Event-Detection Deep Neural Network for OTDR Trace Analysis

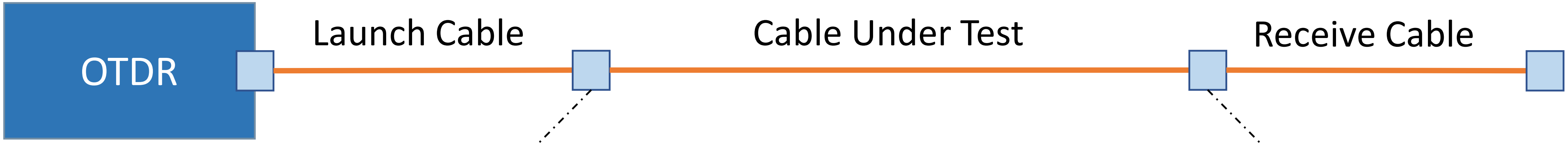
Davide Rutigliano, Giacomo Boracchi, Pietro Invernizzi, Enrico Sozio, Cesare Alippi, Stefano Binetti

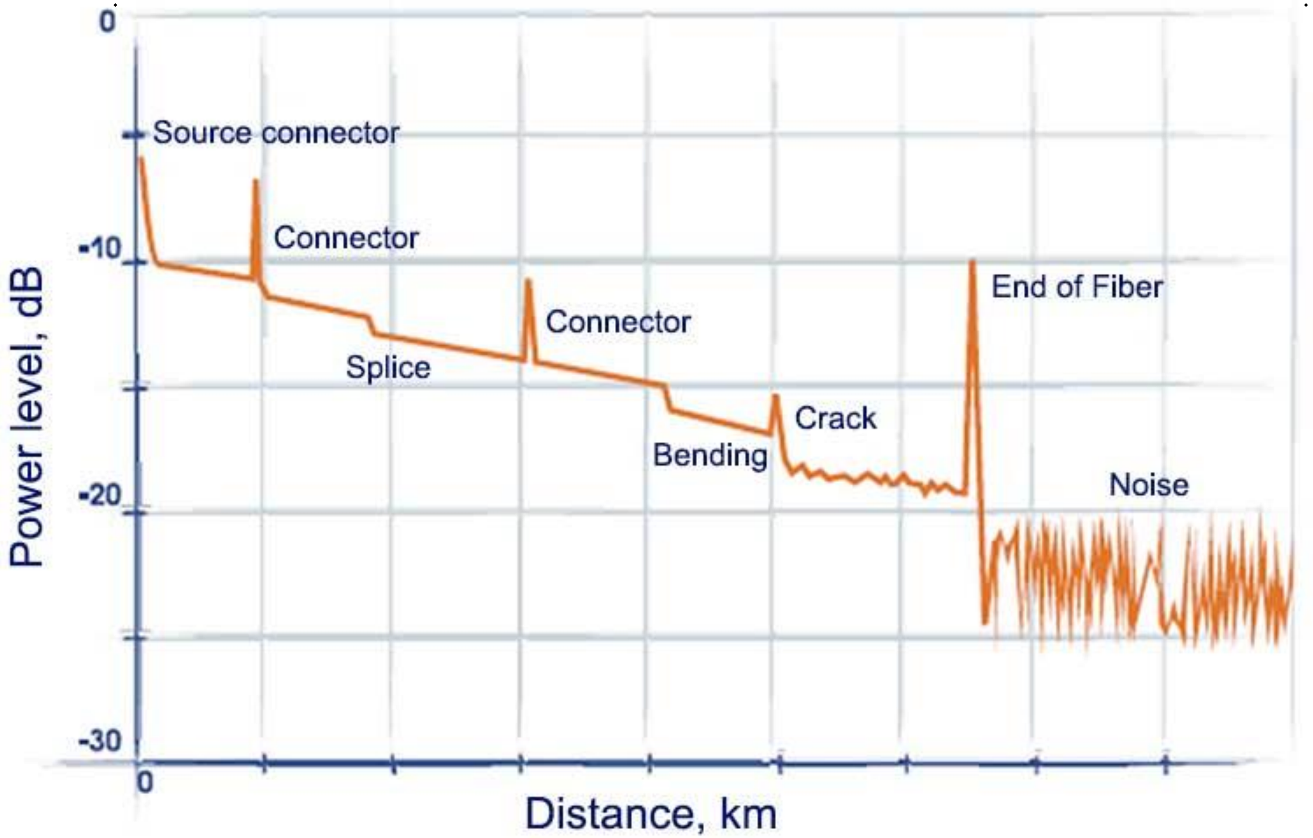
Cisco Photonics, Vimercate, Italy
DEIB - Politecnico di Milano, Milan, Italy

Engineering Applications of Neural Networks (EANN), 2021

Optical Time Domain Reflectometer (OTDR)

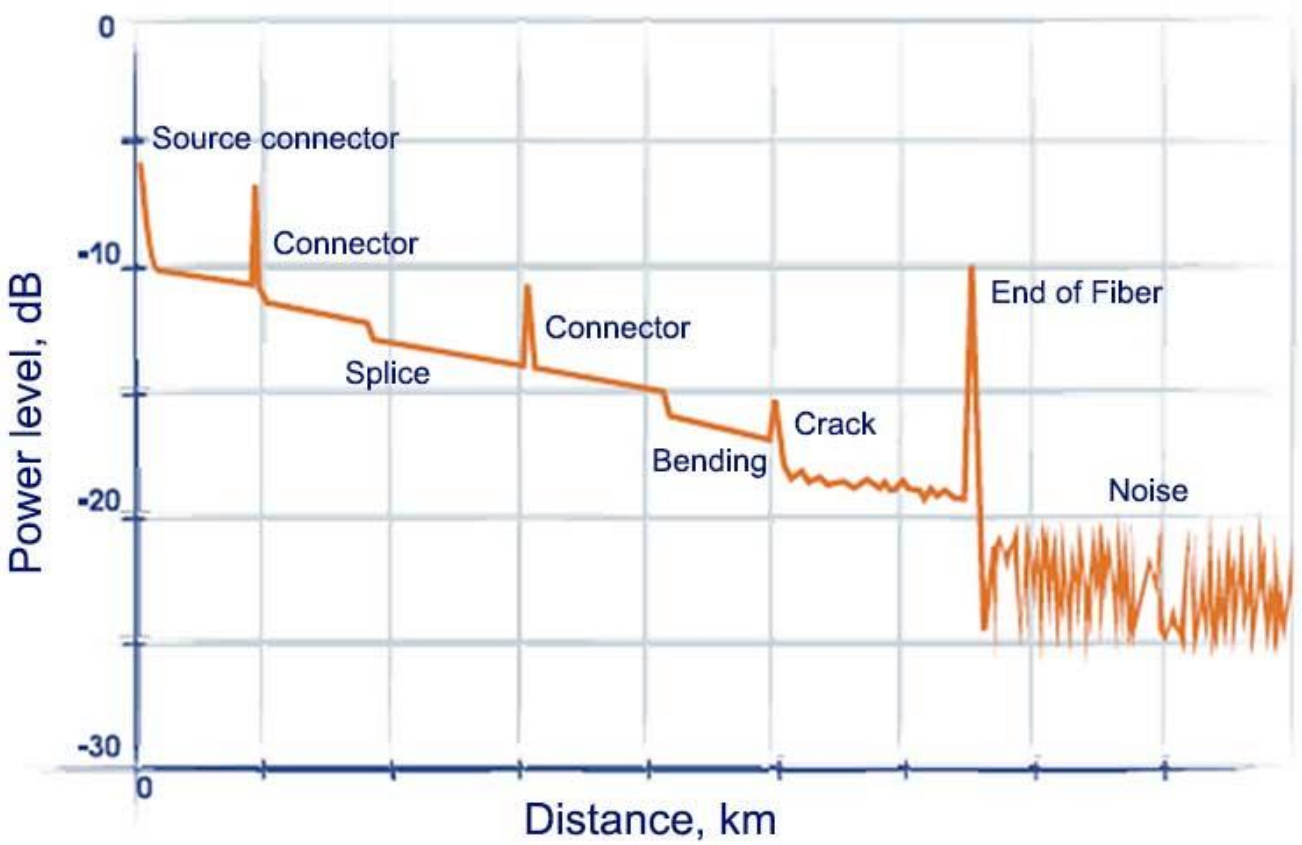
Optical Time Domain Reflectometer (OTDR)



