SPARK+AI
SUMMIT EUROPE

Streaming Random Forest Learning in Spark and StreamDM

Heitor Murilo Gomes and Albert Bifet
Télécom ParisTech

#SAISEco7

Agenda

ML for streaming data

Streaming Random Forest

StreamDM and implementations

Future

About me

- Heitor Murilo Gomes
- PhD in Computer Science
- ML Researcher at Télécom ParisTech
- Contribute to StreamDM and MOA

- heitor.gomes@telecom-paristech.fr
- Website: www.heitorgomes.com
- Linkedin: www.linkedin.com/in/hmgomes/

Data streams

You **can’t** store all the data

Data streams

You **don't want to** store all the data

ML for Streaming data

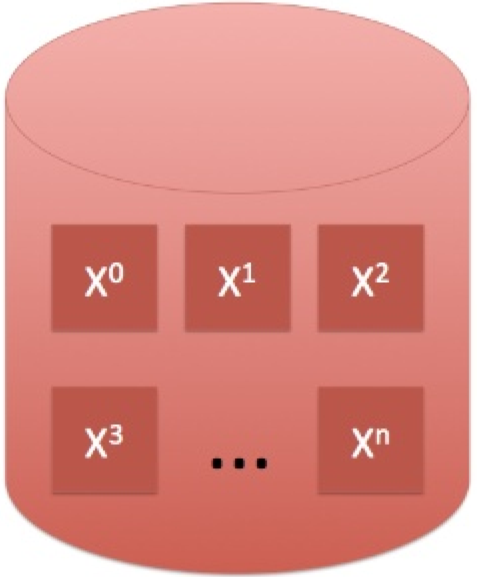
Machine Learning for...

Batch data X **Streaming** data

Batch data

Well defined training phase

Random access to instances

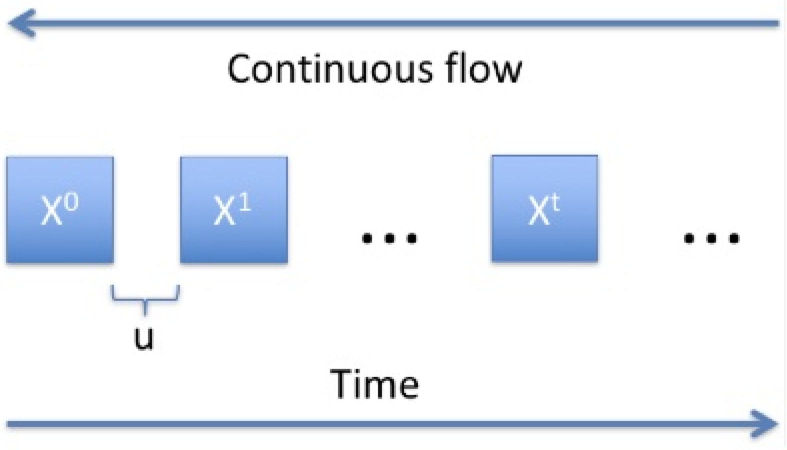


Challenges: missing data, noise, imbalance, high dimensionality, ...

Streaming data

Sequential access only

Strict time/memory requirements



Non-stationary data distribution

Challenges: inherit those from batch + concept drifts, feature evolution, ...

Batch x Streaming

Batch data

The output is a **trained model**

Streaming data

…

The output is a **trainable model**

Definitions / Assumptions

Independent and identically distributed (iid)

(x^t, y^t) does not influence (x^{t+1}, y^{t+1})

Immediate vs. delayed labeling

immediate: y^t is available before x^{t+1} arrives
delayed: y^t may be available at some point in the future

