OIL
Online Incremental Learning

Fabrice Harel-Canada
Migyeong (Mimi) Gwak

Roadmap

- Conceptual Overview
  - What is it
  - Why is it important
  - Applications
  - Key Challenges
  - Models
- Examples
  - Learn++
  - ADAIN

What is OIL?

- Continuous model adaptation (i.e. learning) from streaming data that arrives over time
- Formally, an OIL system must do all of the following:
  - Learns from new data, generally as soon as it becomes available
  - Does not require access to previously processed data (one pass)
  - Preserves previously acquired knowledge
  - Accommodates new classes

Why is OIL important?

- Handles limited upfront data
  - Expensive or time consuming
  - Unknown size or distribution
- Lower memory consumption
  - Doesn’t need to store previously seen data for reprocessing
- Foundation for greater system autonomy

Applications

- Big Data analytics
- Robotics
- Self-Driving Transport
- Image/Video Processing
- Automated Annotation
- Outlier Detection

What are the key challenges?

- Concept drift
- Stability-plasticity dilemma
- Adaptive model complexity
- Efficient memory representation
- Model benchmarking

Concept Drift

Changes in the distribution of data over time

- What is changing?
  - The input distribution ($\mathbf{x}$) → virtual / covariate drift
  - The function itself $p(y|\mathbf{x})$ → real drift
- How fast is the data changing?
  - Slow → drift
  - Fast → shift
- How widespread is the data changing?
  - Only some areas → localized drift
  - Everything → systemic drift

The Stability-Plasticity Dilemma

- Learning new things tends to overwrite previously acquired knowledge
- Update too quickly and previously encoded data will be forgotten constantly, sometimes catastrophically
- Update too slowly and impaired adaptation hurts performance
- Moderation between these two is key!

Adaptive Model Complexity

- Unknown data means that model complexity and meta-params cannot be specified upfront
- Reduces critical meta-params like learning rate into another model parameter to be tuned automatically

Efficient Memory Representation

- Compact representations of knowledge
  - Invariants
    - E.g. classification error for a drift detector
  - Implicit representations params
    - Most common
  - Explicit
    - E.g. limited set of typical examples (prototypes / exemplars)

Model Benchmarking

- Incremental vs Non-incremental
  - Task accuracy of the models using data from the same timeframe
  - The one-pass nature of most stream processing is a critical constraint in these comparisons
- Incremental vs Incremental
  - Measures of robustness and classification error against concept drift

Types of OIL Models

- Support Vector Machines (SVM)
- Connectionist Models
  - Multilayer perceptrons (MLP) / Neural Networks
- Input Partitioners
  - Trees / windows
- Prototype-based
- Ensembles
  - A little bit of everything

Learn++: An Incremental Learning Algorithm for Supervised Neural Networks

...

