ML based RPC Current quality monitoring (Development status)

University of Sofia (Bulgaria)

CMS RPC detector - monitoring

RPC current:

- Depends on environmental parameters, applied HV, LHC parameters, …
- Each chamber has unique behavior

Aim: Development of an Automatic Monitoring Tool able to spot abnormal RPC current behavior and warn for possible hardware problems.

Model version 1

To predict the RPC HV channel current taking into account Inst. Luminosity, Working Point and environmental parameters.:

$$I_{pred} = C_0 + C_1*L_{inst} + C_2*HV + C_3*T + C_4*L_{inst}*e^{(HV/P)} + C_5*RH + C_6*P + C_7*\Delta t$$

C_i – parameters specific for each chamber

L_{inst} – instantaneous luminosity

HV – applied high voltage

T – environmental temperature

P – environmental pressure

RH – environmental relative humidity

Δt – the time interval since the origin for a given year

Model version 2

Removing the time dependent term and added a term including integrated luminosity and a term proportional to the time when the chamber is ON but the instantaneous luminosity is 0.

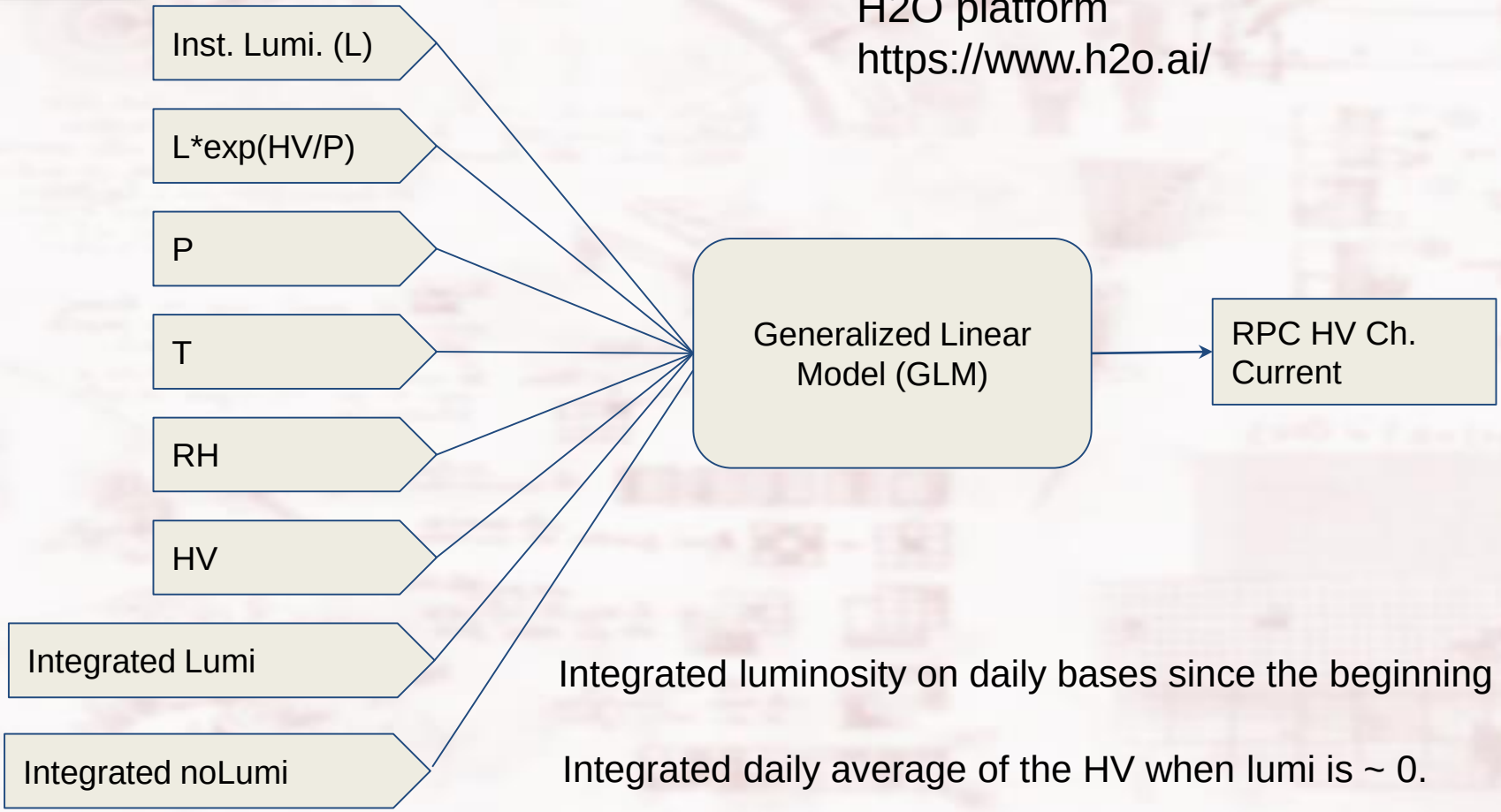
$$I_{pred} = C_0 + C_1*L_{inst} + C_2*HV + C_3*T + C_4*L_{inst}*e^{(HV/P)} + C_5*RH + C_6*P + C'_7*[\text{integrated luminosity}] + C8*[\text{time: ON and no Lumi}]$$