Stationary and non-stationary

Stationary data

The data distribution is unknown, but it doesn't change

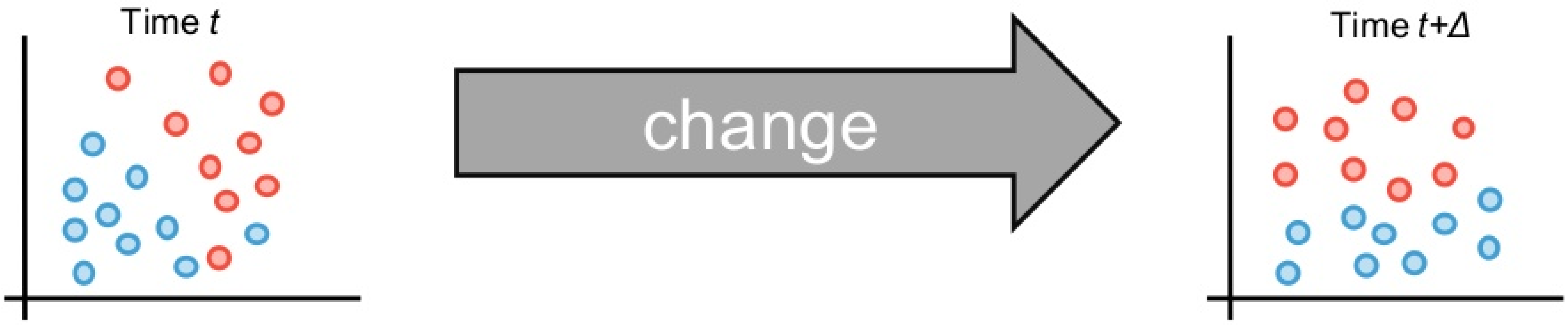
Non-stationary (evolving) data

The data distribution is unknown and it may change

Concept Drift

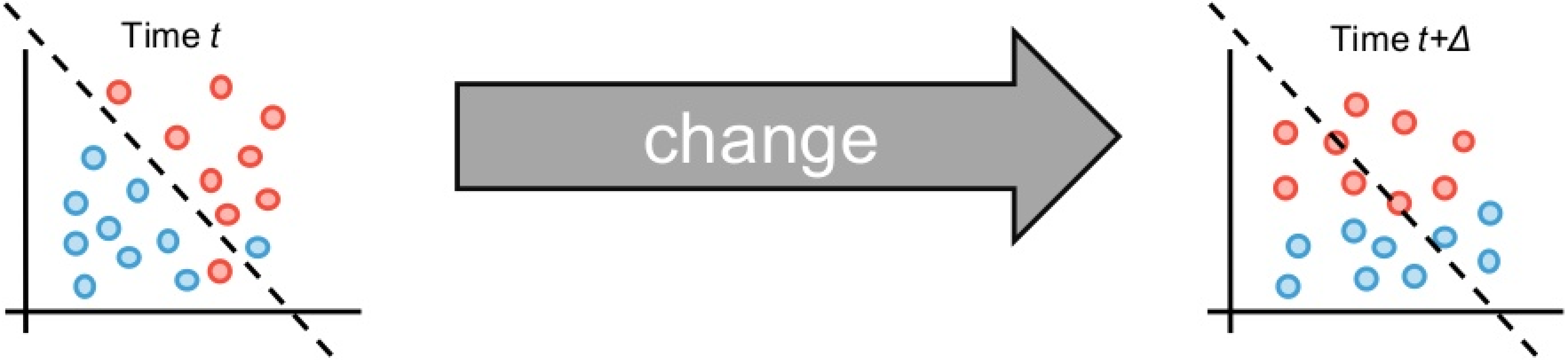
Concept drift

The data distribution may change overtime



Concept drift

The data distribution may change overtime

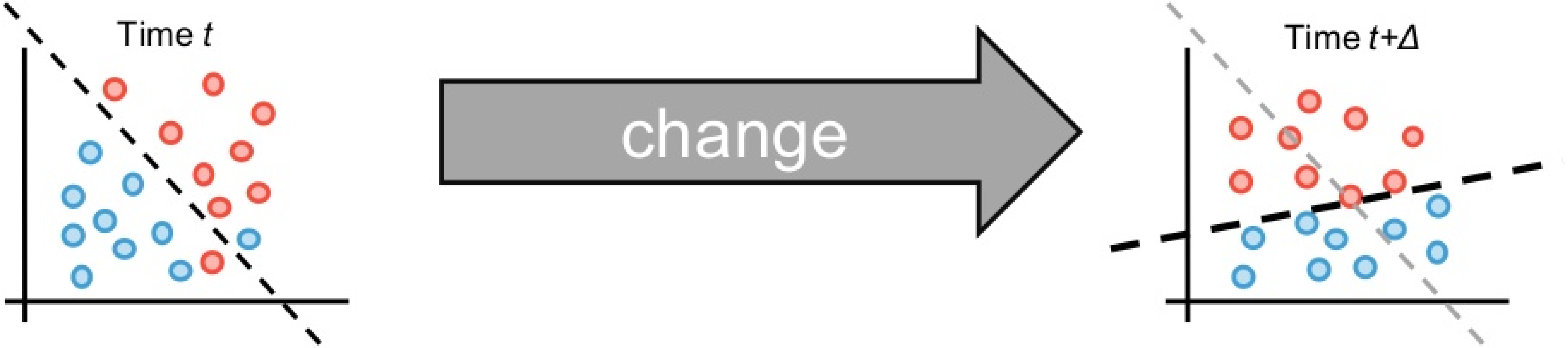


An accurate model

Not anymore...

