Big Data Stream Analytics for Cyber Security

Professor Latifur Khan

Department of Computer Science

University of Texas at Dallas, USA

ACM Distinguish Scientist
IBM Faculty Award Winner 2016 (Research)
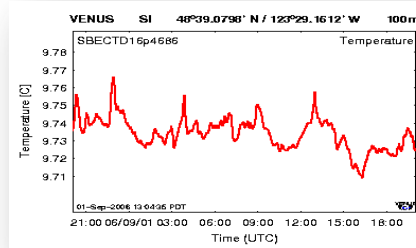
lkhan@utdallas.edu

Outline

- Data Streams: Single Stream and Multi-stream
- Novel Class Detection & Zero Day Attack
- Multi-Stream
- Domain Adaptation
- Big Text Analytics: Political Report
- On-Line Metric Learning
- Website Fingerprinting

Data Streams

➢ Data Stream:

- is continuous flow of data.
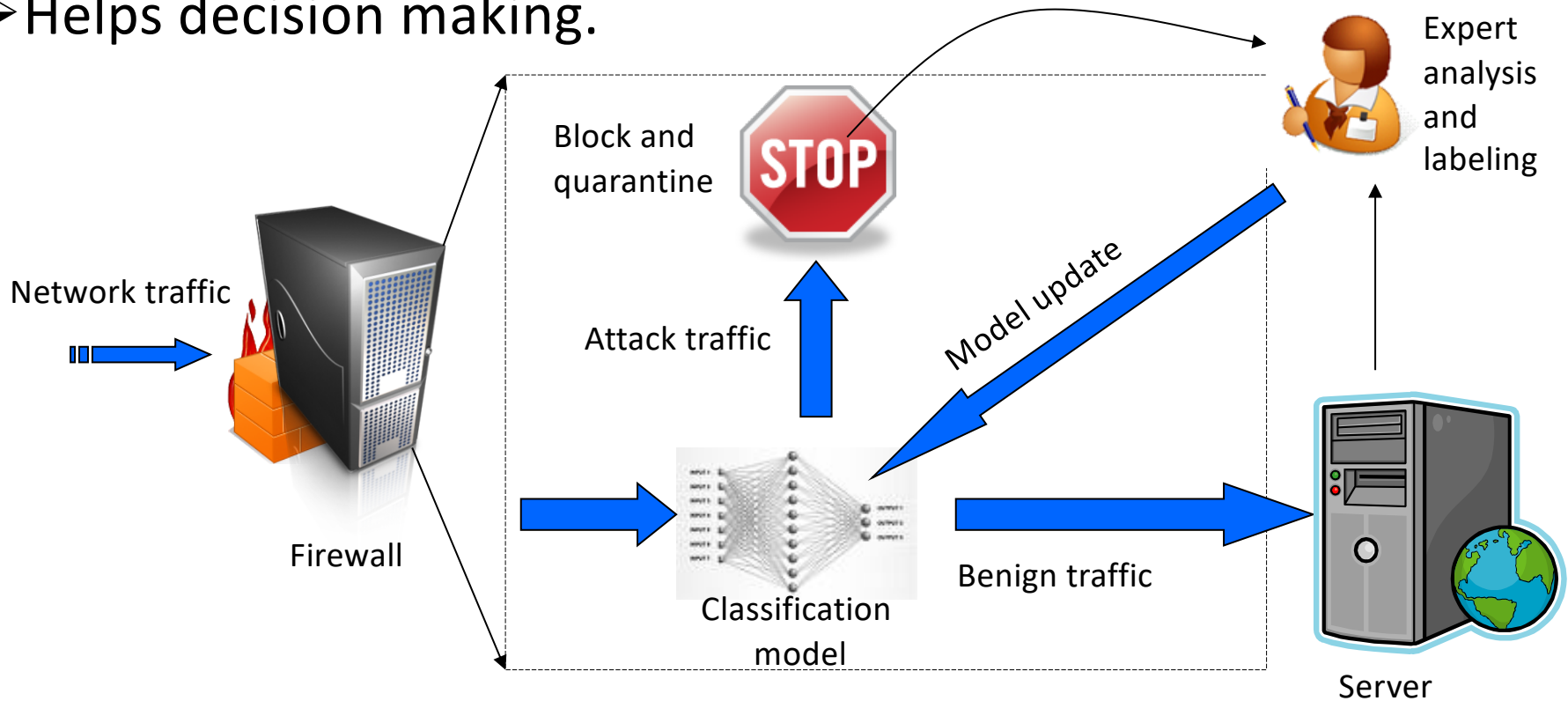- very common in today's connected digital world.

