Investigating Failure Patterns in Particle Accelerator Infrastructures with Explainable Deep Learning

Thomas Cartier-Michaud, Lukas Felsberger, Andrea Apollonio, Andreas Müller, Benjamin Todd, Dieter Kranzlmüller

Structure

Introduction

Methodology

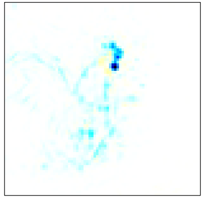
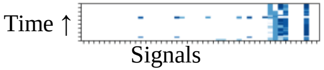
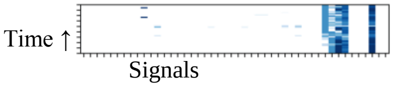
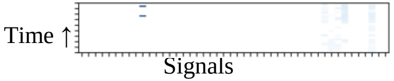
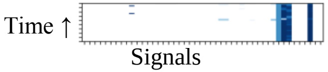
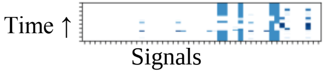
Experiments and Results

Outlook

Introduction

- Large infrastructures hard to diagnose
- ML approaches scale to very high dimensionality
- Can they be useful to help operators?

Introduction - Idea

Data		**Data Driven Model Prediction**		**Explanation**
Input (image of animal)	**Label (species)**	**Input**	**Prediction**	
	cock hammerhead hare		Cock	
Input (past monitoring signals)	**Label (leading to alarm in future?)**	**Input**	**Prediction**	
	No		Yes	
	Yes			
	No			

Introduction - Challenges

Image recognition

- Large data sets
- Ground truth usually known
- Explanation easy to interpret

Failure pattern mining

- Small data sets
- Ground truth hard to come by
- Explanation interpretable?

Methodology

Adapting well established machine learning approach:

Steps	**Data Collection**	**ML Problem Formulation**	**Filtering & Subsampling**	**Algorithm Selection**	**Training**	**Model Selection**
Computation	LASER (SQL), CALS (pytimber)	Python	Python	Literature research	Sklearn, keras, tensorflow	Expert discussions
Duration (indicative)	3 months	1 month	1 week	1 week	2 months	In progress

Data Collection: LASER

- Centralized service capturing/notifying/storing anomalies for the whole accelerator chain + TI
- Alarms are raised for operators ➔ not an interlock system ➔ need for human (slow) intervention
- 30 fields, of which important ones are:
  - FAULT_FAMILY/_MEMBER_/CODE = pointer to the component and the fault
  - PRIORITY = severity of the fault = 0, 1, 2, 3 ➔ we predict priority 3 alarms, supervised problem
  - SYSTEM_TS = time stamp = events are recorded, no continuous signals
- For PSB (same building / components of the machine): bug investigation for 2018

Year	2018	2017	2016	2015	2014	2013	2012	2011
# lines	235 M	305 K	473 K	235 K	1 767 K	12 K	1 054 K	504 K

LASER description

CERN ALARMS DATA MANAGEMENT: STATE AND IMPROVEMENTS

Data Collection: extraction from LASER

- Accessing the TN using a Virtual Machine (~1 month)

- Retrieving data from SQL database (~2 months: 1.7TB)
  ➔ automation using bash scripts in parallel sessions
  ➔ process “artisanal” as LASER is not meant for such large requests

- Storing the data using a cernbox account (<50GB once zipped)

Data Collection: extraction from CALS

- Additional signals fetched from CALS based on expert recommendation
- Using pytimber
  - Maximal data size per request limited
    - Requires splitting of requests and subsequent merging