Concept drift

The data distribution may change overtime



An accurate model

An updated model!

Concept drift

There are many ways to categorize concept drifts…

… Abrupt, gradual, incremental …

A survey on concept drift adaptation. Gama, Žliobaitė, Bifet, Pechenizkiy, Bouchachia (2014). *Computing Surveys (CSUR), ACM.*

… in general, we care about those that **disturb our model**

Addressing concept drifts

Focus on supervised learning*

Some strategies:

Reactive: ensemble learner
Active: drift detector + learner

Addressing concept drifts

Why use an ensemble to cope with concept drift?

Flexibility. Relatively easy to forget and learn new concepts by removing/resetting base models and training new ones.

A survey on ensemble learning for data stream classification. Gomes, Barddal, Enembreck, Bifet (2017). *Computing Surveys (CSUR), ACM.*

Are there other challenges?

Yes, unfortunately (or fortunately)

Concept evolution
Feature evolution
Continual preprocessing
Verification latency (delayed labeling)
…

Our current approach

Ensemble: **StreamingRandomForest**

Base learner: **StreamingDecisionTree**

Implementation compatible with **mllib**

Another implementation in the **StreamDM** framework

HoeffdingTree / StreamingDecisionTree

Incremental decision tree learning algorithm

Intuition: Splitting after observing a small amount of data

Also known as Very Fast Decision Tree (VFDT)

Mining High-Speed Data Streams. Domingos, Hulten (2000). *KDD, ACM.*

StreamingRandomForest

Streaming version of the original Random Forest by Breiman

[**Random forests.** Breiman (2001). *Machine learning, Springer*.]

Main differences:

Bootstrap aggregation and the **base learner**

Overview:

1. *Online bagging*
2. Random subset of features

Streaming Random Forest

1. Online bagging

"Standard" bootstrap aggregation is inviable

Issue: Can't resample from the dataset as it is not stored

Solution: Approximate it using a Poisson distribution with $\lambda=1$

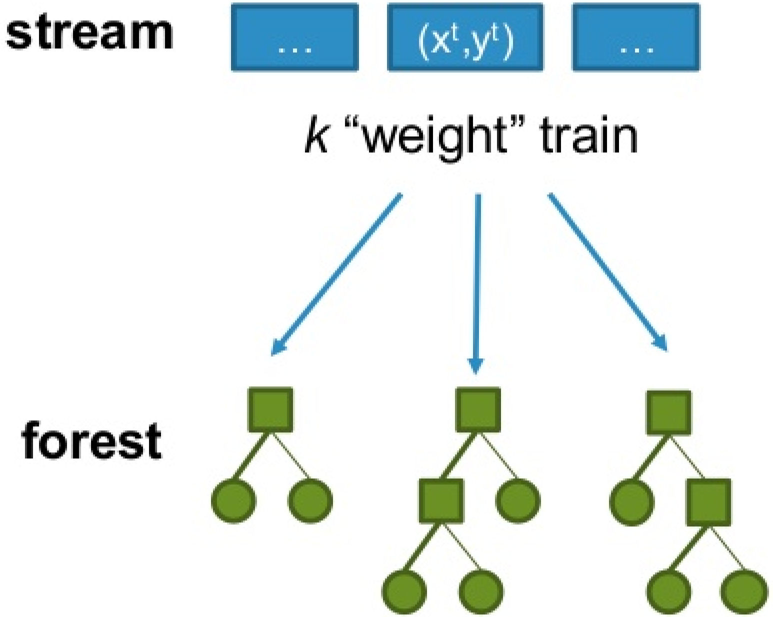
Online bagging and boosting. Oza (2005).

