Learn++

Authors:
CAO Anh Quan
Fei LIANG
Li XU

Supervisor: Prof. Albert Bifet

Overview

- Introduction
- Implementation details
- Evaluation
- Conclusion

1. Introduction

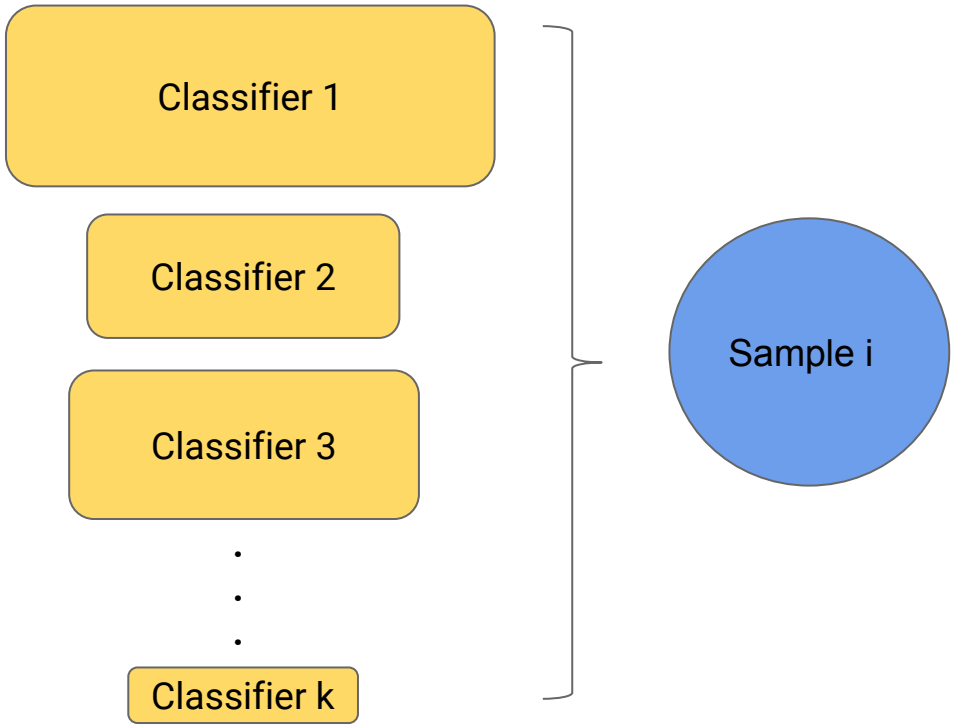
Motivation

Learn additional information

No need for old data

Preserve previously acquired knowledge

Incremental learning



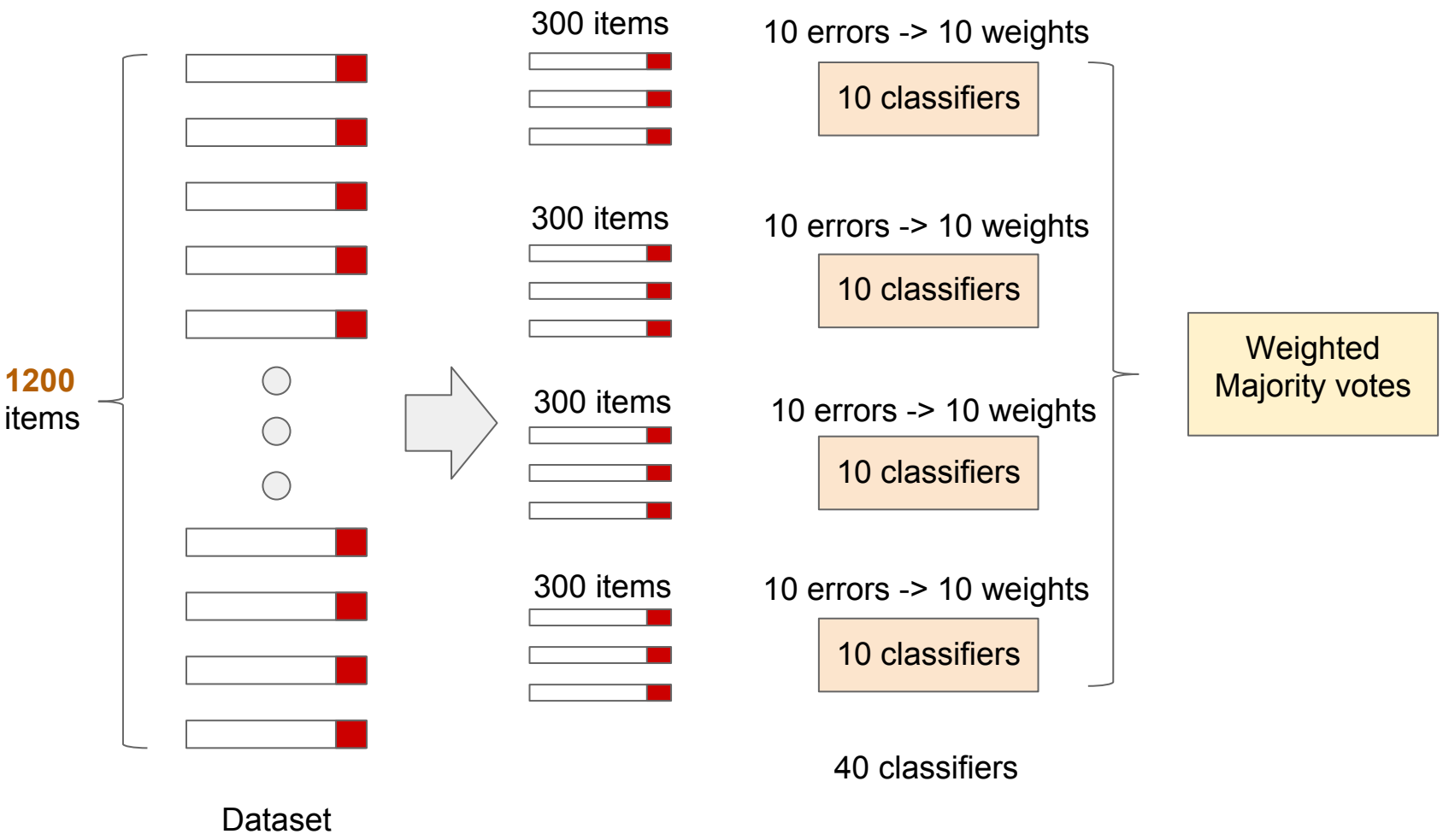
Data set

Sample 3
Sample i
Sample 156
...
Sample 55

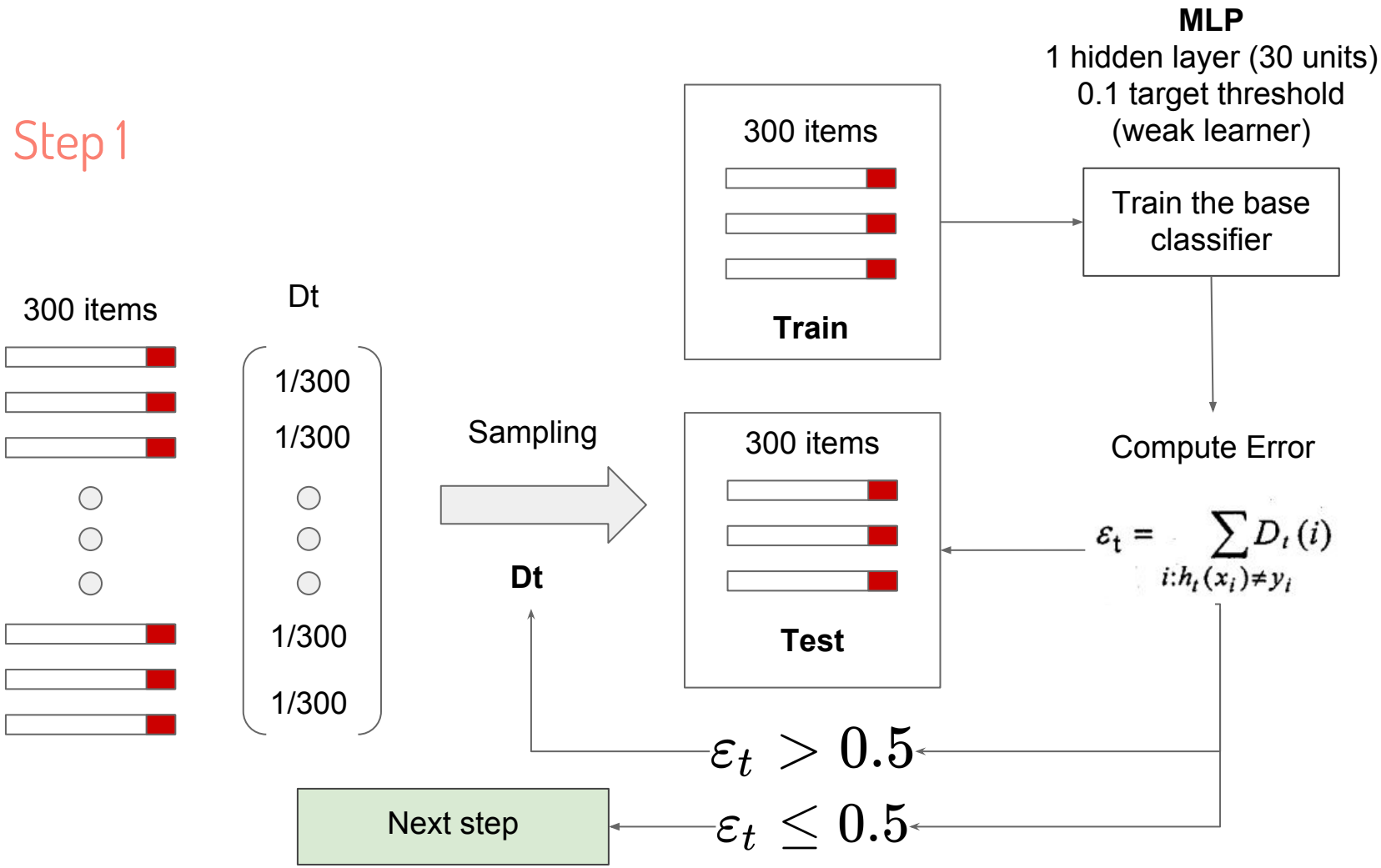
Sample 77
Sample i
Sample i
...
Sample i

Implementation details

We implement the algorithm from scratch


1200
items
Dataset
300 items
300 items
300 items
300 items
10 errors -> 10 weights
10 classifiers
10 errors -> 10 weights
10 classifiers
10 errors -> 10 weights
10 classifiers
10 errors -> 10 weights
10 classifiers
40 classifiers
Weighted
Majority votes

Step 1


MLP
1 hidden layer (30 units)
0.1 target threshold
(weak learner)
300 items
Train
Train the base classifier
300 items
Dt
1/300
1/300
1/300
1/300
Sampling
Dt
300 items
Test
Compute Error
$\varepsilon_t = \sum_{i:h_t(x_i)\neq y_i} D_t(i)$
$\varepsilon_t > 0.5$
$\varepsilon_t \leq 0.5$
Next step

Step 2

Normalized error

$$\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$$

Classifier weight

$$log(\frac{1}{\beta_t})$$

Step 3

Save the classifier and normalized error

Step 1

$E_t > 0.5$

$E_t \leq 0.5$

Composite Error

E.g. Iteration 3 (t = 3)