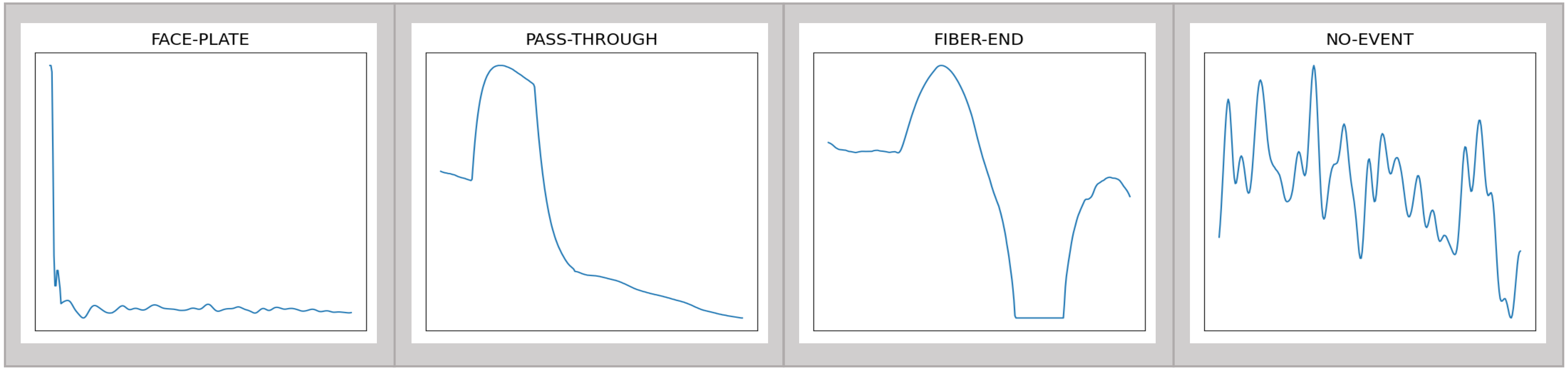


OTDR Traces

- Fiber optic performance are evaluated by measuring the Optical Return Loss (ORL)
- Allow to create a ''virtual'' picture of the fiber link
- Measure the attenuation of the signal due to the fiber dispersion constant
- Highlight multiple events caused by physical devices or flaws



OTDR Events



- Optic events in OTDR traces are caused by devices or defects present on the link
- Each visual pattern identifies a specific event type
- OTDR vendors provide expert-driven solutions to make events localization process automatic

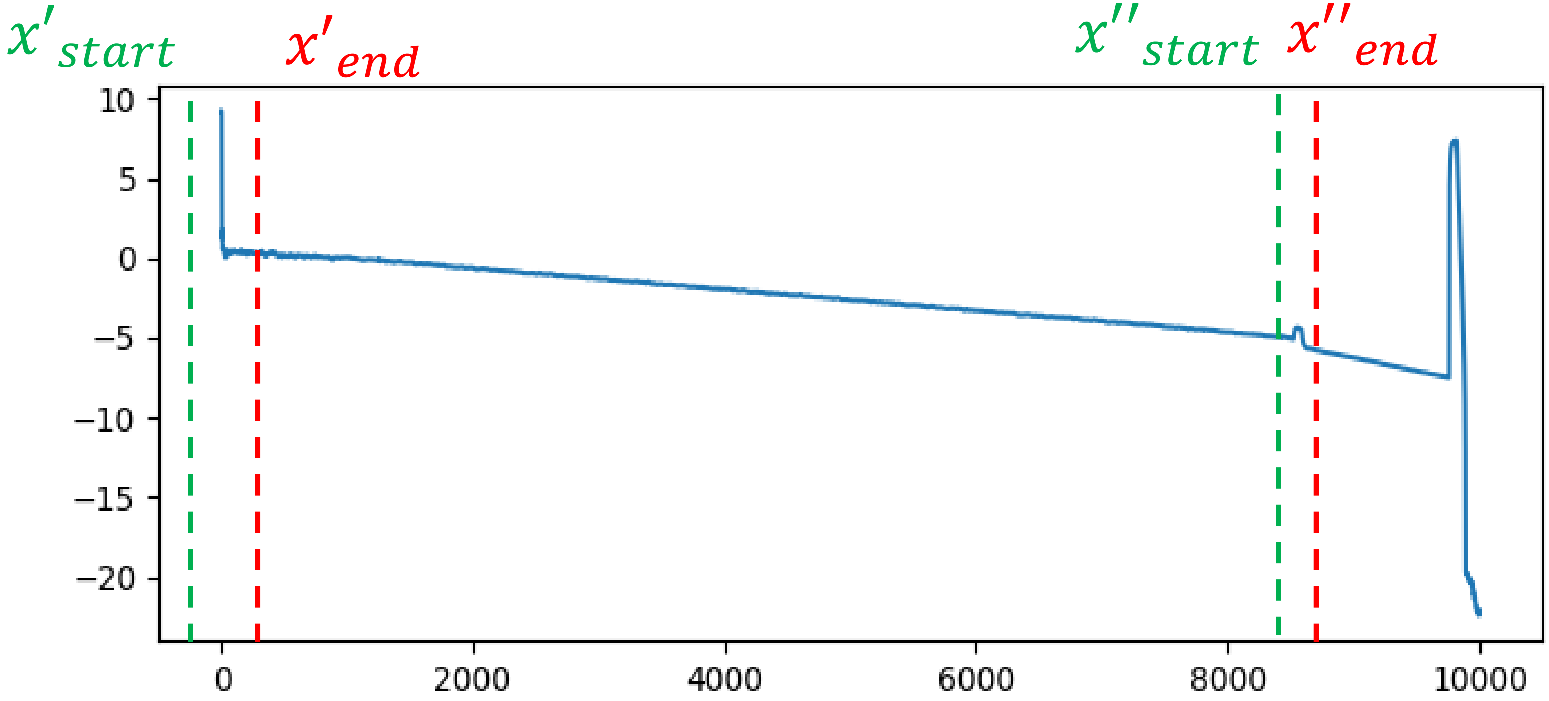
Problem Formulation and Baseline Solution

Problem Formulation

Each OTDR Trace is encoded as a vector $\boldsymbol{T} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \dots, \boldsymbol{x_n}\}$, and might contain

Any number of events, represented by the triplet $\boldsymbol{e} = (\boldsymbol{y}, \boldsymbol{start}, \boldsymbol{end})$

Where: $\boldsymbol{y} \in \{\boldsymbol{Face\text{–}Plate}, \boldsymbol{Fiber\text{–}Cut}, \boldsymbol{Fiber\text{–}End}, \boldsymbol{Pass\text{–}Through}\}$

