# NSL: PASSIVE APPROACHES

Giacomo Boracchi[1] and Gregory Ditzler[2]

[1] Politecnico di Milano
Dipartimento Elettronica e Informazione
Milano, Italy

[2] The University of Arizona
Department of Electrical & Computer Engineering
Tucson, AZ USA

giacomo.boracchi@polimi.it, ditzler@email.arizona.edu

# NSL: A PASSIVE VIEW

# (BACK TO)ADAPTATION STRATEGIES

- Active: the learner $K$ is combined with a tool to detect concept drift that pilots adaptation.
- Passive: assume the process undergoes continuous adaptation.

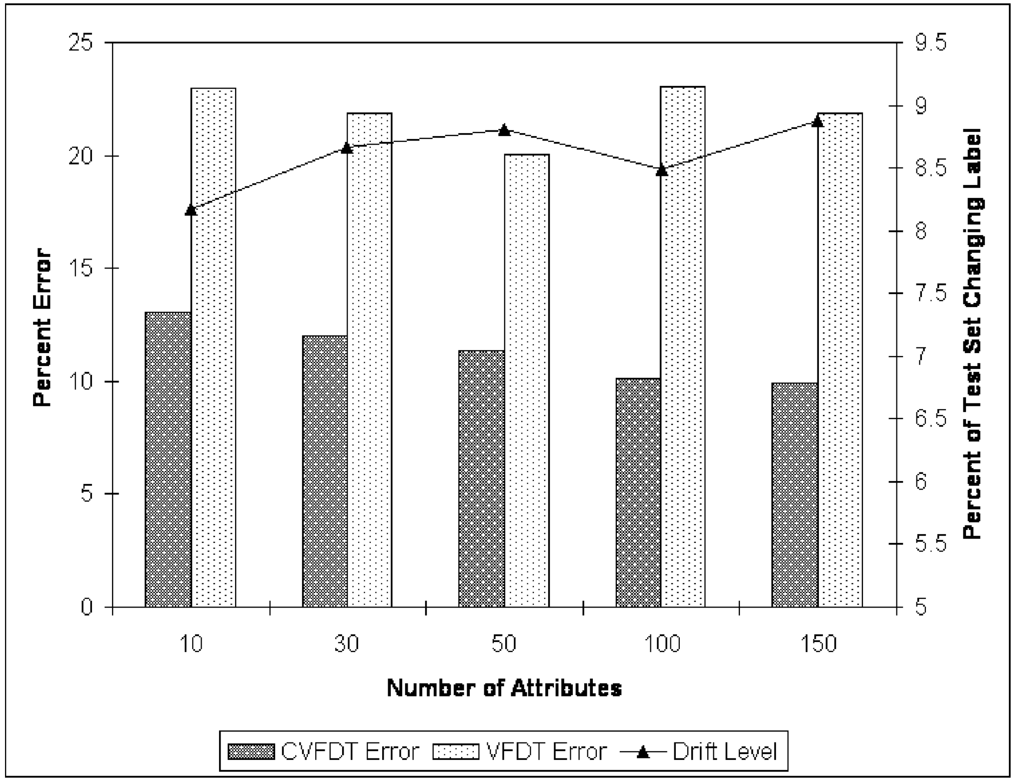
# WHAT IS PASSIVE LEARNING?

- It's all about the adaptation mechanism employed to cope with the change!
  - **active approaches** rely on an explicit detection of the change in the data distribution to activate an adaptation mechanism
  - **passive approaches** continuously update the model over time (without requiring an explicit detection of the change)
    - Passively assume that some type of change is present in the data stream
- Is one approach more correct than another? No!
  - Benchmarking Dilemma – What makes an algorithm successful? Detection delay? Classification error? Can we make the comparison fair?
- Dicotomy of Passive Learning
  - Single Classifier
  - Ensemble
  - Batch versus Online

# SINGLE CLASSIFIER MODELS

# CONCEPT-ADAPTING VERY FAST DECISION TREE

- Decision tree models are very popular with traditional supervised learning, but how can we use them with concept drift in the data stream?
- **Concept-adapting Very Fast Decision Tree** learner (CVFDT) is an online decision tree algorithm
- CVFDT attempts to stay current by growing alternative sub-trees until an old sub-tree is accurate then it is replaced
  - The current split at a node in the tree may not be optimal throughout all of time
  - Use windows of samples to evaluate the tree to remove the low quality sub-trees
  - Time complexity of O(1)



# STOCHASTIC GRADIENT DESCENT & ELM

- **Stochastic gradient descent (SGD)** is an online representation of classical "batch" gradient descent algorithm
  - Commonly used to train neural networks
  - Mini-batches are sometimes used at each timestep to achieve a smoother convergence in the function being minimized
- SGD has been implemented using a linear classifier minimizing a hinge loss function for learning in nonstationary environment
  - Massive Online Analysis (more about this later) has an implementation of this approach in their software package
- An **Extreme Learning Machine** has been also combined with a time-varying NN for learning in nonstationary environments

# ONLINE INFORMATION NETWORK

- OLIN (**online information network**) is fuzzy-logic algorithm that repeatedly learns a from sliding window of examples in order to update the existing model
  - Replace it by a former model
  - Or construct a new model if a major concept drift is detected
  - Borrow from active and passive approaches
- OLIN updates the fuzzy-info network by identifying non-relevant nodes, adding new layers and replacing the output layer when new data arrive. If a new concept was presented then a new fuzzy-info network is learned.
  - A new concept could be identified by a statistically significant drop is the classification accuracy on the latest labeled data

# A DILEMMA OF SORTS

**Stability**

- The ability of an algorithm to recall old information that it has learned in the past

**Plasticity**

- The ability for an algorithm to learn new information when data are available

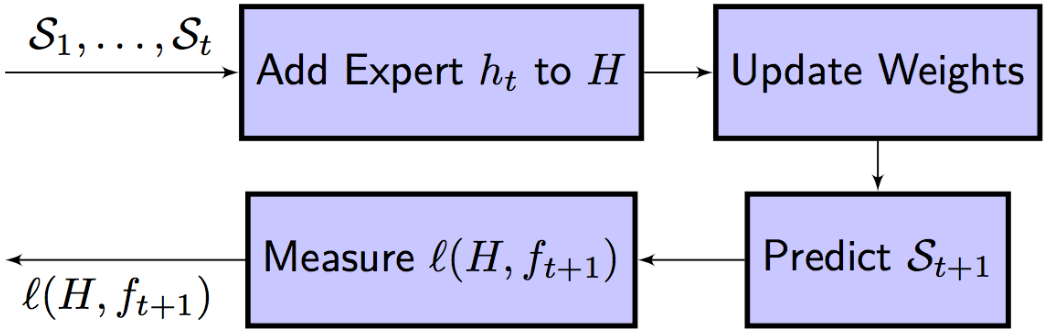
**Sounds like we could have two opposing ideas!**

# ENSEMBLE MODELS

Balancing the stability-plasticity dilemma

# ENSEMBLE MODELS

- Ensemble based approaches provide a natural fit to the problem of learning in a nonstationary setting
  - Ensembles tend to be more accurate than single classifier-based systems due to reduction in the variance of the error
  - (Stability) They have the flexibility to easily incorporate new data into a classification model when new data are presented, simply by adding new members to the ensemble
  - (Plasticity) They provide a natural mechanism to forget irrelevant knowledge, simply by removing the corresponding old classifier(s) from the ensemble



# ENSEMBLE MODELS

An **ensemble** of **multiple models** is preserved in memory

$$\mathcal{H} = \{h_0, \dots, h_N\}$$

Each **individual** $h_i, i = 1, \dots, N$ is typically trained from a different training set and could be from a different model

**Final prediction** of the ensemble is given by (weighted) **aggregation of the individual predictions**

$$H(\boldsymbol{x_t}) = \underset{\boldsymbol{\omega \in \Lambda}}{\operatorname{argmax}} \sum_{\boldsymbol{h_i \in \mathcal{H}}} \alpha_i \left[h_i(\boldsymbol{x}_t) = \omega\right]$$

Typically, one assumes data arrives in **batches** and each classifier is trained over a batch

# ENSEMBLE MODELS

- **Each individual** implicitly refers to a component of a mixture distribution characterizing a **concept**
- In practice, often ensemble methods assume data (supervised and unsupervised) are provided in batches
- **Adaptation** can be achieved by:
  - **updating each individual**: either in batch or online manner
  - **dynamic aggregation**: adaptively defining weights $\omega_i$
  - **structural update**: including/removing new individuals in the ensemble, possibly recovering past ones that are useful in case of recurrent concepts

# TAXONOMY OF CD ADAPTATION IN ENSEMBLE

Ensemble based approaches provide a **natural fit** to the problem **of learning in nonstationary settings**,

- Ensembles tend to be more accurate than single classifier-based systems due to **reduction in the variance of the error**
- **Stability:** flexible to easily incorporate new data into a classification model, simply by **adding new individuals** to the ensemble (or updating individuals)
- **Plasticity:** provide a natural mechanism **to forget irrelevant knowledge**, simply by **removing** the corresponding old **individual(s)** from the ensemble
- They can operate in continuously drifting environments

- Apadtive strategies can be applied to add/remove classifiers by on individual classifier and the ensemble error

# SEA

A **fixed-size ensemble** that performs

- batch learning
- structural update to adapt to concept drift

When a new batch $S = \{(\boldsymbol{x}_0^t, y_0^t), (\boldsymbol{x}_1^t, y_1^t), \dots, (\boldsymbol{x}_B^t, y_B^t)\}$ arrives

- train $h_t$ on $S$
- test $h_{t-1}$ on $S$
- If the ensemble is not full ($\#\mathcal{H} < N$), **add** $h_{t-1}$ to $\mathcal{H}$
- Otherwise, **remove** $h_i \in \mathcal{H}$ that is **less accurate** on $S$ (as far as this is worst than $h_{t-1}$)

**Prune** the ensmeble to improve the performance

# DWM

- Littlestone's **Weighted Majority** algorithm is an ensemble **online algorithm** for combining multiple classifiers learning from a stream of data; however, keeping the same classifiers in the ensemble for the duration of the stream could be suboptimal in a nonstationary setting

**Dynamic weighted majority** (DWM) algorithm is an ensemble method where:

  - Individuals classifiers are trained on incoming data
  - Each **individual is associated to a weight**
  - **Weights are decreased** to individuals that are **not accurate** on the samples of the current batch
  - Individuals having **low weights are dropped**
  - Individuals are **created** at each error of the ensemble
  - Predictions are made by **weighted majority voting**

- Ensemble size is not fixed

J. Zico Kolter, M.A. Maloof, "Dynamic Weighted Majority: An Ensemble Method for Drifting Concepts," *Journal of Machine Learning Research 8*, 2007.

# DIVERSITY, DIVERSITY, DIVERSITY!

- **Diversity** in an ensemble has been shown to be **benefical in ensembles** that have been learned from a static distribution, but how can diversity be used in a nonstationary stream
- Diversity for Dealing with Drifts (DDD) combines **two ensembles:**
  - An **High diversity** ensemble
  - A **Low diversity** ensemble

  **and a concept-drift detection** method.
- Online bagging is used to control ensemble diversity
- In **stationary conditions**, predictions are made by **low-diversity ensemble**
- **After concept drift**, the ensembles are updated and predictions are made by the **high-diversity ensemble**.

MINKU, L. L.; YAO, X. . "DDD: A New Ensemble Approach For Dealing With Concept Drift.", *IEEE Transactions on Knowledge and Data Engineering,* IEEE, v. 24, n. 4, p. 619-633, April 2012

Add Oza reference

# ONLINE BAGGING & BOOSTING

- **Central Concept**
  - Bagging and boosting works well for batches of data; however, these approaches are not equipped to handle data streams. Oza developed approaches to perform this batch to online version
  - Issue: Online bagging and boosting does not account for concept drift
  - Parallel to Active: Implement Change detection to reset classifiers

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in \mathscr{X}$, $y_i \in \{-1, +1\}$.
Initialize: $D_1(i) = 1/m$ for $i = 1, \ldots, m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution $D_t$.
- Get weak hypothesis $h_t : \mathscr{X} \to \{-1, +1\}$.
- Aim: select $h_t$ with low weighted error:

$$\varepsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$.
- Update, for $i = 1, \ldots, m$:

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where $Z_t$ is a normalization factor (chosen so that $D_{t+1}$ will be a distribution).