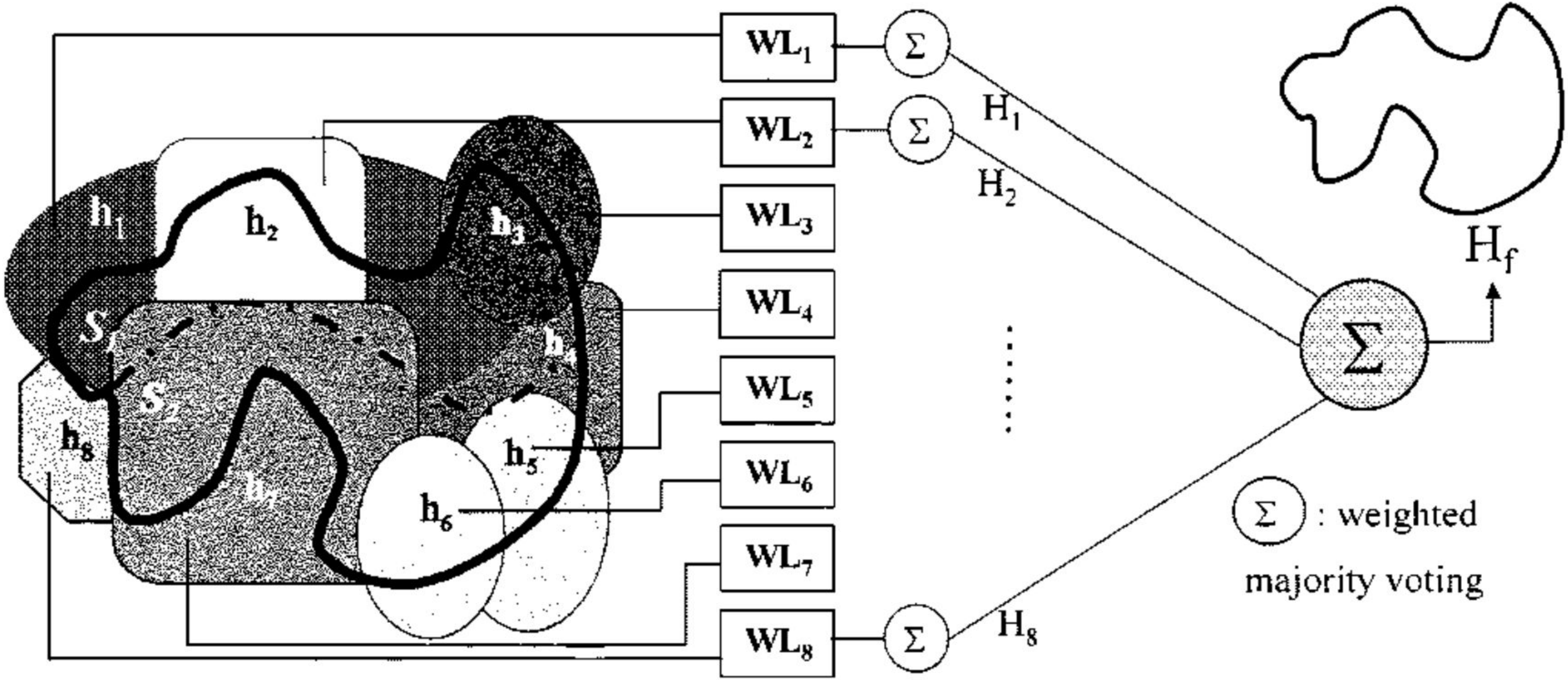
Robi Polikar | Lalita Udpa | Satish S. Udpa | Vasant Honavar

IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS VOL. 31, NO. 4, NOVEMBER 2001

Learn++

- Early example of an ensemble approach to incremental learning for classification tasks
- Builds upon the concept of boosting “weak learner” into “strong learner” by combining them together
- Uses weighted majority voting between these boosted classifiers to make the final output class prediction
- Focuses on novelty by selecting more strongly for incorrectly classified data