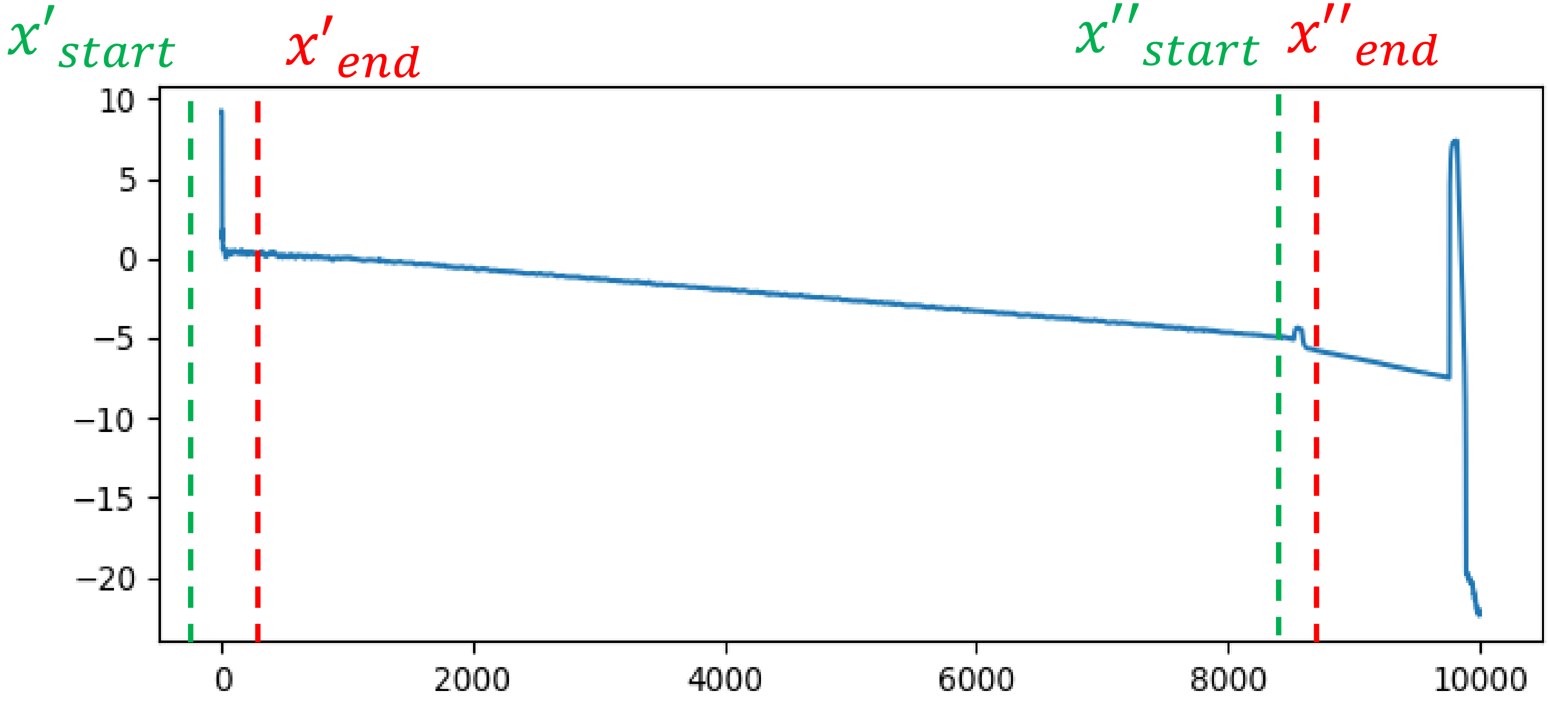


Problem Formulation

Our goal is to design a model able to detect each event in an input trace $\boldsymbol{T}$

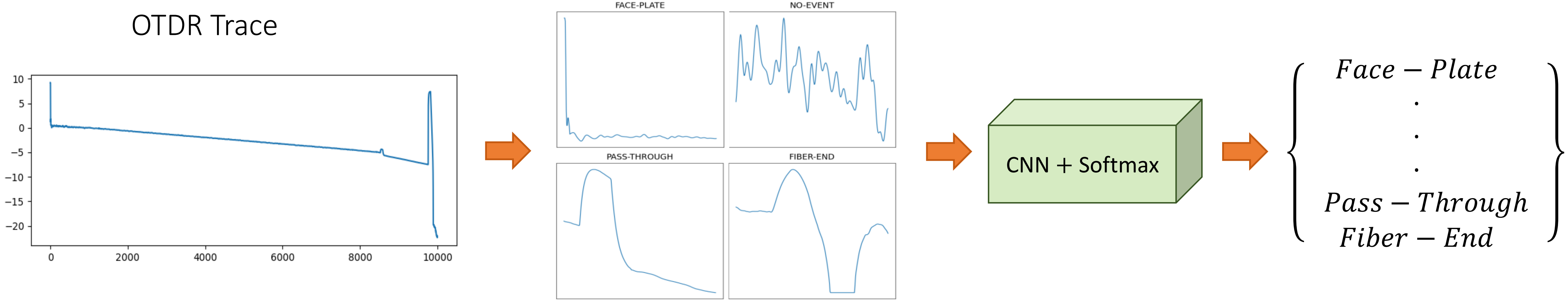
To train our model we assume a labeled training set $\boldsymbol{D}$ containing $\boldsymbol{N}$ traces

Where each trace is associated with the set of annotated events $\boldsymbol{E}$ over the trace $\boldsymbol{T}$



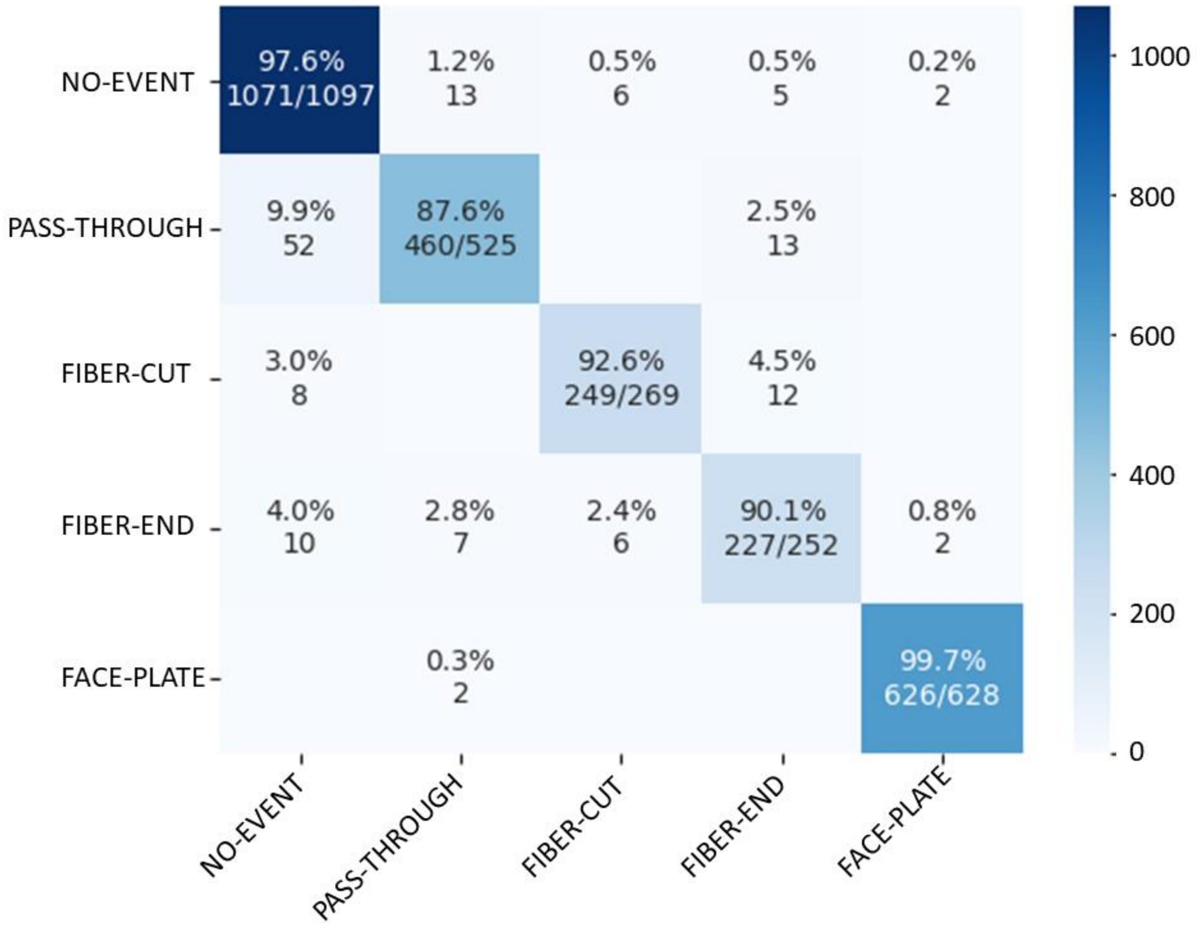
Sliding Window Classifier (Baseline)

- Takes as input windows extracted from an OTDR trace
- Applied on fixed-size windows at inference time
- CNN Classifier trained to predict an event type for each window



Sliding Window Classifier - Drawbacks

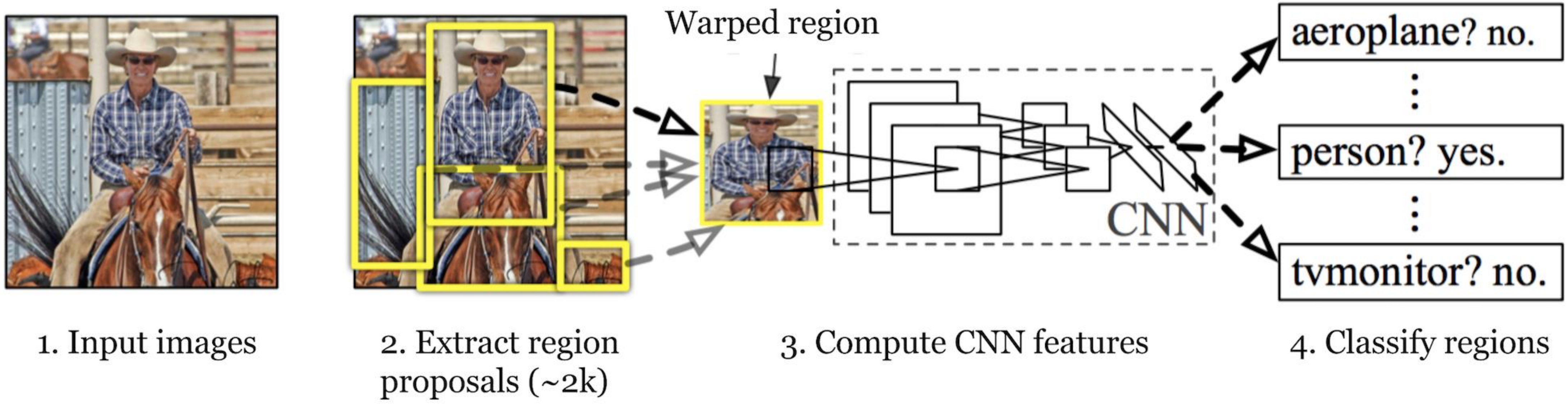
- Requires an additional step to split OTDR traces into windows
- Limited to single-scale events resolution
- Works under the assumption that each window includes at most one event
- Does not consider events position
- Does not share computations