Output the final hypothesis:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

**OnlineBoosting**($\mathbf{h_M}$, $OnlineBase$, $d$)

- Set the example's "weight" $\lambda_d = 1$.
- For each base model $h_m$, ($m \in \{1, 2, \ldots, M\}$) in the ensemble,
  - 1. Set $k$ according to $Poisson(\lambda_d)$.
  - 2. Do $k$ times
    $h_m = OnlineBase(h_m, d)$
  - 3. If $h_m(d)$ is the correct label,
    - then
      - $\lambda_m^{sc} \longleftarrow \lambda_m^{sc} + \lambda_d$
      - $\lambda_d \longleftarrow \lambda_d \left(\frac{N}{2\lambda_m^{sc}}\right)$
    - else
      - $\lambda_m^{sw} \longleftarrow \lambda_m^{sw} + \lambda_d$
      - $\lambda_d \longleftarrow \lambda_d \left(\frac{N}{2\lambda_m^{sw}}\right)$

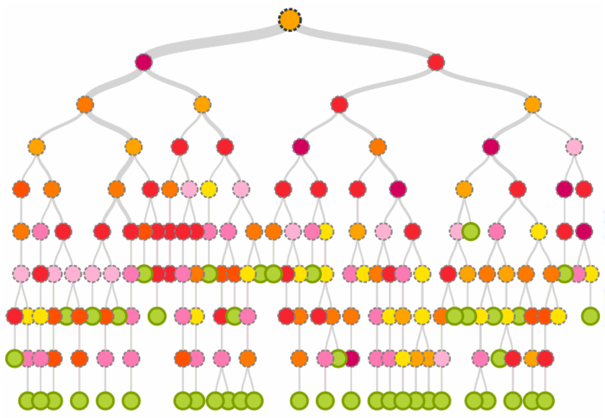
To classify new examples:

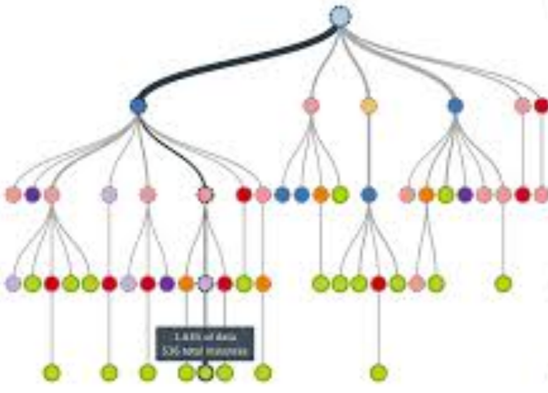
- For each $m \in \{1, 2, \ldots, M\}$
  Calculate $\epsilon_m = \frac{\lambda_m^{sw}}{\lambda_m^{sc} + \lambda_m^{sw}}$ and $\beta_m = \frac{\epsilon_m}{1-\epsilon_m}$
- Return $h(x) = argmax_{c \in C} \sum_{m: h_m(x) = y} log \frac{1}{\beta_m}$.

Add bifet reference

# ADAPTIVE SIZE HOEFFDING TREES (ASHT)

- Online bagging & boost are not designed for learning in nonstationary environments. How can we modify the base classifier to learn concepts over time?
- Thoughts on pruning trees
  - In a slowly changing environment, larger trees can better capture the concepts
  - In a quickly changing environment, smaller trees can adapt quickly to new concepts with new examples from the stream
  - Solution: Ensembles with different size Hoeffding trees
- Examples from the stream are learned by different size Hoeffding trees. These trees are "reset" at different time intervals
  - Short trees are reset more often than larger trees
  - Opportunity: Active approach to tell us when to reset a tree (small and large)





# ACCURACY UPDATED ENSEMBLE

- **Motivation**
  - Build a data stream mining algorithm that is not specific to one type of drift, but rather robust on
  - The "size" of a classifier is important for many application, not only for memory limitations, but also for learning in a nonstationary environment
- The Accuracy Updated Ensemble (AUE2) learns a classifiers on the most recent data and weights them according to their accuracy; however, the base classifier is limited in its memory footprint to avoid unnecessarily large models
  - Weaker classifiers are discarded with the most recent (accurate) classifier
  - Potentially incorporates catastrophic forgetting

D. Brzezinski and J. Stefanowski, "Reacting to Different Types of Concept Drift: The Accuracy Updated Ensemble Algorithm," IEEE Transactions on Neural Networks, vol. 25, no. 1, 2014

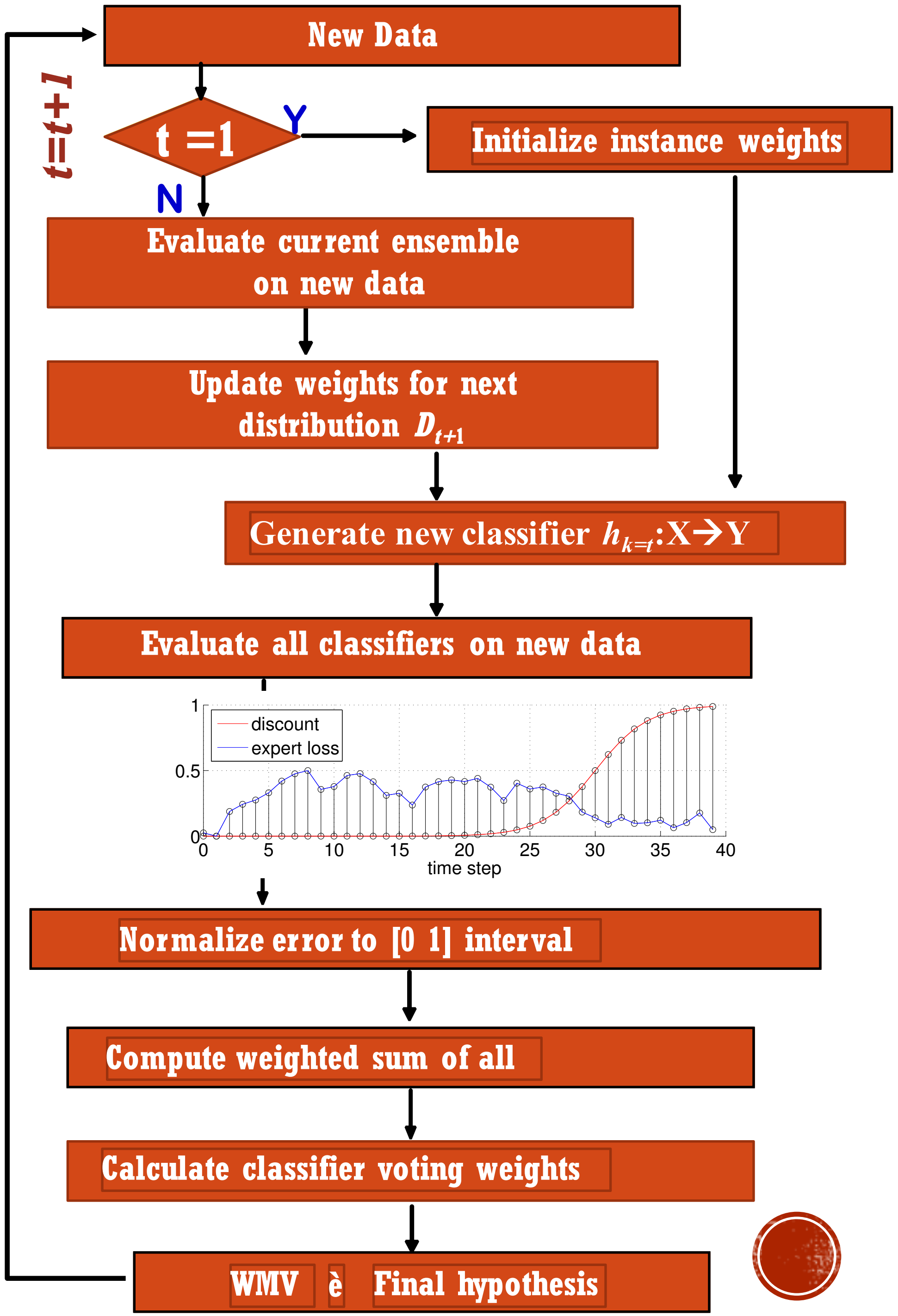
# LEARN++.NSE

- **Learn++** is an incremental learning ensemble to learn classifiers from streaming batches of data
  - Think boosting, but for incremental learning
  - Batches of data are assumed to be sampled iid from a distribution
- Learn++ works well on static distributions; however, **classifier weights remain fixed**, which is an ill-advised strategy if each batch of data is not sampled iid. Especially the testing data!
- **Solution**: Learn++.NSE: Similar to DWM, Learn++.NSE extends Learn++ for learning in nonstationary environments (NSE)

Polikar R., Udpa L., Udpa, S., Honavar, V., "Learn++: An incremental learning algorithm for supervised neural networks," IEEE Transactions on System, Man and Cybernetics (C), vol. 31, no. 4, pp. 497-508, 2001

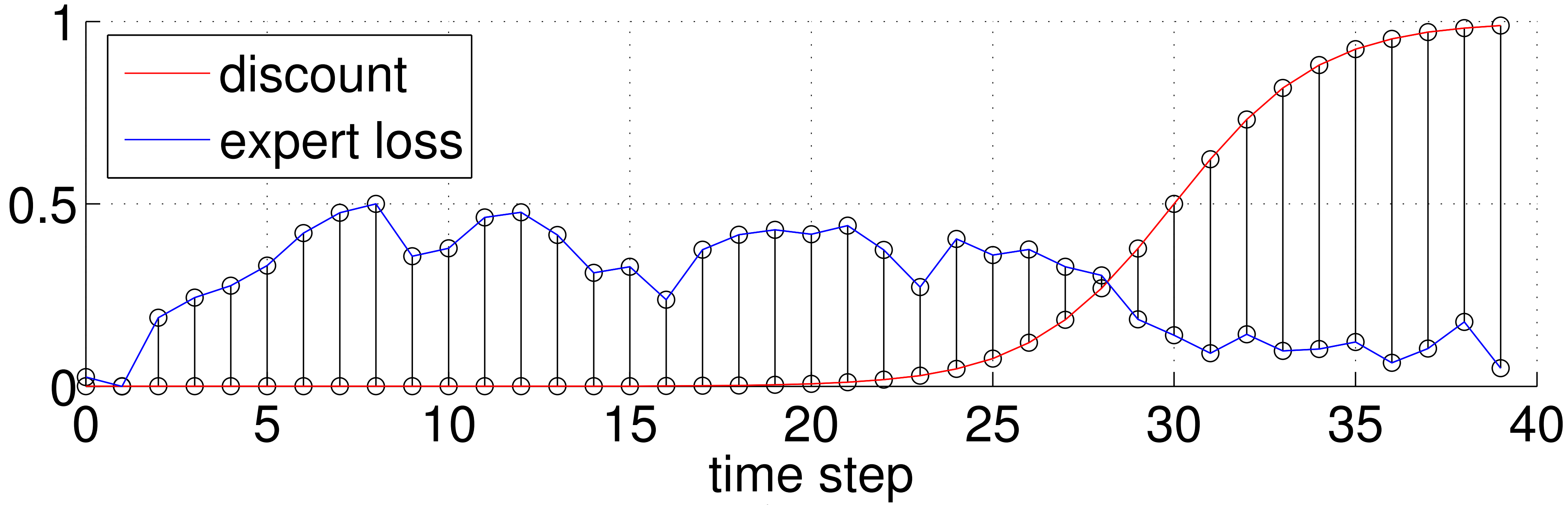
# LEARN++.NSE

- **Learn$^{++}$.NSE**: incremental learning algorithm for concept drift
  - Generate a classifier with each new batch of data, compute a pseudo error for each classifier, apply time-adjusted weighting mechanism, and call a weighted majority vote for an ensemble decision
    - Recent pseudo errors are weighted heavier than old errors
  - Works very well on a broad range of concept drift problems



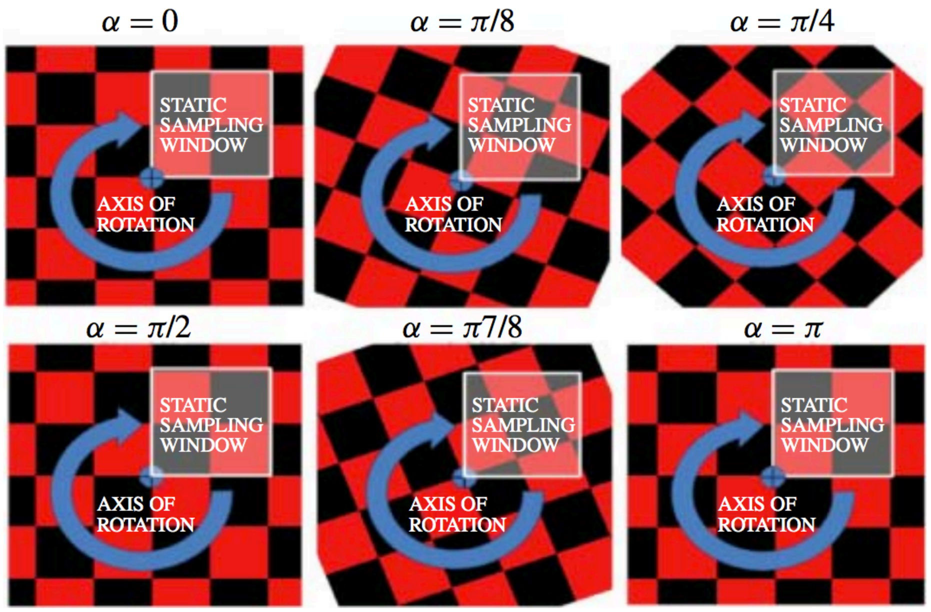

Elwell R. and Polikar R., "Incremental Learning of Concept Drift in Nonstationary Environments" IEEE Transactions on Neural Networks, vol. 22, no. 10, pp. 1517-1531, 2011.