Streaming Random Forest

1. Online bagging

$k \leftarrow Poisson$ **(λ=1)**
$\textbf{if } k > 0 \textbf{ then}$
$\quad l \leftarrow FindLeaf(t, x)$
$\quad UpdateLeafCounts(l, x, k)$

Practical effect: train trees with different subsets of instances.



Streaming Random Forest

1. Online bagging

Leveraging [online] bagging

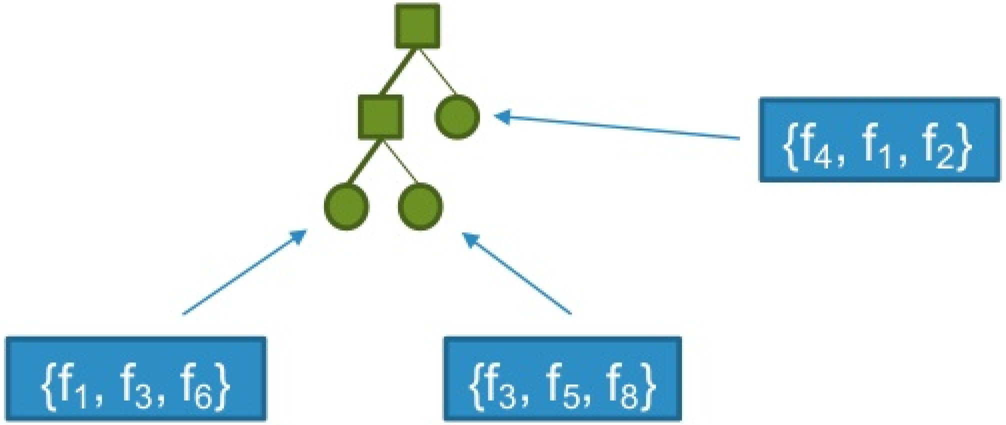
Augment λ (usually 6), such that instances are used more often

Leveraging bagging for evolving data streams. Bifet, Holmes, Pfahringer (2010). *ECML-PKDD, Springer*.

Streaming Random Forest

2. Randomly select subsets of features for splits

Use a **subset of randomly selected features** for each **split**



Implementation

Mllib streaming algorithm:
    StreamingLogisticRegressionWithSGD

StreamingDecisionTree
- Implementation of the HoeffdingTree algorithm

StreamingRandomForest
- Implementation of the Random Forest algorithm

Similar API

```
val streamLR: StreamingLogisticRegressionWithSGD =
  new StreamingLogisticRegressionWithSGD()
    .setInitialWeights(Vectors.zeros(elecConfig.numFeatures))
…
streamLR.trainOn(examples)
```

```
val streamDT: StreamingDecisionTree =
  new StreamingDecisionTree(instanceSchema).
    setModel().setMaxDepth(Option(10))
…
streamDT.trainOn(examples)
```

StreamingDT implementation

Similar coding standard as
StreamingLogisticRegressionWithSGD

Handles both nominal and numeric attributes
- Numeric using Gaussian estimator

Training
- Statistics are computed in the workers
- Splits are decided and performed in the driver