Classifier 1 $log(\frac{1}{\beta_1})$

Classifier 2 $log(\frac{1}{\beta_2})$

Classifier 3 $log(\frac{1}{\beta_3})$

Weighted Majority votes

$H_3(x)$

$$E_t = \sum_{i=1}^{m} D_t(i)[\![H_t(x_i) \neq y_i]\!]$$

Step 3

Composite Error

$$E_t = \sum_{i=1}^{m} Dt(i)[H_t(x_i) \neq y_i]$$

Normalized Composite Error

$$B_t = \frac{1}{1-E_t}$$

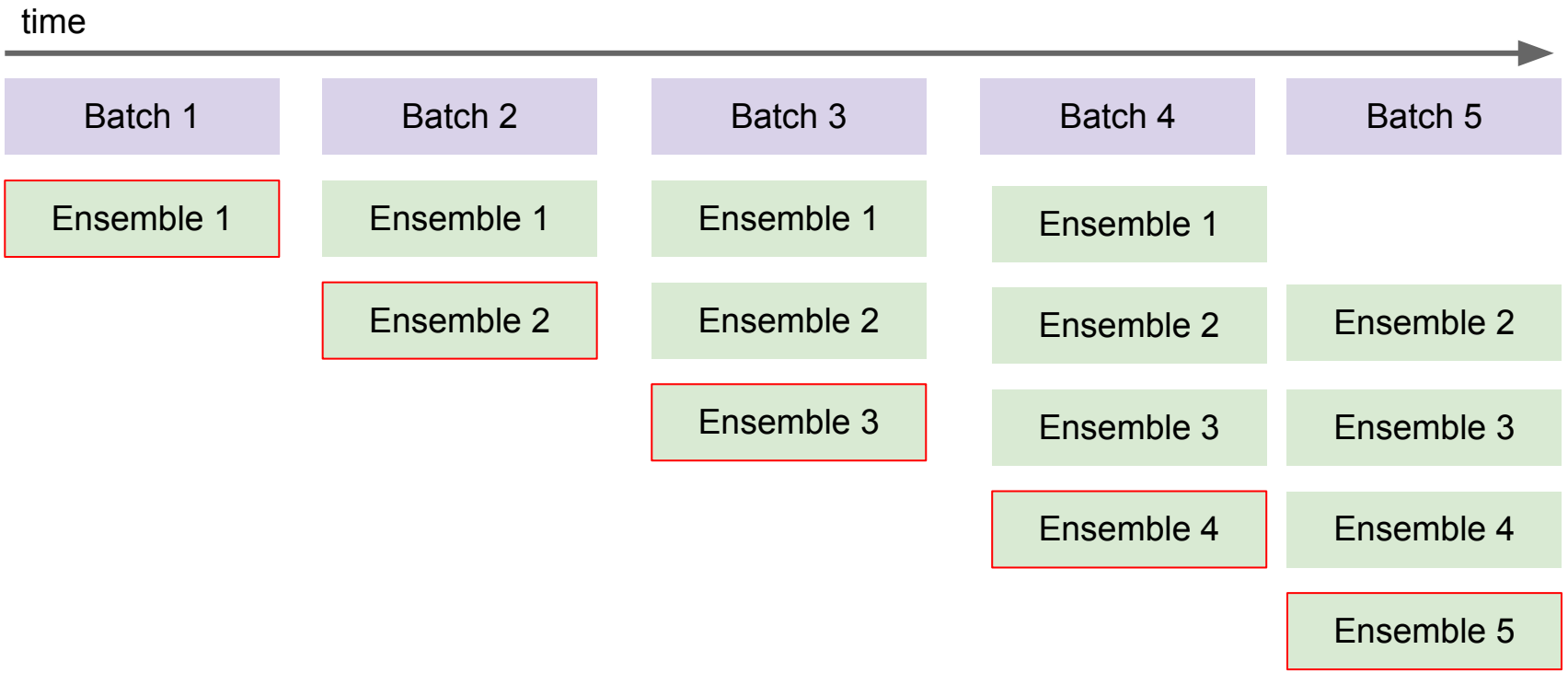
Update Dt

$$Dt = Dt \times \begin{cases} B_t, \ if \ H_t(x_i) = y_i \\ 1, \ otherwise \end{cases}$$

