation.

Courtesy of Marco Ruffini and Irene Macaluso
- **Semi-Supervised learning**
  - Hybrid of previous two categories
  - **Main objective**: most of the training samples are unlabeled, only few are labeled.
    - Common when labeled data are scarce or expensive
  - Self-training: start with labeled data, then label unlabeled data based on first phase

- **Reinforcement learning**
  - Available data is not “labeled”
  - **Main objective**: learn a policy, i.e., a mapping between inputs/states and actions. Behavior is refined through rewards
  - Methodologically similar to «optimal control theory» or «dynamic programming»
  - Q-learning
Reinforcement learning: the algorithm learns by receiving feedback on the effect of modifying some parameters, e.g. the power and the modulation.

<table>
<thead>
<tr>
<th>Initial state</th>
<th>Action</th>
<th>State</th>
<th>Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda=1550\text{nm}$, nodes A-B-D-E, Mod QPSK, BER=10^{-3}</td>
<td>Change: Mod DPQPSK</td>
<td>BER= 10^{-3}</td>
<td>0</td>
</tr>
<tr>
<td>$\lambda=1550\text{nm}$, nodes A-B-D-E, Mod QPSK, BER=10^{-3}</td>
<td>Change: output power channel +5 dBm</td>
<td>BER= 10^{-2}</td>
<td>-1</td>
</tr>
<tr>
<td>$\lambda=1550\text{nm}$, nodes A-B-D-E, Mod QPSK, BER=10^{-3}</td>
<td>Change: Mod BPSK</td>
<td>BER= 10^{-4}</td>
<td>+1</td>
</tr>
</tbody>
</table>

**Reinforcement Learning**: the algorithm learns by receiving feedback on the effect of modifying some parameters, e.g. the power and the modulation.
Some basic concepts in ML

- Data shall be manipulated with care!
  - Overfitting vs Underfitting
  - Linear vs Non Linear models
An example of fitting a model

- Suppose we want to predict a house price given its size.
• Linear, quadratic, polynomial, non linear

SIDE MESSAGE:
Increasing model flexibility might lead to overfitting!!!
Predictions can get worse if the model is too flexible (counter-intuitive!)
Some algorithms

- Supervised
  - Parametric
    - Linear regression
    - Logistic regression
    - Neural Networks
    - SVM
  - Non parametric
    - K-nearest neighbor
    - Random Forest
- Unsupervised
  - Clustering
    - K-means
    - Gaussian Mixture Models
• Simplest model
  - $h(x)$ is a **linear** function
  - $h(x)$ has only one variable (univariate), i.e., feature $x_1$

  \[
h(x) = h(x_1) = \theta_0 + \theta_1 x_1\]

  - $\theta_0$ and $\theta_1$ are the “weights”

  How to choose $\theta_0$ and $\theta_1$?

Minimize the training mean-square error (MSE)

\[
\min_{\theta_0, \theta_1} \left\{ \text{MSE} = \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2 \right\}
\]
Multivariate and Polynomial Regression

- Multivariate
  - We now have a features vector \( \mathbf{x} = (x_1, x_2, \ldots, x_n) \)
  - \( h(\mathbf{x}) = h(x_1 \ldots x_N) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_N x_N \)
    - \( \theta_0 \ldots \theta_1 \) are the “weights” chosen by the algorithm

- Polynomial
  - We now increasing the order of polynomials in \( h(\mathbf{x}) \)
  - \( h(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_{12} x_1 x_2 + \theta_2 (x_2)^2 \ldots + \theta_n x_n \)
Logistic regression [Classification!]

- Output $h(x)$ takes only **discrete** values
  - Ex: $y=${0;1}, e.g., yes/no, good/bad, spam/non-spam…
  - Multiclass classifier: $y=${A,B,C,…}

- A good candidate function $h$ for
  - $h(z) = \frac{1}{1 + e^{-z}}$ is the “logistic” (or “sigmoid”) function
    - for $z \to -\infty$: $h(z)\to0$
    - for $z \to +\infty$: $h(z)\to1$
    - for $z=0$: $h(z)=0.5$
Why do we need a new algorithm?