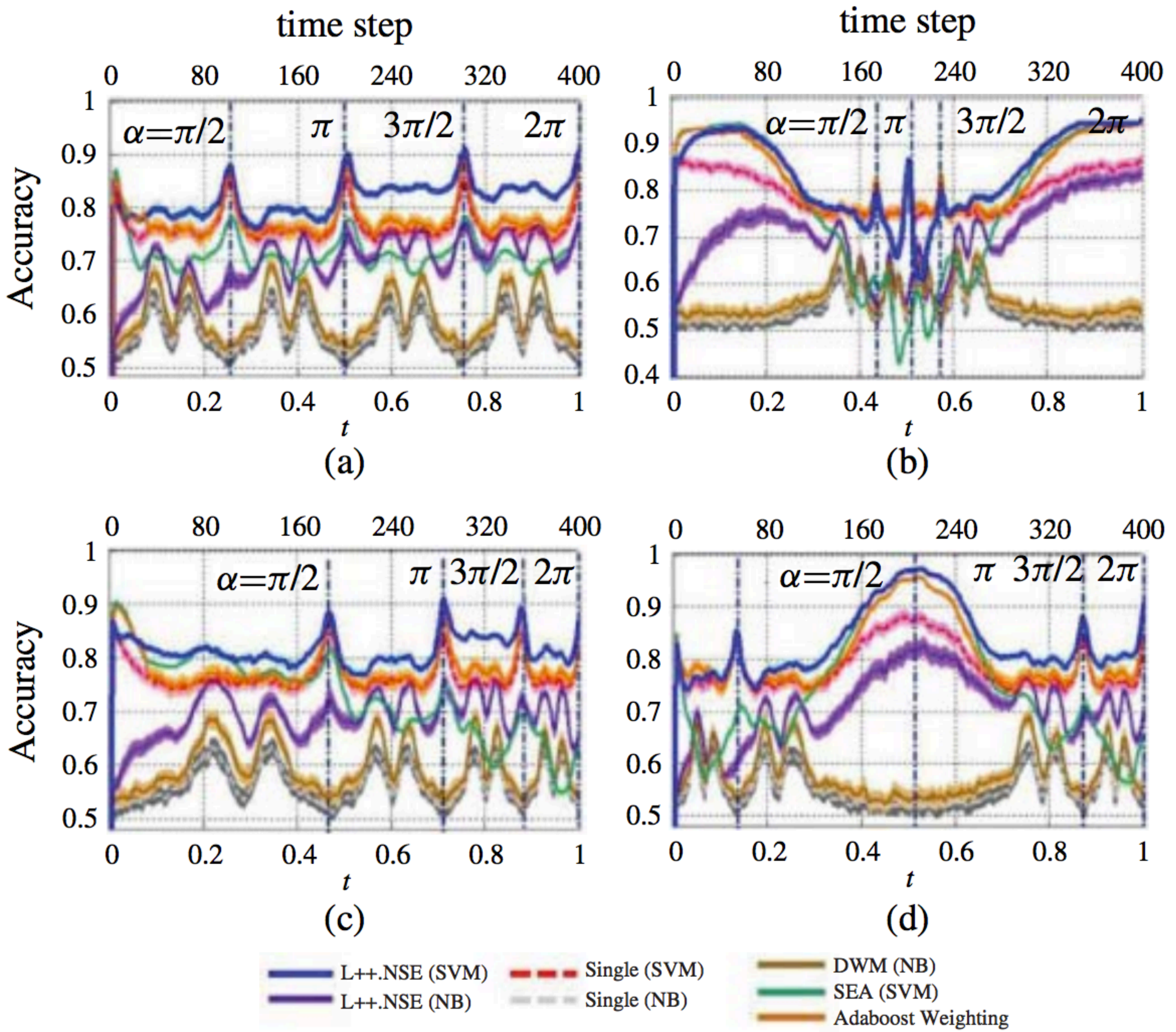
# DISCOUNTED LOSS (ERROR)




$$\hat{\ell}_t(h_j) = \sum_{k=1}^{t} \rho_{t-k} \ell_k(h_j)$$

# LEARN++.NSE IN ACTION

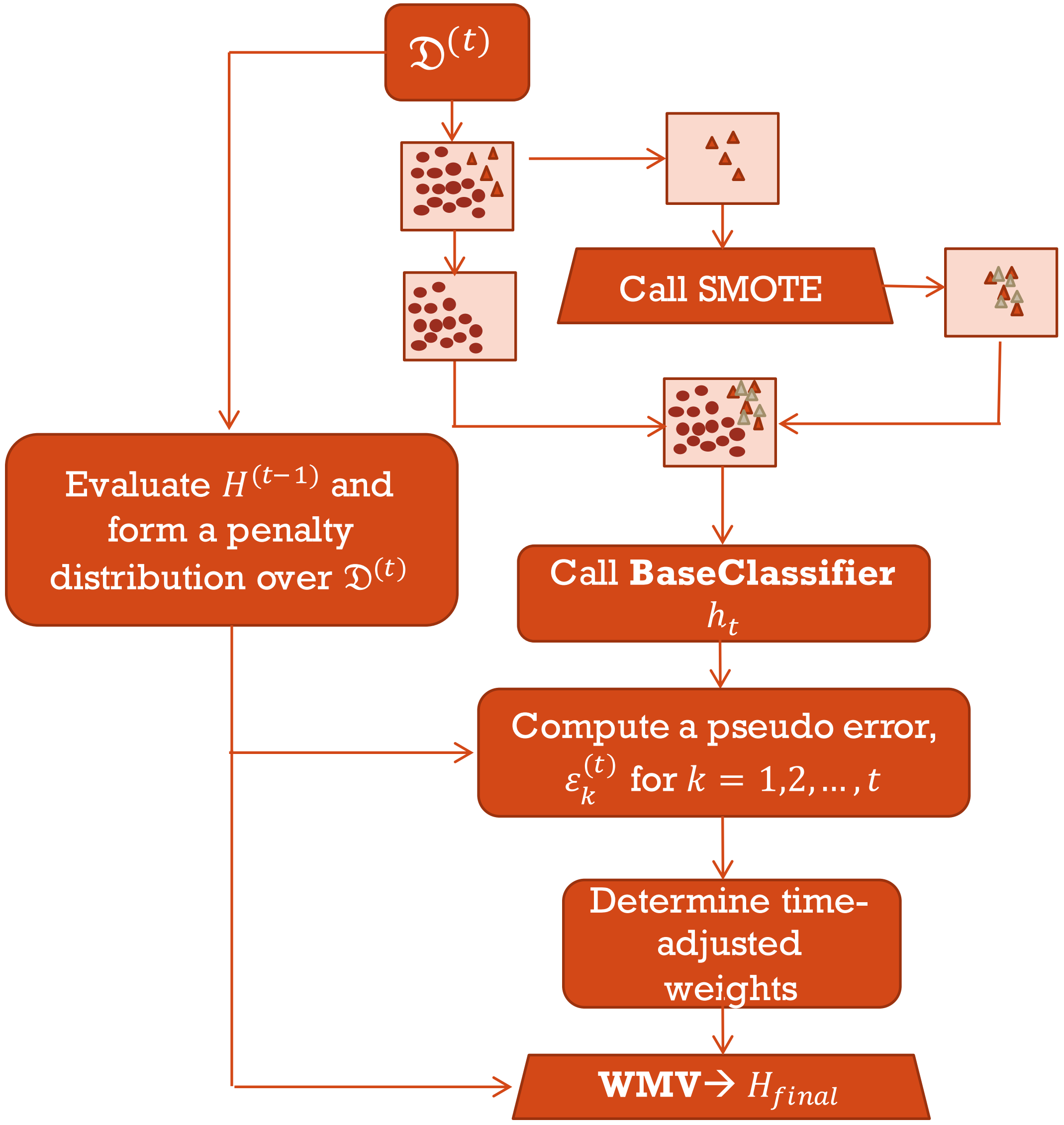


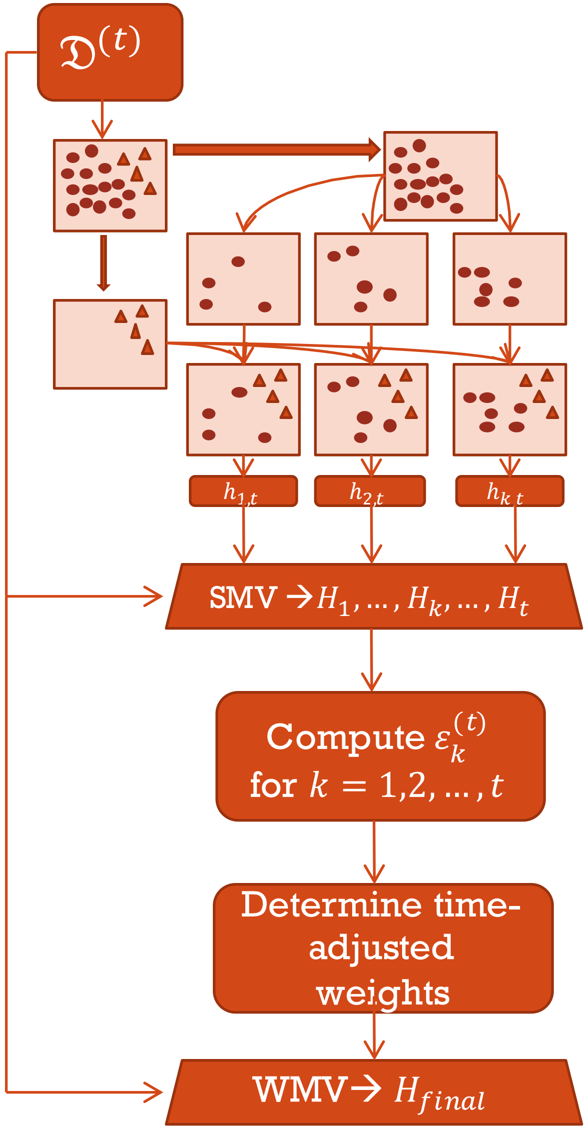
# IMBALANCED & NSE

- **Class Imbalance**
  - Learners tend to bias themselves towards the majority class
    - Minority class is typically of great importance
  - Many concept drift algorithms tend to use **error or a figure of merit derived from error** to adapt to a nonstationary environment
- **Learn$^{++}$.CDS** {*Concept Drift with SMOTE*}
  - Apply SMOTE to Learn$^{++}$.NSE
    - Learn$^{++}$.NSE works well on problems involving concept drift
    - SMOTE works well at increasing the recall of a minority class
- **Learn$^{++}$.NIE** {*Nonstationary and Imbalanced Environments*}
  - Classifiers are replaced with sub-ensembles
    - Sub-ensemble is applied to learn a minority class
  - Voting weights are assigned based on figures of merit besides a class independent error
- Other Approaches: SERA, UCB, muSERA, REA