Introduction to Object Detection

Deep Neural Network For Object Detection in Images: R-CNN

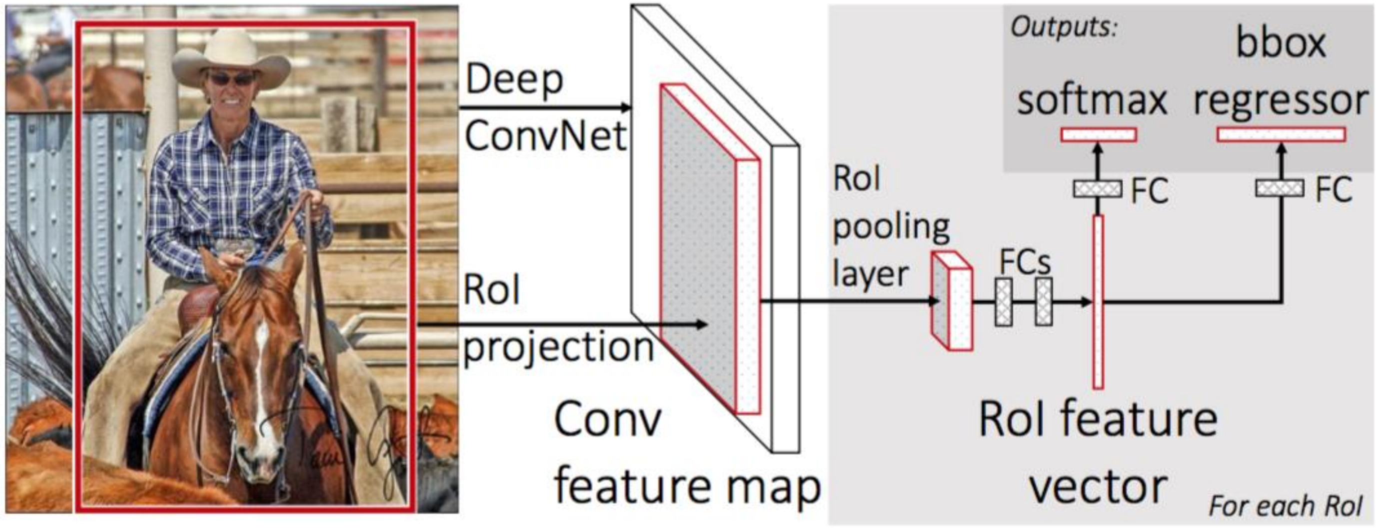
- A Region Proposals algorithm extract region proposals from the input image
- Then a Convolutional Neural Network is used to compute a latent representation of each region
- CNN Features are fed as input to an SVM classifier that predicts the object class



R-CNN Architecture - **Rich feature hierarchies for accurate object detection and semantic segmentation**

Deep Neural Network For Object Detection in Images: Fast R-CNN

- Introduce the concept of Region of Interest (RoI)
- The Region Proposals are projected onto the CNN feature map
- RoI Pooling extract a feature vector for each RoI projected on the CNN feature map



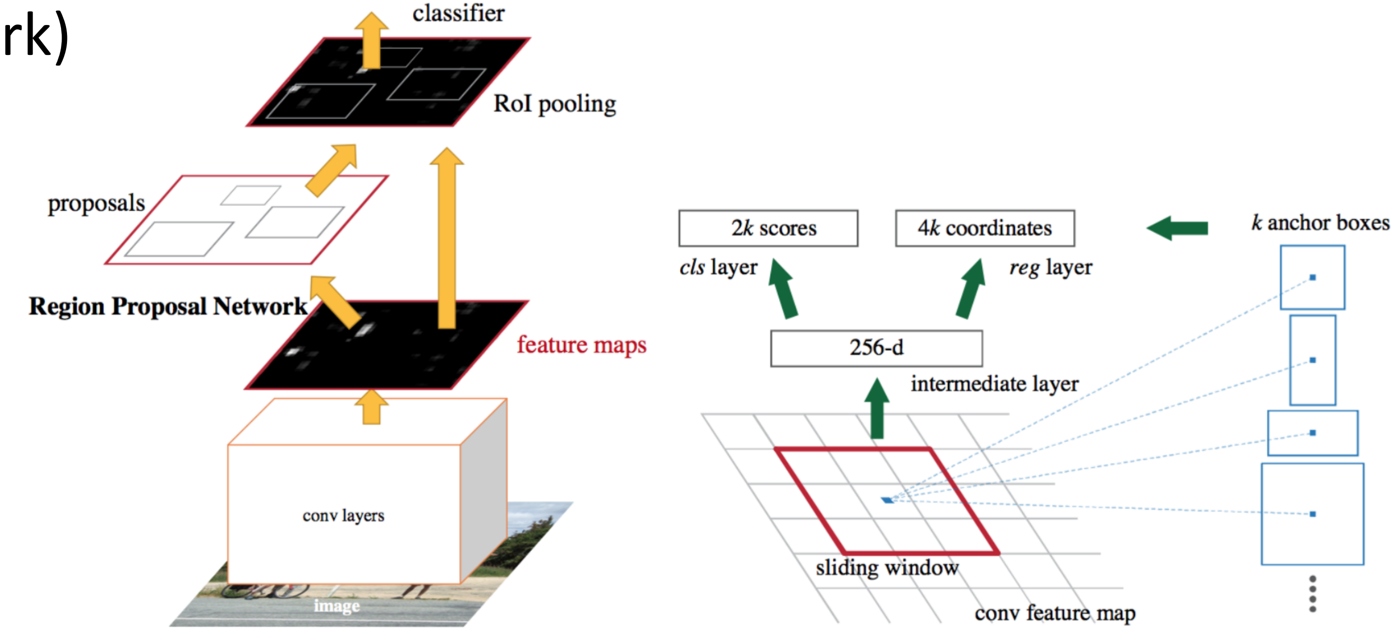
Fast R-CNN Architecture - **Fast R-CNN**

Deep Neural Network For Object Detection in Images: Faster R-CNN

- Introduce the concept of Region Proposal Network (RPN)

- The Region Proposals are computed from the CNN feature map

- RPN and Detection Head share the same convolutional layers backbone (Features Extractor Network)