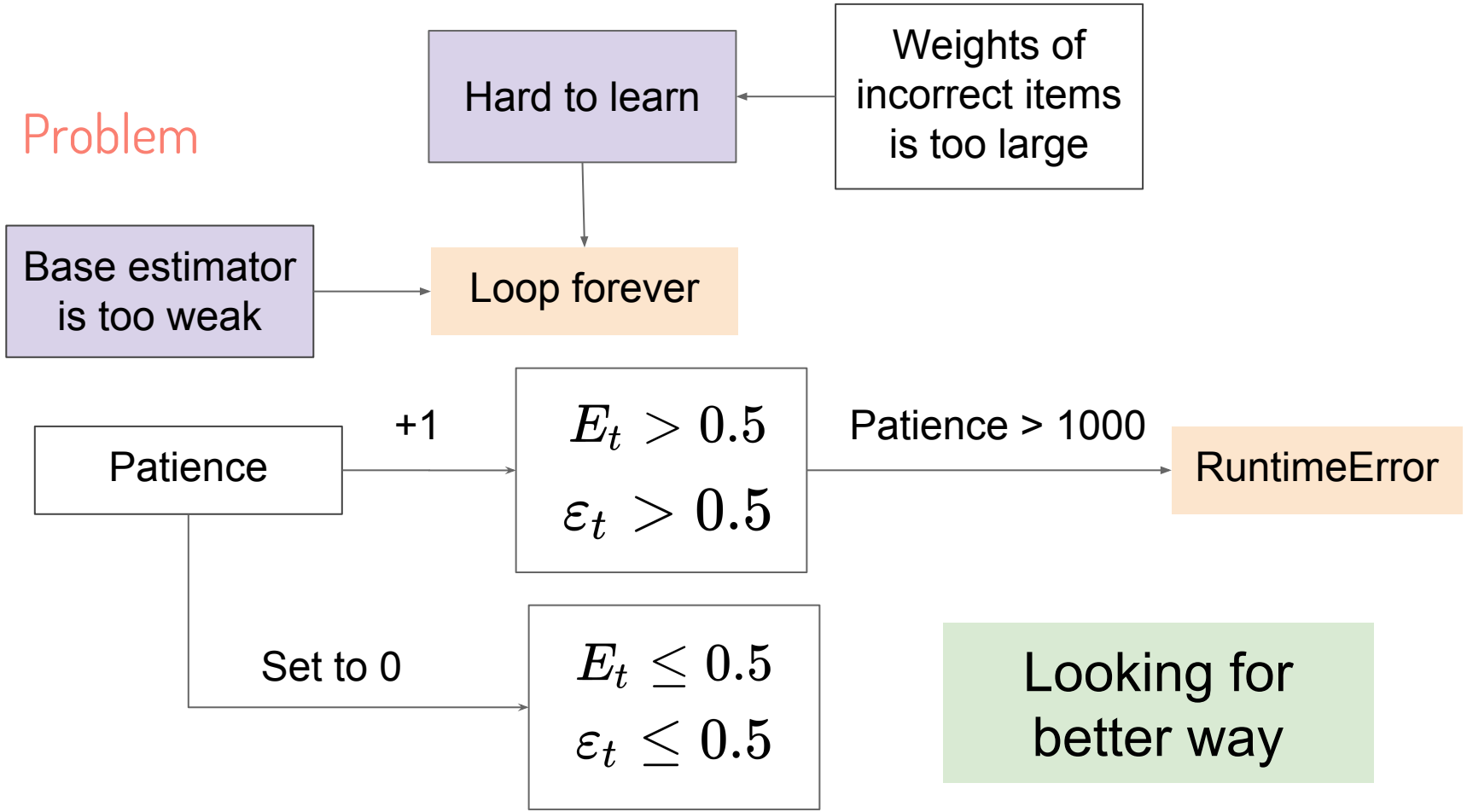
Next iteration

Adapt to data stream

N_ensembles = 3



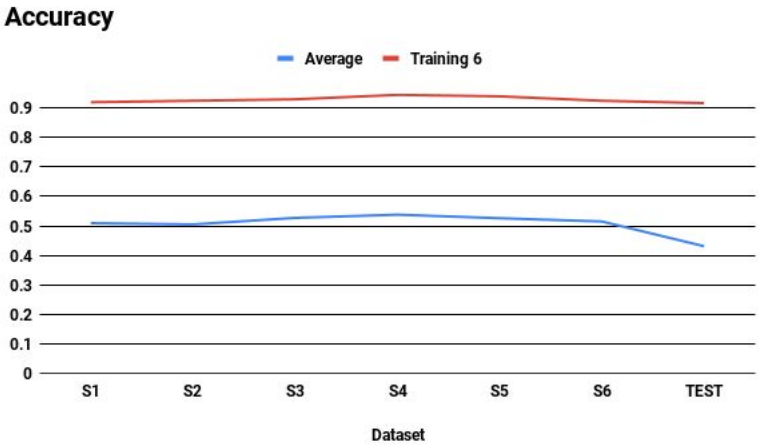
Problem


Hard to learn
Weights of incorrect items is too large
Base estimator is too weak
Loop forever
Patience
+1
$E_t > 0.5$
$\varepsilon_t > 0.5$
Patience > 1000
RuntimeError
Set to 0
$E_t \leq 0.5$
$\varepsilon_t \leq 0.5$
Looking for better way

3.
