Unified Streaming/Batch Learning and Explainable Multi-output Prediction

Jesse Read

1. Data Streams as Time Series
2. Chaining Methods for Multi-Output Learning
3. Applications of Chaining in Data Streams

Outline

1. Data Streams as Time Series
2. Chaining Methods for Multi-Output Learning
3. Applications of Chaining in Data Streams

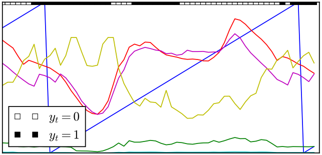
Data Streams

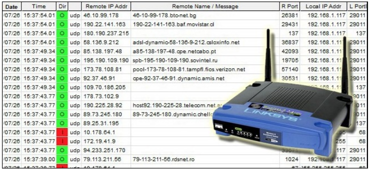
A data stream,

$$x_1, x_2, \ldots, x_t, \ldots$$

where, at real time t we observe x_t, which comes from some concept (which we don't observe directly):

$$x_t \sim P_t$$





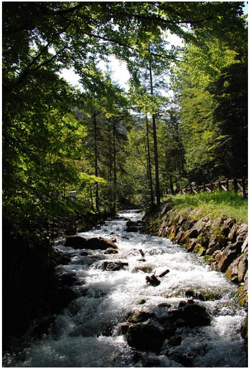
Electricity dataset (left), image [1] (right)

Applications: IoT, energy/traffic and demand prediction, monitoring and tracking, event and fraud detection, click/web logs, finance, reinforcement learning, ...

Requirements

To deploy a model in the data stream setting, we require:

1. Prediction/action done immediately ($\widehat{y}_t = h_t(x_t)$ at time t)
2. Computational time spent per instance must be less that the rate of arrival



Streaming Classification

Supervised ML models are often studied in the context of streams.

Common assumptions found in the literature

1. Speed and size of stream implies instance-incremental learning (*at most* a single look at each data point)
2. The true label of data points become available (providing a stream of training examples)
3. No temporal dependence
4. Concept drift will occur

[1] e.g., predicting the weather – true label comes the next day

Streaming Classification

Supervised ML models are often studied in the context of streams.

Common assumptions found in the literature

1. Speed and size of stream implies instance-incremental learning (*at most* a single look at each data point)
2. The true label of data points become available (providing a stream of training examples)
3. No temporal dependence
4. Concept drift will occur

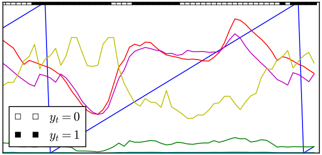
Some observations:

- Assumption 1 is unnecessary
- Assumption 2: Where do true labels from?
  - A human – then contradicts 1. (in most cases)
  - The future[1] – then contradicts 3. – it is a time series
- Assumptions 3 and 4 are contradictory

[1] e.g., predicting the weather – true label comes the next day

Data Streams as Time Series

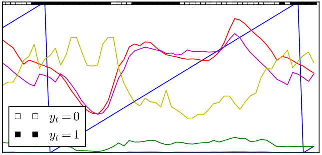
Benchmark datasets often look like time series:



Prediction of Electricity demand

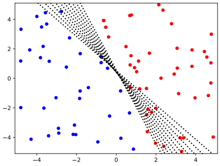
Data Streams as Time Series

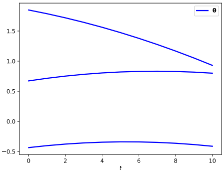
Benchmark datasets often look like time series:

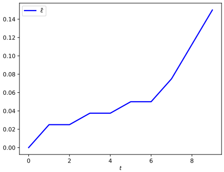


Prediction of Electricity demand

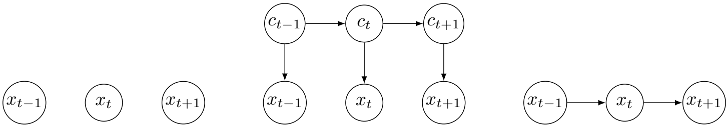
Even when points sampled iid wrt current concept, a time series forms in the coefficients, and/or in the error signal:







Concept drift ⇒ temporal dependence:



Time series tasks:

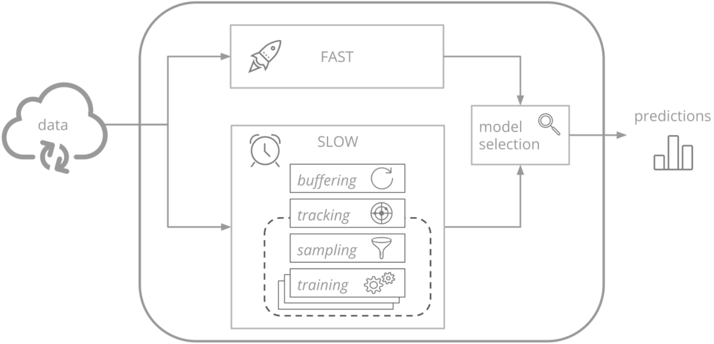
- Filtering
- Forecasting
- State labelling/change point detection
- Event/anomaly detection
- ...

(no supervised streaming classification!)

A data stream is a time series with constraints (prediction required *now*, update faster than rate of arrival).

Fast and Slow Learning

- A framework for Fast and Slow learning
- Invest in higher level (slow) processes
- Batch and stream learning need not be mutually exclusive
- Time series methods, weakly labeled and unlabeled data
- Awareness of multi-input *multi-output* setting



Built into Scikit MultiFlow framework: `https://scikit-multiflow.github.io/`

Montiel et al. 2018a

Outline

1. Data Streams as Time Series
2. Chaining Methods for Multi-Output Learning
3. Applications of Chaining in Data Streams

Multi-label Learning

Input, e.g.,

$\mathbf{x} =$ 

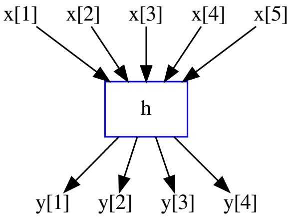
Prediction/output, e.g.,

$$\widehat{\mathbf{y}} = [1, 0, 1, 0, 0] \Leftrightarrow \{\text{beach}, \text{foliage}\}$$

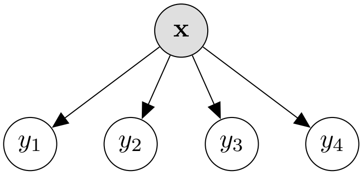
i.e., multiple outputs *per instance*.

Multi-label Problem $[Y_1, \ldots, Y_L] \in \{0,1\}^L$

X_1	X_2	X_3	X_4	X_5	Y_1	Y_2	Y_3	Y_4
1	0.1	3	A	NO	0	1	1	0
0	0.9	1	C	YES	1	0	0	0
0	0.0	1	A	NO	0	1	0	0
1	0.8	2	B	YES	1	0	0	1
1	0.0	2	B	YES	0	0	0	1
0	0.0	3	A	YES	?	?	?	?



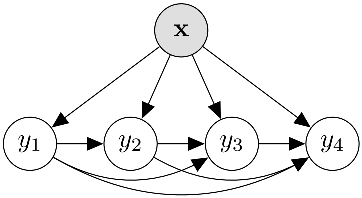
Why not Independent Classifiers?



If we model labels together, we can achieve

- Better predictive performance
- Better computational performance
- Interpret relationships among labels (i.e., interpretability)
- Approach structured-output prediction tasks

Classifier Chains



- Predictions cascade along a chain (as additional features)
- Has a probabilistic interpretation:

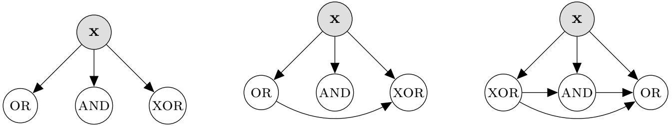
$$\widehat{\mathbf{y}} = \underset{\mathbf{y}\in\{0,1\}^L}{\operatorname{argmax}} P(y_1|\mathbf{x}) \prod_{j=2}^{L} P(y_j|\mathbf{x}, y_1, \ldots, y_{j-1})$$

- Inference becomes a search (for best $\widehat{\mathbf{y}}$, in $\{0,1\}^L$ space); e.g., greedy, Monte Carlo search, ϵ-greedy, beam search.

