Run-Time Adaptation of Complex Event Forecasting

Manolis Pitsikalis[1], **Elias Alevizos**[1,2], **Nikos Giatrakos**[3] **and Alexander Artikis**[1,4]

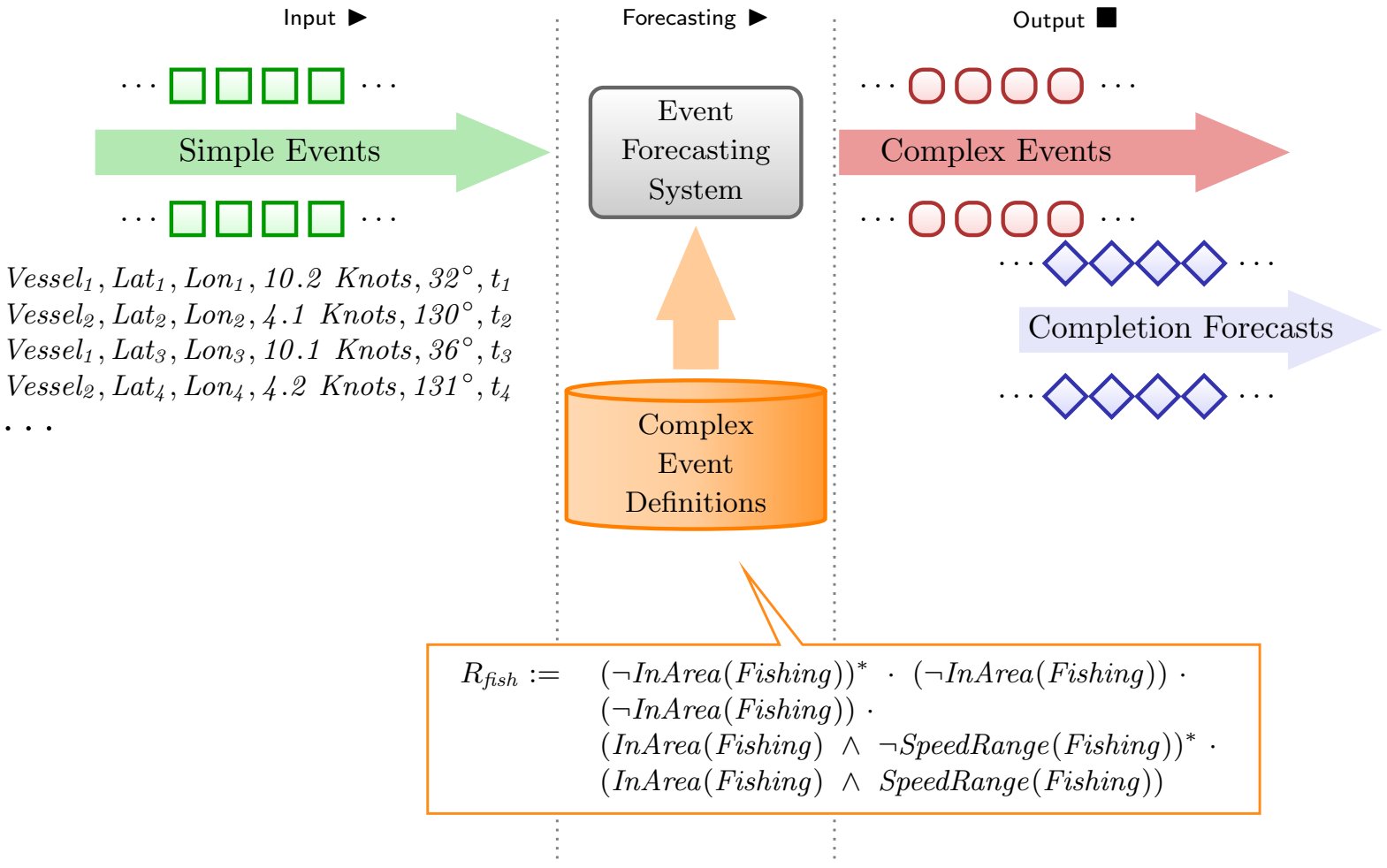
[1]NCSR Demokritos, [2]The American College of Greece,
[3]Technical University of Crete, [4] University of Piraeus

{manospits, alevizos.elias, a.artikis}@iit.demokritos.gr,
ngiatrakos@tuc.gr

DEBS 2025

CREXDATA
Critical Action Planning over Extreme-Scale Data

Complex Event Forecasting



Wayeb

Wayeb[1] is our CEF/R engine.

- CE patterns are defined as regular expressions
- they are compiled into automata
- we can use the automata to perform CER

CE Forecasting: forecast when a CE pattern is expected to be fully matched.

- Wayeb employs prediction suffix trees (PST) for computing the probability of an event appearing in the input.
- Automata runs + PSTs $\rightarrow$ CEF

[1] Alevizos E., et al., Complex Event Forecasting with Prediction Suffix Trees. VLDBJ, 2022.

Problem statement

Wayeb's forecasting accuracy is influenced by the hyper-parameters used during PST training.

- Solution: Hyper-parameter optimisation[2]

What about **input data evolutions** that invalidate the deployed model?

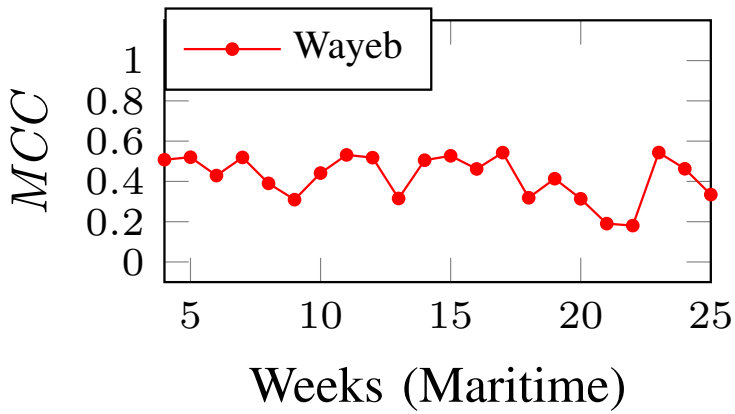
[2]Stavropoulos V. et al. Optimizing Complex Event Forecasting. DEBS 2022.

Problem statement

Wayeb's forecasting accuracy is influenced by the hyper-parameters used during PST training.