Formulation of a Supervised Machine Learning Problem

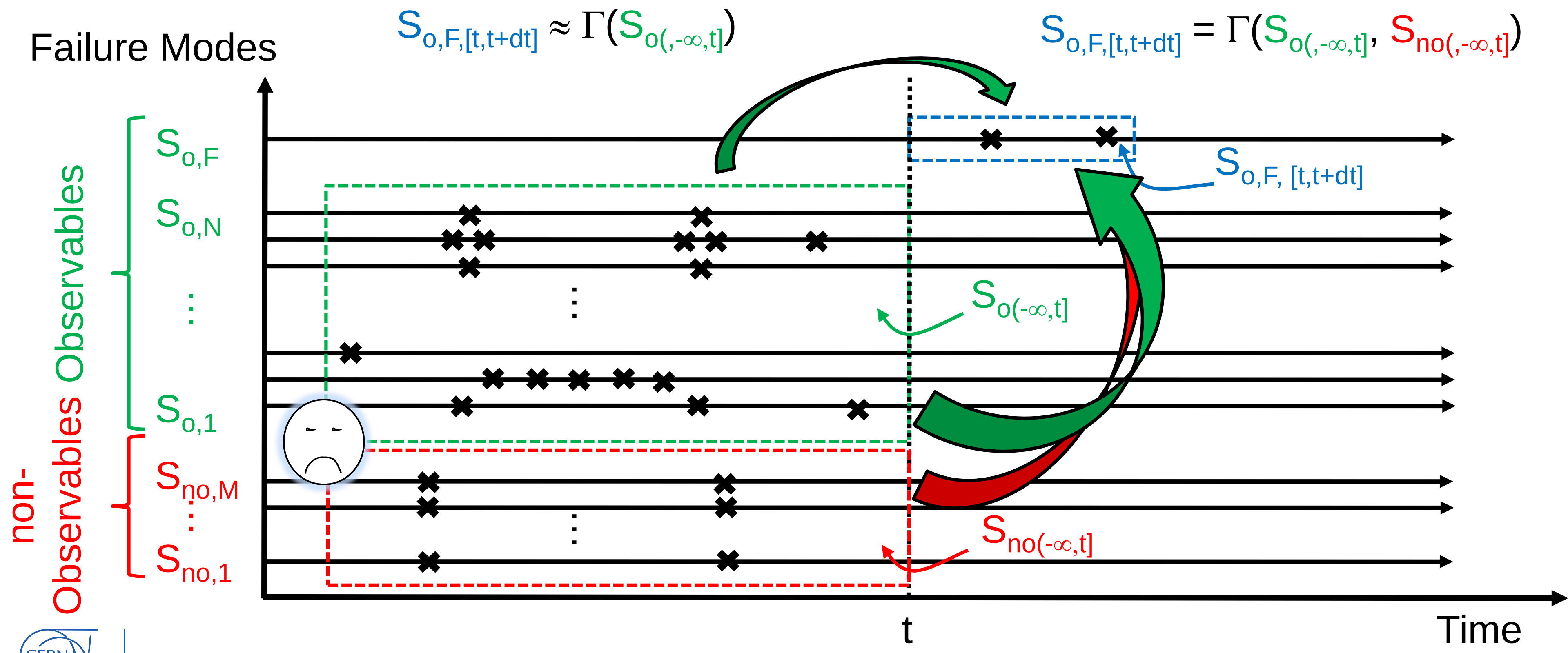
Supervised ML needs:

- Data
  - Input
  - Label/Output
- Model linking in- and outputs
  - Model structure
  - Parameter optimization

We have:

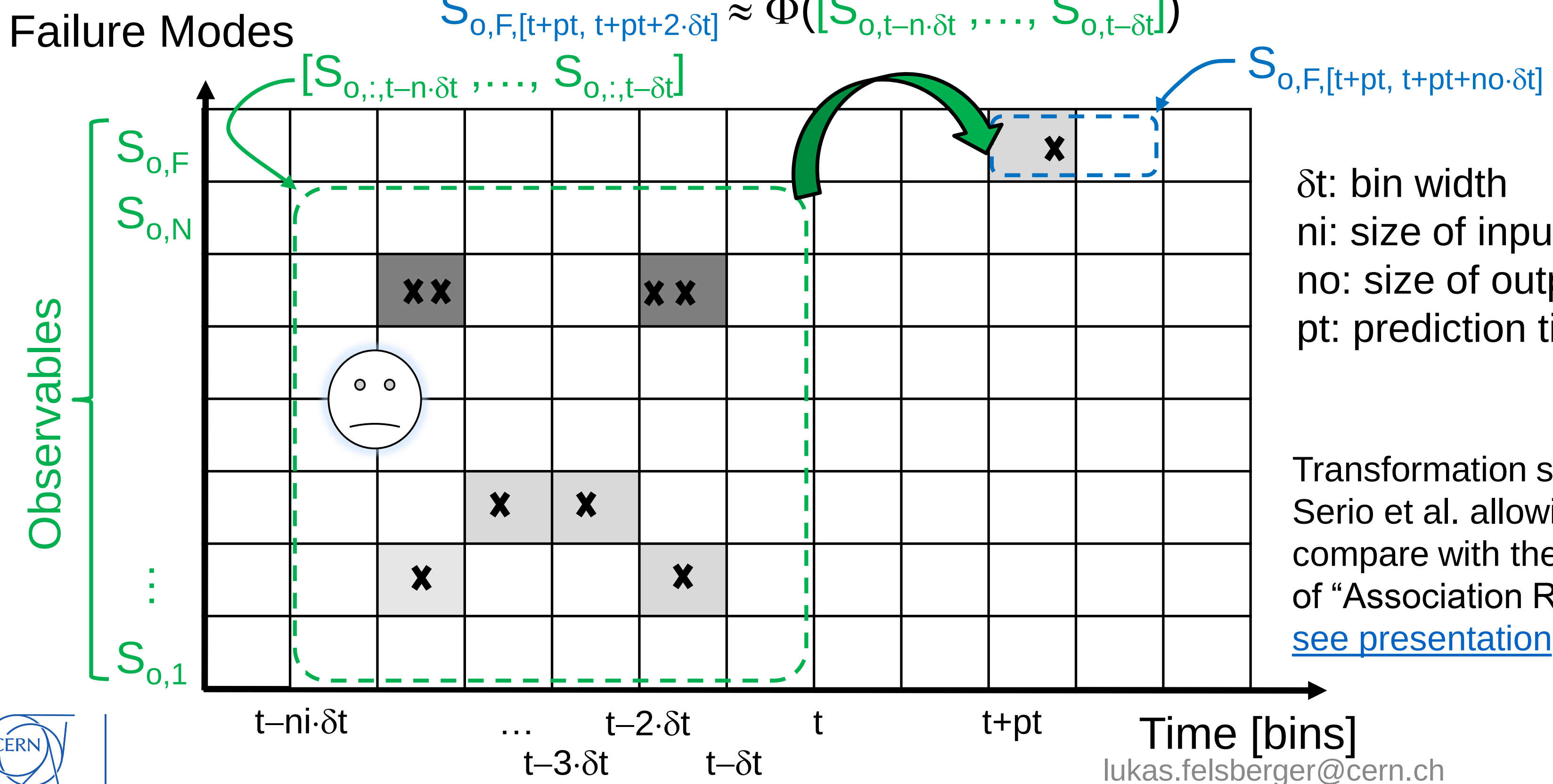
- Alarms in time
  - Alarms in the past
  - Priority 3 alarm in the future
- Model linking past and future alarms
  - Choice of ML models
  - Choice of optimizers