- Some problems are just too complex
  - Many features can have a role \( \rightarrow \) increased features space
- Difficult for a human to even know which features are important

\[
h(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \ldots)
\]

Suppose we have 100 different features and we want to add all quadratic terms:

- \( x_1^2, x_1 x_2, \ldots, x_1 x_{100} \)
- \( x_2^2, \ldots, x_2 x_{100} \)
- \( \ldots \)
- \( x_{99}^2, x_{99} x_{100} \)
- \( x_{100}^2 \)

n “original” features require \( O(n^2) \) quadratic terms!
**Neural networks representation**

*Logistic unit or “neuron”*

- The simplest neural network

\[
h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-(\theta^T x)}}
\]

Diagram:
- Input layer: \(x_0, x_1, x_2, \ldots, x_n\)
- Output layer: \(h_\theta(x)\)
- Neural network representation
  - Logistic unit or “neuron”
  - \(h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-(\theta^T x)}}\)
A “collection” of interacting neurons

3 observations:
1. NN can capture any relation between x and y
2. Hyperparameters: # of hidden layers, # neurons per hidden layer. Who decides them?
3. Deep Learning: the more layer, the less decisions shall be taken by a programmer
Use of data in Machine Learning

Training, testing, validation

Data-set

Training set

Validation set*

Test set

Training phase: fit the ML model

Validation phase: optimize the model

Test phase: assess model performance

Algorithm optimization loop

FINAL MODEL

Results!

*Using validation set is **only one** of the approaches to optimize a ML algorithm.
K-Nearest Neighbors

- Used for classification and regression
- Decision based on the K nearest points in the training sets
  - Need to choose K

- Example 1: classification (K=3)
  - Choose the most frequent class among the KNN \(\rightarrow\) predict class 1
  - Changing the value of K (e.g. K=5) may affect the result \(\rightarrow\) predict class 2
• Let us focus only on classifiers
  ▪ Note: the output produced by the classifier is $P_{pos}$
    ○ Ideally, we want $P_{pos}$ close to 1

• Metrics used in the following
  ○ Accuracy
    – Fraction of test instances correctly classified
    – Affected by by relative frequency
  ○ Area under the ROC curve

• Other metrics
  ○ Precision
  ○ Recall
Several questions to be addressed

- Which ML algorithm best describes our problem?
- Which data/features should we consider to make predictions?
- Is it worth collecting as much data as possible? Is there any irrelevant parameter we can (or should) neglect?
- What is the performance of our learning algorithm?
- And what is its complexity?
- ...
1. ML for QoT Estimation for Unestablished Lightpaths

2. ML for Soft-Failure Identification

3. An overview of other applications at network layer
Outline: Part 2
Applications

1. ML for QoT Estimation for Unestablished Lightpaths

2. ML for Soft-Failure Identification

3. An overview of other applications at network layer
   - Javier Mata, et a., Artificial intelligence (AI) methods in optical networks: A comprehensive survey, Optical Switching and Networking, Volume 28, 2018, pp. 43-57
Motivation

Increasing «degrees of freedom»

- A wider range of **degrees of freedom** (parameters) is available to system engineers:
  - path
  - spectrum
  - modulation format
  - baud rate
  - FEC coding
  - single/multicarrier transmission
  - nonlinearity mitigation solution
  - adaptive channel spacing
  - …

- Combinations of these lightpath parameters grow dramatically
- Possibly, for all of these combinations, we shall calculate a QoT
Existing (pre-deployment) estimation techniques for lightpath QoT

• “Exact” analytical models estimating physical layer impairments (e.g., split-step Fourier method…)
  - Accurate results
  - Heavy computational requirements
  - Not scalable to large networks and real time estimations

• Marginated formulas (Power Budget, Gaussian model…)
  - Faster and more scalable
  - Inaccurate, high margination, underutilization of network resources (up to extra 2 dB for design margins [1])

Machine Learning as an alternative approach?

- Machine Learning exploits knowledge extracted from field data…
  - QoT of established lightpaths, e.g. using monitors (OPMs) at the receiver
  - …. to predict the QoT of unestablished lightpaths

😊 No need for complex analytical models
😊 Fast and scalable
😊 Requires training phase with historical data
  - How long must the training phase be?
  - How accurate will the estimation be?
  - Objectives of our numerical analysis…. 
Whole framework: RSA + QoT estimation

Traffic request

Routing and spectrum assignment algorithm

Query input (set of features)

Answer (estimated BER/OSNR)

ML Classifier

Classifier training

Measured BER/OSNR

Lightpath deployment

Selection

Traffic request

DECISION PROCESS

Marginated BER/OSNR calculations

Query input (set of features)

Measured BER/OSNR

Lightpath deployment

Selection
How our proposed ML classifier works

Case 1

Output: probability that BER ≤ T*
Input: set of lightpath features

The classifier is trained on a set of L experiments to generate the ground truth

(Case of only local knowledge)
How our proposed ML classifier works

Case 2

• To the previous 6 feature we add, for the «most interfering left and right neighbors»:
  • guardband
  • traffic volume
  • modulation format

(Case of complete knowledge)

• Note: these additional six features are chosen with the intent to capture cross-channel nonlinear effects
How to generate synthetic field data?