- Solution: Hyper-parameter optimisation[2]

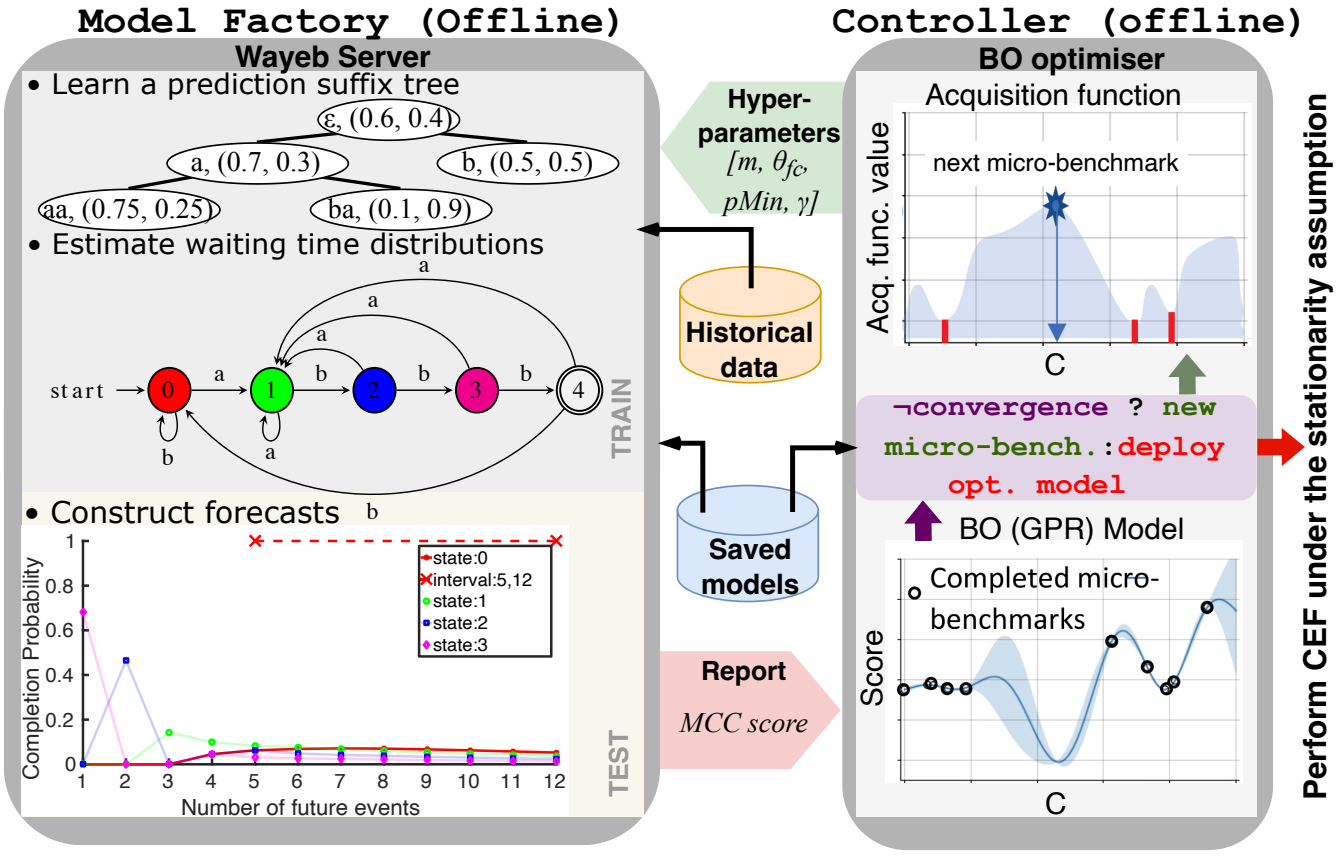
What about **input data evolutions** that invalidate the deployed model?



[2]Stavropoulos V. et al. Optimizing Complex Event Forecasting. DEBS 2022.

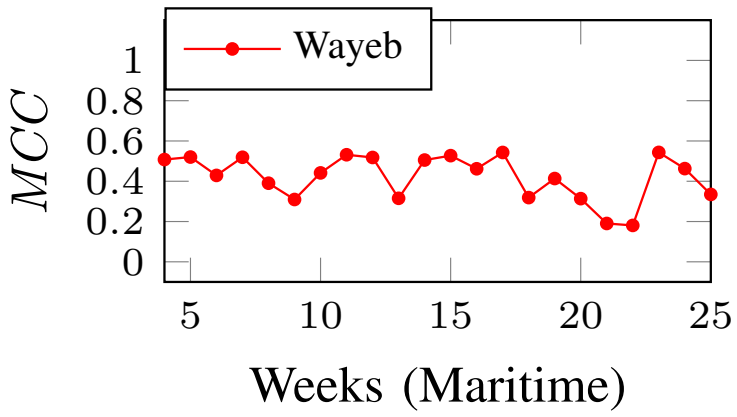
Stationarity assumption: offline optimisation

offCEF



Offline optimisation

Recall...



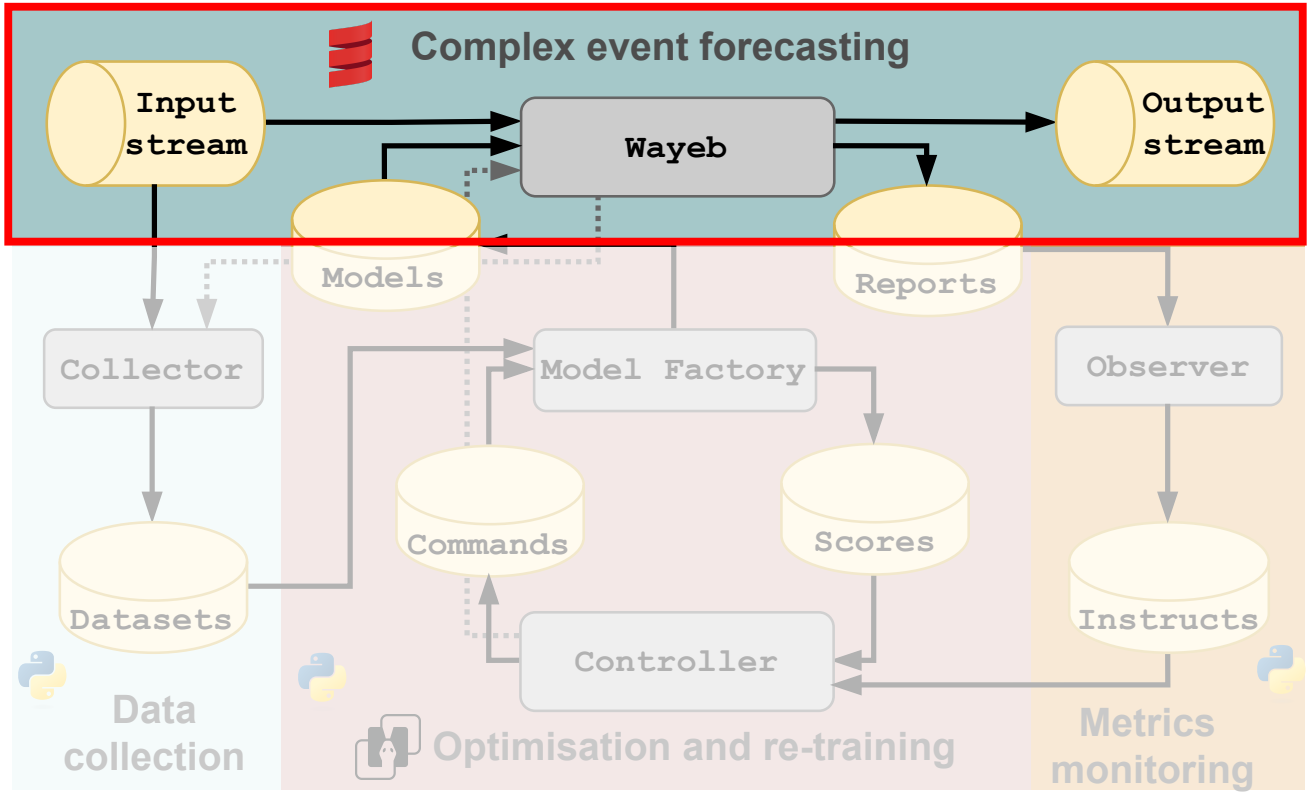
→ `offCEF` **is not suitable** for CEF over event streams where input-data evolutions are present.