Evaluation

Optical digits dataset

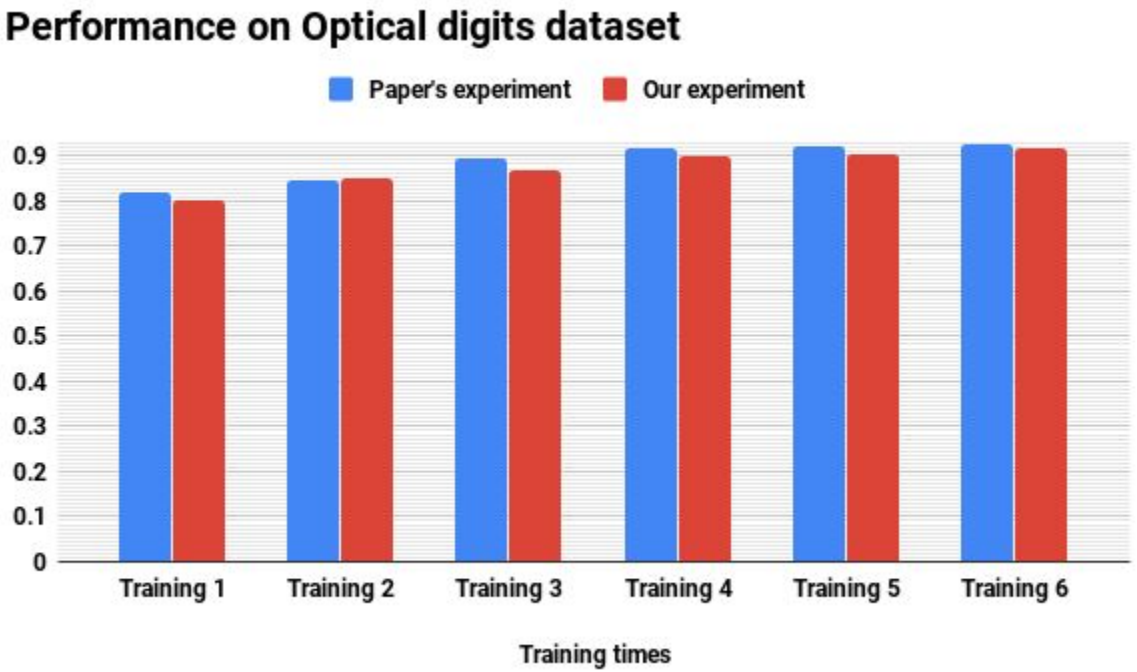
Dataset	Average	Training 1	Training 2	Training 3	Training 4	Training 5	Training 6
S1	0.5095	0.915	0.935	0.905	0.905	0.91	0.92
S2	0.5053		0.895	0.915	0.92	0.9	0.925
S3	0.5275			0.925	0.93	0.925	0.93
S4	0.5383				0.945	0.94	0.945
S5	0.5263					0.95	0.94
S6	0.5153						0.925
TEST	0.4313	0.803	0.852	0.867	0.899	0.903	0.917