Sensor Data

Network Traffic

- important source of knowledge that enables to take extremely important decisions in (near) real time.

➢ Hence, data stream mining is very important.
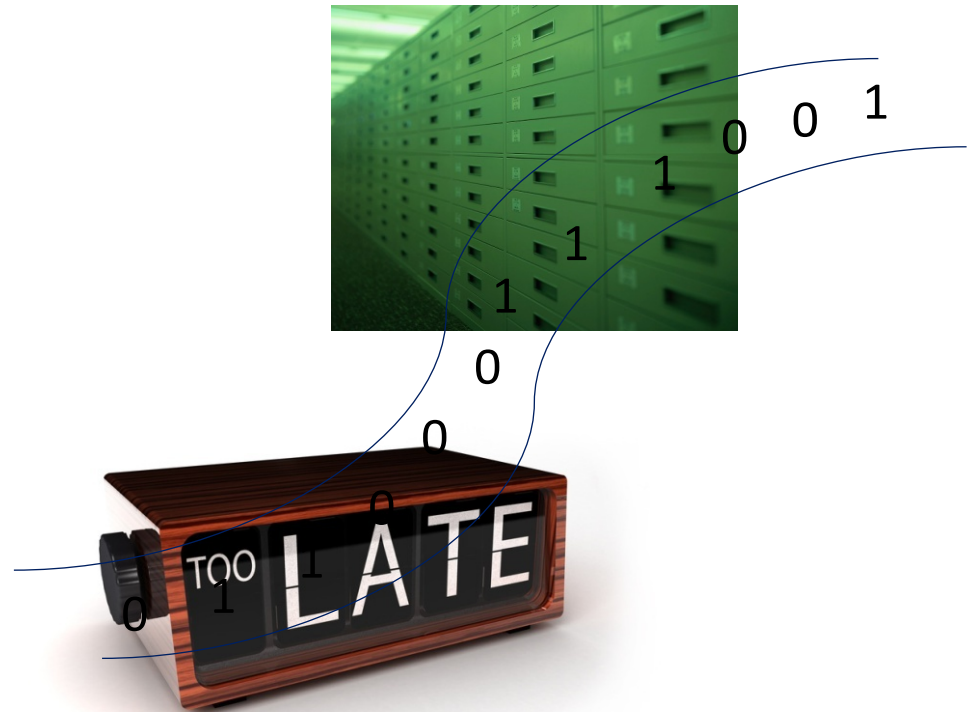
Data Stream Classification

➢ Uses past data to build classification model.