Gas mixture quality studies
for the CMS RPC detectors during LHC Run 2

Roberto Guida, Beatrice Mandelli
CERN
Andrea Gelmi on behalf of the CMS collaboration
(Università e INFN Bari)

ML approach

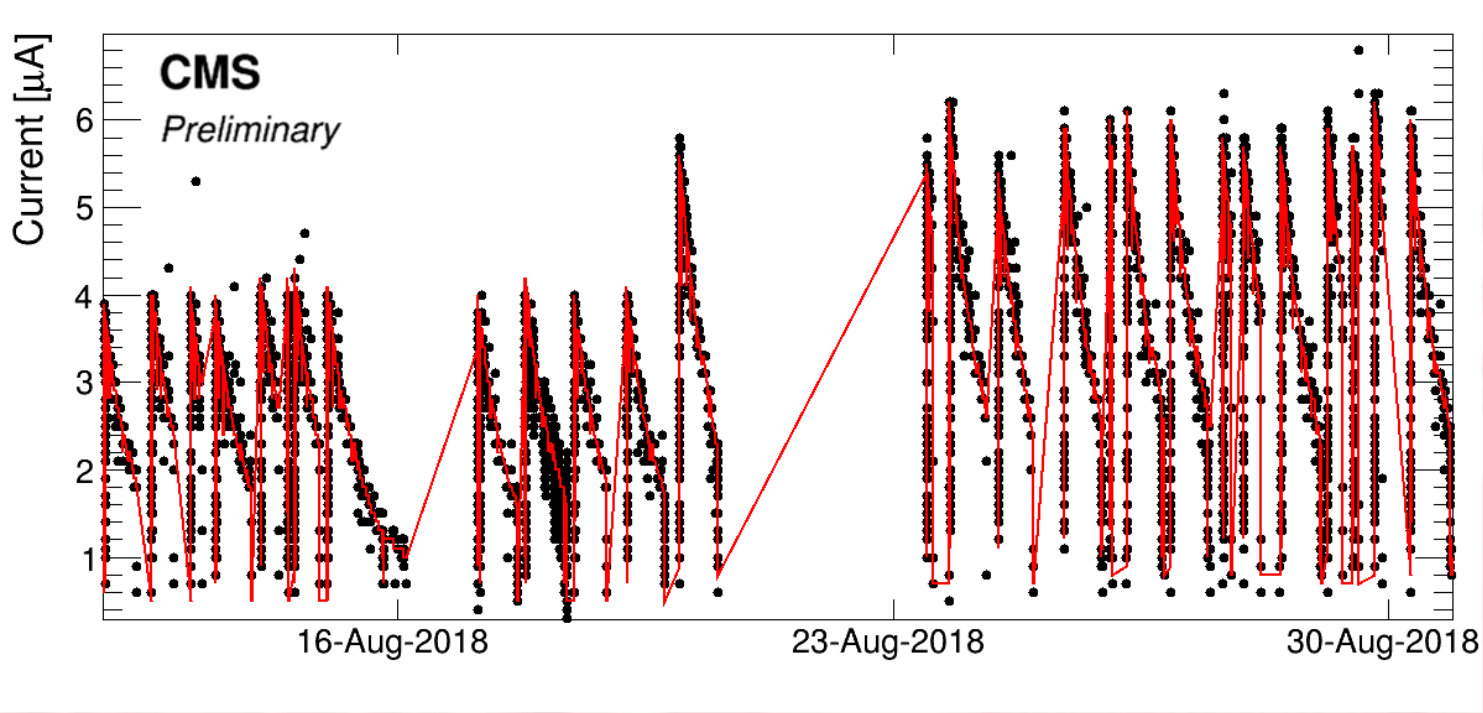
H2O platform
https://www.h2o.ai/



Integrated luminosity on daily bases since the beginning of 2016

Integrated daily average of the HV when lumi is ~ 0.