Base Learner: MLP Classifier
hidden_layer_sizes=(30,)
tol=0.1
max_iter=default

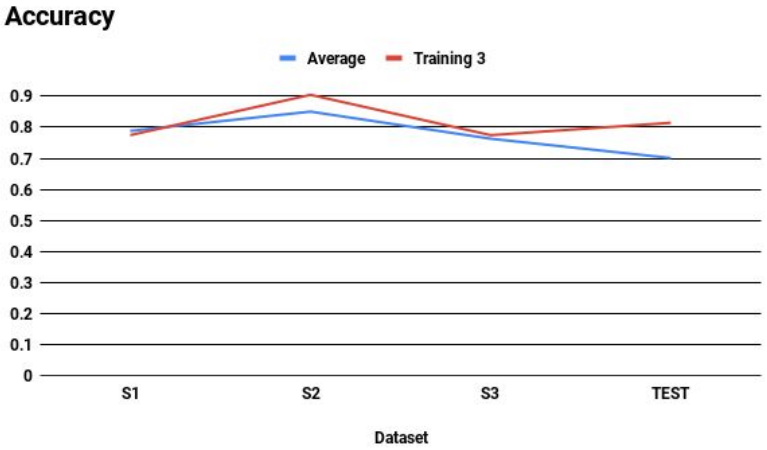
Number of estimators: 30

Performance comparison



Vehicle dataset

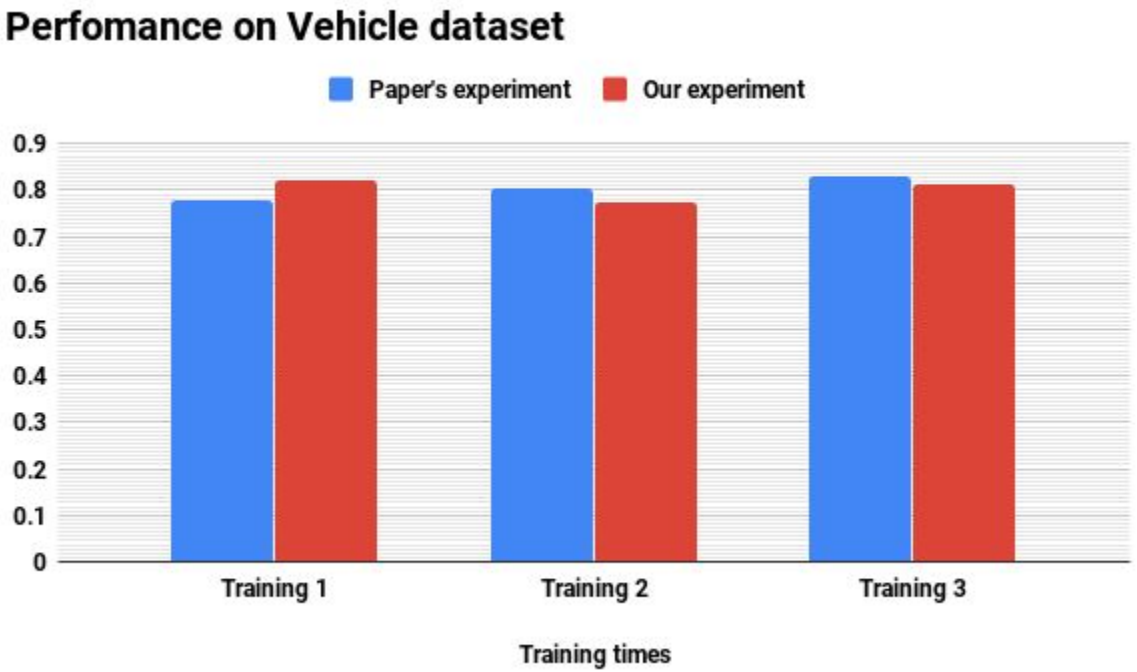
Dataset	Average	Training 1	Training 2	Training 3
S1	0.7886	0.915	0.795	0.775
S2	0.8509		0.895	0.905
S3	0.7635			0.775
TEST	0.7012	0.8194	0.7731	0.8148