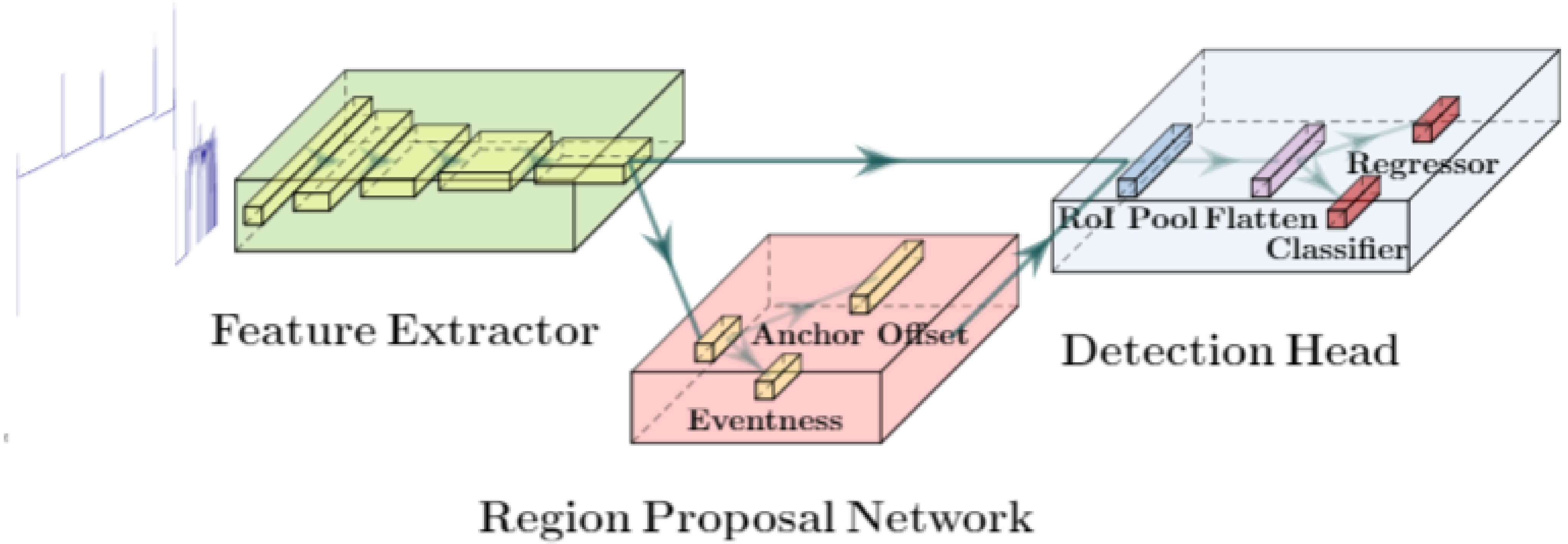


Faster R-CNN Architecture - **Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks**

Deep Neural Network for OTDR Events Detection

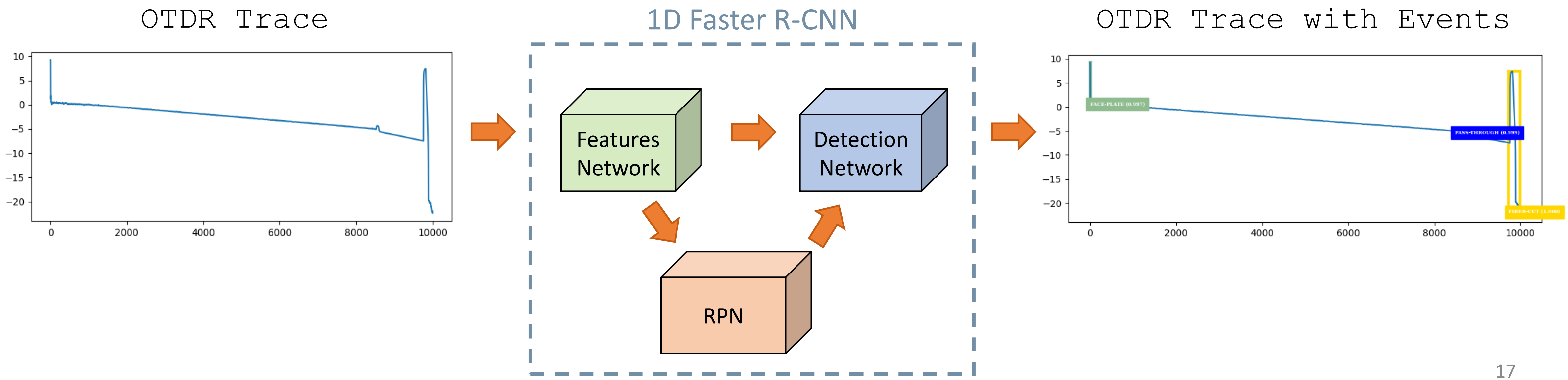
Deep Neural Network For Optical Events Detection

- Inspired by Faster R-CNN
- Designed to process 1D inputs and detect specific shape pattens in 1D data
- Composed of 3 Convolutional Neural Networks

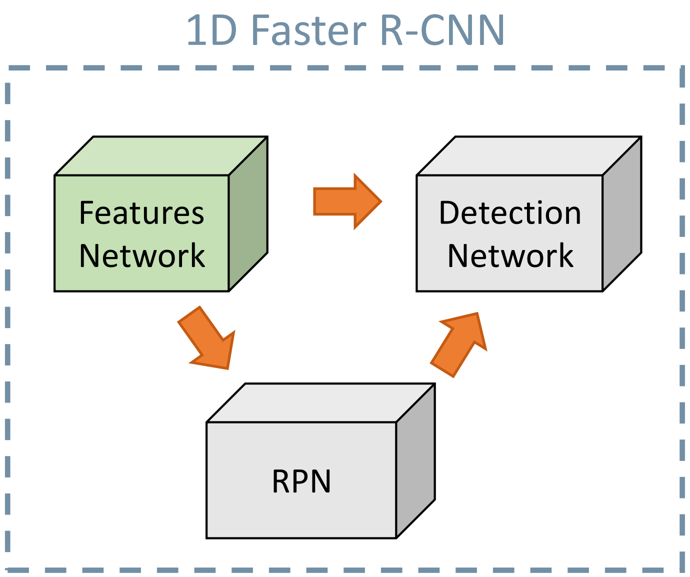


Deep Neural Network For Optical Events Detection

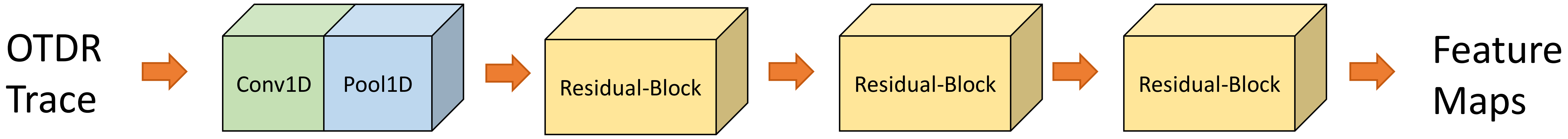
- Inspired by Faster R-CNN
- Designed to process 1D inputs and detect specific shape pattens in 1D data
- Composed of 3 Convolutional Neural Networks



Features Extractor Network



- Takes as input the entire OTDR trace
- Outputs a **latent representation** of the input
- Pre-Trained on Sliding Window Classification
- Inspired by ResNet Architecture



Region Proposal Network

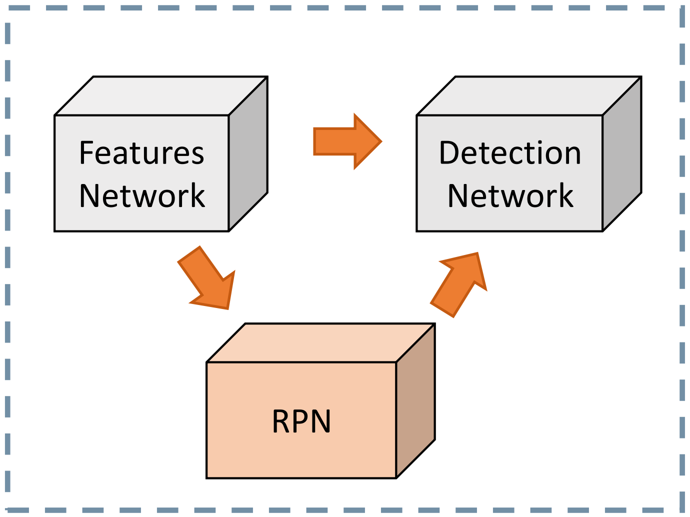
- Initialized by Features Extractor Network

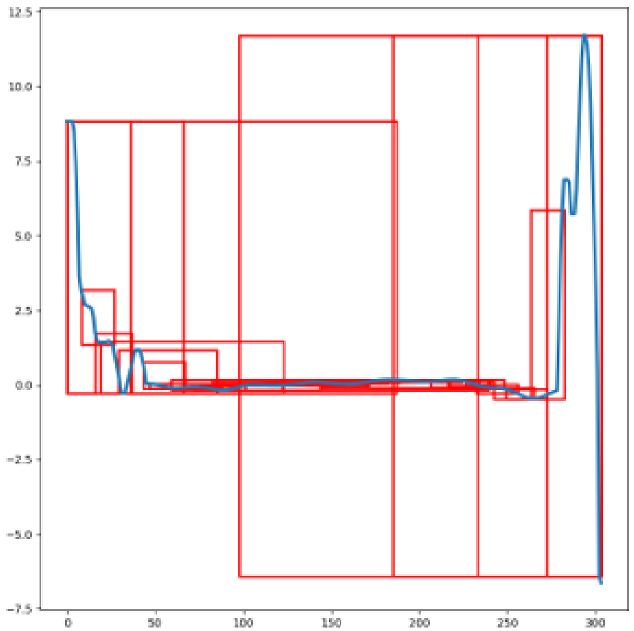
- Generate a set of **multi-scale region proposals** for each spatial location in the *input feature maps*