Run-Time Adaptation of CEF

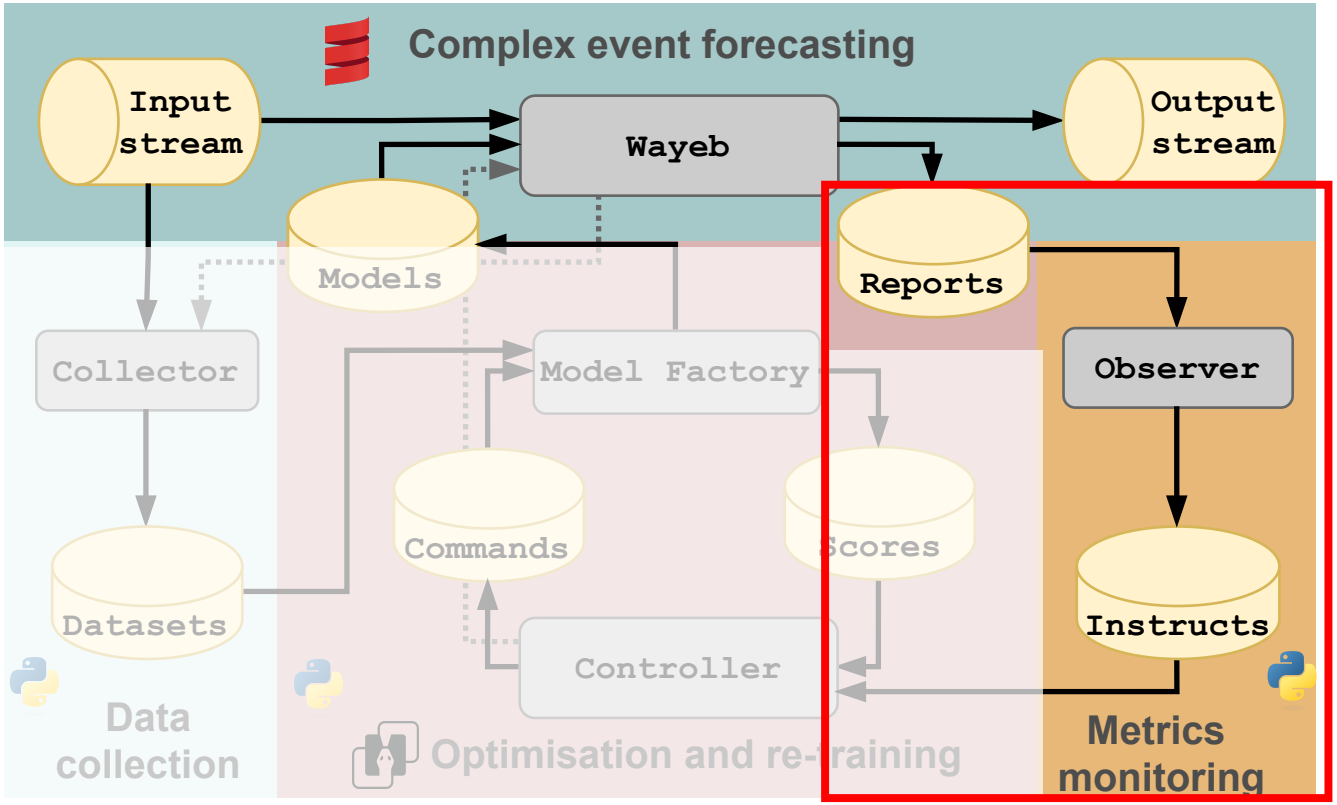
We developed `RTCEF`, a framework for **run-time** adaptation of CEF that offers:

- a distributed architecture,
- retrain vs re-optimize selection policies,
- little to no interruption in forecasting,

Architecture of RTCEF



Architecture of RTCEF



Metrics monitoring

CEF scores degradation $\rightarrow$ deployed CEF model no longer suitable

What should we do?

- Retrain (same hyper-parameters)
- Hyper-parameter optimisation (new hyper-parameters)

Metrics monitoring

CEF scores degradation $\rightarrow$ deployed CEF model no longer suitable

What should we do?

- Retrain (same hyper-parameters)
- Hyper-parameter optimisation (new hyper-parameters)

Hyper-parameter optimisation in general yields "better" models than retraining, however it is **much more** expensive.

Metrics monitoring

CEF scores degradation $\rightarrow$ deployed CEF model no longer suitable

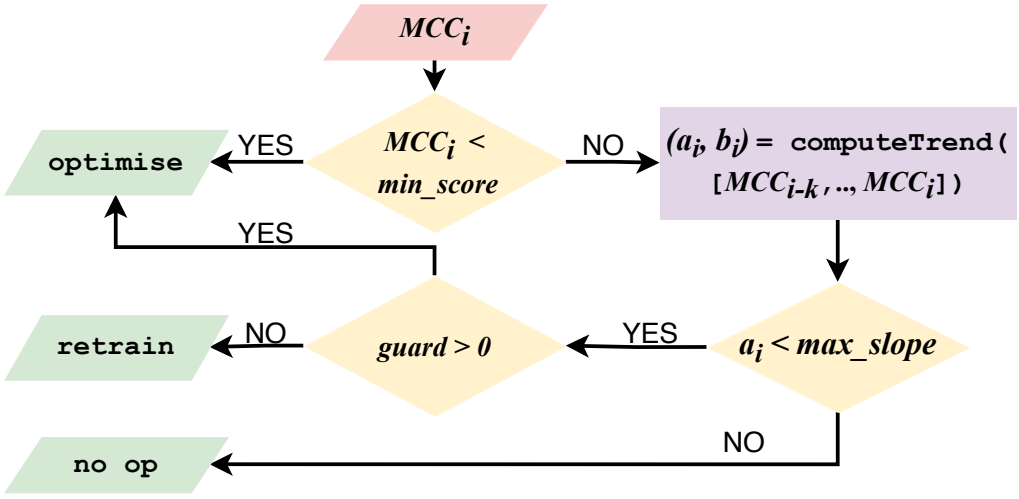
What should we do?