ML Problem Formulation: existence of a model



ML Problem Formulation: Discretization

$S_{o,F,[t+pt,\ t+pt+2\cdot\delta t]} \approx \Phi([S_{o,t-n\cdot\delta t}, \ldots, S_{o,t-\delta t}])$

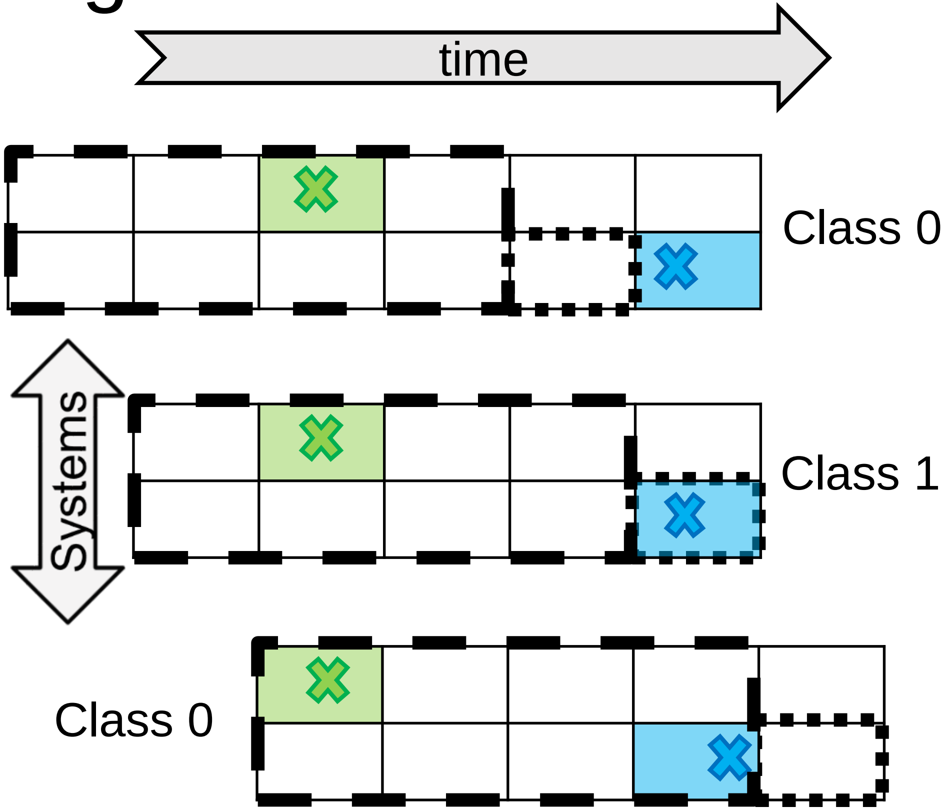


δt: bin width
ni: size of input window
no: size of output window
pt: prediction time

Transformation similar to L. Serio et al. allowing to compare with their approach of “Association Rules Mining” see presentation

Filtering and Subsampling

- Focus on EPC of PSB ➔ reducing the number inputs
- Failures are rare (< 30) ➔
  - Filter out signals with too low or without activity
  - balancing class 0 (no failure) and class 1 (failure) elements by subsampling class 0
  - forcing contrast