Incremental learning via weighted weak classifiers

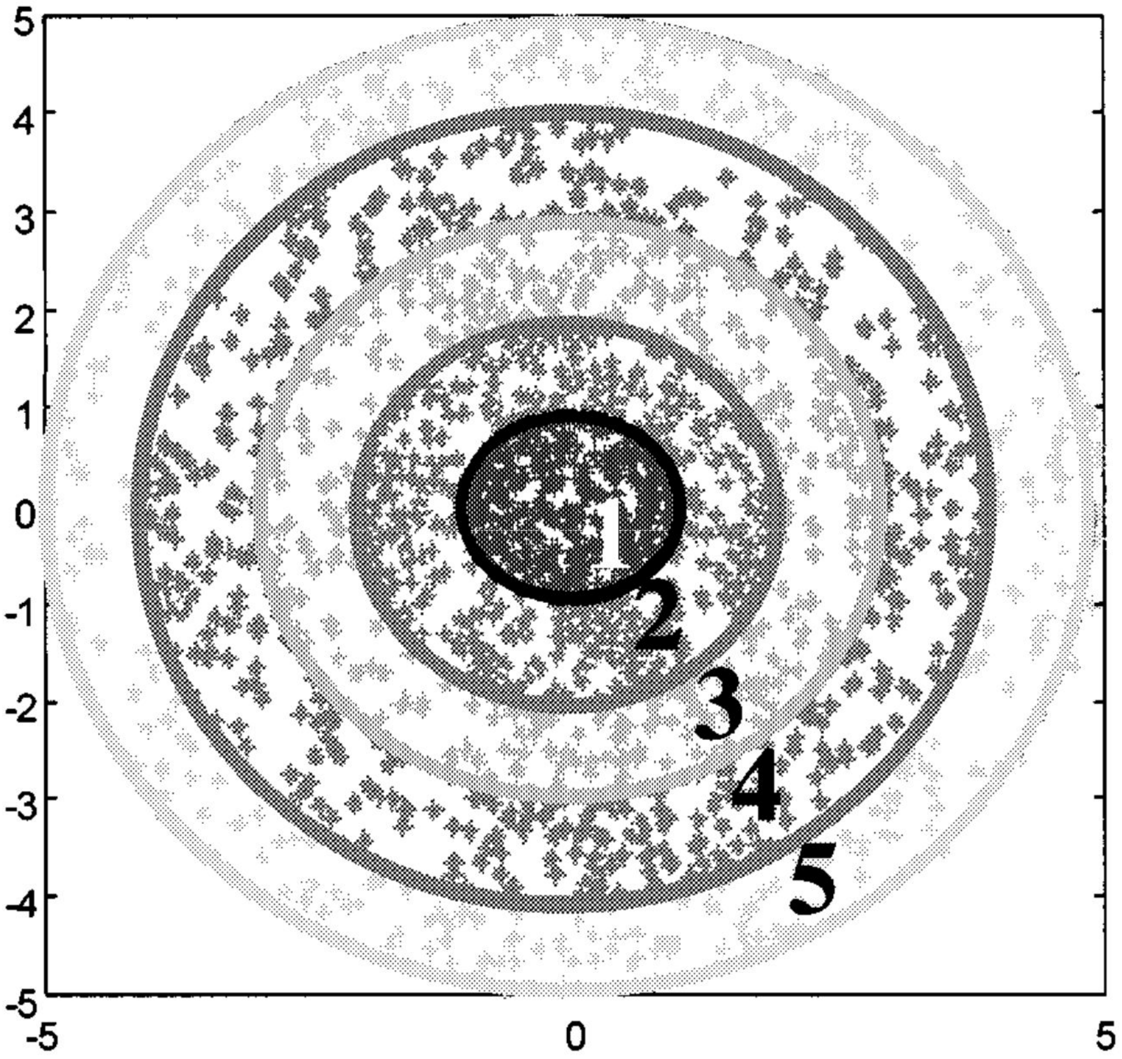


How does Learn++ overcome the challenges of OIL?

- It handles concept drift by focusing on hard / novel data
- It handles the stability-plasticity dilemma by using weighted majority voting so that new classes can be accommodated when encountered, while still strongly weighting the learners that were good at classifying the previous set of classes
- It handles adaptive model complexity by adding new classifiers until performance can no longer be improved
- It compactly represents knowledge in params / weights

Concentric Circles

- 2 attributes (x, y)
- 5 classes
- Goal
  - Initialize the OIL with data from the 1, 2, 3
  - Start including examples from new classes



Results

TRAINING AND GENERALIZATION PERFORMANCE OF LEARN++ ON CONCENTRIC CIRCLES DATABASE

Inc. Train→ ↓ Dataset	Training1 (10)	Training 2 (10)	Training 3 (13)	Training 4 (3)	Training 5 (15)	Training 6 (7)	Last 7
S_1	98.7%	96.7%	91.4%	91.4%	95.3%	95.3%	41.7%
S_2	---	96.1%	87.1%	85.8%	92.2%	91.6%	40.6%
S_3	---	---	98.3%	98.3%	72%	90.8%	51.5%
S_4	---	---	---	93.6%	77%	88.4%	49.8%
S_5	---	---	---	---	88%	95.2%	60.4%
S_6	---	---	---	---	---	96.4%	53.6%
TEST	**55.6%**	**56.8%**	**73.2%**	**74.4%**	**85.8%**	**89.6%**	52.8%

- Test shows monotonic improvement as it learns how to classify unknown classes
- Large accuracy improvements coincide with new class introductions in 3 and 5
- Last 7 highlights the importance of previously learned classifications