## Learn++.NIE




## Learn++.CDS



# A GLIMPSE AT ENSEMBLE ERROR

- Assuming the loss is convex in its first argument, the ensemble's expected loss becomes

$$\mathbb{E}_T[\ell(H, f_T)] \leq \sum_{k=1}^{t} w_{k,t}\mathbb{E}_T[\ell(h_k, f_T)]$$

  which is a weighted average each expert's loss
  - Generally not computable and is difficult to interpret
- Decomposing the loss using existing domain adaptation theory with prior work leads to the bound

$$\mathbb{E}_T[\ell(H, f_T)] \leq \sum_{k=1}^{t} w_{k,t}\mathbb{E}_T[\ell(h_k, f_T)]$$

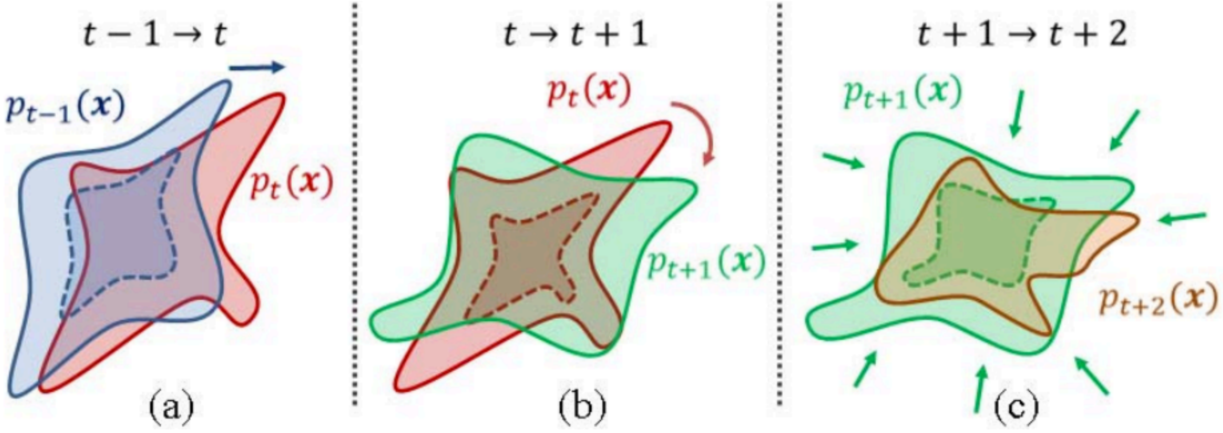
$$\leq \sum_{k=1}^{t} w_{k,t}\left(\mathbb{E}_k[\ell(h_k, f_k)] + \mathbb{E}_T[\ell(f_k, f_T)] + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}\left(\mathcal{U}_k, \mathcal{U}_T\right)\right)$$

- Looks bad! But interpretable!
  Loss(ensemble) < weightedSum(training loss + labeling disagreement + divergence of unlabeled data)
  - Real & virtual drifts have been defined in literature, but now related to loss!

# INITIALLY LABELED ENVIRONMENTS

# INITIALLY LABELED ENVIRONMENTS

- All previous learning scenarios assumed a supervised learning setting
  - Transductive and Semi-Supervised was discussed
  - What if we're only provide some labeled data at $T_0$ and all future time points are unlabeled?
  - Active Learning versus Learning in Initially Labeled Environments
    - AL: Assume that we have access to an oracle that can label the unlabeled data at a cost
    - ILNSE: Extreme latency verification! No labeled data are received after $T_0$




Progression of a single class experiencing (a) translational, (b) rotational, and (c) volumetric drift.

# COMPACTED OBJECT SAMPLE EXTRACTION

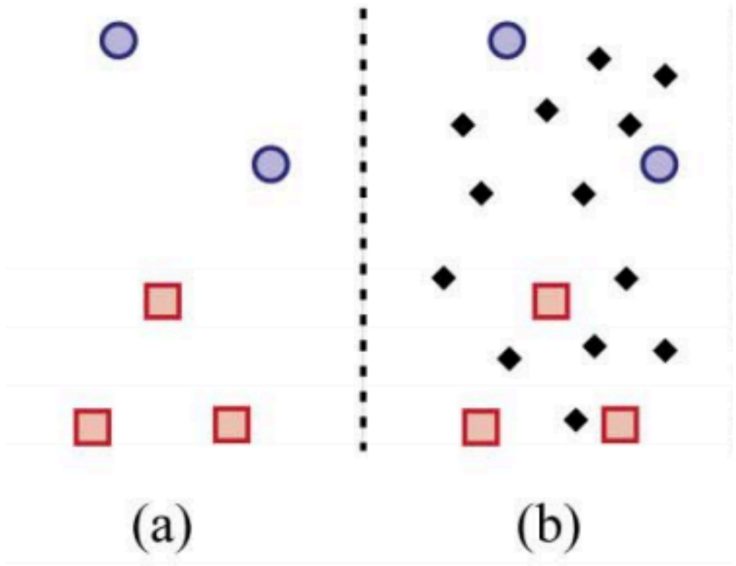

(a)

Initially Labeled Data

Compose

Dyer K., Capo R., Polikar R., "COMPOSE: A Semi-Supervised Learning Framework for Initially Labeled Non-Stationary Streaming Data" IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014

# COMPACTED OBJECT SAMPLE EXTRACTION



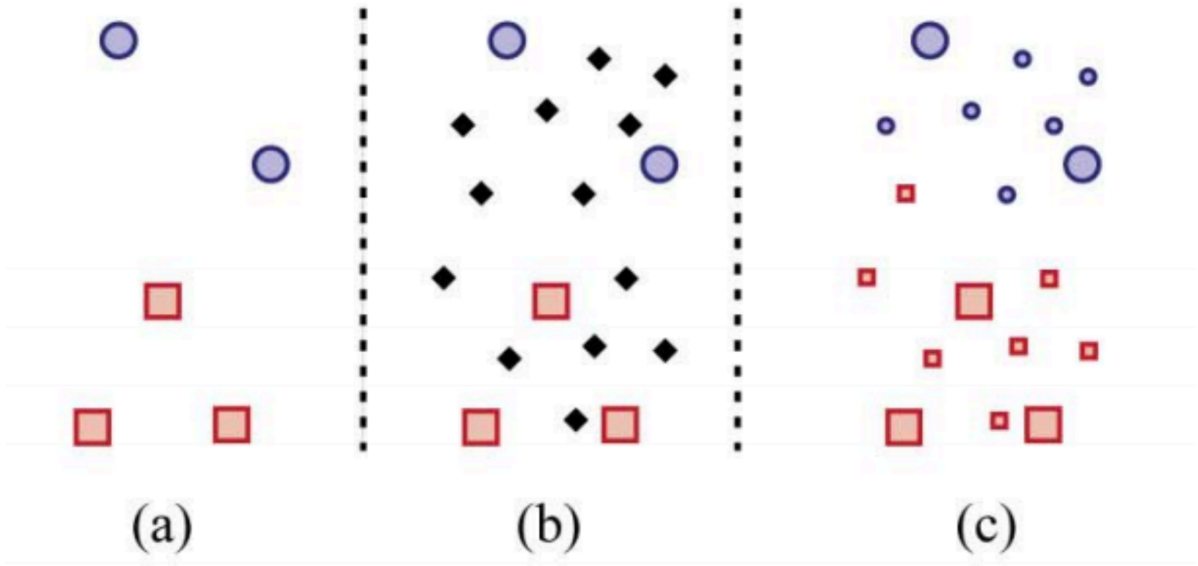

Initially Labeled Data

Receive Unlabeled Data

**Compose**

Dyer K., Capo R., Polikar R., "COMPOSE: A Semi-Supervised Learning Framework for Initially Labeled Non-Stationary Streaming Data" IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014

# COMPACTED OBJECT SAMPLE EXTRACTION




Initially Labeled Data

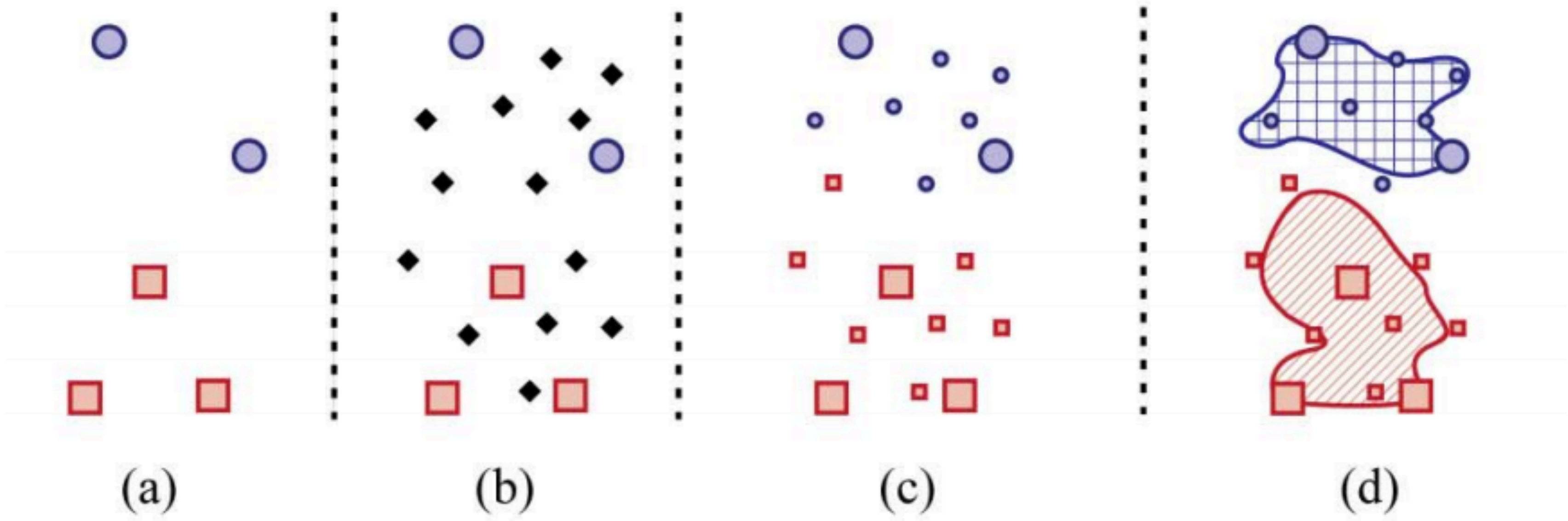
Receive Unlabeled Data

Classify Using SSL

**Compose**

Dyer K., Capo R., Polikar R., "COMPOSE: A Semi-Supervised Learning Framework for Initially Labeled Non-Stationary Streaming Data" IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014

# COMPACTED OBJECT SAMPLE EXTRACTION

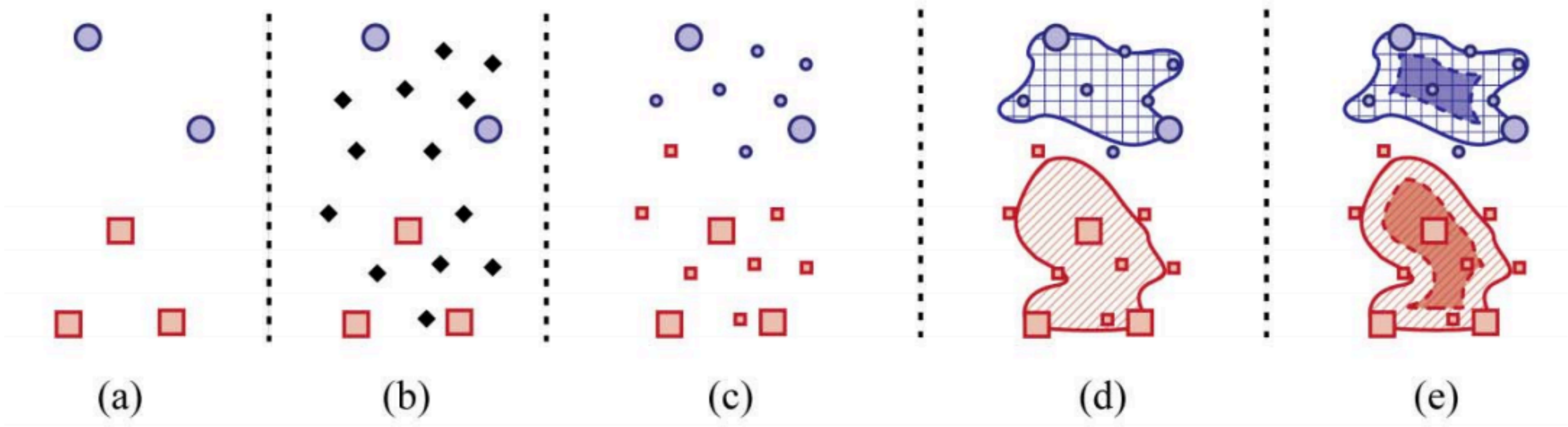



Initially Labeled Data | Receive Unlabeled Data | Classify Using SSL | Construct a Boundary