We use a Bit Error Rate Estimation Tool (ETool) that on input of …

- a candidate lightpath
- a modulation format

.. and under assumption of..

- AWGN channel
- back-to-back penalties
- a random system margin expneg distributed with mean 2 dB
  - Expneg provides a worst case

... calculates ...

- BER measured at the input of the channel decoder
Which Machine Learning Algorithm?

- We use a Random Forest (RF) classifier with 25 estimators.
- To take this choice, we have compared:
  - 5 RF classifiers
  - 3 k-Nearest-Neighbor classifiers

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Training time (s)</th>
<th>Test time (s)</th>
<th>AUC</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dummy classifier</td>
<td>0.048979</td>
<td>3.83 e-07</td>
<td>0.501</td>
<td>0.539</td>
</tr>
<tr>
<td>1 Nearest Neighbor</td>
<td>1.183121</td>
<td>4.83 e-05</td>
<td>0.959</td>
<td>0.957</td>
</tr>
<tr>
<td>5 Nearest Neighbor</td>
<td>1.085116</td>
<td>5.05 e-05</td>
<td>0.991</td>
<td>0.965</td>
</tr>
<tr>
<td>25 Nearest Neighbor</td>
<td>1.211694</td>
<td>6.91 e-05</td>
<td>0.996</td>
<td>0.965</td>
</tr>
<tr>
<td>Random Forest 1 tree</td>
<td>0.076944</td>
<td>3.96 e-07</td>
<td>0.991</td>
<td>0.965</td>
</tr>
<tr>
<td>Random Forest 5 trees</td>
<td>0.180835</td>
<td>6.24 e-07</td>
<td>0.995</td>
<td>0.970</td>
</tr>
<tr>
<td>Random Forest 25 trees</td>
<td>0.721042</td>
<td>1.56 e-06</td>
<td>0.996</td>
<td>0.968</td>
</tr>
<tr>
<td>Random Forest 100 trees</td>
<td>2.830545</td>
<td>5.32 e-06</td>
<td>0.996</td>
<td>0.966</td>
</tr>
<tr>
<td>Random Forest 500 trees</td>
<td>14.052182</td>
<td>2.63 e-05</td>
<td>0.996</td>
<td>0.966</td>
</tr>
</tbody>
</table>

- RF with 25 estimators provided the best trade-off between performance and computational time.
Training and Testing Scenario

- Japanese and NSF optical network
- Flexgrid @ 12.5 GHz
- Transceivers @ 28 GBaud
- 6 Modulation formats
  - (DP) BPSK, QPSK, 8-QAM to 64-QAM,
- Traffic requests: [50;1000] Gbps
- 3 candidate paths per node pair
- BER threshold $T = 4 \times 10^{-3}$
How long shall training phase be?

(1) Accuracy vs training set size

- «ROC» curve
- Area under the ROC curve (AUC)

Take-Away 1: Training phase has a reasonable duration
How long shall training phase be?

(2) Effect of topology on Accuracy

Take-Away 2: easier to classify on a large network (less options!)
How to build the training dataset?

- Use historical data
  - We will never observe samples with too high BER!!

- Use random probes:
  - Very costly (high spectrum occupation)

- Use selective probes:
  - Lower spectrum occupation, good accuracy

**TABLE V: AUC comparison of probing approaches**

<table>
<thead>
<tr>
<th>Training set</th>
<th>AUC (full testing dataset)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C (historical)</td>
<td>0.77</td>
</tr>
<tr>
<td>C (selective, 5% probes)</td>
<td>0.85</td>
</tr>
<tr>
<td>C (selective, 10% probes)</td>
<td>0.87</td>
</tr>
<tr>
<td>C (selective, 25% probes)</td>
<td>0.89</td>
</tr>
<tr>
<td>C (selective, 50% probes)</td>
<td>0.89</td>
</tr>
<tr>
<td>A (random)</td>
<td>0.98</td>
</tr>
</tbody>
</table>
Analysis of feature relevance

- Removing irrelevant «ML-input features» makes the system less costly and less complex to manage.