Algorithm Selection

- Goal was to test explainable deep learning
  - But will be compared against “traditional” ML algorithms

Traditional ML

(based on popular choice for universal learners):

- SVM with linear kernel
- Random Forest
- K Nearest Neighbour

Deep Learning

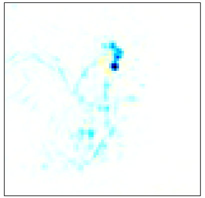
(based on latest review papers for time series problems):

- Fully Convolutional Network
- Fully Convolutional Networks with Dropout Regularization
- time-Convolutional Neural Network

References in paper

Training: Implementation

- python3 + 2 main libraries:
  - Learning: https://github.com/hfawaz/dl-4-tsc
  - Explaining: https://github.com/albermax/innvestigate
  - Implemented in keras and tensorflow



Training: Computation

- Not computationally intensive but many trainings (~100 000) for different meta parameters

  ➔ No use of GPU as reading / transforming / writing would have been the bottleneck

- Generation of thousands of jobs using 1 core each and executed in parallel on CERN Cluster

  ➔ 220 000 cores in the cluster, usage of up to 1000 cores at once

- Limit of 2GB/core the cluster

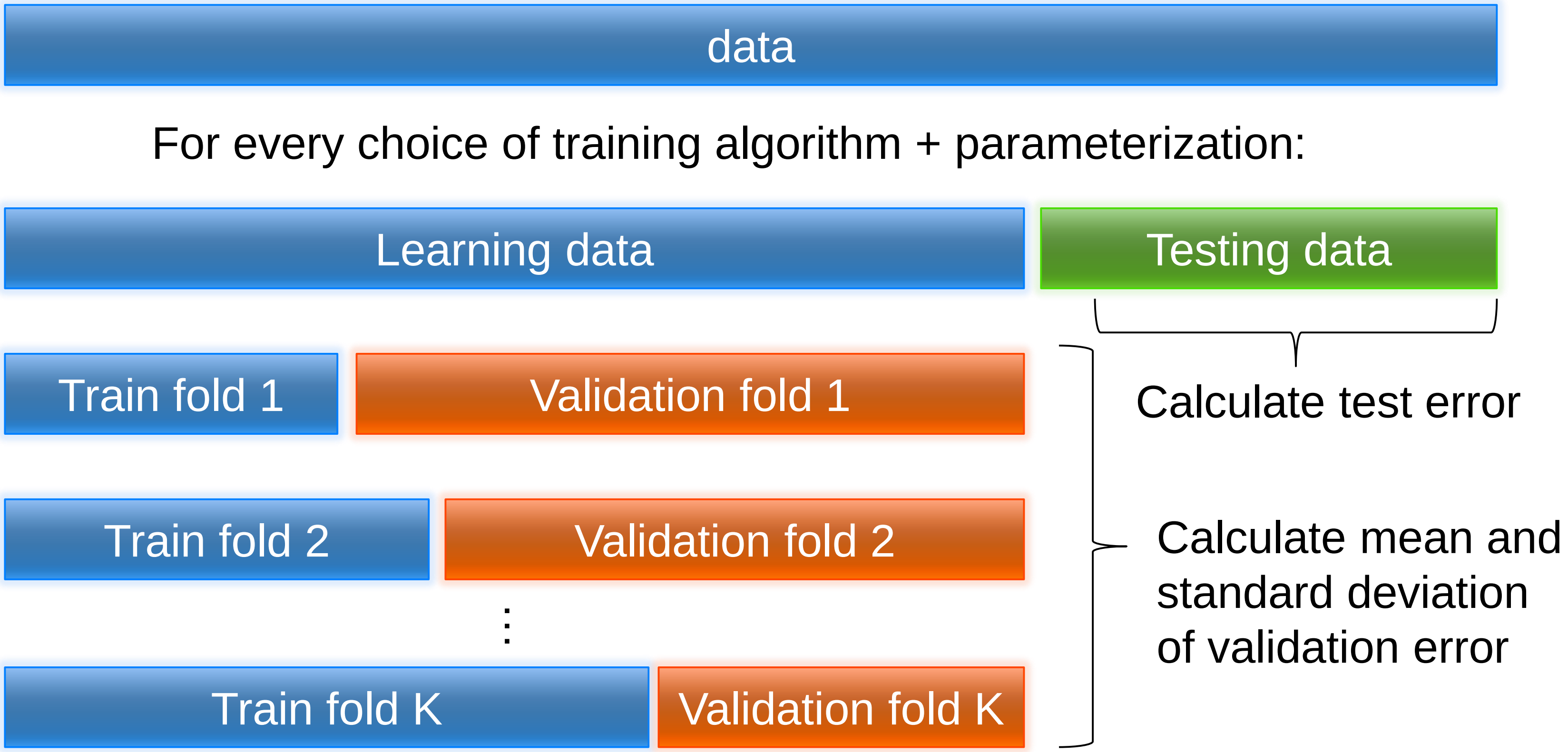
  ➔ Necessity to rewrite the transformation algorithm