**Compose**

Dyer K., Capo R., Polikar R., "COMPOSE: A Semi-Supervised Learning Framework for Initially Labeled Non-Stationary Streaming Data" IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014

# COMPACTED OBJECT SAMPLE EXTRACTION




Initially Labeled Data

Receive Unlabeled Data

Classify Using SSL

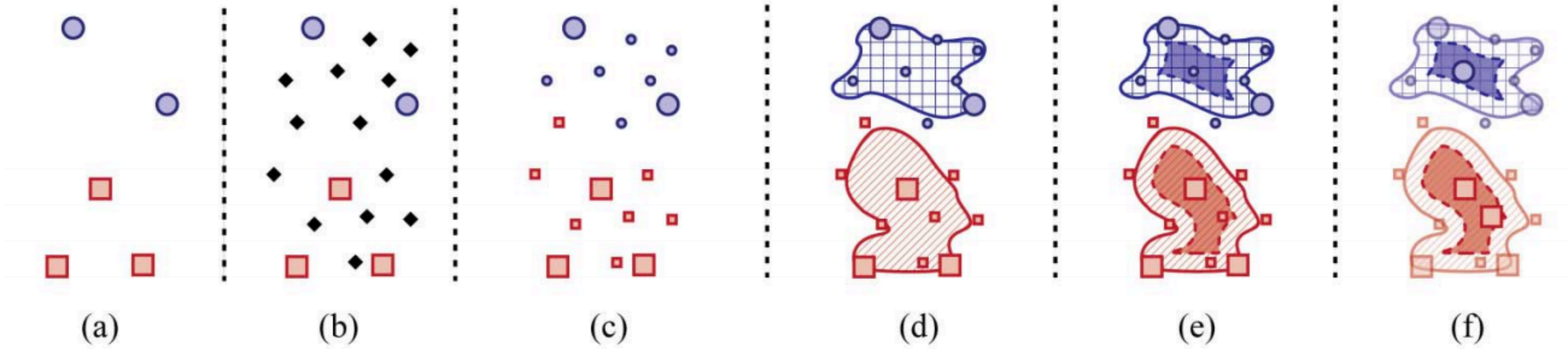
Construct a Boundary

Compact the Boundary

**Compose**

Dyer K., Capo R., Polikar R., "COMPOSE: A Semi-Supervised Learning Framework for Initially Labeled Non-Stationary Streaming Data" IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014

# COMPACTED OBJECT SAMPLE EXTRACTION

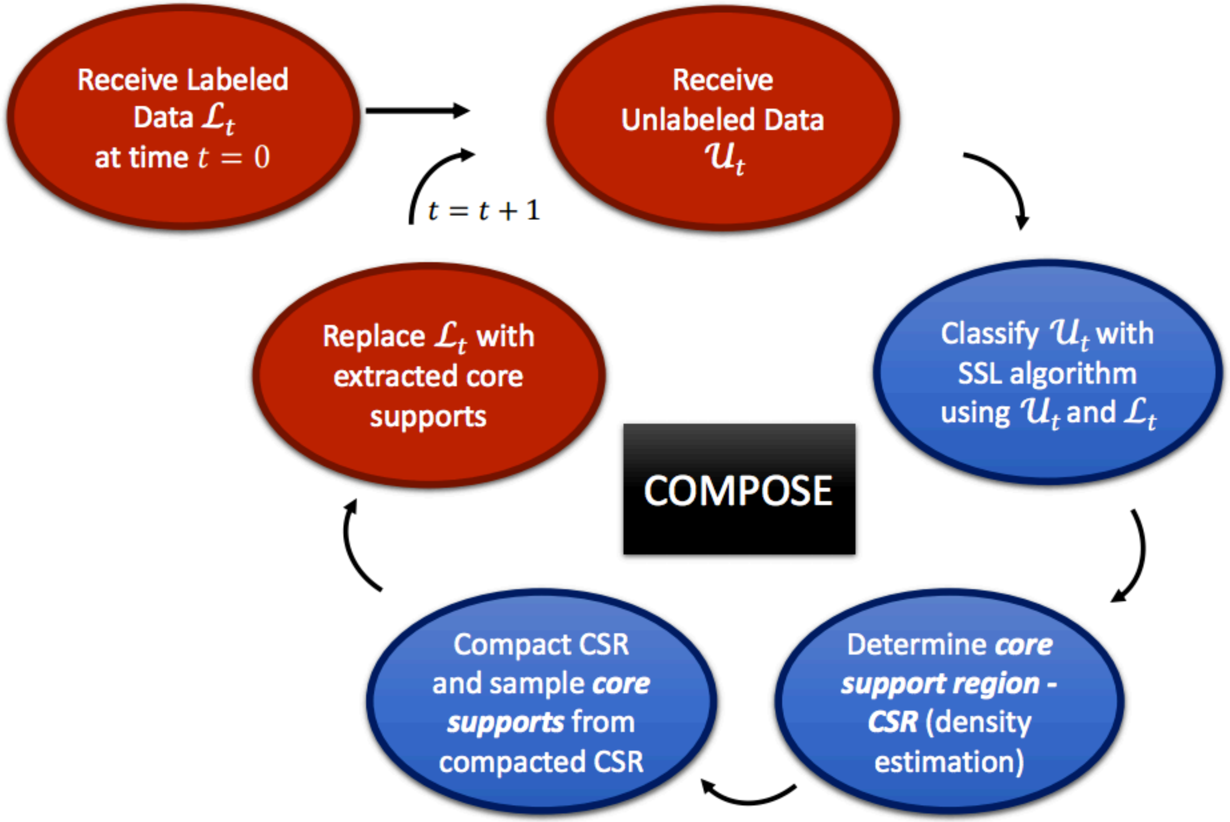



Initially Labeled Data | Receive Unlabeled Data | Classify Using SSL | Construct a Boundary | Compact the Boundary | Extract Core Set

**Compose**

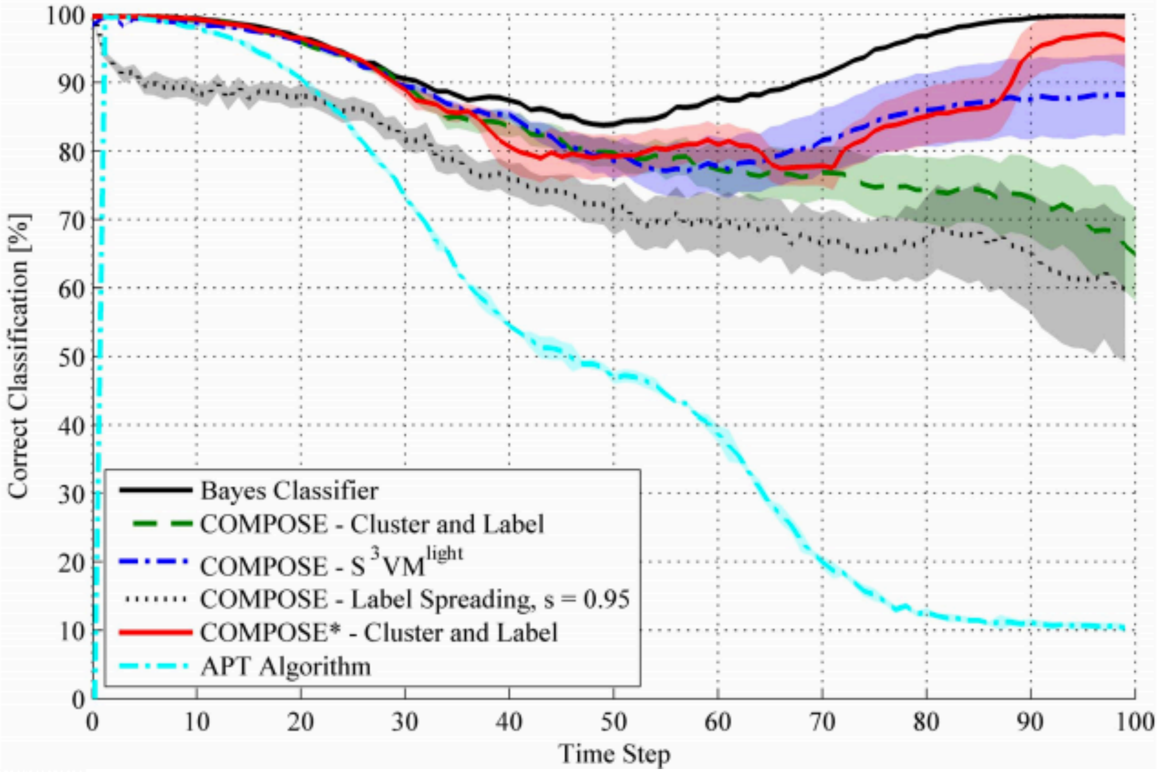
Dyer K., Capo R., Polikar R., "COMPOSE: A Semi-Supervised Learning Framework for Initially Labeled Non-Stationary Streaming Data" IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014

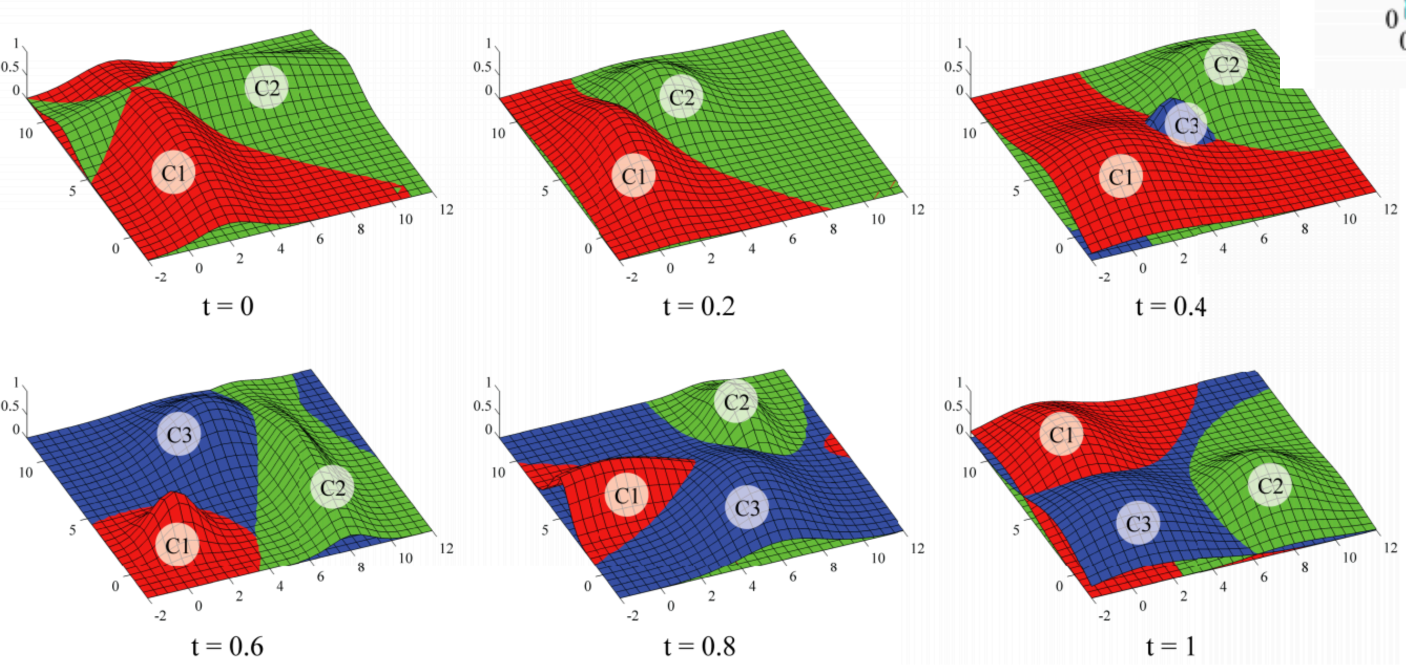
# COMPOSE




Sarnelle J., Sanchez A., Capo R., Haas J. and Polikar R., "Quantifying the Limited and Gradual Concept Drift Assumption," Int. Joint Conf. on Neural Networks (IJCNN), 2015.