**TABLE IV: The considered feature subsets**

<table>
<thead>
<tr>
<th>Feature</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>S7</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of links</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>link path length</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>length of longest link</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>traffic volume</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>modulation format</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>guardband, modulation format and traffic volume of nearest left and right neighbor</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Graph:**
- Full vs. Near to T accuracy for different subsets (S1 to S7)
Ok, by automating QoT estimation we can save workforce/Opex and decrease setup times. But, what is the impact on resource saving?

Output: probability that BER ≤ T*

<table>
<thead>
<tr>
<th>γ</th>
<th>Average reduction in number of installed transceivers [%]</th>
<th>Average reduction in overall spectrum occupation [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>17.14</td>
<td>16.70</td>
</tr>
<tr>
<td>0.6</td>
<td>15.84</td>
<td>15.44</td>
</tr>
<tr>
<td>0.7</td>
<td>14.51</td>
<td>13.88</td>
</tr>
<tr>
<td>0.8</td>
<td>11.41</td>
<td>10.73</td>
</tr>
<tr>
<td>0.9</td>
<td>6.90</td>
<td>6.31</td>
</tr>
<tr>
<td>1</td>
<td>1.17</td>
<td>0.96</td>
</tr>
</tbody>
</table>

γ: threshold you are willing to accept
Outline: Part 2
Applications

1. ML for QoT Estimation for Unestablished Lightpaths

2. ML for Soft-Failure Identification

3. An overview of other applications at network layer
   - Javier Mata, et a., Artificial intelligence (AI) methods in optical networks: A comprehensive survey, Optical Switching and Networking, Volume 28, 2018, pp. 43-57
Two main failure types in optical networks

- **Hard-failures**
  - Sudden events, e.g., fiber cuts, power outages, etc.
  - Unpredictable, require «protection» (reactive procedures)

- **Soft-failures:**
  - Gradual transmission degradation due to equipment malfunctioning, filter shrinking/misalignment…
  - Trigger early network reconfiguration (proactive procedures)
• How can we *predict* soft-failures?

Perform continuous monitoring of BER at the receiver…
… until some “anomalies” are detected

Early-detection helps *preventing* service disruption (e.g., through proactive reconfiguration)

---

• How can we identify the cause of the failure?
  § Failures can be caused by different sources
    o Filters shrinking/misalignment
    o Amplifier malfunctioning
    o Fiber bends
    o …

Different sources of failure can be distinguished via the different effects on BER (i.e., via different BER “features”)

Phases of our study

Overall view

- **BER monitoring and data collection**
  - “normal” data
  - anomaly

- **Data preprocessing**
  - Outliers removal
  - Training, cross-validation and test sets formation (75% + 15% + 15% of the original data-set)

- **BER window formation**
  - Select:
    - BER sampling time ($T_{BER}$)
    - window size (duration of observation)

- **Features extraction**
  - BER statistics:
    - mean
    - min/max
    - standard dev.
    - Window spectral components

- **ML algorithm training**
  - Fault detection:
    - Binary SVM
    - Random Forest
    - Multiclass SVM
    - Neural Network
  - Fault identification:
    - Neural Network

- **ML algorithm optimization loop**

- **Detect-Failure (DET-F)**

- **Identify-Failure (IDENT-F)**

- **Prediction & Evaluation**
2nd Phase of our study
Deciding ML algorithm, Train. & Valid.

1. Data Retrieval

3 decisions

BER window
- Select:
  - BER sampling time \(T_{BER}\)
  - window size (duration of observation)

Features
- BER statistics:
  - mean
  - min/max
  - standard dev.
- Window spectral components
- Feature Scaling

ML algorithm
- Fault detection:
  - Binary SVM
  - Random Forest
  - Multiclass SVM
- Neural Network

Fault identification:
- Neural Network

Validation (optimization of hyperparameters)

3. Prediction and Evaluation

BER window \((x, y)\)
Testbed setup

- Testbed for real BER traces
  - Ericsson 380 km transmission system
    - 24 hours BER monitoring
    - 3 seconds sampling interval
  - PM-QPSK modulation @ 100Gb/s
  - 6 Erbium Doped Fiber Amplifiers (EDFA) followed by Variable Optical Attenuators (VOAs)
  - Bandwidth-Variable Wavelength Selective Switch (BV-WSS) is used to emulate 2 types of BER degradation:
    - Filter misalignment
    - Additional attenuation in intermediate span (e.g., due to EDFA gain-reduction)
Numerical results: *Detection*

Accuracy vs window features

- Binary SVM

**Take-away 1:** Higher performance for with low sampling time → Fast monitoring equipment is required

**Take-away 2:** For increasing sampling time, longer “Windows” are needed for high accuracy
Numerical results: *Identification*
Accuracy vs window features