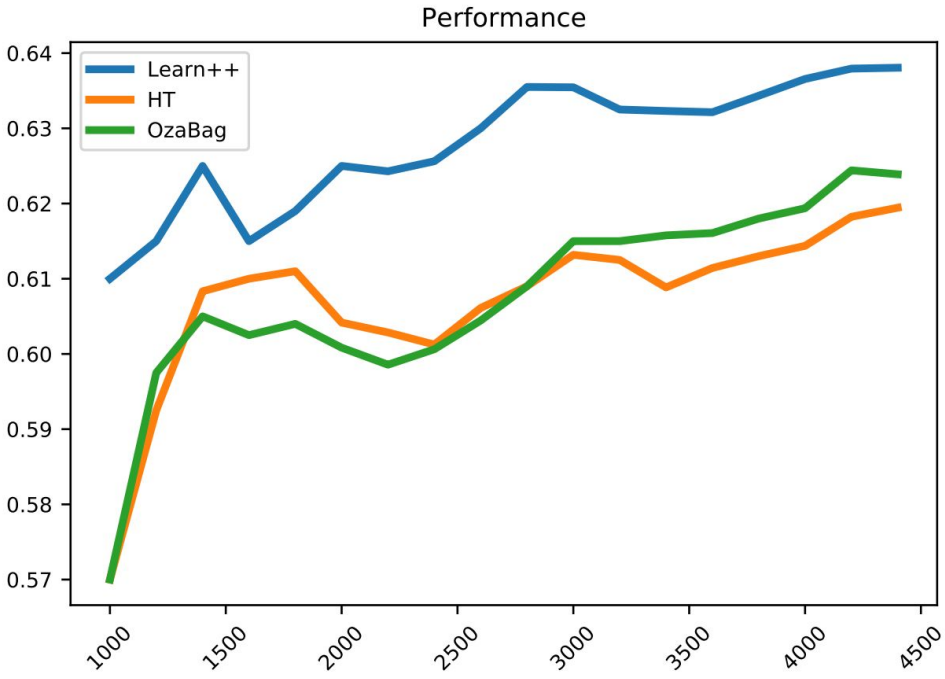
Base Learner: MLPClassifier
hidden_layer_sizes=(100,)
tol=1e-3,
max_iter=500

Number of estimators: 30

Performance comparison



Performance on Data Stream



Configuration:

Learn++: MLP (128 hidden layer)
30 estimator

OzaBagging: Hoeffding Tree
5 estimator

Stream: RandomTreeGenerator

4.
Conclusion

Summary

1. Motivation
2. Explanation of main idea
3. Implementation
4. Evaluation on different dataset

Future work

1. Make more tests with data stream
2. Optimize the codes.
3. Better way to adapt to data stream.
4. Integrate in Scikit Multiflow.