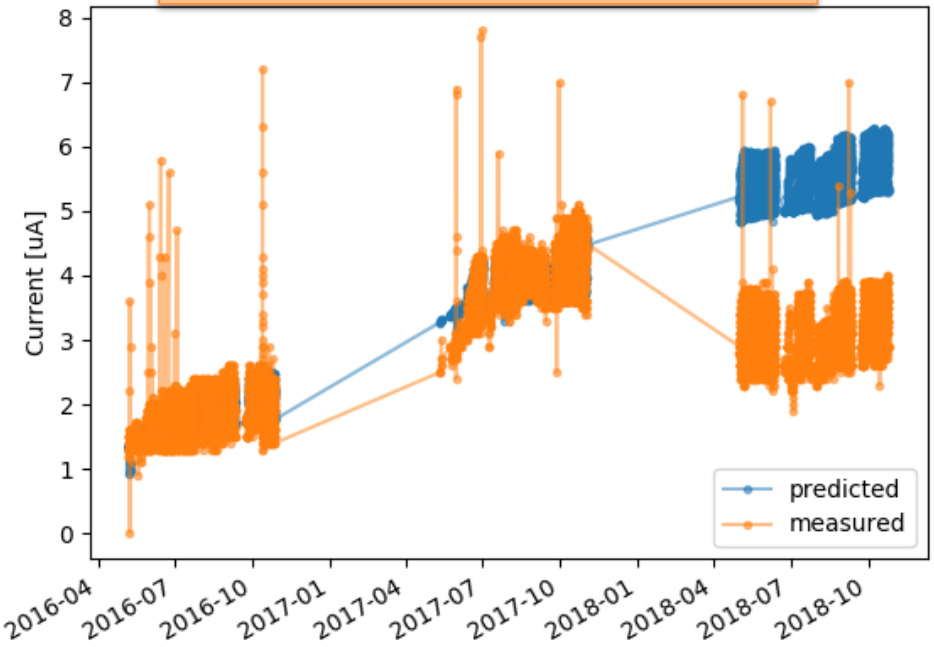
Predicted current follows the data points – version 1



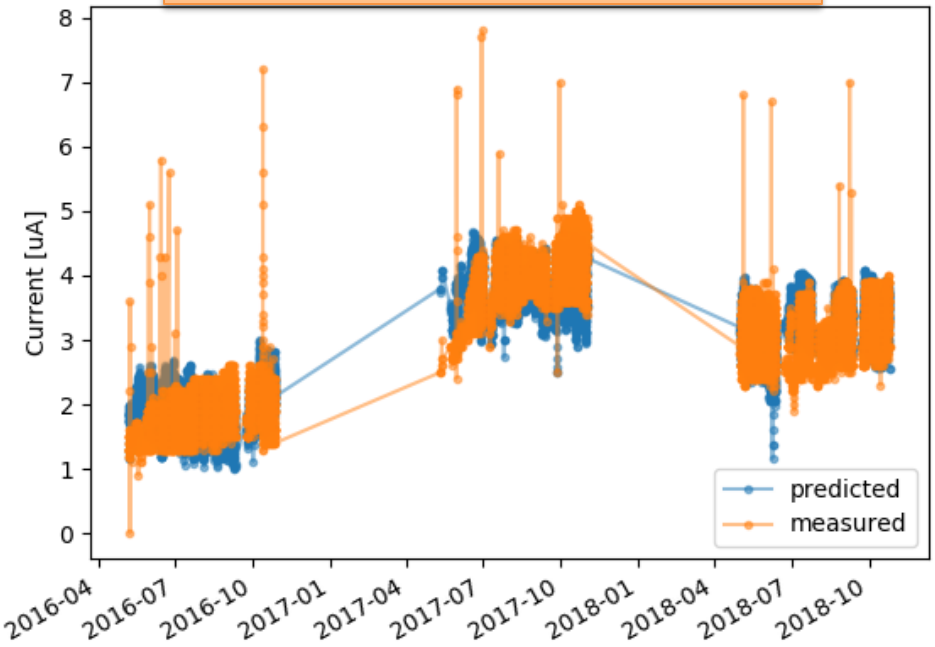
Predicted (red line) and the measured (black points) current are in good agreement although HV working point is changed by 200V on 19 Aug 2018.