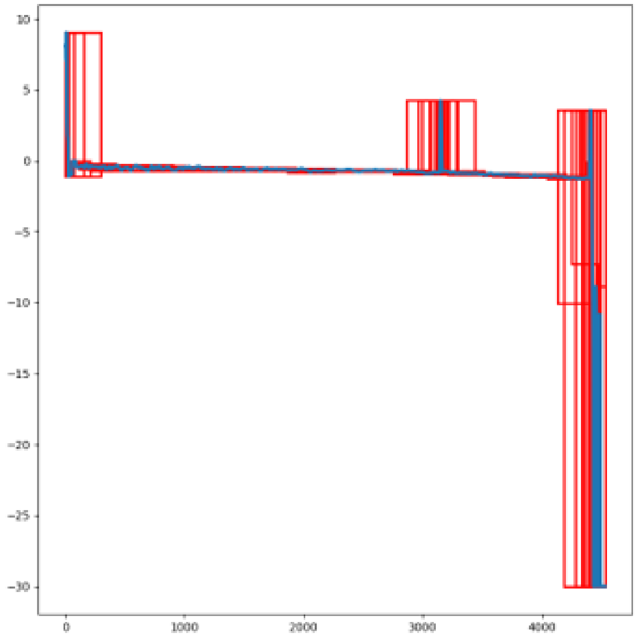
- Each proposal is associated with an ***eventness score***

- Translation invariant approach

1D Faster R-CNN

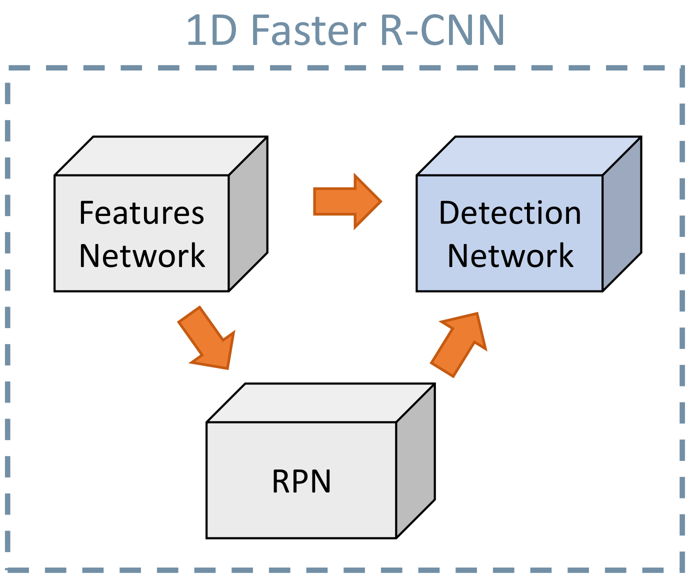


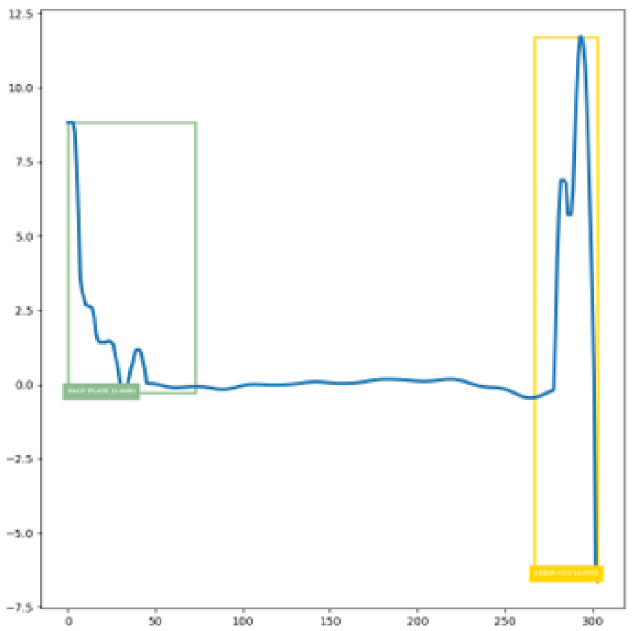


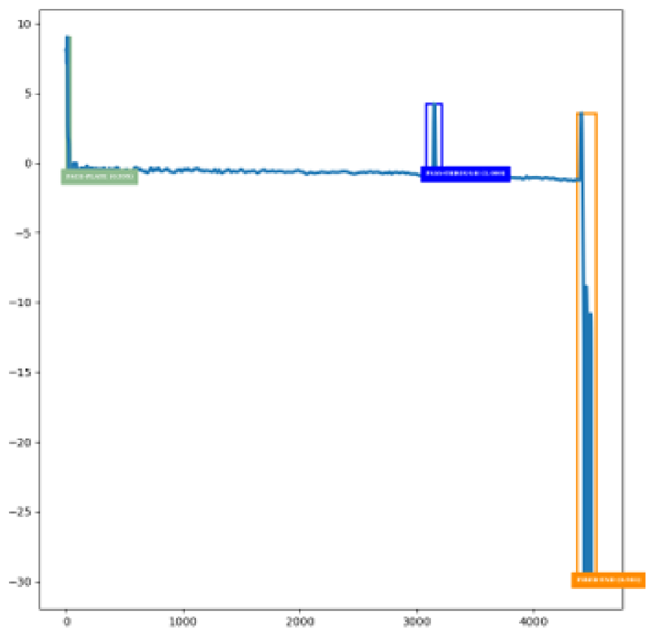


Detection Head

- Initialized by Features Extractor and Region Proposal Networks
- Extract Region of Interest (RoIs) from the input feature maps and region proposals
- Classify each RoI with an **event type**
- Refine RoI coordinates to match a true **event location**







Training

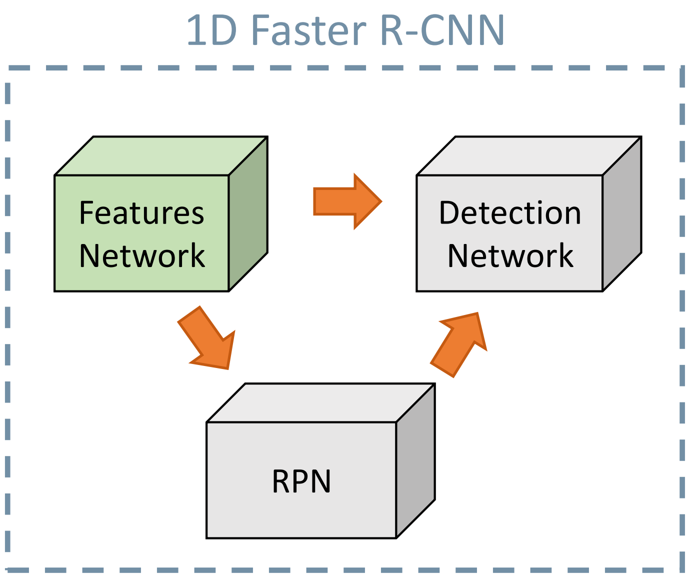
Training follows Faster R-CNN *alternate training procedure:*

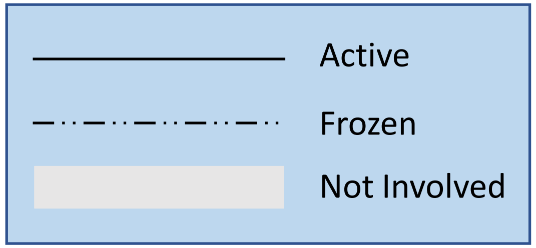
1. Training Region Proposal Network (RPN) layers initialized by the Features Extractor Network, keeping its weights fixed