# COMPOSE IN ACTION




Dyer K., Capo R., Polikar R., "COMPOSE: A Semi-Supervised Learning Framework for Initially Labeled Non-Stationary Streaming Data" IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014 .

# COMMENTS FROM MY EXPERIENCE

- Ensemble classifier approaches have had more success that single classifier implementations for nonstationary environments*
- Hybrid approaches (active & passive) can be beneficial! There is no single best strategy
  - Sometimes we lump these approach in the the active category
- In practice, a weighted majority vote is a better strategy as long as we have a reliable estimate of a classifiers error

* That is not to say there is not single classifier solutions that do not work well.

# NSL: PASSIVE APPROACHES

Giacomo Boracchi[1] and Gregory Ditzler[2]

[1] Politecnico di Milano
Dipartimento Elettronica e Informazione
Milano, Italy

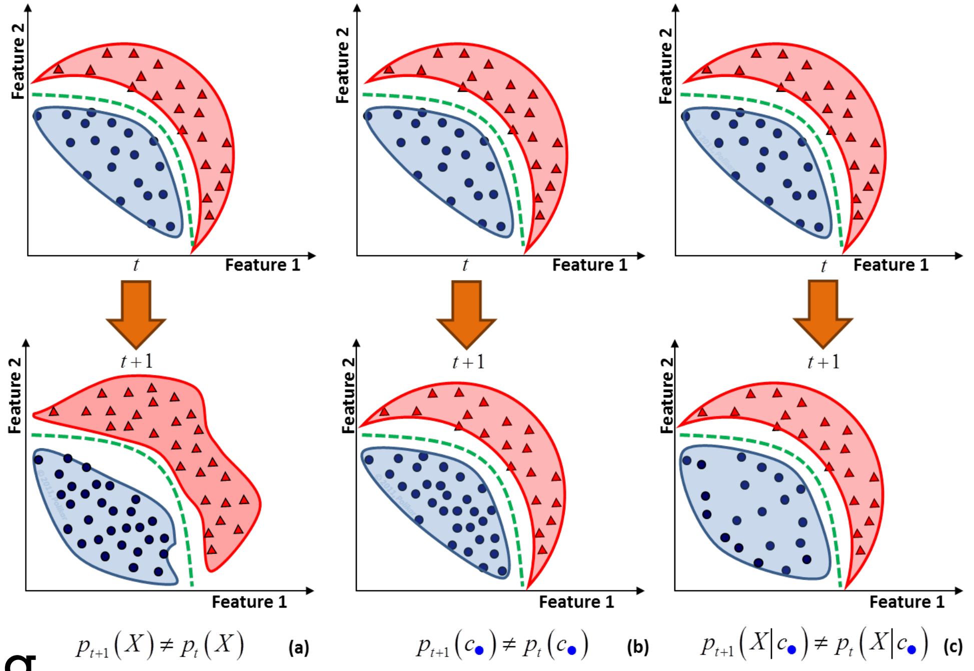
[2] The University of Arizona
Department of Electrical & Computer Engineering
Tucson, AZ USA

giacomo.boracchi@polimi.it, ditzler@email.arizona.edu

# DATA SETS & GENERATORS

# BUILDING A NONSTATIONARY DATA STREAM

- **Recall**: A **concept drift** occurs at time $t$ if
$$\phi_t(\boldsymbol{x}, y) \neq \phi_{t+1}(\boldsymbol{x}, y)$$
(we also say $\mathcal{X}$ becomes **nonstationary**)
- Drift might affect $\phi_t(y|\boldsymbol{x})$ and/or $\phi_t(\boldsymbol{x})$
  - Real and virtual drifts
  - Abrupt, Gradual, Fast
- Synthetic data streams can be generated by sampling data from a distribution that simulates the changes in the probabilities
  - E.g., Data could be sampled from a Gaussian distribution with changing parameters or classes abruptly changed/swap

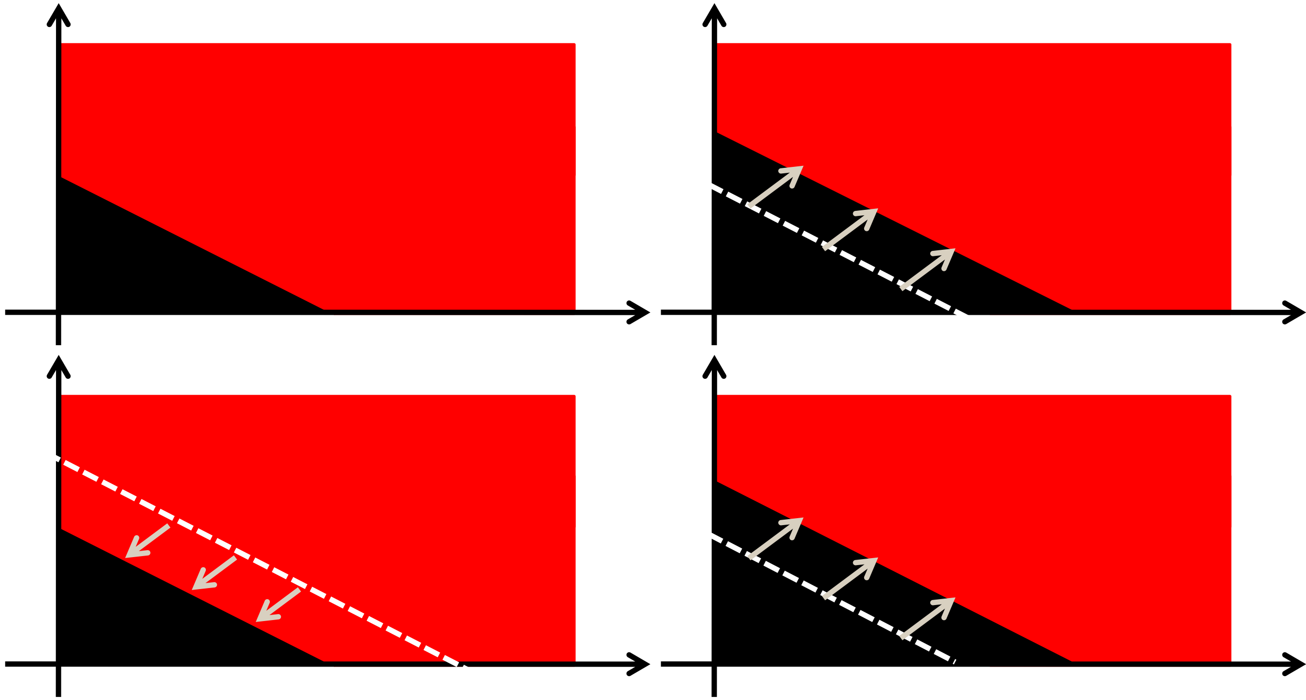


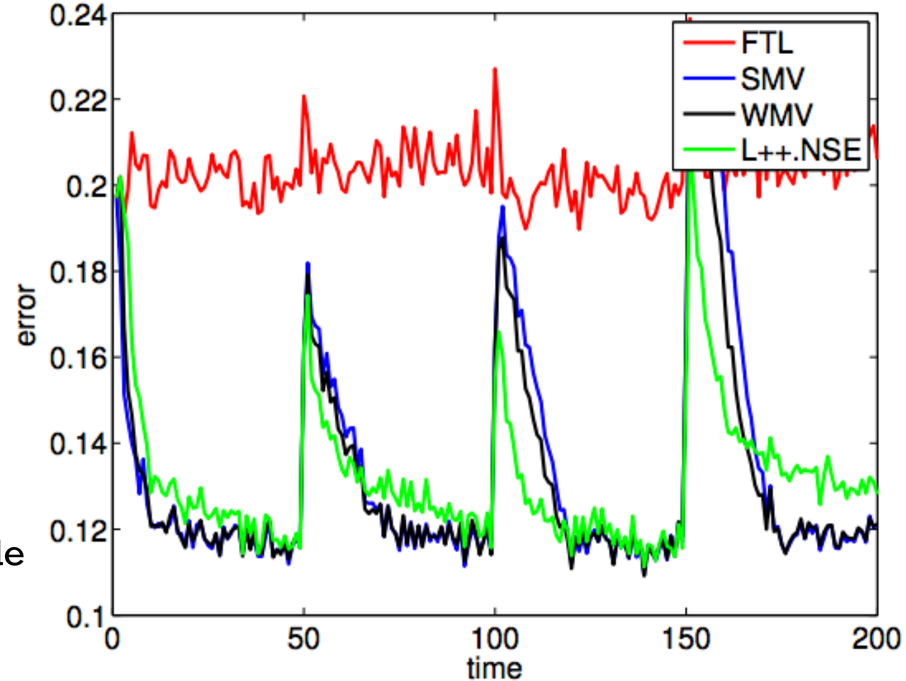
# THE VALUE OF SYNTHETIC & REAL DATA

- Concept drift algorithms have been benchmarked against a vast pool of synthetic and real-world data sets, both of which are of great importance to appropriately benchmarking
  - **Synthetic data** allow us to carefully design experiments to evaluate the limitations of an approach
  - **Real world** data serve as the ultimate benchmark about how we should expect an algorithm to perform when it is deployed

# SEA DATA STREAM

- Hyperplane changes location at three points in time
  - Three features only two of which are relevant. One feature is noise with 10% noise in the labels
  - Class imbalance changes as the plane shift. Thus, change in $\phi(\boldsymbol{x}|y)$ and $\phi(y)$ changes.
    - Dual change



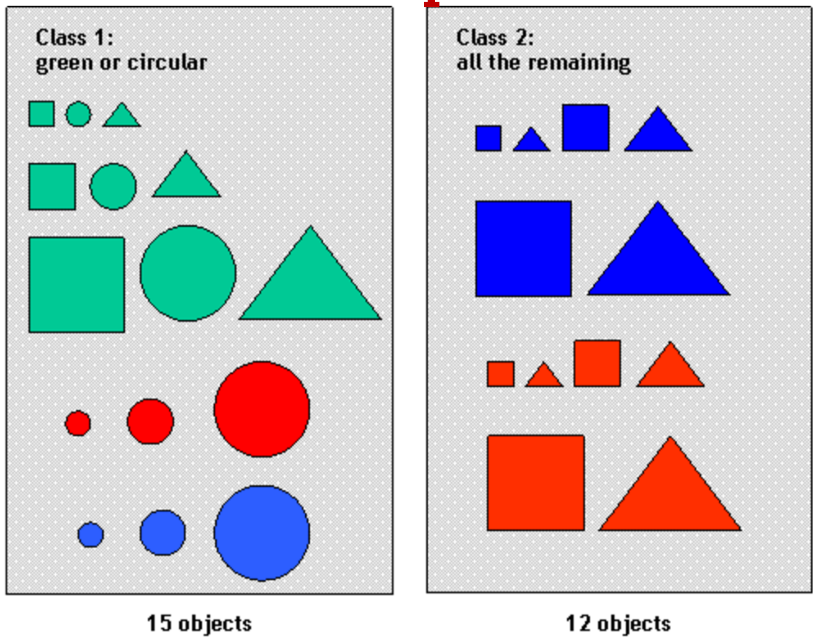



G. Ditzler, G. Rosen, and R. Polikar, "Domain Adaptation Bounds for Multiple Expert Systems Under Concept Drift," International Joint Conference on Neural Networks, 2014. (best student paper)