- Limit of 100GB in AFS/work

  ➔ Easily reached with more than 100 000 combinations of hyper parameters * 1MB

We underestimated the input/output/postprocessing importance for large scans

Model Selection



Model Selection – (Hyper) Parameters

Steps	Data Collection	ML Problem Formulation	Filtering & Subsampling	Algorithm Selection	Training	Model Selection
# Hyper-Paramters	1	4	4	2		3
# ML Parameters					<10^6	
Parameter Cardinality	20	4^4	2	10	Infinite	4

lukas.felsberger@cern.ch

Experiments and Results

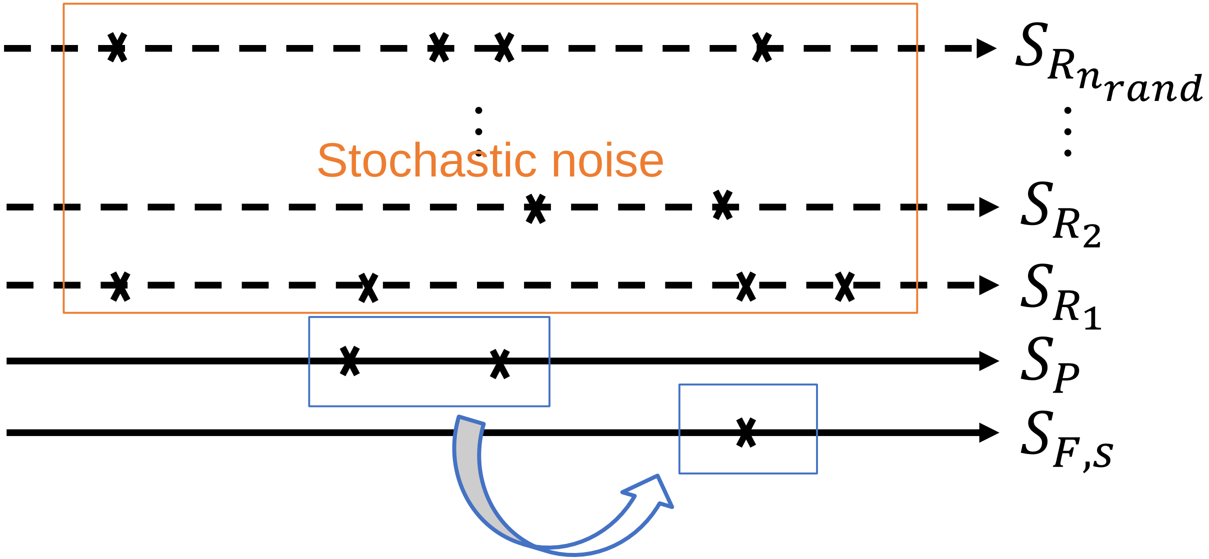
Goal: Use framework to predict and explain accelerator failures.

Using new framework on new problems requires iterative approach:

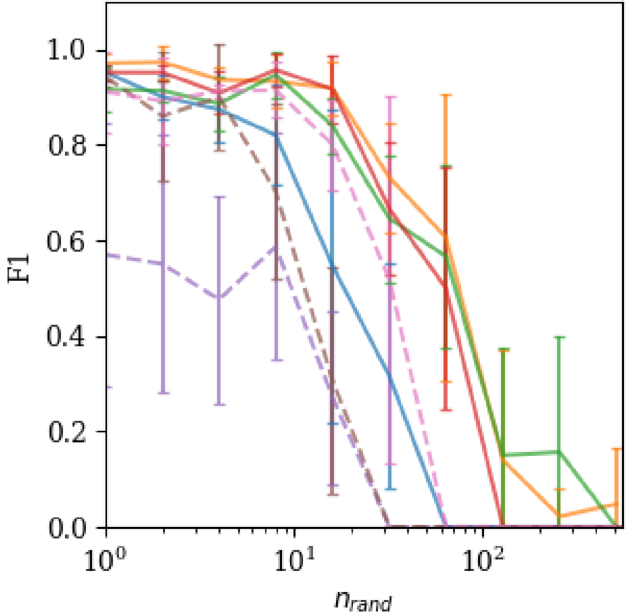
1. Verify and test new framework using synthetic data with known ground truth
2. Attack new problem (predict+explain faults in PSB) with verified framework