➢ Predicts the labels of future instances using the model.
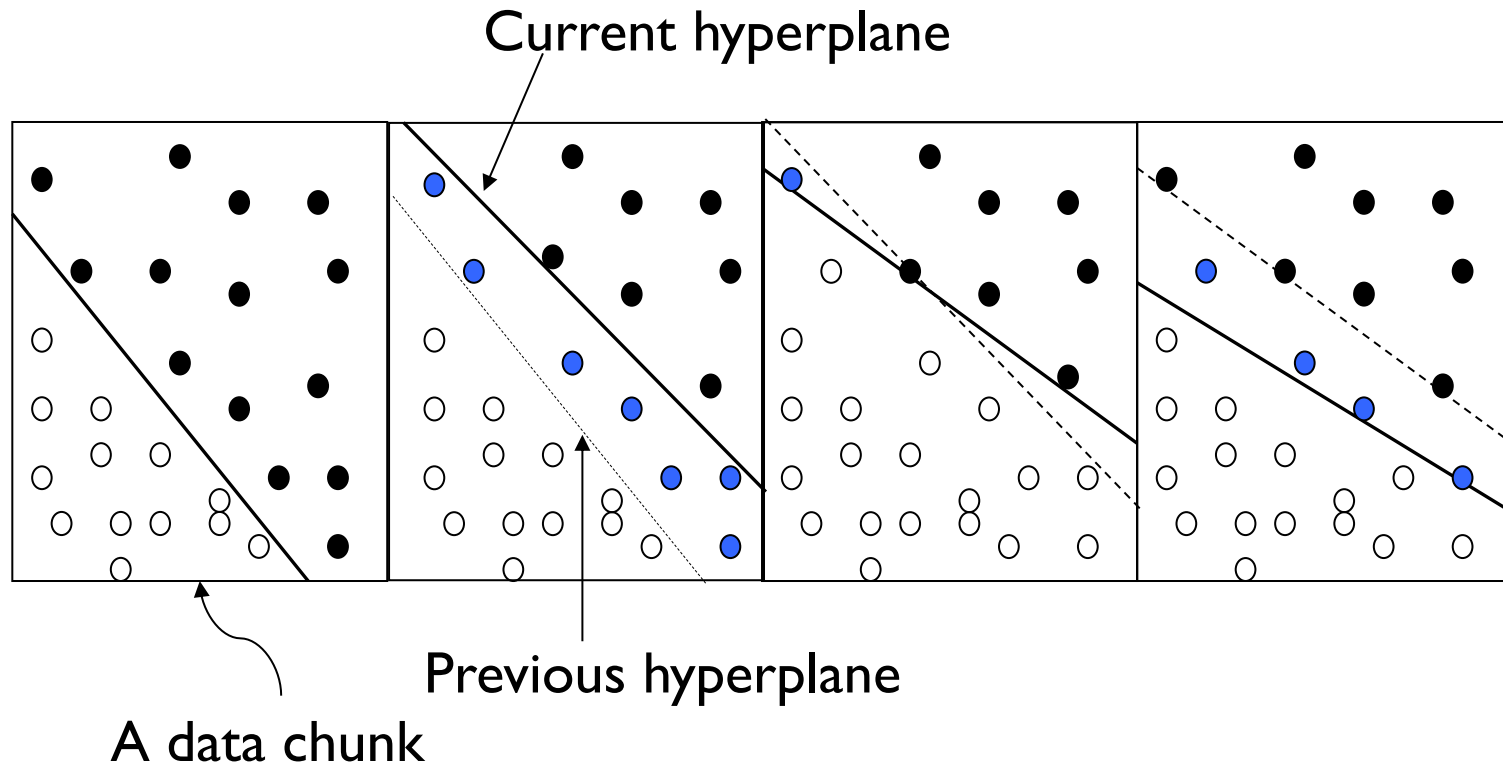
➢ Helps decision making.