Read, Martino, and Luengo, Pat. Rec. 2014

Ordering/Structuring the Labels

1. Existing hierarchy? May not be useful
   - Only models positive dependence (if human-defined)
   - No guarantee of suitability for chosen classifiers
2. Based on label dependence? It depends (on classifiers, inference, . . .); consider:



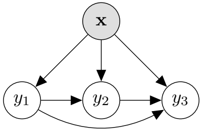
Metric	(left)	(middle)	(right)
Hamming score	0.83	1.00	0.83
Exact match	0.50	1.00	0.50

Logistic regression at each node h_j, greedy inference

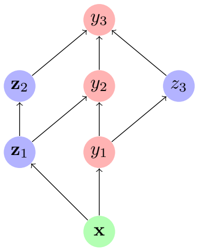
3. Hill-climbing in the label-structure space: Slow(!), but
   - Many local maxima (easy to reach) – i.e., it works!
   - Can make use of sub-optimal models that were trialled

Classifier Chains: Why it Works

As a probabilistic graphical model:

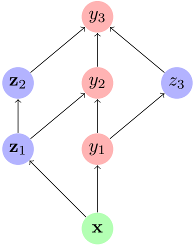


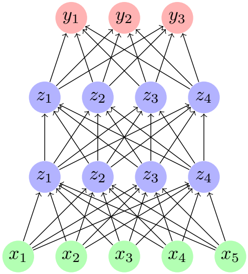
vs as a neural network (z delay nodes simply carry forward value):



it's deep in the label space!

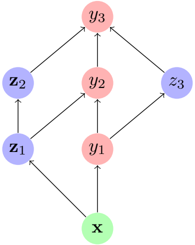
Advantages vs standard neural network?

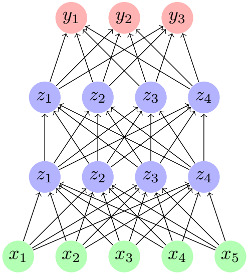




- Just apply 'off-the-shelf' [deep] neural net?
  - Dependence is modelled via the hidden layer(s)
  - Well-established, popular, competitive
- But with classifier chains:
  - The 'hidden' nodes come 'for free' (they're not hidden): faster training, less data required
  - A form of transfer learning

Advantages vs standard neural network?



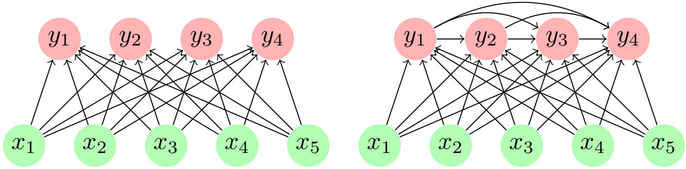


- Just apply 'off-the-shelf' [deep] neural net?
  - Dependence is modelled via the hidden layer(s)
  - Well-established, popular, competitive
- But with classifier chains:
  - The 'hidden' nodes come 'for free' (they're not hidden): faster training, less data required
  - A form of transfer learning

Observation: a bad/outdated prediction does not mean a bad representation!

Multi-Output Regression

X_1	X_2	X_3	X_4	X_5	Y_1	Y_2	Y_3
1	0.1	3	A	NO	37.00	25	0.88
0	0.9	1	C	YES	-22.88	22	0.22
0	0.0	1	A	NO	19.21	12	0.25
1	0.8	2	B	YES	88.23	11	0.77
1	0.0	2	B	YES	?	?	?

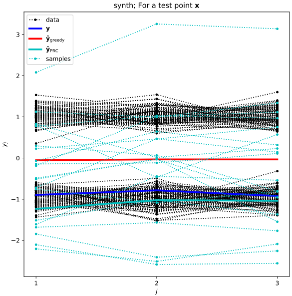


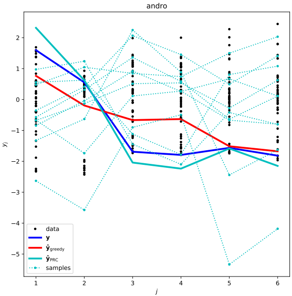
- Individual regressors – directly applicable.
- Chains
  - greedy inference – directly applicable, but may be pointless!
  - with probabilistic inference – not tractable, but we can sample *if we have* $p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1})$.