2. Training Detection Head layers using proposals from RPN trained at step 1

3. Train RPN layers initializing the Feature Network with weights from step 2

4. Training unique layers of the Detection Head, using proposals from RPN trained at step 3, and backbone from step 2





Training

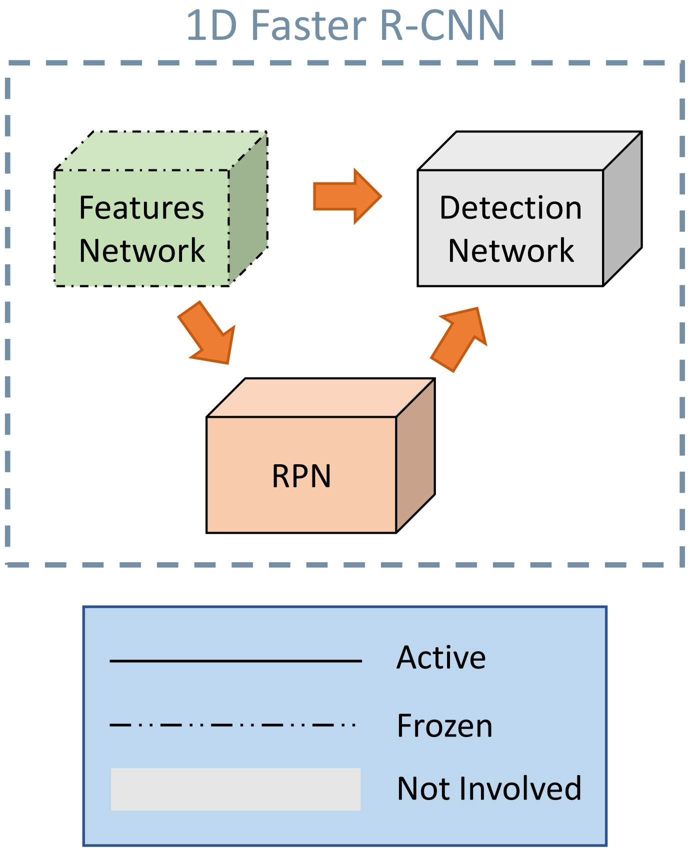
Training follows Faster R-CNN *alternate training procedure:*

1. **Training Region Proposal Network (RPN) layers initialized by the Features Extractor Network, keeping its weights fixed**

2. Training Detection Head layers using proposals from RPN trained at step 1

3. Train RPN layers initializing the Feature Network with weights from step 2

4. Training unique layers of the Detection Head, using proposals from RPN trained at step 3, and backbone from step 2



Training

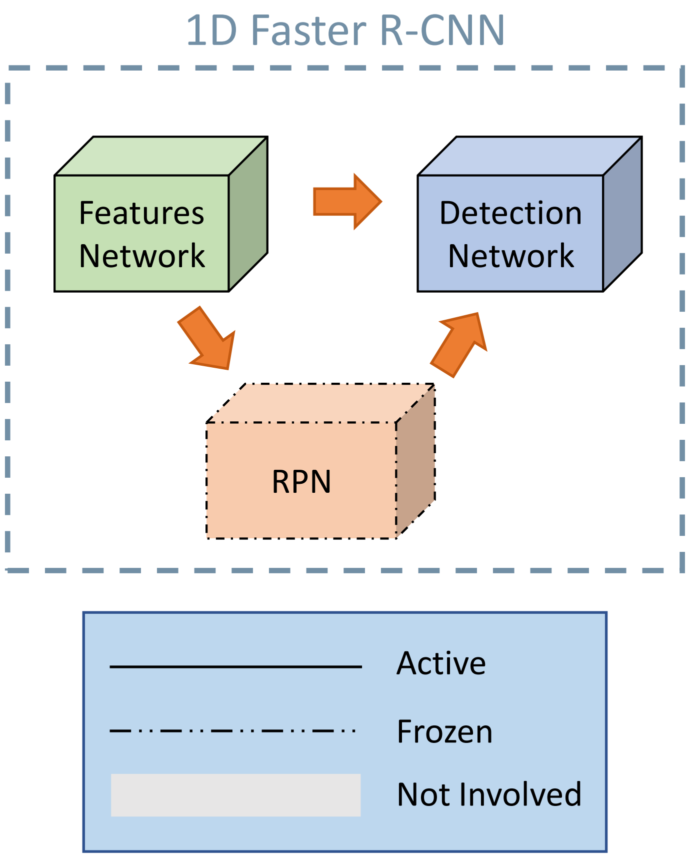
Training follows Faster R-CNN *alternate training procedure:*

1. Training Region Proposal Network (RPN) layers initialized by the Features Extractor Network, keeping its weights fixed

2. **Training Detection Head layers using proposals from RPN trained at step 1**

3. Train RPN layers initializing the Feature Network with weights from step 2

4. Training unique layers of the Detection Head, using proposals from RPN trained at step 3, and backbone from step 2



Training

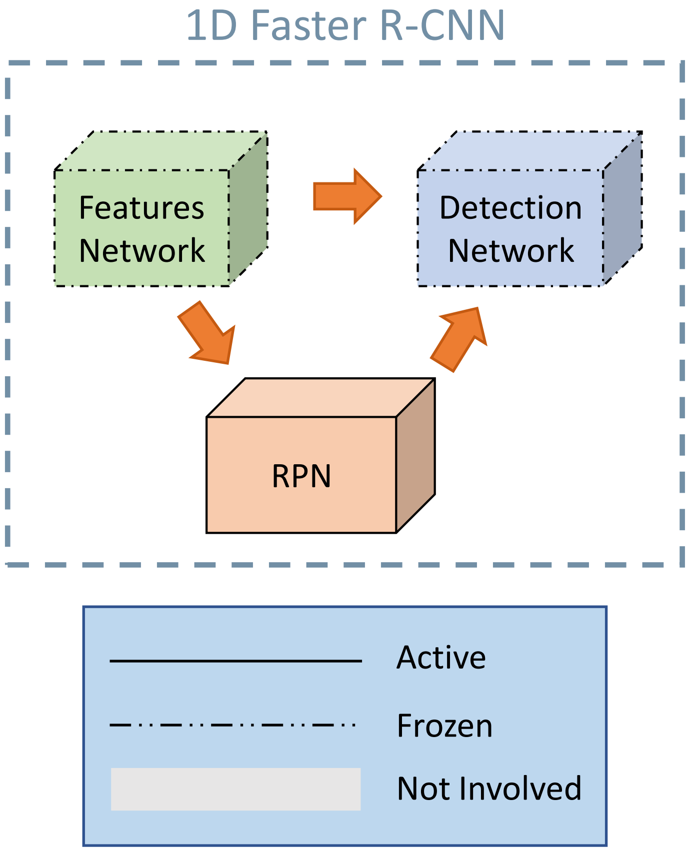
Training follows Faster R-CNN *alternate training procedure:*

1. Training Region Proposal Network (RPN) layers initialized by the Features Extractor Network, keeping its weights fixed

2. Training Detection Head layers using proposals from RPN trained at step 1

3. **Train RPN layers initializing the Feature Network with weights from step 2**

4. Training unique layers of the Detection Head, using proposals from RPN trained at step 3, and backbone from step 2



Training

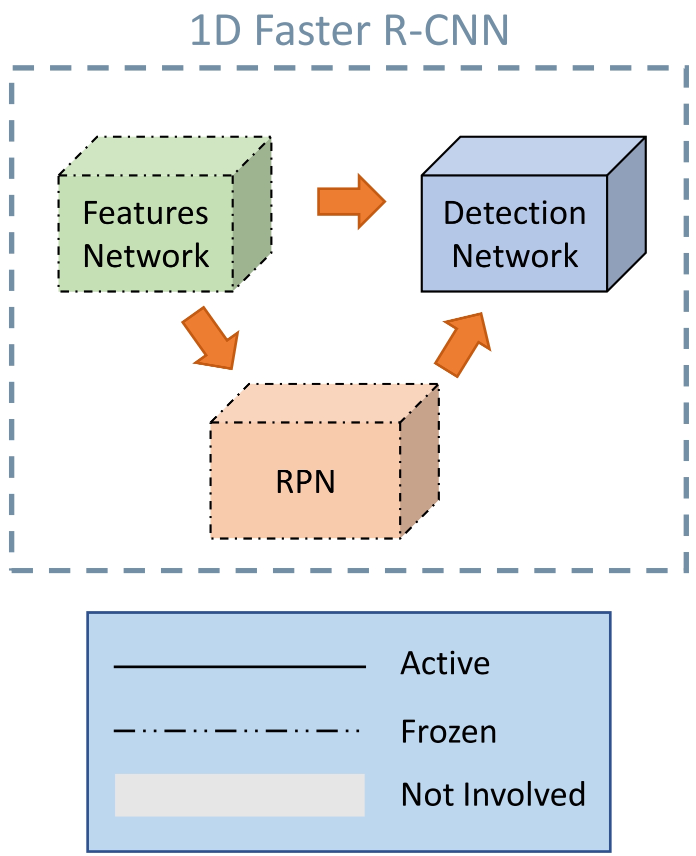
Training follows Faster R-CNN *alternate training procedure:*

1. Training Region Proposal Network (RPN) layers initialized by the Features Extractor Network, keeping its weights fixed