Version 1 vs. Verson 2

VERSION 1

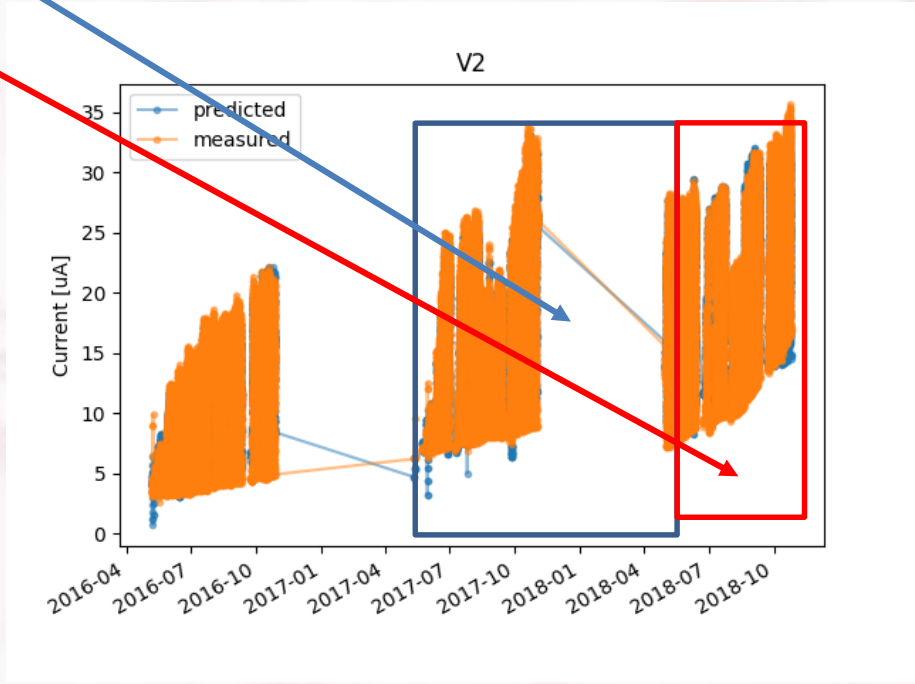


VERSION 2

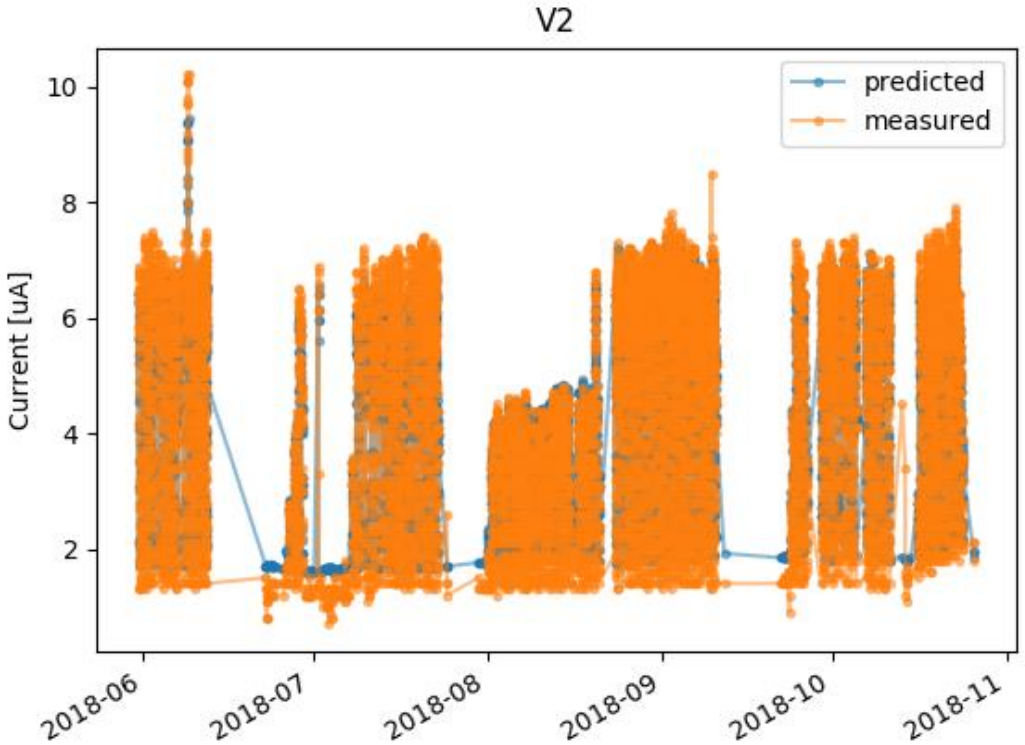


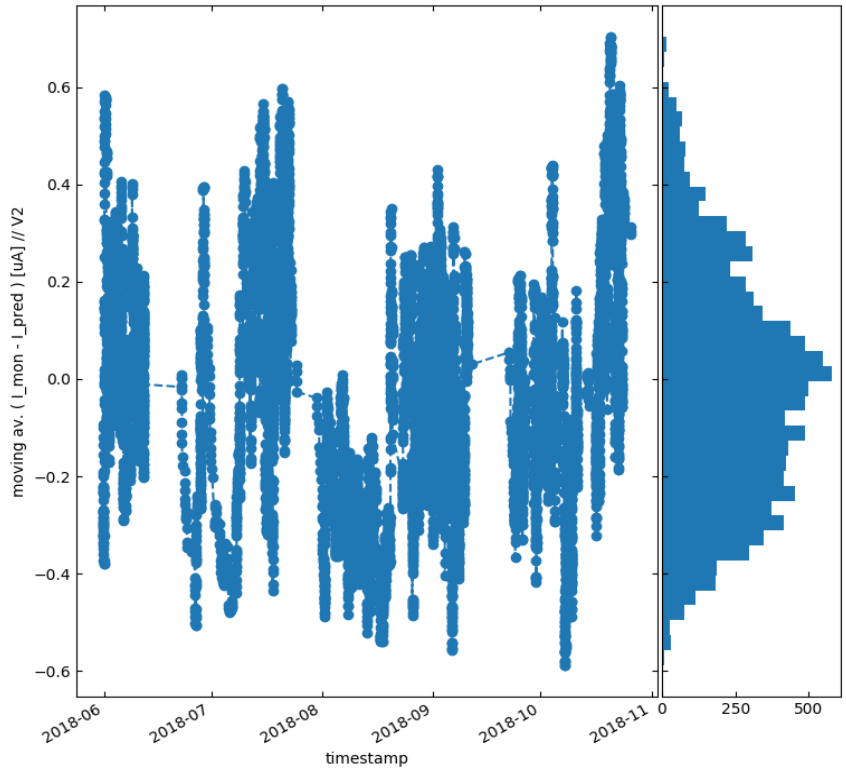
Training strategy

- Training period – from 01 May 2017 till 30 May 2018
- Validation period – from 31 May 2018 till the end of Nov 2018