Network traffic

Firewall

Block and quarantine

STOP

Attack traffic

Model update

Classification model

Benign traffic

Expert analysis and labeling

Server

Challenge: Infinite Length

➢ Impractical to store and use all historical data

- requires infinite storage

- and running time

Challenge: Concept Drift

Current hyperplane

Previous hyperplane

A data chunk

Negative instance ●

Positive instance ○

Instances victim of concept-drift ●

Challenge: Concept Evolution
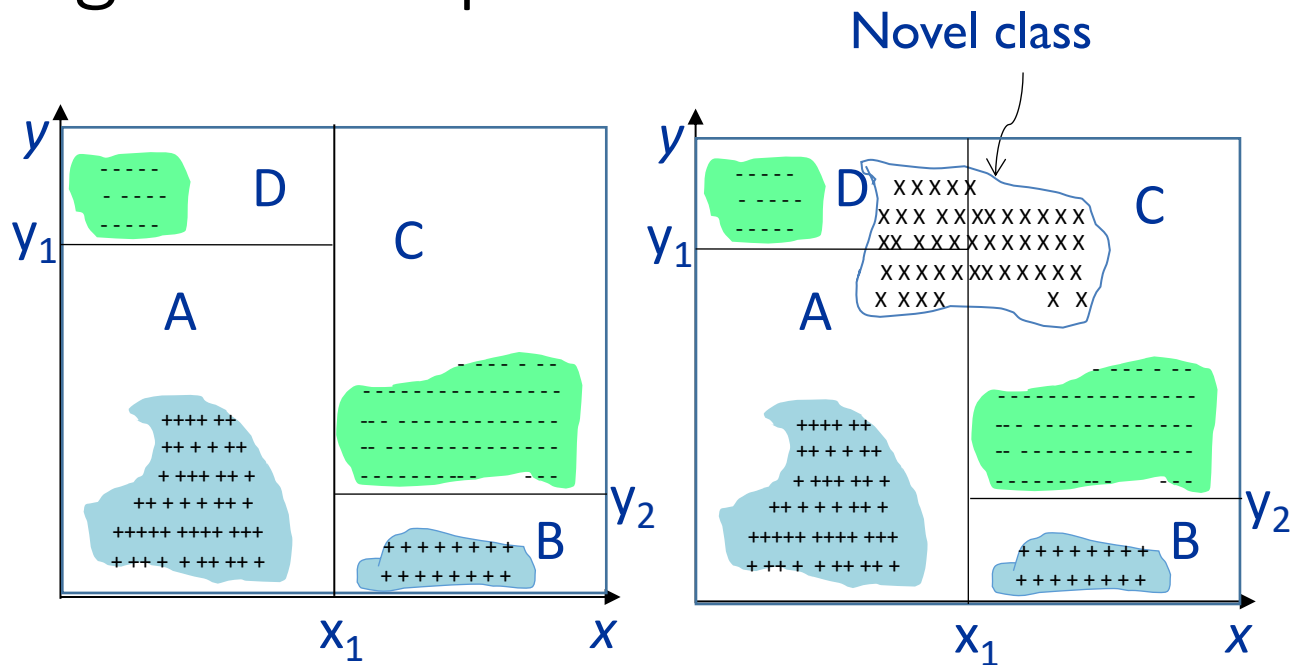
Classification rules:

R1. if ($x > x_1$ and $y < y_2$) or ($x < x_1$ and $y < y_1$) then class = +

R2. if ($x > x_1$ and $y > y_2$) or ($x < x_1$ and $y > y_1$) then class = -

Existing classification models misclassify novel class instances

Concept Evolution

A novel class is a newly emerged class that has not previously been modeled by the classifier over the input stream.

Class Distribution

Novel Class

➢ M. M. Masud, T. M. Al-Khateeb, L. Khan, C. Aggarwal, J. Gao, J. Han, and B. Thuraisingham, "Detecting recurring and novel classes in concept-drifting data streams," in Proc. 11th IEEE Int. Conf. Data Mining (ICDM), 2011, pp. 1176–1181.