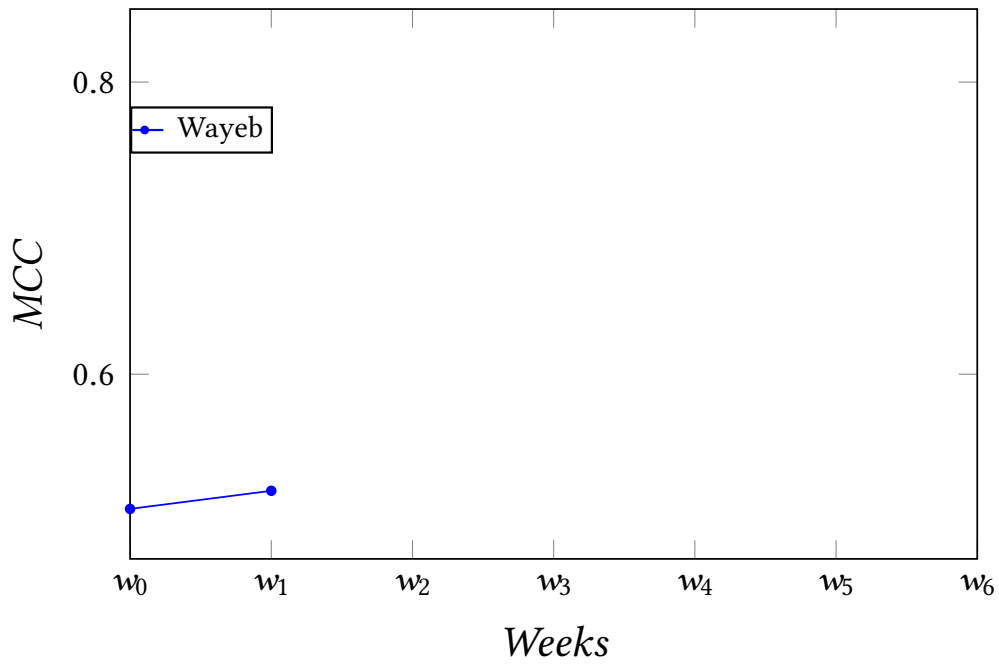
- Retrain (same hyper-parameters)
- Hyper-parameter optimisation (new hyper-parameters)

Hyper-parameter optimisation in general yields "better" models than retraining, however it is **much more** expensive.

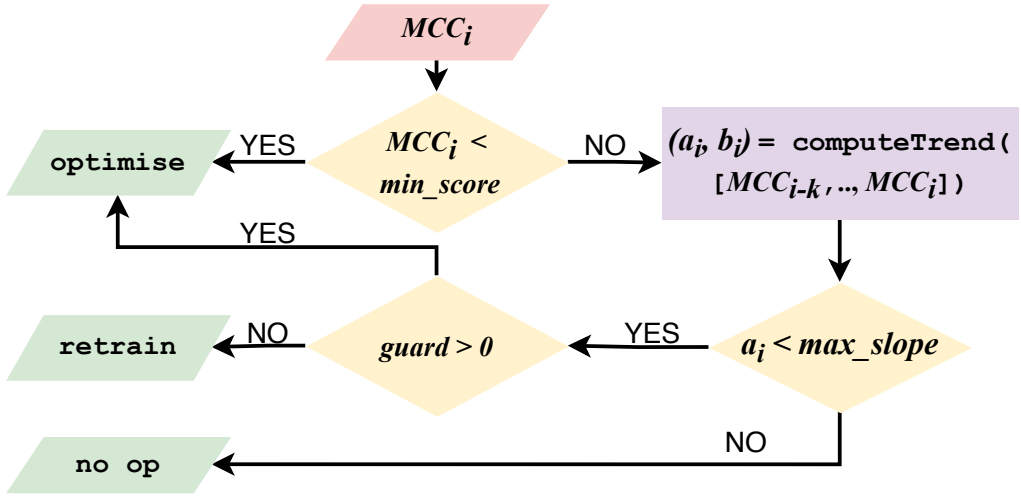
Retrain vs optimise: trend inspired policy

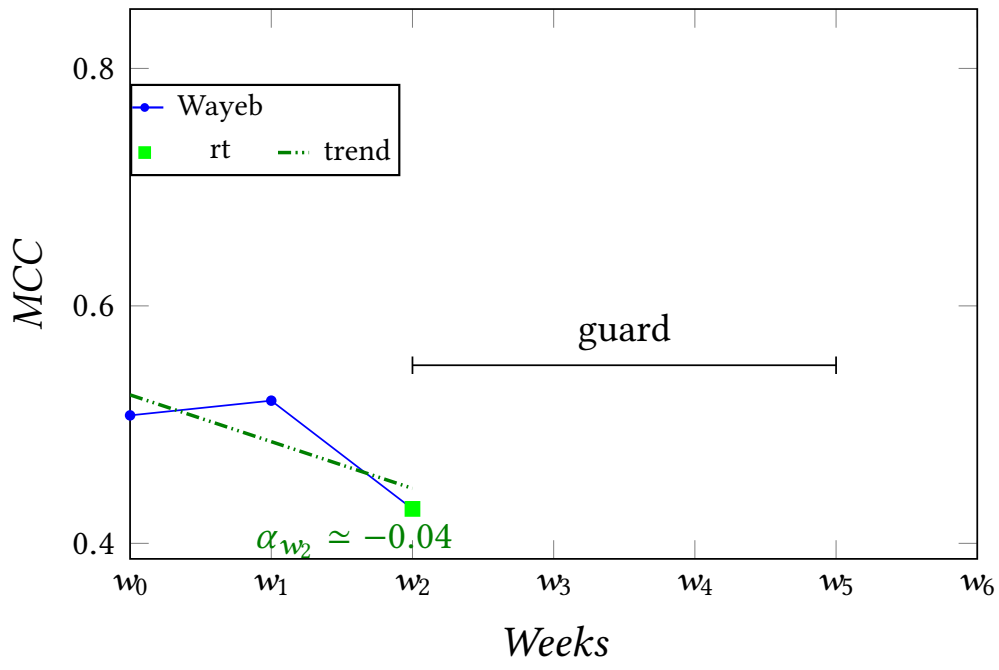
Observer


MCC_i
MCC_i < min_score
YES
optimise
NO
(a_i, b_i) = computeTrend([MCC_{i-k}, .., MCC_i])
YES
$a_i < max\_slope$
YES
$guard > 0$
NO
retrain
NO
no op