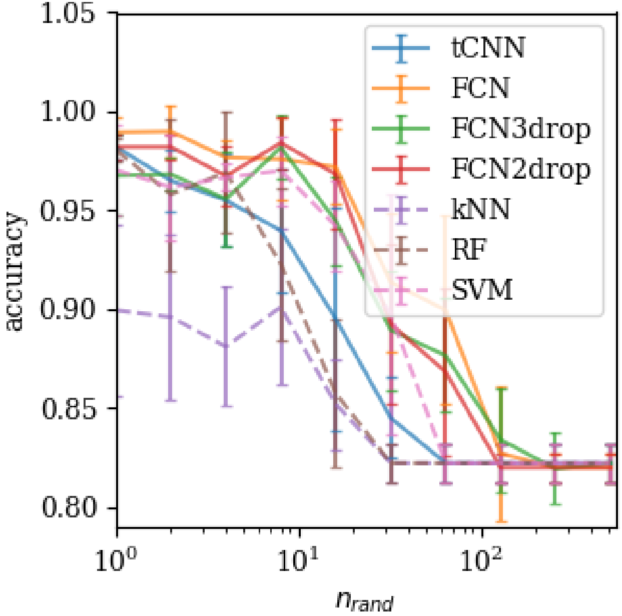
Synthetic Data Experiment

Verify and test new framework using synthetic data with known ground truth:



Synthetic Data Experiment





F1 = 2 * (Precision * Sensitivity) / (Precision + Sensitivity)

Accuracy = (TP + TN) / All Predictions

Sensitivity (Recall) = TP / (FN + TP)

Precision = TP / (TP + FP)

	Predicted Negative	Predicted Positive
Actual Negative	TN	FP
Actual Positive	FN	TP

https://towardsdatascience.com/evaluating-machine-learning-classification-problems-in-python-5-1-metrics-that-matter-792c6faddf5

Synthetic Data Experiment - Discussion

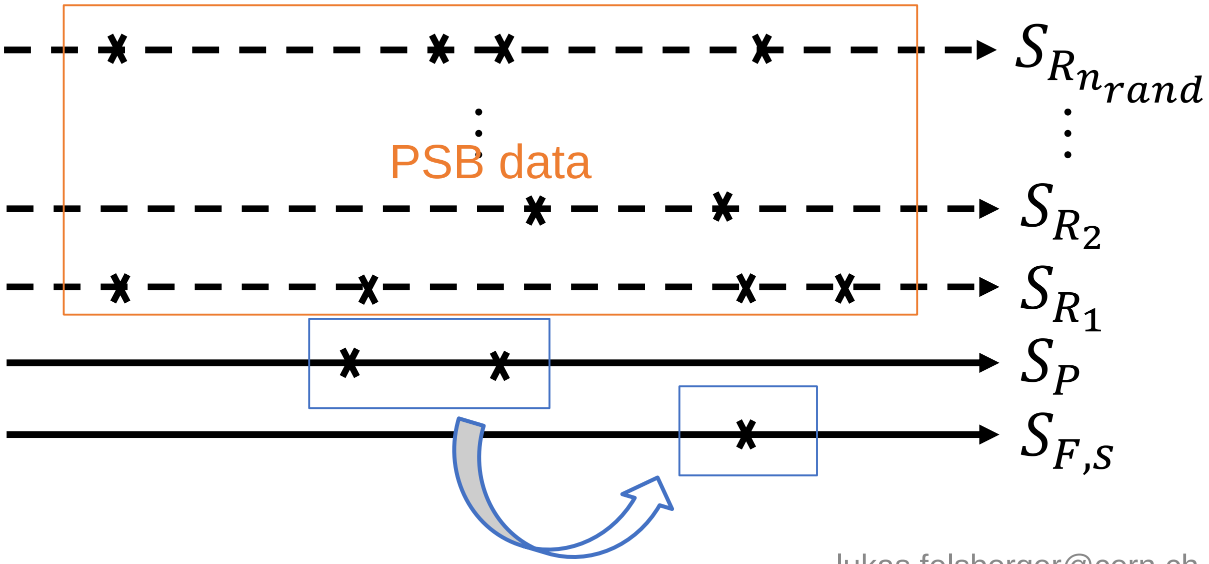
- ✓ Learned predictive models from less than 10 training examples for up to 64 noise channels
- ✓ Framework works in principle

Real Data Experiments

- Test new problem (predict+explain faults in PSB) with validated framework
- Data
  - LASER alarms for power converters in PSB
  - CALS logging
    - External condition signals
    - PSB beam destination signal