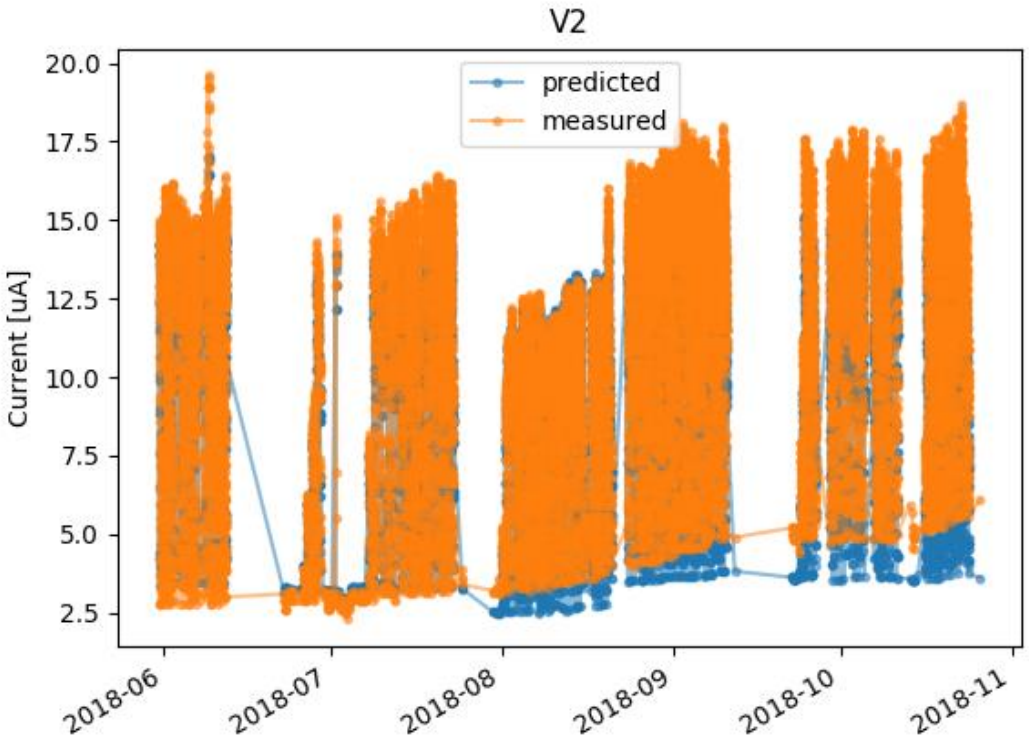


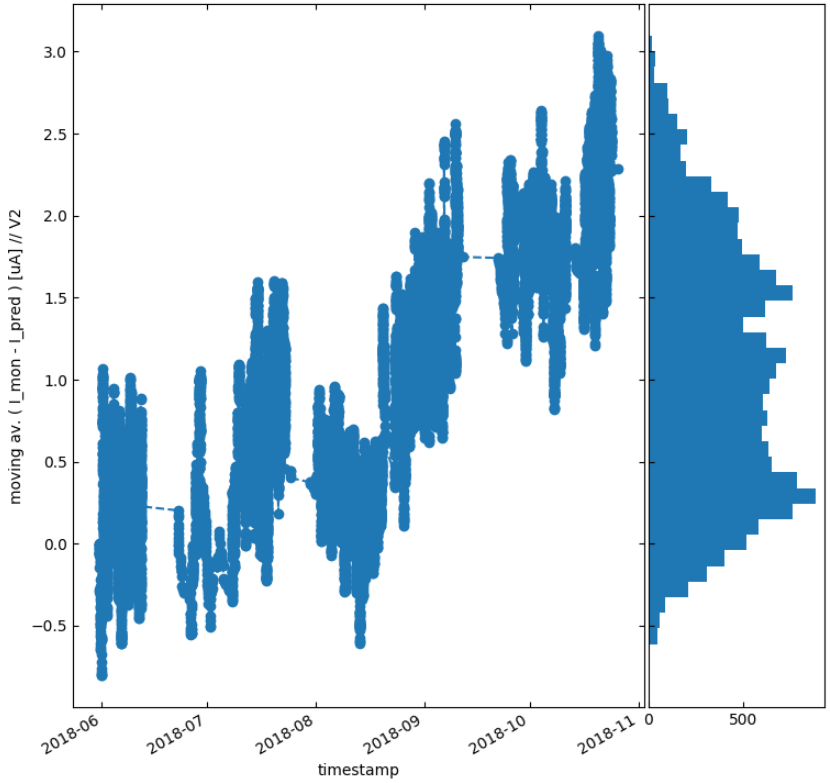
Example 1


V2
predicted
measured
Current [uA]
10
8
6
4
2
2018-06
2018-07
2018-08
2018-09
2018-10
2018-11