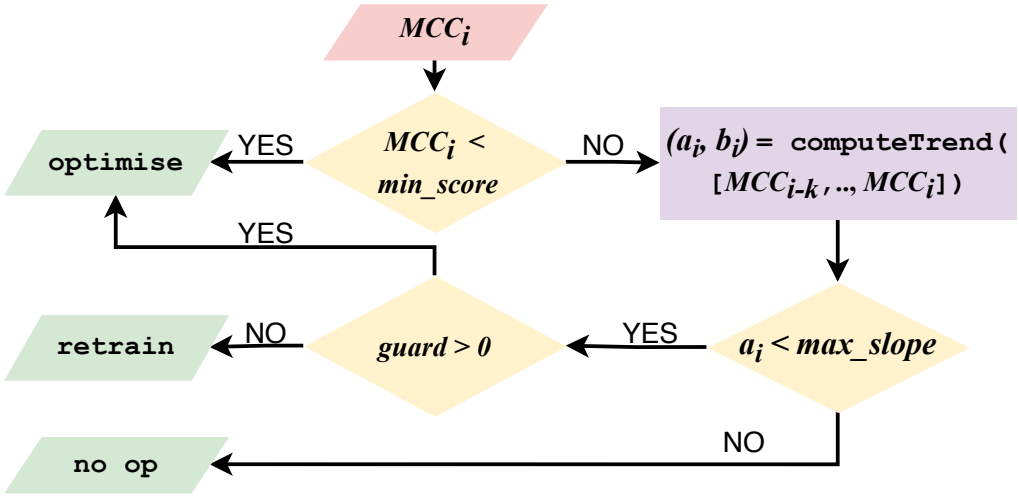

0.8
0.6
MCC
Wayeb
w_0
w_1
w_2
w_3
w_4
w_5
w_6
Weeks

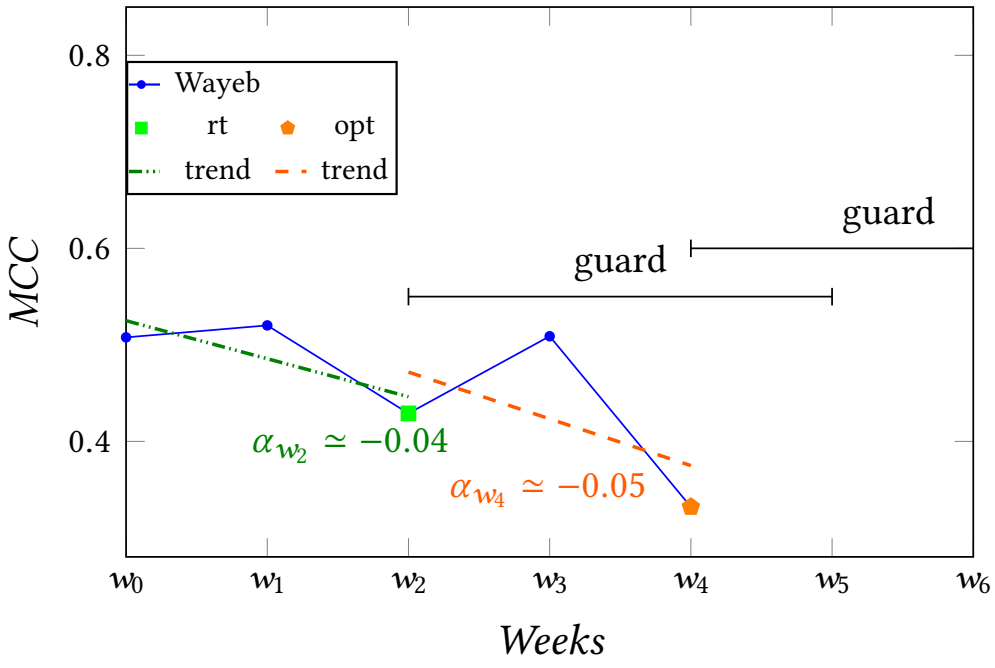
Observer


MCC_i
MCC_i < min_score
YES
optimise
NO
(a_i, b_i) = computeTrend([MCC_{i-k}, .., MCC_i])
YES
$a_i < max\_slope$
YES
$guard > 0$
NO
retrain
NO
no op

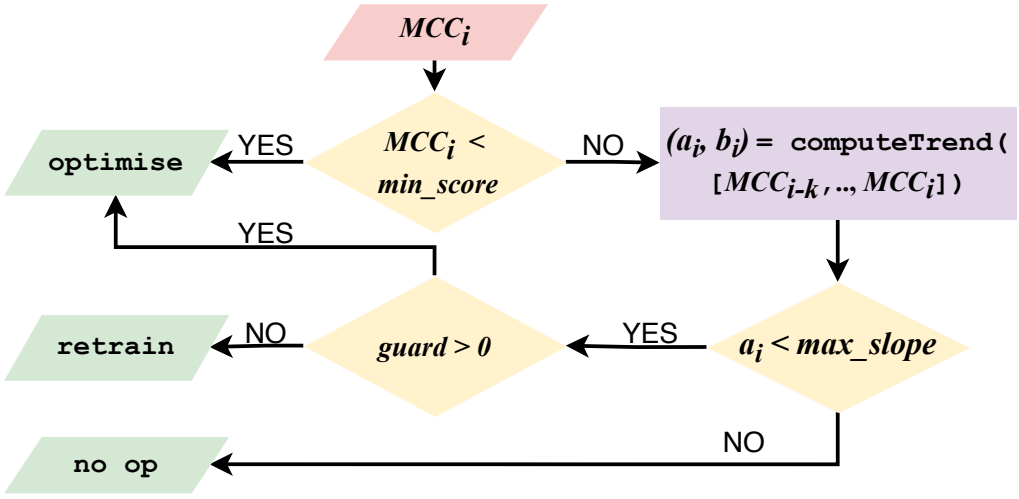

0.8
0.6
0.4
MCC
Wayeb
rt
trend
guard
$\alpha_{w_2} \simeq -0.04$
w_0
w_1
w_2
w_3
w_4
w_5
w_6
Weeks

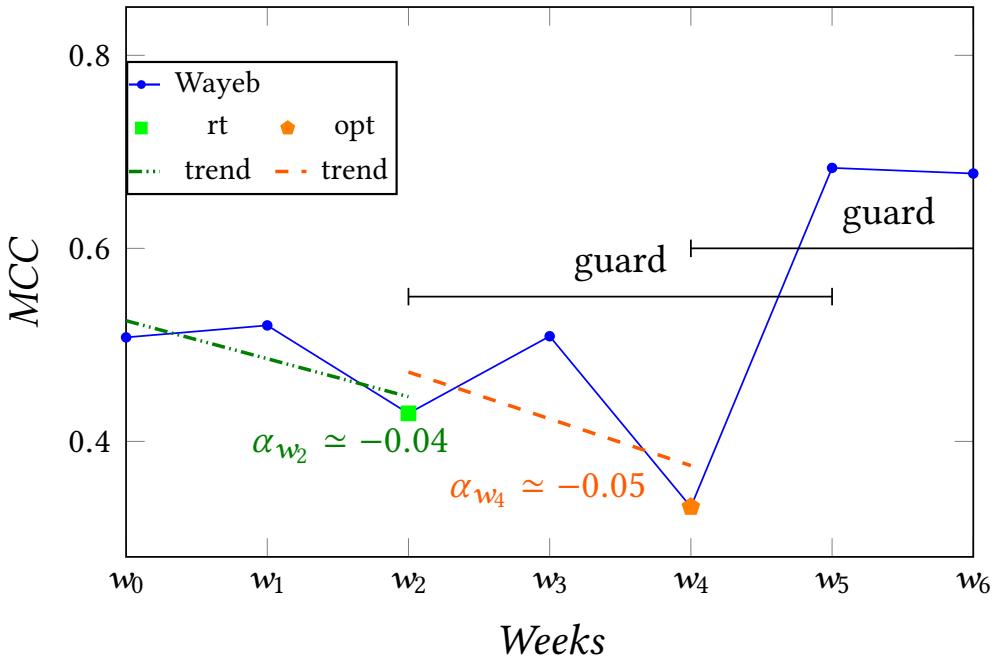
Observer


MCC_i
MCC_i < min_score
YES
optimise
NO
(a_i, b_i) = computeTrend([MCC_{i-k}, .., MCC_i])
YES
$a_i < max\_slope$
YES
$guard > 0$
NO
retrain
NO
no op