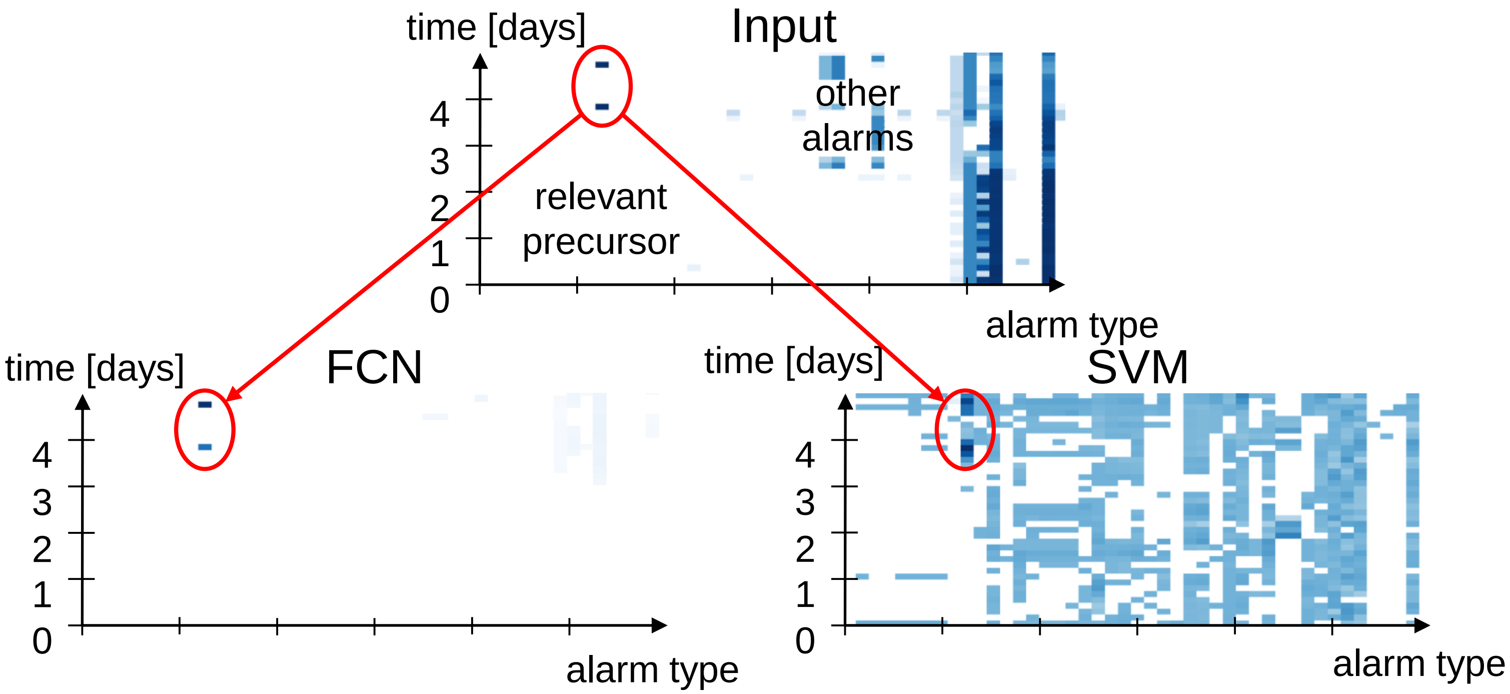
Mixing Synthetic and Real Data

- Preliminary step: Redo previous experiment and replace noise by real data
- Idea: if there was a well defined pattern in PSB data, would we detect it?



Mixing Synthetic and Real Data

- Learned predictive models from less than 10 training examples and for 43 PSB signals (as noise)
- Discovers correct synthetic pattern:



Mixing Synthetic and Real Data - Discussion

- ✓ Should find patterns in PSB data if there are
  - ✓ Learned predictive model from less than 10 training examples for 43 PSB signals (as noise)
  - ✓ Finds correct failure precursors

Real data

- Predict high priority alarms in LASER
- Example
  - Converter: BR3.DVT13L4 (ACAPULCO)
  - Fault code: 20
  - PC Permit not present
- Trained on data from 2015-09-01 to 2016-09-01; tested from 2016-09-12 to 2017-05-31
- Question:
  - Does it predict?
  - Does it explain?

Real data

Does it predict?

- Yes, when there are patterns
- But: there don't seem to be many patterns
- And: We are at small data limit of machine learning → performance estimations uncertain