Incremental Learning from Stream Data

•••

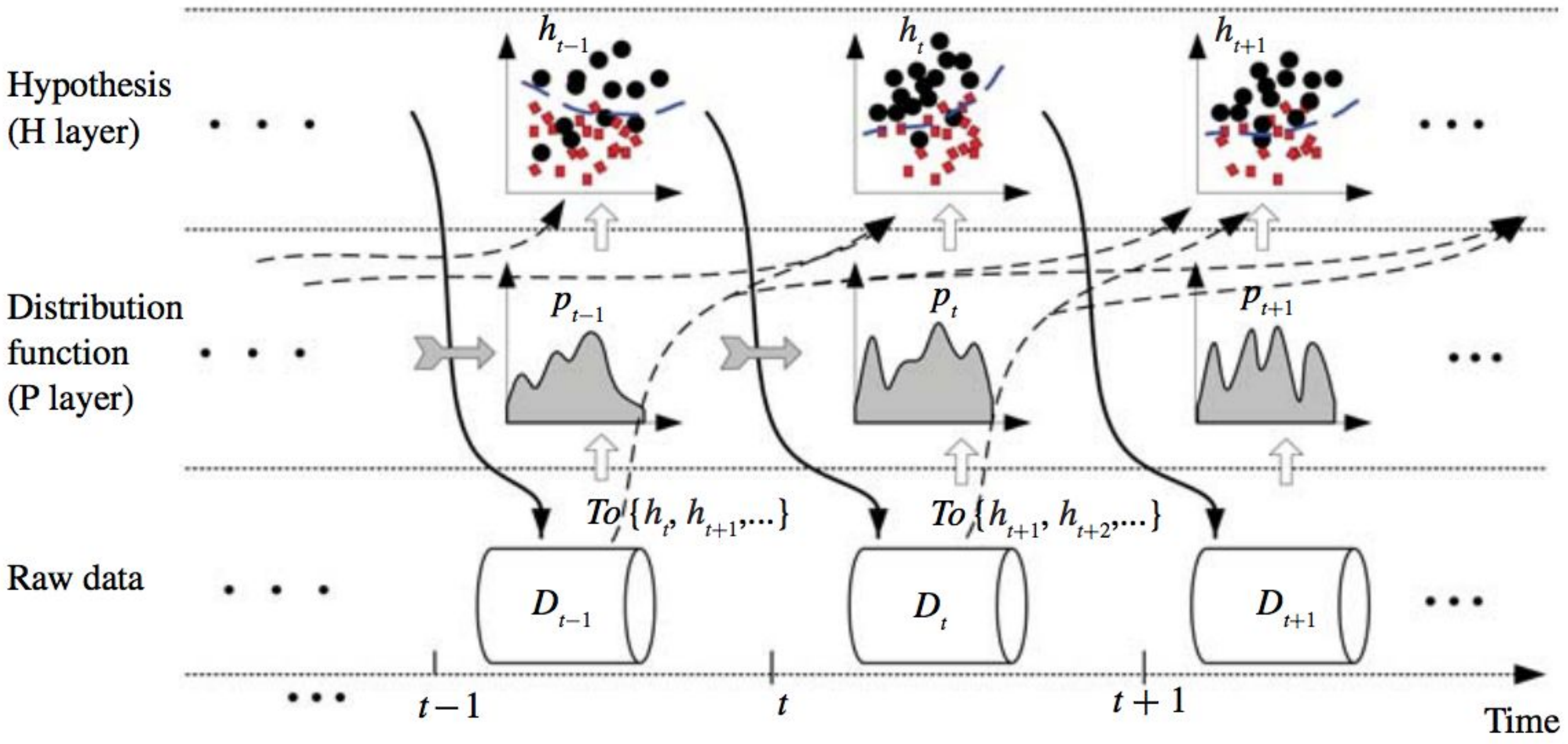
Haibo He | Sheng Chen | Kang Li | Xin Xu

IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 22, NO. 12, DECEMBER 2011

ADAIN

- A general **ADA**ptive **IN**cremental learning framework.
- Motivated by the adaptive boosting principle and ensemble learning methodology.
- Learning from stream data for classification.
- Accumulating experience over time.
- Using such knowledge to improve future learning and prediction performance.
- Different base classifiers can be integrated.