# KUNCHEVA'S DRIFT GENERATOR

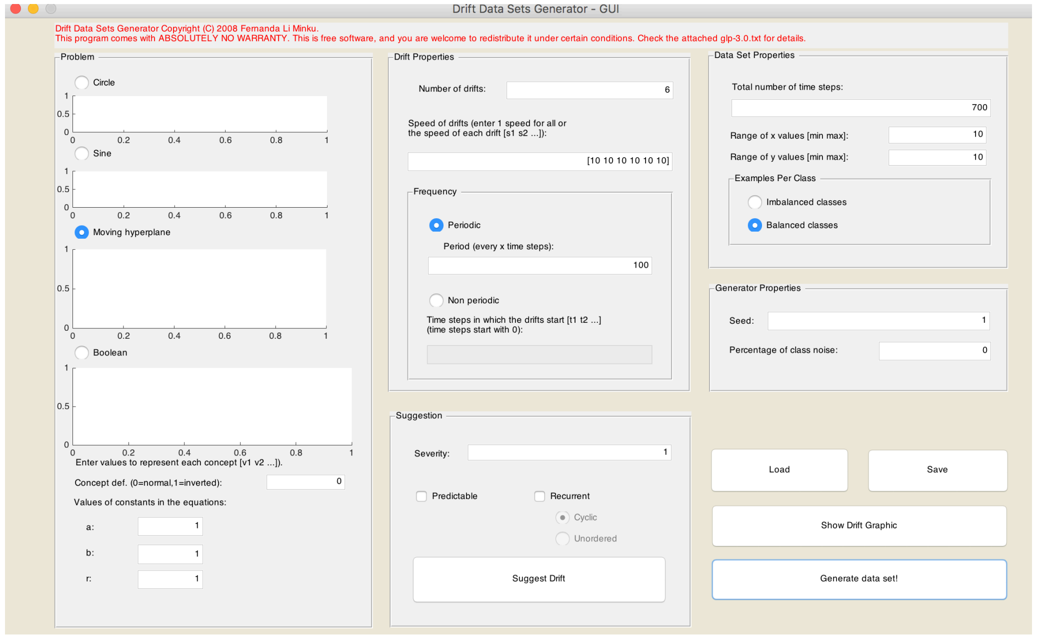
- Narasimhamurthy & Kuncheva (2007) developed a Matlab package for simulating nonstationary data streams. Feature include:
  - STAGGER: The feature space is described by three features: size, color and shape. There are three data sources
    - Target Concept 1 : size = small AND color = red
    - Target Concept 2 : color = green OR shape = circular
    - Target Concept 3 : size = medium OR size = large
  - Drifting Hyperplanes: Similar to SEA with a plane
  - Drifting Multi-Modal Gaussian Distributions:
  - Stationary to Nonstationary Data Stream: The above data streams can be sampled from a static distribution; however, to add in nonstationarities, drift can be simulated in the stream by sampling from different concepts

**STAGGER: Concept 2**




Narasimhamurthy A., L.I. Kuncheva, A framework for generating data to simulate changing environments, *Proc. IASTED, Artificial Intelligence and Applications,* Innsbruck, Austria, 2007, 384-389

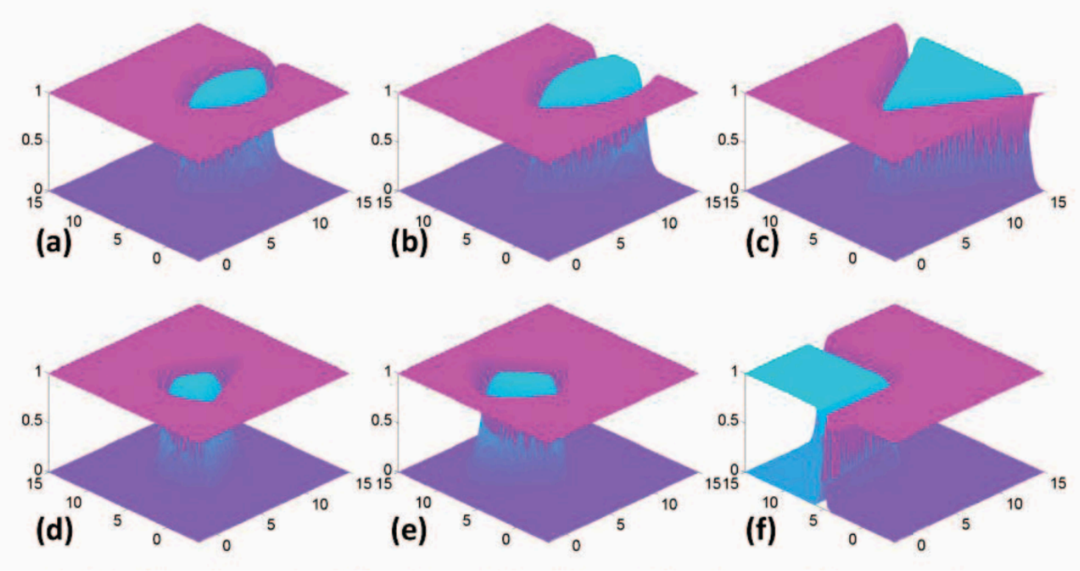
# DDD GENERATOR

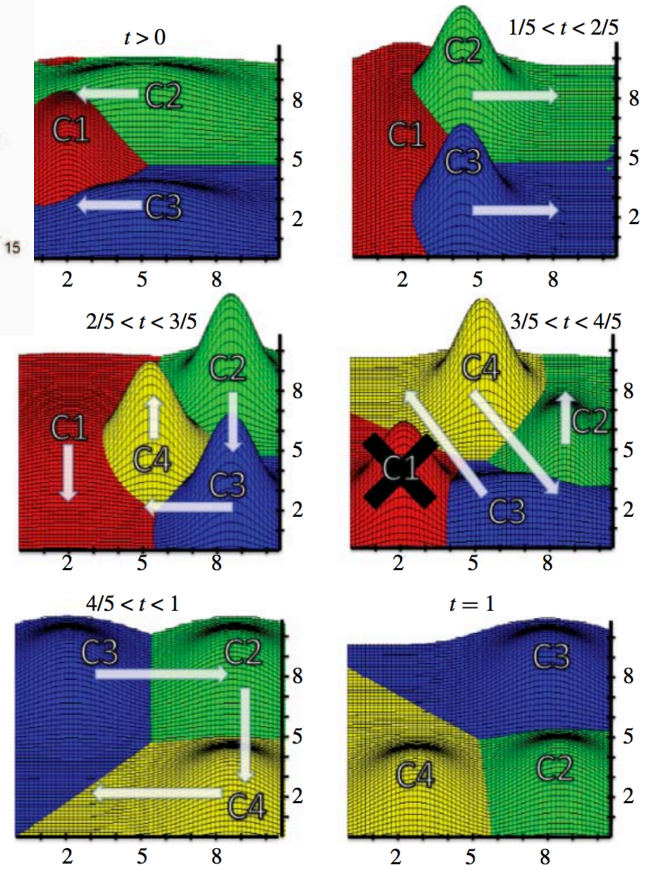


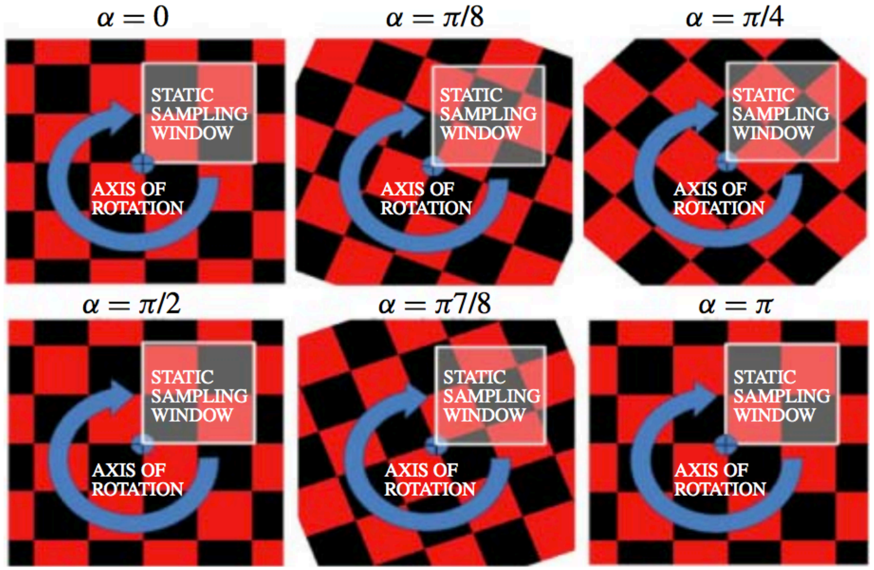

- Minku et al.'s concept drift generator for Matlab
  - **Circle**: Given two variables and a point, do samples fall in or out of a circle with radius $r$? Let $r$ change to simulate drift
  - **Sine**: $y > a\sin(b\ x + c) + d$? Changle the parameters $a$, $b$, $c$, and $d$.
  - **Moving Hyperplane**: Similar to SEA with a 1D line
  - **Boolean**: Modification of a STAGGER themed data set
- You can simulate a lot of different data streams with non-stationarities using the GUI

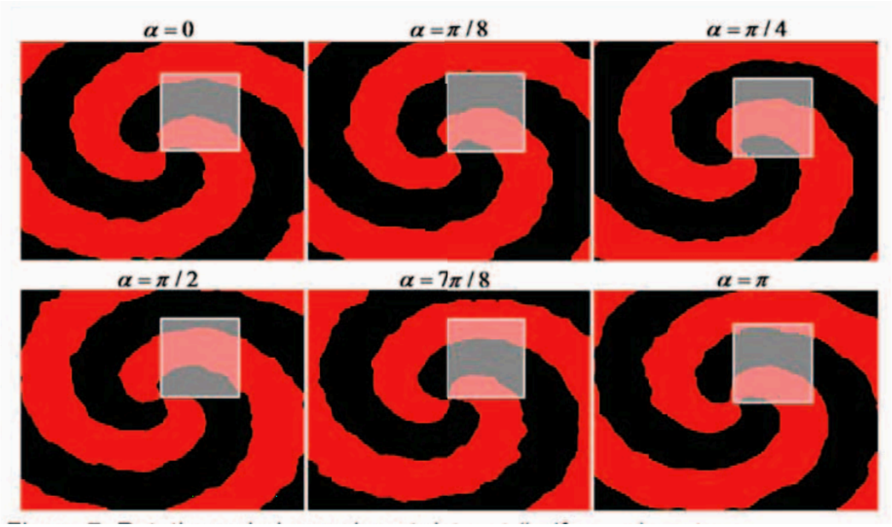
MINKU, L. L.; WHITE, A. P.; YAO, X. . "The Impact of Diversity on On-line Ensemble Learning in the Presence of Concept Drift.", *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 22, n. 5, p. 730-742, May 2010

# OTHER APPROACHES



# REAL-WORLD DATA

# REAL-WORLD DATA

- On **real data** it is sometimes difficult to obtain statistically significant results,
  - How can we quantify delays in change-detection applications on time-dependent data
    - For synthetic data we will know the location of the change; however, this is a bit more ambiguous with real-world data
  - Sometimes time-dependant are data correlated, e.g., the New South Whales electricity data (elec2)
  - Data may not be evolving through a sequence of stationary states
    - This is a problem for active methods
  - Difficult to estimate what could be the performance in real-world because sometimes supervised samples are provided depending on your previous performance
  - Labeled data are not always available to tell us what the current error is for the system to be able to update classifier paremeters (e.g., classifier weights)