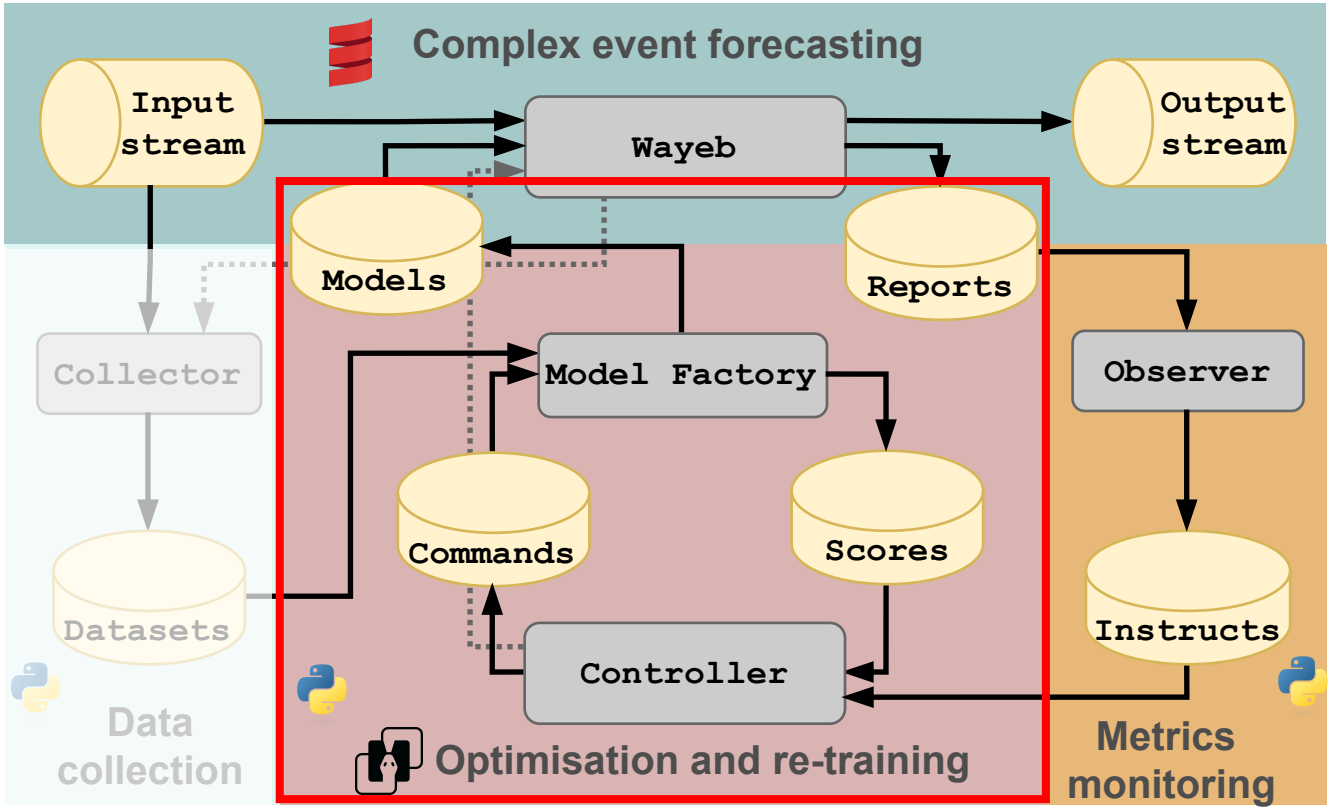

0.8
0.6
0.4
MCC
Wayeb
rt
opt
trend
trend
guard
guard
$\alpha_{w_2} \simeq -0.04$
$\alpha_{w_4} \simeq -0.05$
w_0
w_1
w_2
w_3
w_4
w_5
w_6
Weeks

Observer


MCC_i
MCC_i < min_score
YES
optimise
NO
(a_i, b_i) = computeTrend([MCC_{i-k}, .., MCC_i])
YES
$a_i < max\_slope$
YES
$guard > 0$
NO
retrain
NO
no op


0.8
0.6
0.4
MCC
Wayeb
rt
opt
trend
trend
guard
guard
$\alpha_{w_2} \simeq -0.04$
$\alpha_{w_4} \simeq -0.05$
w_0
w_1
w_2
w_3
w_4
w_5
w_6
Weeks

Architecture of RTCEF



Retraining and Hyper-parameter Optimisation

Retraining: Use the last hyper-parameters and train a new PST.

Hyper-parameter optimisation: similar to `offCEF`...
but `RTCEF` **does not** start each optimisation procedure from scratch. Instead:

- We keep *keep_percentage* points from the previous run,
- and, sample remaining points as usual.

Retraining and Hyper-parameter Optimisation

Retraining: Use the last hyper-parameters and train a new PST.

Hyper-parameter optimisation: similar to `offCEF`...
but `RTCEF` **does not** start each optimisation procedure from scratch. Instead:

- We keep *keep_percentage* points from the previous run,
- and, sample remaining points as usual.

Retraining/Hyperparameter Optimisation **happen in parallel to CEF**. CEF (Wayeb) stops **only** for model replacement.

Model replacement

Simply replacing a deployed model with a revised version leads to:

- a **loss of ongoing automata** runs,
- and therefore, **ongoing forecasts**.

Solution: We theoretically proved that *under certain conditions* model transition is **lossless** and possible in linear time.

Model replacement

Simply replacing a deployed model with a revised version leads to:

- a **loss of ongoing automata** runs,
- and therefore, **ongoing forecasts**.

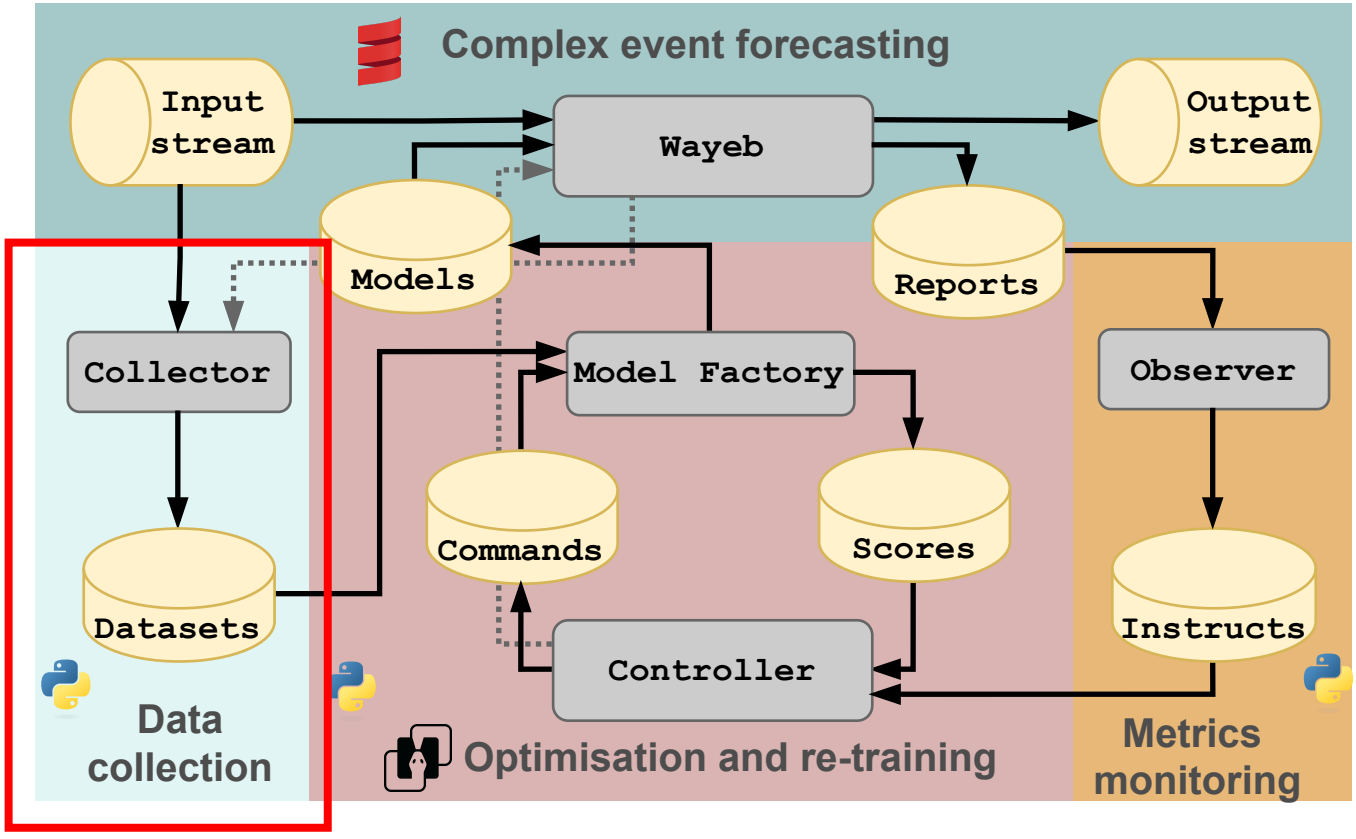
Solution: We theoretically proved that *under certain conditions* model transition is **lossless** and possible in linear time.