- Neural Network

### Take-away 3:
To perform failure-cause identification, much smaller sampling period is needed with respect to failure detection.
Outline: Part 2
Applications

1. ML for QoT Estimation for Unestablished Lightpaths

2. ML for Soft-Failure Identification

3. An overview of other applications at network layer
   • F. Musumeci et al., “A Survey on Application of Machine Learning Techniques in Optical Networks”, Submitted to IEEE communication surveys and tutorials, available in ArXiv
   • Javier Mata, et a., Artificial intelligence (AI) methods in optical networks: A comprehensive survey, Optical Switching and Networking, Volume 28, 2018, pp. 43-57
Overview of other applications

Physical layer
1. Quality of Transmission (QoT) estimation
2. Optical amplifier control
3. Modulation format recognition
4. Nonlinearities mitigation

Network layer
1. Traffic prediction and virtual topology design
2. Failure detection and localization
3. Flow classification

When adding/dropping channels into/from a WDM system, EDFA gain should be adjusted to re-balance output powers.

Analytical models:
- typically not generalizable
- depend on the specific system (gain-control mechanism, EDFA gain tilt, nr of EDFAs...) which use to vary during their activity

ML allows to self-learn typical response patterns.


Bastos et al., “Mapping EDFA Noise Figure and Gain Flatness Over the Power Mask Using Neural Networks”, Journal of Microwaves, Optoelectronics and Electromagnetic Applications, vol. 12, n. SI-2, July 2013
• Elastic transceiver can to operate with different modulation formats

• Traditional MFI requires prior information exchange between end points (from upper layer protocols)
  – additional delay for in signal detection
• ML enables automated MFR from features of the received signal

---

• Optical signals are affected by fiber nonlinearities
  – Kerr effect, self-phase modulation (SPM), cross-phase modulation (XPM)…

• Traditional methods require complex mathematical models and prior information on the traversed channel
• ML enables “safer” decision by learning from actual channel properties

Network layer

Traffic prediction and virtual topology design

- New services with high spatio-temporal traffic dynamics
- No reconfiguration → peak-traffic dimensioning
- ML leverages online (live) traffic monitoring/prediction to avoid overprovisioning

Network layer domain
Flow classification

• Traffic flows can be heterogeneous in terms of:
  § protocols (http, ftp, smtp…)
  § services (fixed vs mobile, VoD, data transfer, text messages…)
  § requirements (latency, bandwidth, jitter…)
  § network “customers” (human end-users, companies, sensors, servers…)
  • E.g., “mice” vs “elephant” flows in Data Centers

Distinguish between different flows is crucial for resources (i.e., capacity) allocation, scheduling, SLAs, QoS…

• ML
  § enables traffic features extraction from direct
  § observation of traffic flows
  § allows simultaneous use of heterogeneous features


• Viljoen et al., “Machine Learning Based Adaptive Flow Classification for Optically Interconnected Data Centers”, in ICTON 2016, July 2016

Conclusion

- Automated QoT estimation via machine learning
  - Necessary in a dynamic context
  - Enables margin compression

- Automated soft-failure detection and identification
  - Successful testing of identification of failure modes
  - Sampling BER each few seconds led to satisfactory accuracies
  - Identification is more complex than detection (to be confirmed..)

- Definitely not a fast learning curve if you do not simply want to use machine learning as a black box
Some material

- Books (general refs. for ML):
  - G. James, D. Witten, T. Hastie, R. Tibshirani, “An Introduction to Statistical Learning with Applications in R”, Ed. Springer

- Prof. Andrew Ng lectures (Stanford University)
- … Google it!
Some publications (1)

Surveys
- F. Musumeci et al., “A Survey on Application of Machine Learning Techniques in Optical Networks”, Submitted to IEEE communication surveys and tutorials
- Javier Mata, et a., Artificial intelligence (AI) methods in optical networks: A comprehensive survey, Optical Switching and Networking, Volume 28, 2018, pp. 43-57

Some Motivations

QoT estimation
Some publications (2)

Failure recovery

Projects
- EU ORCHESTRA and CHRON projects
Thank You!

..and thanks to them!

Achille Pattavina
Francesco Musumeci
Shahin Shahkarami
Luca Barletta

Biswanath Mukherjee, Yu Wu Lin Wang, Sabidur Rehman, Filippo Cugini (CNIT)
Cristina Rottondi and Alessandro Giusti (Dalle Molle Inst.)