# REAL-WORLD DATA

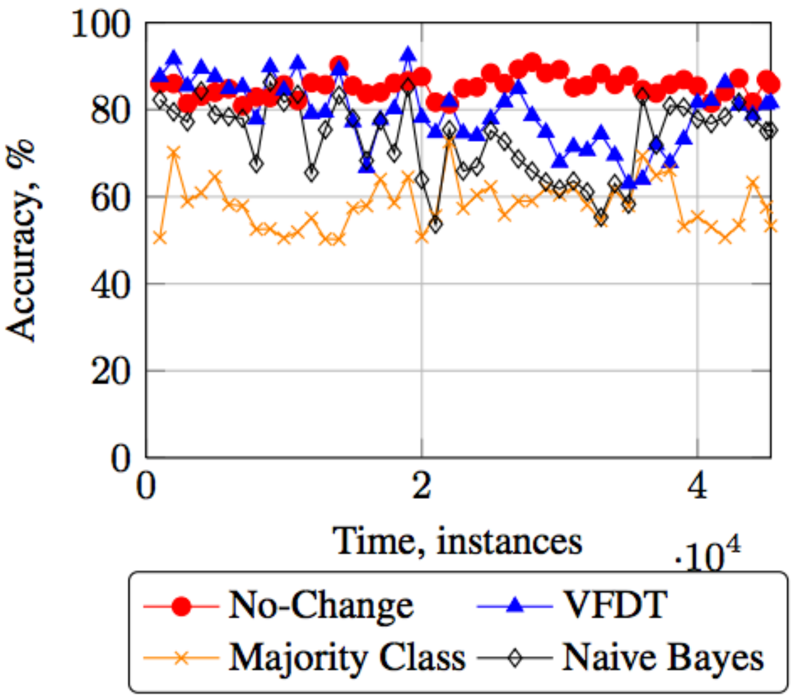
- **Airlines Data**: 100M+ instances contain flight arrival and departure records. The goal is to predict if a flight is delayed.
- **Chess.com**: Game records for a player over approximately three years
- elec2:
- **KDD Cup 1999 Data**: Collection of network intrusion detection data.
- **Luxembourg**: Predict a users internet usage European Social Survey data
- **NOAA**: ~27 years of daily weather measurements from Nebraska. The goal is to predict rainfall.
- **POLIMI Rock Collapse/Landslide Forecasting**: Sensor measurements coming from monitoring systems for rock collapse and landslide forecasting deployed on the Italian Alps.
- **Spam**: Collection of spam & ham emails collected over two years

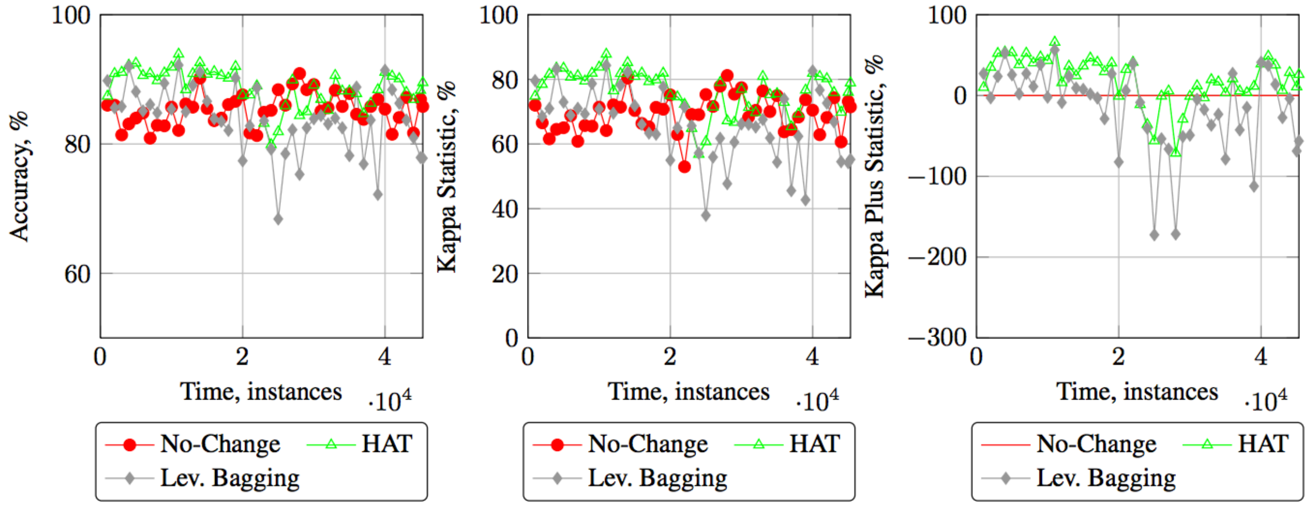
https://github.com/gditzler/ConceptDriftResources

# NOTE ON ELEC2

- Data streams could have a temporal component that is not considered in the evaluation of a classifier(s)
  - Assumption is that data are not sampled iid, but still sampled independently
  - This is an issue if the data are auto-correlated
- Bifet et al. pointed out this flaw in the elec2 data set and presented a new statistic for benchmarking such data sets that have temporal dependence

$$\kappa^{+} = \frac{n}{n-1} \cdot p_0 - \frac{1}{n-1}$$



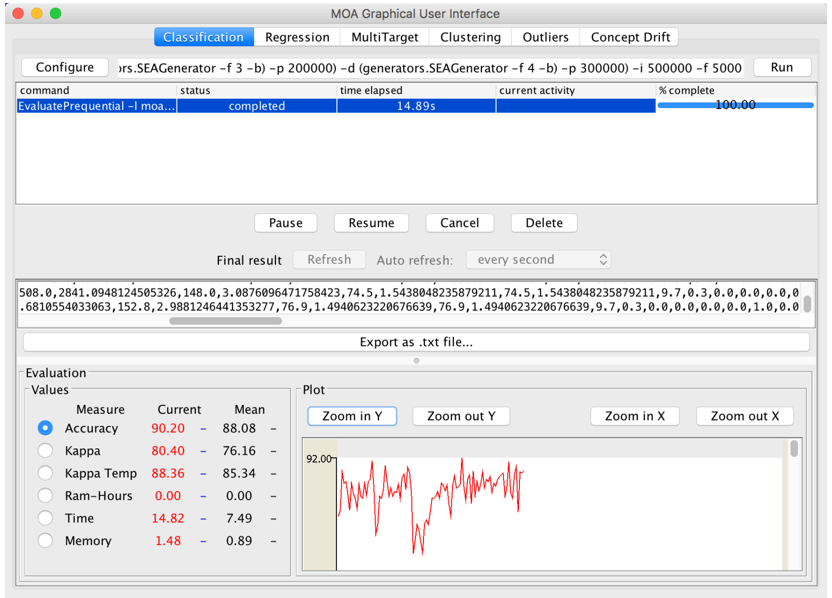




Albert Bifet, Indrė Žliobaitė, Jesse Read, Bernhard Pfahringer and Geoff Holmes: Pitfalls in benchmarking data stream classification and how to avoid them ECML-PKDD 2013

# SOFTWARE

# MASSIVE ONLINE ANALYSIS

- **Massive Online Analysis (MOA)** is a Java software package for developing and benchmarking data stream algorithms
  - **Active approaches**
    - DDM
    - EDDM
    - ADWIN
  - **Passive approaches**
    - ASHT
    - SGD
    - AUE
    - AUE2




- **Extensions**: Scalable MOA (Apache Storm), and StreamDM (Apache Streaming Spark)

# INCREMENTAL LEARNING WITH ENSEMBLES

- Matlab-based toolbox (collection) of scripts that implement several ensemble-based algorithms for learning in nonstationary environments
  - Weighted Majority Ensemble
  - Simple Majority Ensemble
  - Follow the Leader
  - Learn++/Learn++.NSE/Learn++.CDS
  - Example scripts are included

| | Data set | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *rain* | *elec2* | *german* | *ringworm* | *splice* | *twonorm* | *wave* | *chess* | *lux* | rank |
| | **1–0 loss** | | | | | | | | | |
| *nse* | 0.218 (3) | 0.339 (1) | 0.224 (1) | 0.232 (2) | 0.18 (1) | 0.0234 (2) | 0.139 (1) | 0.329 (3) | 0.102 (1) | 1.67 |
| *avg* | 0.216 (2) | 0.365 (4) | 0.259 (4) | 0.242 (3) | 0.206 (3) | 0.0245 (3) | 0.152 (3) | 0.419 (4) | 0.499 (4) | 3.33 |
| *exp* | 0.214 (1) | 0.354 (3) | 0.229 (2) | 0.232 (1) | 0.184 (2) | 0.0231 (1) | 0.144 (2) | 0.31 (2) | 0.485 (3) | 1.89 |
| *ftl* | 0.294 (4) | 0.341 (2) | 0.243 (3) | 0.249 (4) | 0.271 (4) | 0.0387 (4) | 0.162 (4) | 0.288 (1) | 0.13 (2) | 3.11 |
| | **logistic loss** | | | | | | | | | |
| *nse* | 0.781 (3) | 0.897 (2.5) | 0.69 (1.5) | 0.747 (1) | 0.756 (1.5) | 0.489 (1) | 0.637 (1) | 0.879 (3) | 0.992 (2) | 1.83 |
| *avg* | 0.781 (2) | 0.897 (2.5) | 0.69 (1.5) | 0.747 (2) | 0.756 (1.5) | 0.491 (3) | 0.637 (2) | 0.879 (4) | 1.03 (4) | 2.50 |
| *exp* | 0.779 (1) | 0.892 (1) | 0.691 (3) | 0.748 (3) | 0.756 (3) | 0.491 (2) | 0.637 (3) | 0.873 (2) | 0.998 (3) | 2.33 |
| *ftl* | 0.87 (4) | 0.938 (4) | 0.738 (4) | 0.795 (4) | 0.824 (4) | 0.502 (4) | 0.665 (4) | 0.83 (1) | 0.627 (1) | 3.33 |
| | **mse loss** | | | | | | | | | |
| *nse* | 0.652 (3) | 0.896 (3) | 0.716 (1.5) | 0.705 (1.5) | 0.569 (1.5) | 0.0759 (1) | 0.454 (3) | 0.991 (3) | 0.377 (1) | 2.06 |
| *avg* | 0.625 (1) | 0.892 (2) | 0.716 (1.5) | 0.705 (1.5) | 0.569 (1.5) | 0.0768 (3) | 0.41 (2) | 0.933 (2) | 1.47 (4) | 2.06 |
| *exp* | 0.63 (2) | 0.872 (1) | 0.719 (3) | 0.707 (3) | 0.572 (3) | 0.0766 (2) | 0.409 (1) | 0.918 (1) | 1.1 (3) | 2.11 |
| *ftl* | 1.18 (4) | 1.37 (4) | 0.971 (4) | 0.997 (4) | 1.09 (4) | 0.155 (4) | 0.646 (4) | 1.15 (4) | 0.52 (2) | 3.78 |

G. Ditzler, G. Rosen and R. Polikar, "Discounted expert weighting for concept drift," International Symposium on Computational Intelligence in Dynamic and Uncertain Environments, 2013.

# COMPARING MULTIPLE CLASSIFIERS

- Comparing multiple classifiers on multiple datasets is not a trivial problem
  - Confidence intervals will only allow for the comparison of multiple classifiers on a single dataset
- The rank based Friedman test can determine if classifiers are performing equally across multiple dataset
  - Apply ranks to the average of each measure on a dataset
  - Standard deviation of the measure is not used in the Friedman test

$$\chi_F^2 = \frac{12N}{k(k+1)}\left(\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right)$$

$$F_f = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}$$

- $z$-scores can be computed from the ranks in the Friedman test
  - The $\alpha$-level or critical value must be adjusted based on the multiple comparisons being made
  - Bonferroni-Dunn procedure adjusts $\alpha$ to $\alpha\,/(k-1)$

$$z_m(i,j) = \frac{R_m(i) - R_m(j)}{\sqrt{\frac{k(k+1)}{6N}}}$$

J. Demsar, "Statistical Comparisons of Classifiers over Multiple Data Sets," J. Machine Learning Research, vol. 7, pp. 1-30, 2006.

# CHALLENGES AND PERSPECTIVES

# CHALLENGES

- Learning in nonstaionary environments is becoming a more mature field; however, there are many sub-problems in the field that still need to be addressed more rigorously
  - **Unbalanced environment**: Data from each of the classes are extremely imbalanced, which is a serious problem if the algorithm is using error to track the environment.
  - **Semisupervised**: How can we best incorporate data that are unlabeled into our model if we cannot assume the data are sampled iid?
    - Consensus Maximization: Train supervised and unsupervised models
    - Semi-supervised vs. Transductive?
  - **Latency verification**: What if we cannot assume that the classifier will receive immediate feedback?
    - A study of extreme latency verification and how to perform benchmarks
  - **Error estimation**: How can error be accurately estimated in the precense of non-stationary data streams when an concept is abruptly re-introduced

# CHALLENGES

- A **theoretical framework** is lacking that incorporates drift information in the labeled and unlabeled data
  - Less heuristics more statistics!
- Integration of expert-driven knowledge with data-driven knowledge