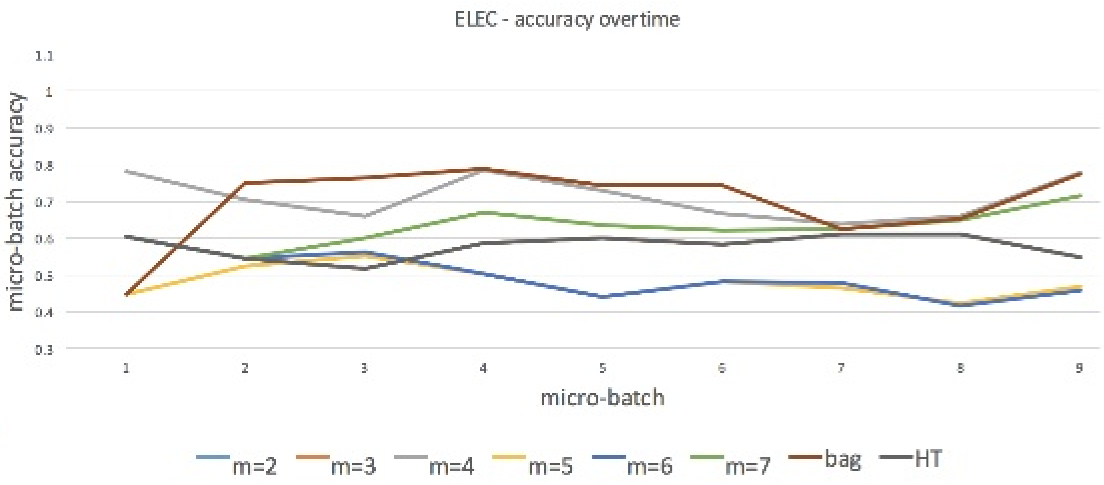
StreamingRF implementation

Similar to StreamingDecisionTree

Training

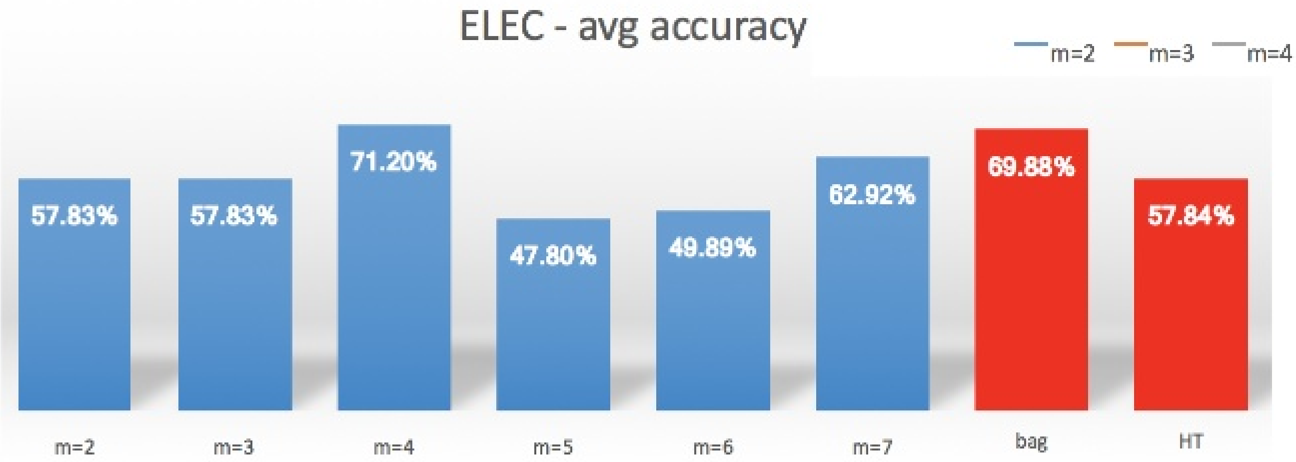
- Delegates training to the underlying tree models
- Uses online bagging

Some results



Electricity dataset

~50k instances
8 features
2 class labels



Some results

Covertype dataset

~600k instances
54 features
7 class labels

Streaming Random forest

MaxDepth = 20
NumTrees = 10 and 100

StreamDM

Started in Huawei Noah's Ark Lab

Collaboration between **Huawei Shenzhen** and **Télécom ParisTech**

Open source

Built on top of **Spark Streaming***

- Experimental **Structured Streaming version**

Extensible to include new tasks/algorithms

Website: http://huawei-noah.github.io/streamDM/

GitHub: https://github.com/huawei-noah/streamDM

GitHub (structured streaming): https://github.com/hmgomes/streamDM/tree/structured-streaming

StreamDM

Stream readers/writers

Classes for reading data in and outputting results

Tasks

Setting up the learning cycle (e.g. train/predict/evaluate)

Methods

Supervised and unsupervised learning algorithms. Hoeffding Tree, CluStream, Random Forest*, Bagging, …

Base/other classes

Instance and Example representation, Feature specification, parameter handling, …

DEMO

Wrap-up / Future

Machine learning for streaming data

- Non-stationary (and concept drift)

Streaming Decision Trees and Random Forest implementations

Future:

- StreamDM Structured streaming version
- Improve performance (both decision tree and random forest)
- More methods (anomaly detection, multi-output, …)

Further reading / links

Machine Learning for Data Streams
Albert Bifet, Ricard Gavaldà, Geoffrey Holmes, Bernhard Pfahringer

Contact
heitor.gomes@telecom-paristech.fr

Repository (main)
https://github.com/huawei-noah/streamDM
Repository (structured streaming)
https://github.com/hmgomes/streamDM/tree/structured-streaming