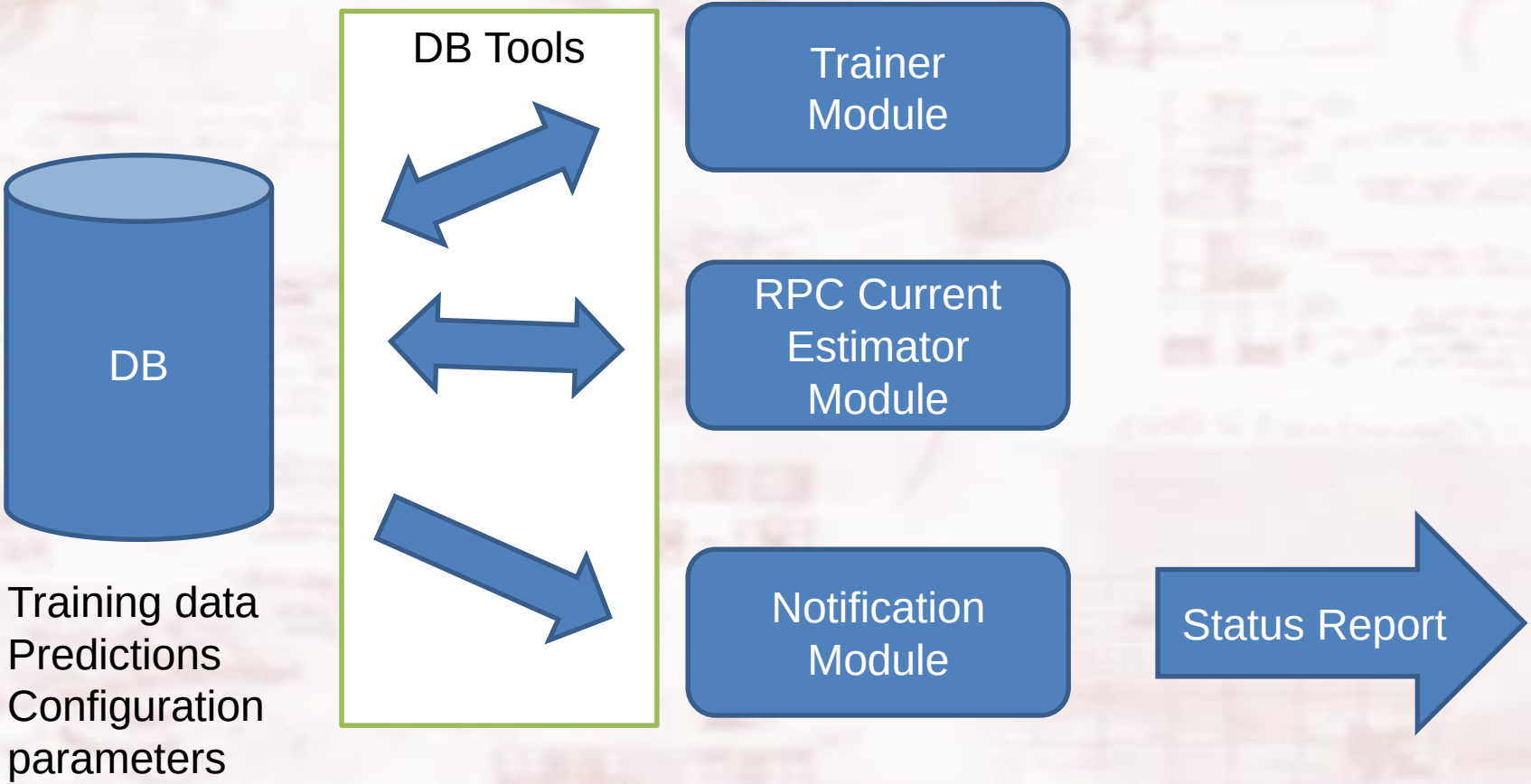

moving av. (I_mon - I_pred) [uA] // V2
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
2018-06
2018-07
2018-08
2018-09
2018-10
2018-11
timestamp
0
250
500

Example 2


V2
predicted
measured
Current [uA]
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
2018-06
2018-07
2018-08
2018-09
2018-10
2018-11


moving av. (I_mon - I_pred) [uA] // V2
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
2018-06
2018-07
2018-08
2018-09
2018-10
2018-11
timestamp
0
500

Implementation layout



Development strategy

- Programming Language – Python
- DB independent
- ML platform independent
- Low DB load – avoid performance demanding SQL queries