Conditions:

- the CEF models must have an order $\in [m_l, m_u]$, and
- the last m_u symbols of the input stream are available.

`RTCEF` performs **lossless** model replacement (proof in the paper).

Architecture of RTCEF



Experimental setup

We evaluate RTCEF on maritime situational awareness...

$$R_{port} := (\neg InsidePort(Brest)) * \cdot (\neg InsidePort(Brest)) \cdot (\neg InsidePort(Brest)) \cdot (InsidePort(Brest))$$

...and in credit card fraud management.

$$R_{cards} := (amDiff > 0) \cdot (amDiff > 0) \cdot (amDiff > 0) \cdot (amDiff > 0) \cdot (amDiff > 0) \cdot (amDiff > 0) \cdot (amDiff > 0)$$

Experimental setup (cont'd)

Maritime: Our dataset[3] contains 18M AIS messages transmitted between October 1st 2015 and 31st March 2016 (6 months)

Credit card fraud: We use a sythetic dataset[4] containing 1M transactions over a period of 21 months.

To validate our results, we create 6 and 21 versions of the datasets:

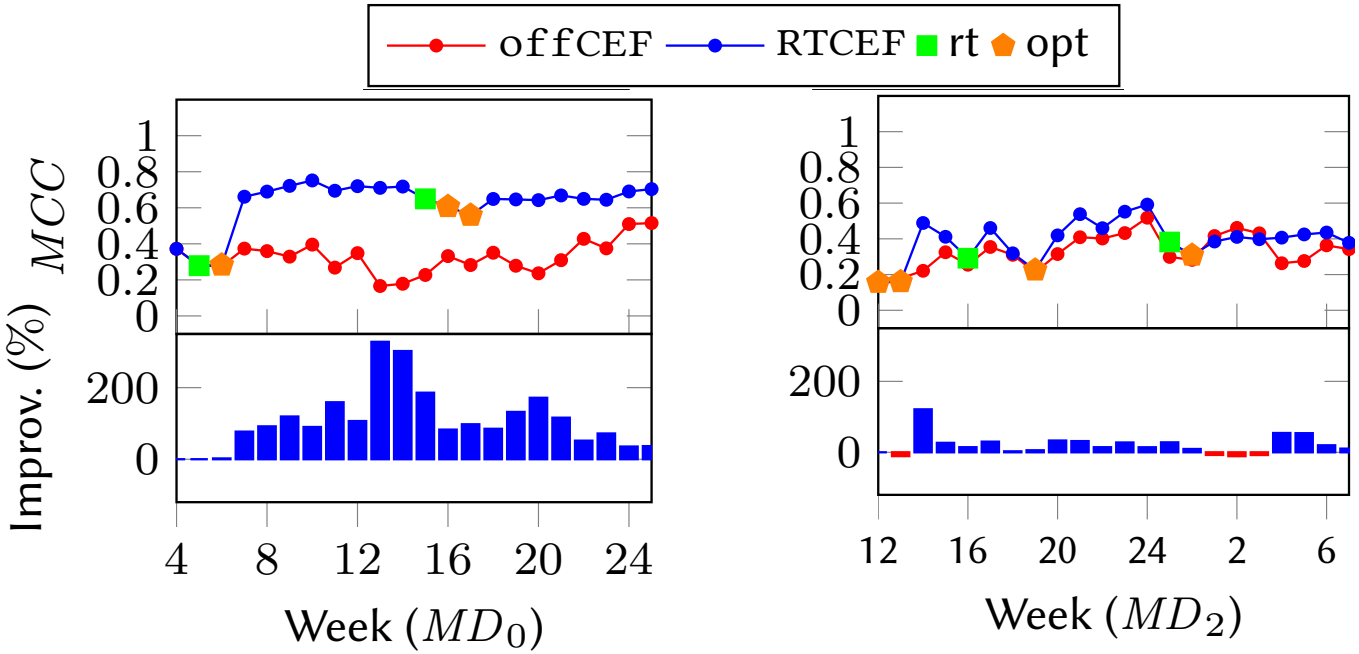
$$\{MD, FD\}_i = \Big\|_{j=0}^{j=\{5,20\}} Month_{(j+i) \bmod \{6,21\}}$$

...and we compare the performance of `RTCEF` against `offCEF`.