System Architecture



Learning Procedure

1. Estimate the initial distribution function for the current data set (D_t) using a mapping function.

$$\hat{\boldsymbol{P}}_{t-1} = \varphi(\mathcal{D}_{t-1}, \mathcal{D}_t, P_{t-1})$$

2. Apply previous hypothesis (h_{t-1}) to D_t, calculate the pseudo-error of h_{t-1}.

$$\varepsilon_{t-1} = \sum_{j:h_{t-1}(\boldsymbol{x}_j) \neq y_j} \hat{\boldsymbol{P}}_{t-1}(j)$$

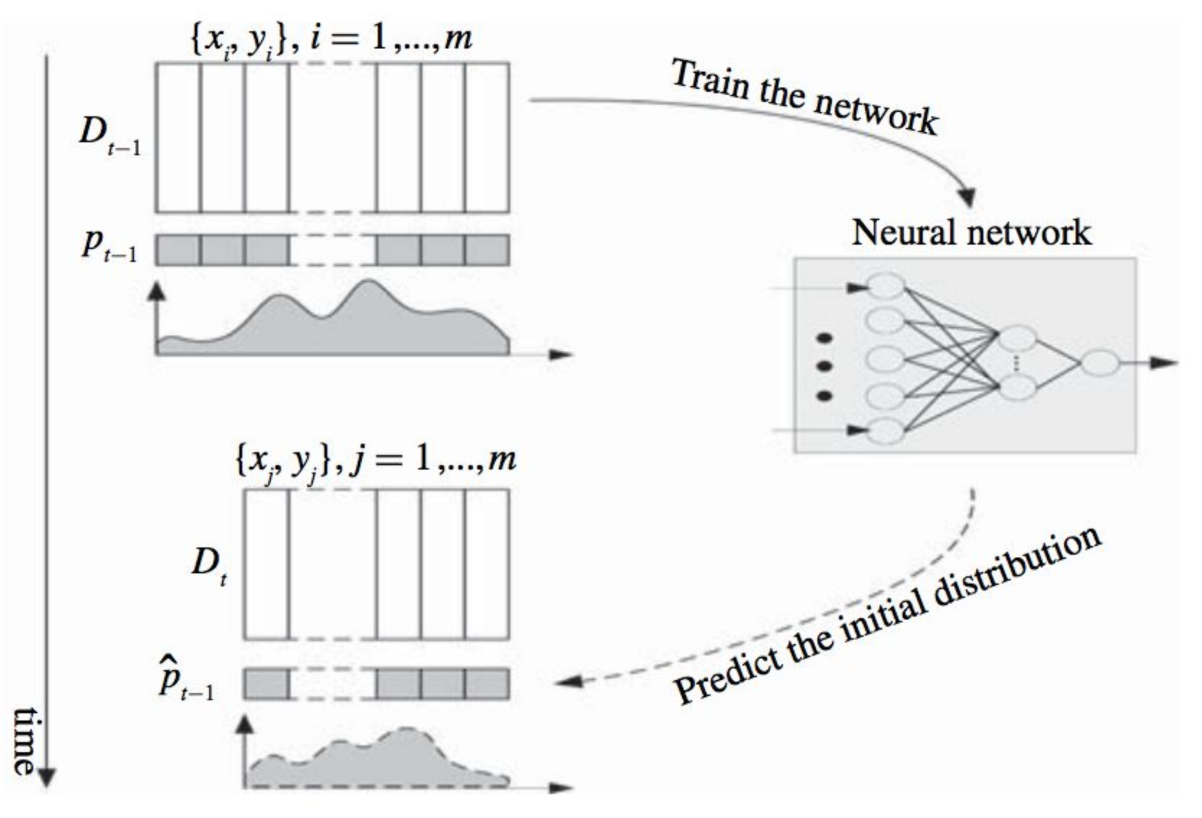
3. Update the distribution(P_t) for D_t.

$$\boldsymbol{P}_t(j) = \frac{\hat{\boldsymbol{P}}_{t-1}(j)}{Z_t} \times \begin{cases} \beta_{t-1} & \text{if } h_{t-1}(\boldsymbol{x}_j) = y_j \\ 1 & \text{otherwise} \end{cases}$$

4. Develop a hypothesis (h_t) based on D_t with P_t.
5. Repeat the procedure when the next chunk of data set (D_{t+1}) is received.