- Performance optimization on low performance virtual machine with locally installed MariaDB server - openstack.cern.ch

Important DB Tables objects

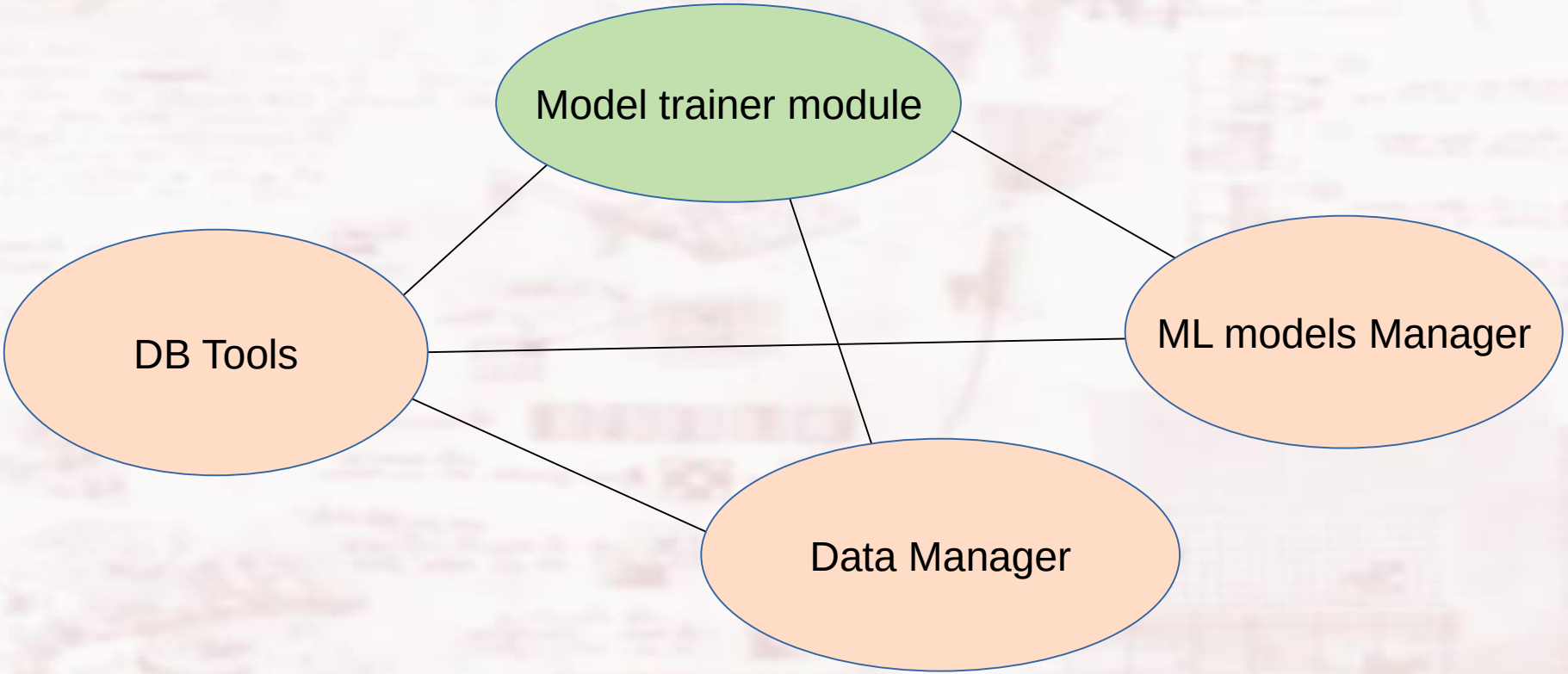
TrainingDataTable
change_date : str dpid : str flag : str hours_without_lumi : str imon : str instant_lumi : str integrated_lumi : str uxcDP : str uxcP : str uxcRH : str uxcT : str vmon : str
get_insert_data_query(chdate, imon, vmon, dpid, flag, inst_lumi, p, t, rh, int_lumi, hrs_wo_lumi) set_change_date_col(name, type) set_flag_col(name, type) set_hours_without_lumi(name, type) set_imon_col(name, type) set_inst_lumi_col(name, type) set_integrated_lumi(name, type) set_pdid_col(name, type) set_uxc_dp(name, type) set_uxc_p(name, type) set_uxc_rh(name, type) set_uxc_t(name, type) set_vmon_col(name, type)