8

Existing Techniques: Ensemble based Approaches

Masud et al. [1][2]
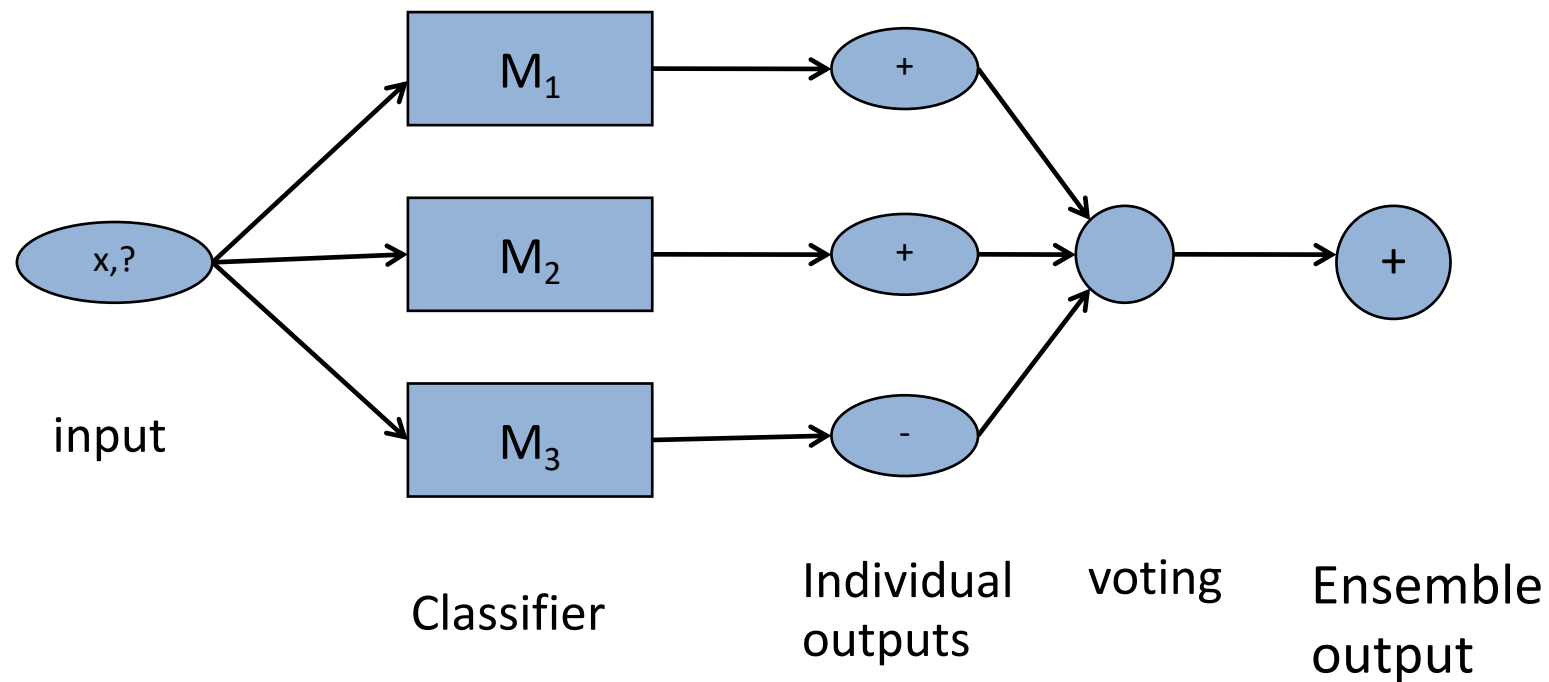
input

Classifier

Individual outputs

voting

Ensemble output

[1] Mohammad M. Masud, Jing Gao, Latifur Khan, Jiawei Han, Bhavani M. Thuraisingham: A Practical Approach to Classify Evolving Data Streams: Training with Limited Amount of Labeled Data. ICDM 2008: 929-934
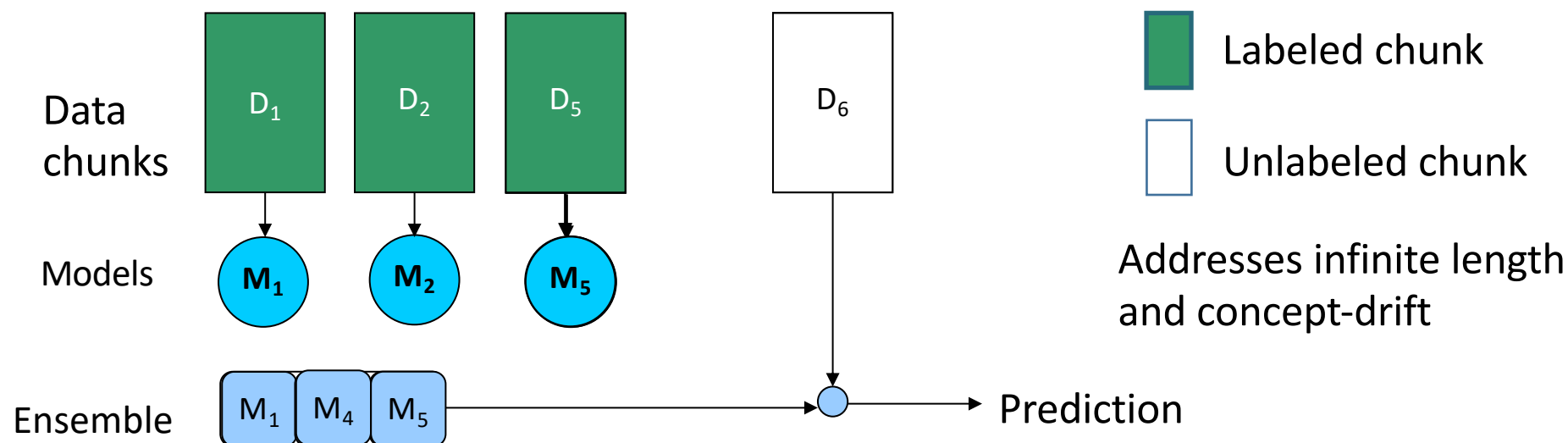[2] Mohammad M. Masud, Clay Woolam, Jing Gao, Latifur Khan, Jiawei Han, Kevin W. Hamlen, Nikunj C. Oza: Facing the reality of data stream classification: coping with scarcity of labeled data. Knowl. Inf. Syst. 33(1): 213-244 (2011)