Mapping Function

- Transforms the knowledge from the current data chunk into the learning process of the future data chunks
- Provides a connection from past experience to the newly received data



Simulation Results

TABLE III

RUNNING TIME FOR ADAIN.SVR AND LEARN^{++} (*in seconds*)

Data set	ADAIN.SVR		Learn^{++}	
	training	*testing*	*training*	*testing*
Spambase	128.42	2.19	600.26	246.62
Magic	7256.18	14.36	107 256.33	1623.74
Waveform	163.51	3.39	548.84	238.1016
Sat	352.31	4.95	1246.62	346.53

TABLE II

AVERAGED PREDICTION ACCURACY

Data sets	Methods	Prediction accuracy						
		class 1	class 2	class 3	class 4	class 5	class 6	Overall
Spambase	ADAIN.MLP	0.8820	0.9352	–	–	–	–	0.9142
	ADAIN.SVR	0.8990	0.9205	–	–	–	–	0.9120
	IMORL	**0.9106**	0.8929	–	–	–	–	0.9000
	Accumulation	0.8803	0.9190	–	–	–	–	0.9038
	Learn^{++}	0.8532	**0.9561**	–	–	–	–	**0.9143**
Magic	ADAIN.MLP	0.9315	0.7137	–	–	–	–	0.8549
	ADAIN.SVR	0.9319	0.7395	–	–	–	–	**0.8644**
	IMORL	0.8404	**0.7836**	–	–	–	–	0.8205
	Accumulation	0.8670	0.7410	–	–	–	–	0.8268
	Learn^{++}	**0.9523**	0.6786	–	–	–	–	0.8547
Waveform	ADAIN.MLP	0.7843	0.8230	0.8193	–	–	–	0.8132
	ADAIN.SVR	0.7576	0.8198	0.8474	–	–	–	0.8077
	IMORL	0.7575	0.8000	0.8009	–	–	–	0.7814
	Accumulation	0.7070	0.7558	0.7534	–	–	–	0.7384
	Learn^{++}	**0.7870**	**0.8360**	**0.9072**	–	–	–	**0.8428**
Sat	ADAIN.MLP	0.9602	**0.9131**	0.9169	0.4837	0.6417	0.8494	0.8387
	ADAIN.SVR	0.9584	0.8889	0.9305	0.5180	0.7235	0.8345	0.8471
	IMORL	0.9000	0.8918	0.8566	**0.5653**	0.6841	0.7897	0.8079
	Accumulation	0.9452	0.9473	0.8697	0.5316	**0.7971**	0.8499	0.8454
	Learn^{++}	**0.9696**	0.8860	**0.9327**	0.5651	0.6958	**0.8545**	**0.8558**

TABLE VI

OVERALL PREDICTION ACCURACY AND AUC FOR ADAIN.MLP AND BOOSTONLINE

Data set	ADAIN.MLP		BoostOnline		VW	
	OA	*AUC*	*OA*	*AUC*	*OA*	*AUC*
Spambase	**0.9302**	0.8777	0.9003	0.8709	0.9212	**0.9736**
Magic	**0.8553**	**0.9100**	0.7448	0.8056	0.8223	0.8649
Waveform	**0.8530**	**0.9649**	0.7393	0.8949	–	–
Sat	**0.8484**	**0.9781**	0.7211	0.9393	–	–

How does ADAIN overcome the challenges of OIL?

- It handles concept drift by focusing on **hard / novel data**
- It compactly represents knowledge in a single **hypothesis**
- It handles the stability-plasticity dilemma by **setting incrementally developed hypotheses** and **pruning the obsolete hypothesis** to better keep tuned on the evolving data stream
- It handles adaptive model complexity by the **mapping function**

Learn++

- Use weighted majority voting between these boosted classifiers
- Add new classifiers until performance can no longer be improved

ADAIN

- Obtain hypothesis and distribution function from the current chunk of data and improves upon it
- Use mapping function to obtain weights of instances in a new chunks of the data


QUESTIONS?

References

1. Alexander Gepperth, Barbara Hammer. Incremental learning algorithms and applications. European Symposium on Artificial Neural Networks (ESANN), 2016, Bruges, Belgium.
2. R. Polikar, L. Upda, S. S. Upda and V. Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 31, no. 4, pp. 497-508, Nov. 2001.
3. H. He, S. Chen, K. Li and X. Xu, "Incremental Learning From Stream Data," in IEEE Transactions on Neural Networks, vol. 22, no. 12, pp. 1901-1914, Dec. 2011.