MLModels
dpid : str model_id : str model_path : str modelconf_id : str mojo_path : str mse : str r2 : str
get_insert_query(modelconf_id, dpid, r2, mse, model_path, mojo_path) get_model_query(modelconf_id, dpid) get_update_model_query(model_id, modelconf_id, dpid, r2, mse, model_path, mojo_path) set_dpid(name, type) set_model_path(name, type) set_modelconf_id(name, type) set_mojo_path(name, type) set_mse(name, type) set_r2(name, type)

PredictedCurrentsTable
dpid : str measured_value : str model_id : str predicted_for : str predicted_value : str predicted_value_error : str
get_insert_query(model_id, dpid, predicted_for, predicted_value, predicted_value_error, measured_value) get_select_by_dpid_model_id_dpid_timewindow_query(dpid, model_id, begw, endw) set_dpid(name, type) set_measured_value(name, type) set_model_id(name, type) set_predicted_for(name, type) set_predicted_value(name, type)

MLModelsConf
input_cols : str last_update : str mlclass : str modelconf_id : str name : str output_cols : str test_from : str test_to : str train_from : str train_to : str
get_insert_query(name, mlclass, input_cols, output_cols, train_from, train_to, test_from, test_to) get_select_query_by_model_name(name) get_select_query_by_modelconf_id(modelconf_id) set_input_cols(name, type) set_mlclass(name, type) set_name(name, type) set_output_cols(name, type) set_test_from(name, type) set_test_to(name, type) set_train_from(name, type) set_train_to(name, type) update_by_modelconf_id_query(modelconf_id, name, mlclass, input_cols, output_cols, train_from, train_to, test_from, test_to) update_by_name_query(name, mlclass, input_cols, output_cols, train_from, train_to, test_from, test_to)

Trainer Module



ML Model Conf Object

- It has an unique name and defines the set of parameters:
  - ML model class – important for preparing the input data
  - Input column names of the Training Table
  - Output column name of the Training Table
  - Begining and end of the trainig period
  - Begining and end of the test period –optional

ML Model Object

Uniquely determined by the ML Model Conf and the DPID of the HV Channel:

- modelconf_id
- dpid
- r2
- mse
- model_path
- mojo_path

How to train a model - example

Create ML model configuration if not present in the DB:

```
rpccurrml = dbase.mysql_dbConnector(host='localhost',user='********',password='*********')
rpccurrml.connect_to_db('RPCCURRML')
mconf_manager = MLModelsConfManager(rpccurrml,table_mlmodelsconf)
mconf = MLModelConf()
mconf.name = "test_conf"; mconf.mlclass = "GLM_V2"
mconf.input_cols = ",".join([table_training.vmon,table_training.uxcP, table_training.uxcT,
    table_training.uxcRH, table_training.instant_lumi, table_training.integrated_lumi,
    table_training.hours_without_lumi])
mconf.output_cols = table_training.imon
mconf.train_from = "2016-05-01"; mconf.train_to = "2016-11-30"
mconf.test_from = "2017-06-01"; mconf.test_to = "2017-11-30"
mconf_manager.RegisterMLModelConf(mconf)
```

Write the model conf to DB

How to train a model – exmaple – cont.

```
import h2o
import RPCHVChannelModel

h2o.init()

RPCHVChannelModel.init("test_conf")

dpid = 315
flag = 56

RPCHVChannelModel.train_and_register_for_dpid(dpid,flag)
```

Summary

- Source code repo (public): https://github.com/peicho-petkov/rpc-currents-ml
- Training data preparation => Ready
- Trainer Module => Ready
- Different ML methods can be easily implemented
- Current Estimator module => Ongoing

Next steps...

- Implement Current Estimetor module
- Notification manager


感謝你
Danke
Gracias
Grazie
Merci
Thanks!
THANK YOU!
alamy stock photo
GJRB3M
www.alamy.com