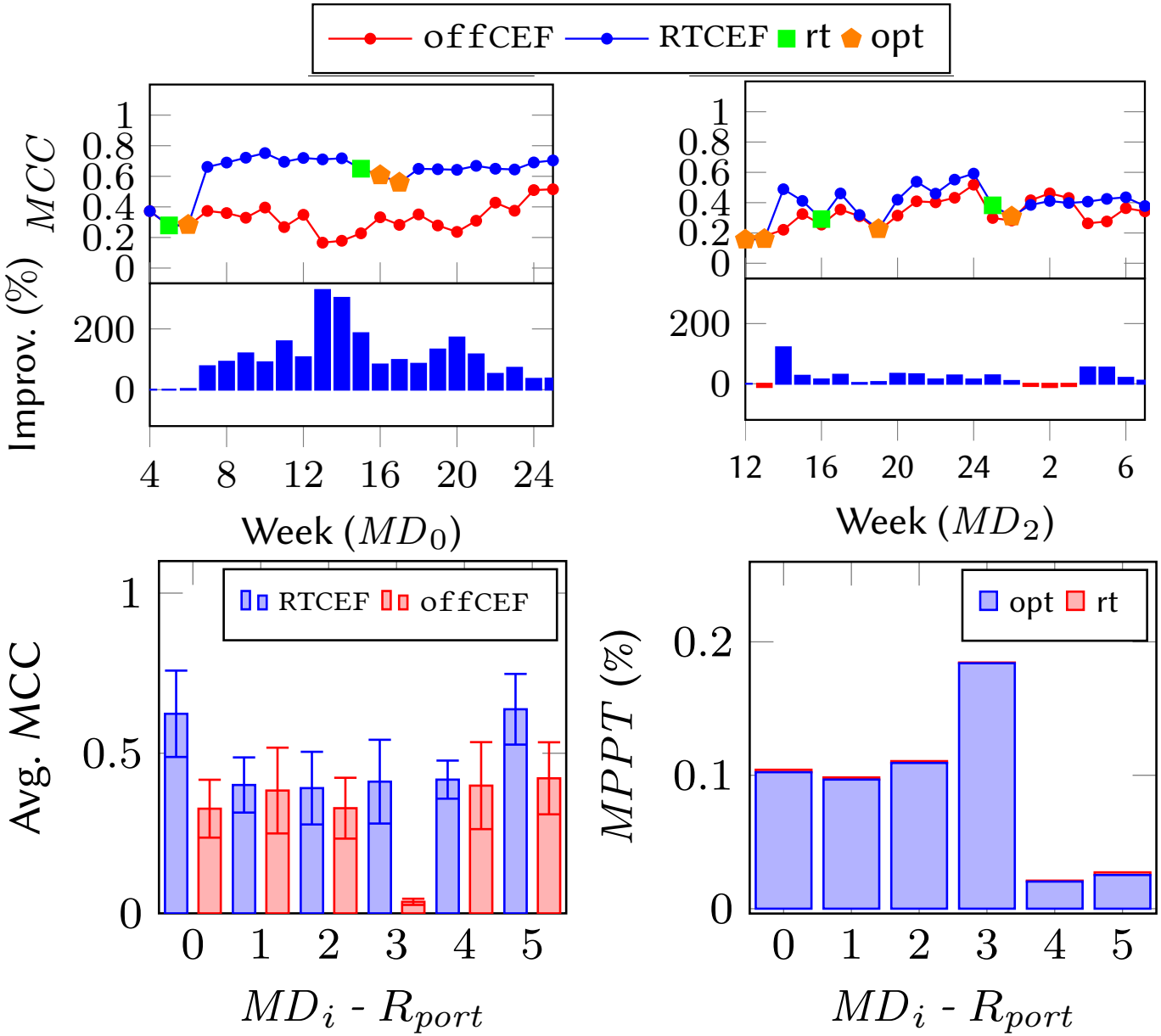
[3] `https://zenodo.org/records/1167595`

[4] `https://feedzai.com/`

Experimental results: maritime

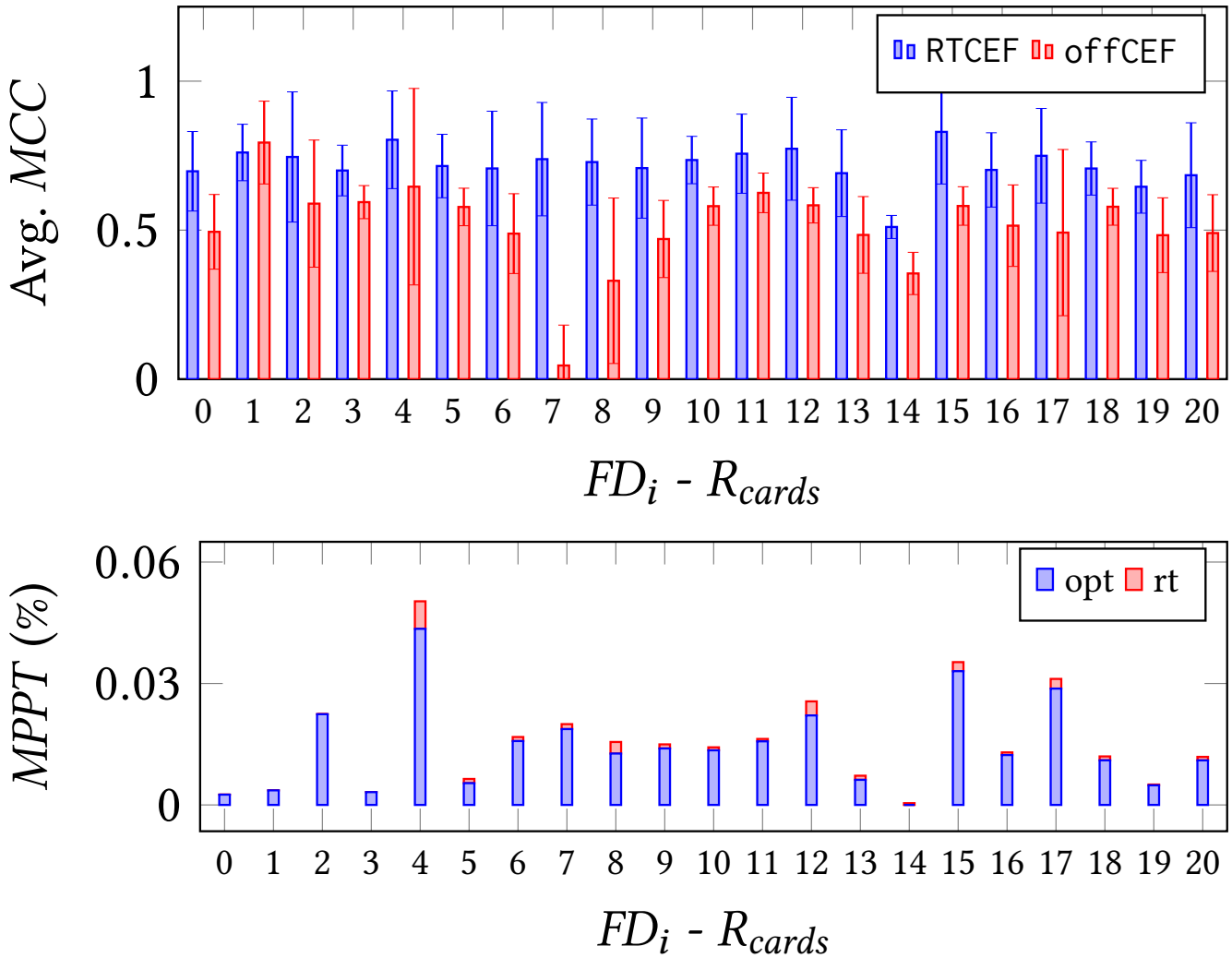


Experimental results: maritime



**$MCC \in [-1, 1]$

Experimental results: credit card fraud



Outro

We presented `RTCEF`:

- a novel framework for **run-time adaptation** of CEF,
- involving services running synergistically for **undisrupted** CEF,
- with **clear benefits** over the offline approach.

Future work:

- Investigate additional collection and retrain vs optimise policies.
- Distribute further the most "expensive" services of the framework.
- Application of `RTCEF` on other tasks such as run-time CER query optimisation.

Thank you!

https://github.com/manospits/rtcef