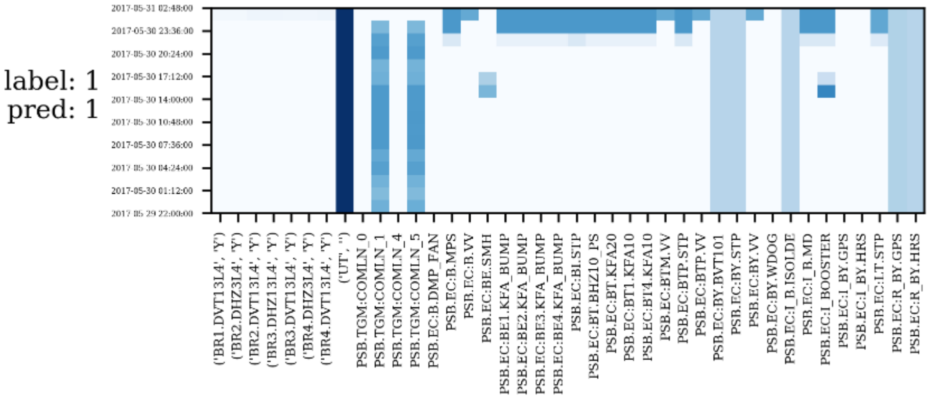
score	# alarms Test	# alarms Train	fcn	fcn_3d	**fcn_2d**	kNN	random_forest	svm
acc_test	3	7	0.96	0.96	**1**	0.92	0.84	0.88
acc_val	2.4	4.6	0.84	0.90	**0.93**	0.82	0.68	0.73
F1_test	3	7	0.8	0.8	**1**	0.5	0	0
F1_val	2.4	4.6	0.27	0.4	**0.68**	0	0.073	0.29

Real data

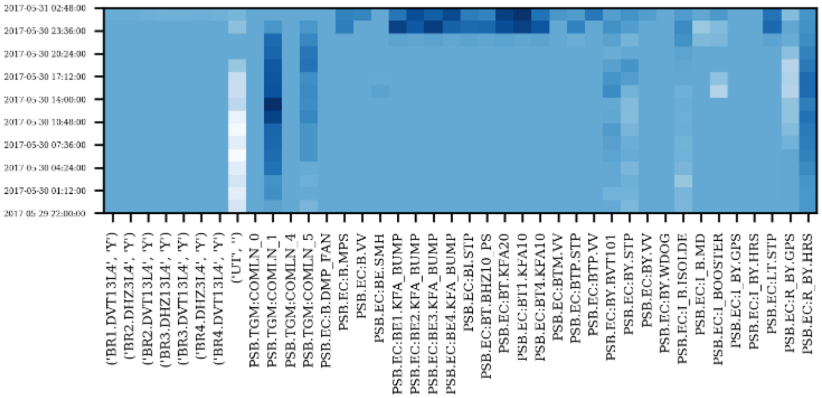
Does it explain?

- It gives hints
- Complex to interpret and ambiguous

Input



FCN based explanation



Real data - Discussion

- ✓ Achieves good predictive performance when patterns exist
- ✓ Learned patterns are hard to interpret
- ✓ Interpretation when combined with logbook data easier
- ✓ Too little data to draw conclusions with certainty
  - ✓ Approaching limits for machine learning
  - ✓ → Improve data selection/pre-processing/problem formulation

Conclusions and Outlook

Conclusions - Computation

- Underestimation of
  - Effort required to download the data
  - Execution time apart from training
- Benefitted from
  - Usage of well known libraries
  - Computation using CERN cluster
- Useful tools
  - Swan notebooks to prototype and share scripts (but terrible for debugging)
  - CERN cluster to do massively parallel computation
  - Mattermost to communicate (2 of us in Meyrin + 1 in Prévessin)
  - cernbox to share data
  - GitLab to share code: https://gitlab.cern.ch/tcartier/mlcern

Conclusions - Results

- Framework predicted and explained failure patterns correctly for generated test cases
  - Detected patterns from fewer than 10 training examples within up to 10^2 signals
- For PSB data experiments
  - Predictions were accurate if well defined patterns existed
  - Explanations were hard to interpret for studied cases
    - Could be improved by different choice of input/output data
- Deep learning approaches outperformed traditional ML in presented cases
  - Random Forest outperformed everything else in traffic jam prediction problem (check paper)

Outlook

- Will implement a more data effective representation for learning
- Synthetic data: More complex patterns to better study failure mechanism explanation (e.g. Boolean logic between signals)
- Real data: Reformulate prediction problem and find less complex application scenarios with the goal of
  - Having more examples to learn from
  - Learning models of more controlled systems
- Framework is modular and re-usable
  - Clone our code and investigate how air traffic explains spread of Corona

Thank you for your attention!

Further details in:

- Publication
  - Felsberger L., Apollonio, A., Cartier-Michaud, T., Mueller, A., Todd B., Kranzlmüller, D. (2020) Analyzing Failure Mechanisms in Complex Infrastructures. Lecture Notes in Computer Science, submitted. Preprint
- Code
  - https://github.com/lfelsber/alarmsMining (public)
  - https://gitlab.cern.ch/tcartier/mlcern (CERN)

Training

- Concept of machine learning: Train on observed data, apply to unseen data
- Have to find best predictive model by systematic search over
  - Input data selection
  - Algorithm selection
  - Problem parameter selection
- → requires (more than a bit of) computation