2. Training Detection Head layers using proposals from RPN trained at step 1

3. Train RPN layers initializing the Feature Network with weights from step 2

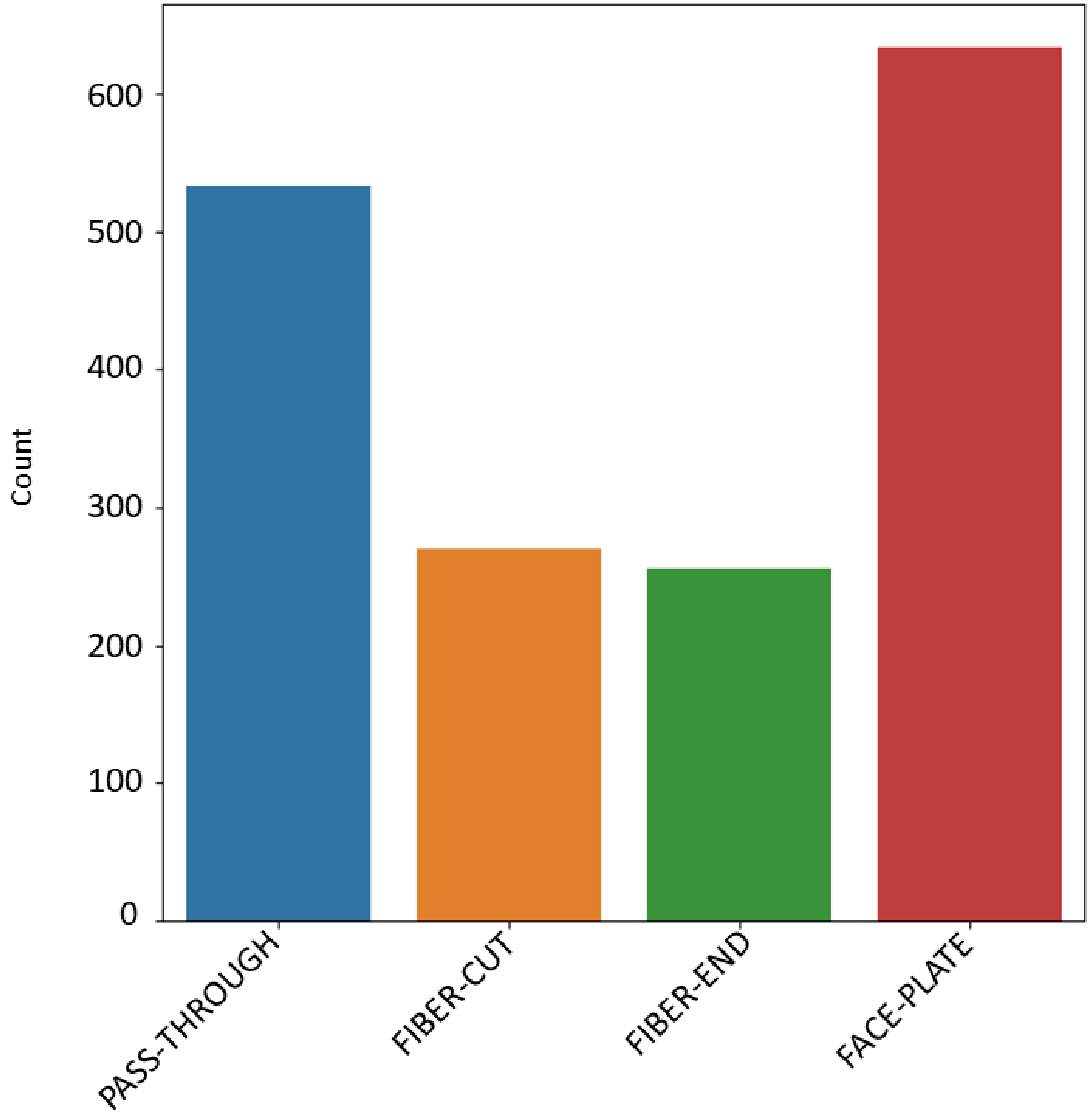
4. **Training unique layers of the Detection Head, using proposals from RPN trained at step 3, and backbone from step 2**



Experiments and Comparison with Cisco NCS-1K

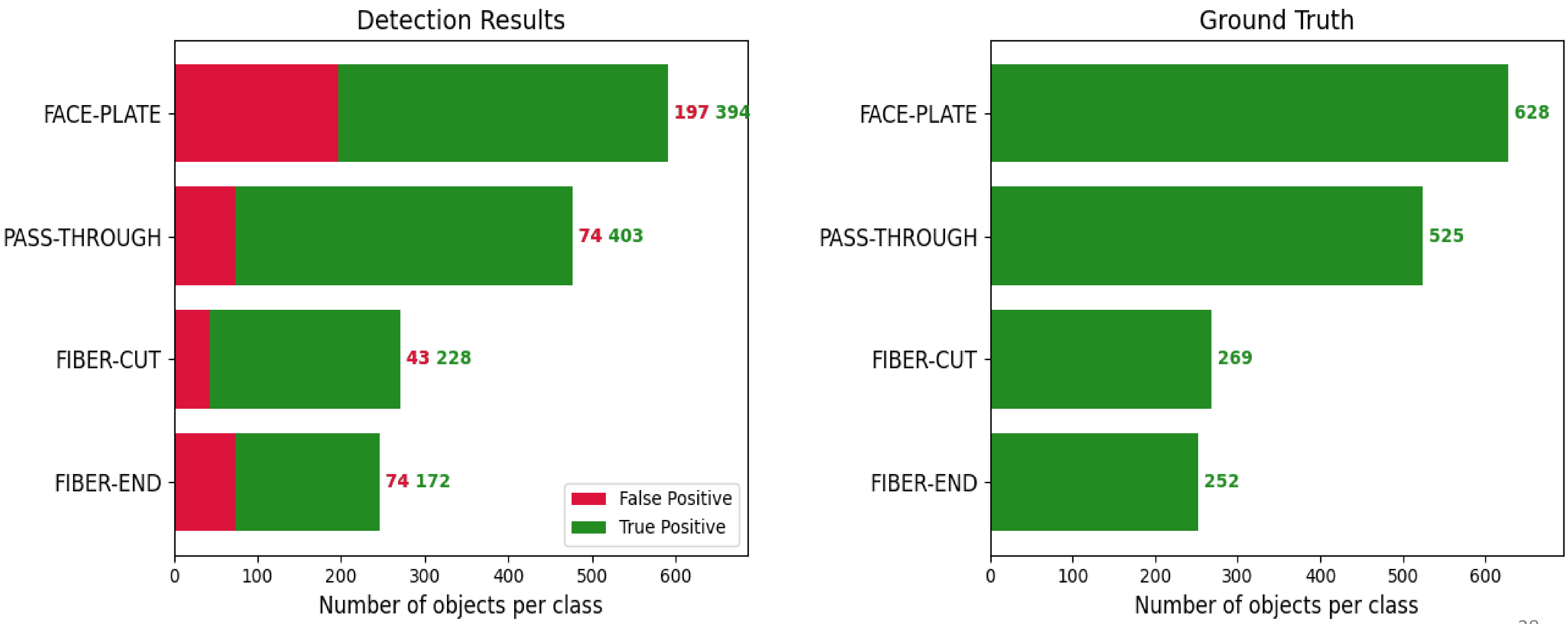
Dataset

- 628 OTDR traces acquired from several fiber setup
- 1674 events labeled together with Cisco experts
- OTDR events belonging to the most common cases
- Ongoing activities to extend the dataset with “less common” events



Results

- Mean Average Precision (mAP) score of 85%



Comparison with Cisco NCS-1K embedded OTDR

- Expert Driven Approach
  - Limited to few event types
  - High number of false positives
  - Needs of human expert

- Machine Learning Approach
  - Trainable on any set of event types
  - Precise localization and classification
  - Completely automatic

Approach	mAP score (%)	Reflective Events	Non-Reflective Events	Fiber Termination
Expert Driven	50.61	25.43	77.16	49.25
Ours	**77.43**	**76.97**	**78.40**	**76.91**

Conclusion and Future Work

- Conclusions
  - Accurate classification and precise localization of OTDR events
  - Models implemented in Python and TensorFlow, Inference in TensorFlow Lite
  - Running into a real-world application on Cisco NCS-1K routing platforms
- Future Developments
  - Benchmark results with other detection architectures applied on time-series data
  - Train the algorithm on an extended dataset with larger set of classes (i.e.,

Event-Detection Deep Neural Network for OTDR Trace Analysis

Thanks For Your Attention