Regressor Chains

Results of chains under MSE (mean squared error) no better than using individual models / not interesting, *unless*

- Predictions provide an improved (non-linear) representation.
- Non-isotropic (state space models; where x_j is seen at 'time' j)
- We are interested in interpretation/explainability, e.g.,
  - Anomaly detection
  - Missing-value imputation
- New label concepts arrive later (we can transfer learning), make computational time savings.



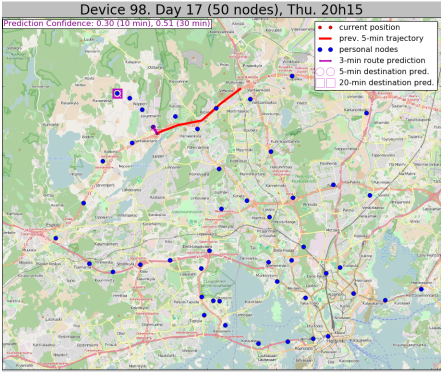


Outline

1. Data Streams as Time Series
2. Chaining Methods for Multi-Output Learning
3. Applications of Chaining in Data Streams

Route Forecasting

- Create 'personal nodes' for a traveller
- Model and predict routes using classifier chains
- An advantage with relatively little training data and vs other methods (e.g., HMM, RNN)

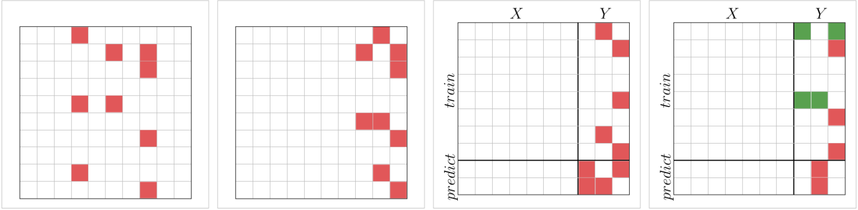


Personal nodes of a traveller and a predicted trajectory

Read, Martino, and Hollmén, Pat. Rec. 2017

Missing-Value Imputation

- Some values in the stream are missing!
- Turn the stream into multi-output samples, train, and predict (*impute*) missing values.
- Related to tasks in recommender systems

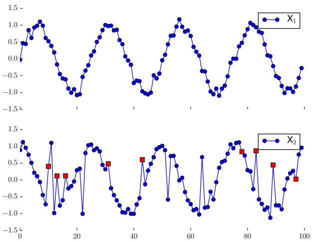


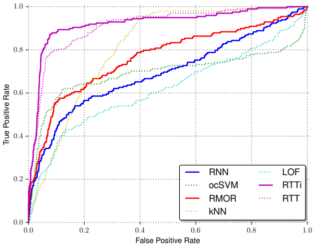
A set/stream of data transformed into a multi-output prediction problem.

Montiel et al., PAKDD 2018, and manuscript under review

Anomaly Detection and Interpretation

- Create 'random threads' (classifier/regressor chain cascades) through feature space and time (window)
- Monitor error spaces for anomalies
- Generate likely paths over the 'gap'
  (expand the number of samples if necessary)
- Impute this (treat it as a missing value) prior to using as a training example

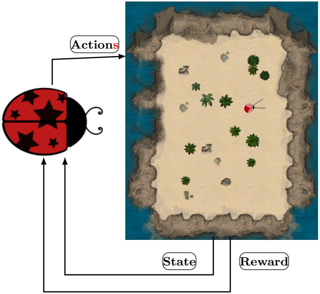




Song et al., ICDM demo 2018, and manuscript in preparation

Continual Learning

In reinforcement learning,



- Reward signal is sparse
- Self-train on own surrogate reward, then use it as a feature.
- Recall: Incorrect predictions are not useless representations
- i.e., build up representation; transfer learning.

Work in progress ...

Summary

1. Data Streams as Time Series
2. Chaining Methods for Multi-Output Learning
3. Applications of Chaining in Data Streams

Unified Streaming/Batch Learning and Explainable Multi-output Prediction

Jesse Read

ÉCOLE POLYTECHNIQUE
UNIVERSITÉ PARIS-SACLAY

DaSciM

http://www.lix.polytechnique.fr/~jread/
http://www.lix.polytechnique.fr/dascim/