Existing Techniques: Ensemble Techniques

➢ Divide the data stream into equal sized chunks
- Train a classifier from each data chunk
- Keep the best *t* such classifier-ensemble
- Example: for *t* = 3

Note: D_i may contain data points from different classes

Data chunks

Models

Ensemble

D_1 D_2 D_5 D_6

M_1 M_2 M_5

M_1 M_4 M_5 →   Prediction

Labeled chunk

Unlabeled chunk

Addresses infinite length and concept-drift

Novel Class Detection

Masud et al. [1][2], *Khateeb et al.* [3]

➢ Non parametric
 – does not assume any underlying model of existing classes

➢ Steps:
  1. Creating and saving decision boundary during training
  2. Detecting and filtering outliers
  3. Measuring cohesion and separation among test and training instances

[1] Mohammad M. Masud, Qing Chen, Latifur Khan, Charu C. Aggarwal, Jing Gao, Jiawei Han, Ashok N. Srivastava, Nikunj C. Oza: Classification and Adaptive Novel Class Detection of Feature-Evolving Data Streams. IEEE Trans. Knowl. Data Eng. 25(7): 1484-1497 (2013)
[2] Mohammad M. Masud, Jing Gao, Latifur Khan, Jiawei Han, Bhavani M. Thuraisingham: Classification and Novel Class Detection in Concept-Drifting Data Streams under Time Constraints. IEEE Trans. Knowl. Data Eng. 23(6): 859-874 (2011)
[3] Tahseen Al-Khateeb, Mohammad M. Masud, Latifur Khan, Charu C. Aggarwal, Jiawei Han, Bhavani M. Thuraisingham: Stream Classification with Recurring and Novel Class Detection Using Class-Based Ensemble. ICDM 2012: 31-40

Training with Semi-Supervised Clustering
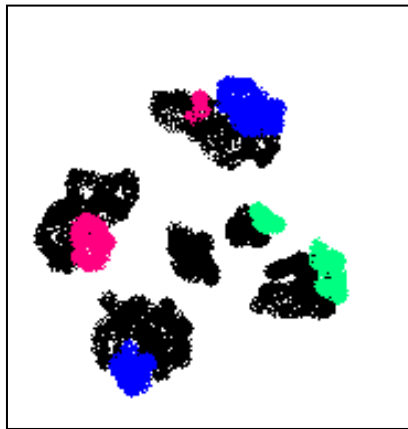
Impurity based Clustering
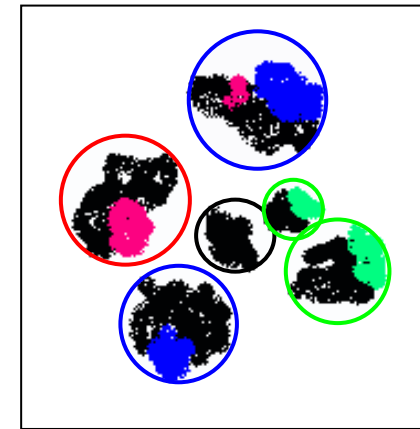
Legend:

Black dots: unlabeled instances

Colored dots: labeled instances

Semi Supervised Clustering

Masud et al. [1][2]

➢Objective function (dual minimization problem)

$$O_{MCIKmeans} = \sum_{i=1}^{K}\left(\sum_{x\in\mathcal{X}_i}||x-u_i||^2 + \sum_{x\in\mathcal{L}_i}||x-u_i||^2 * Imp_i\right)$$

Intra-cluster dispersion   **Cluster impurity**

*Imp_i: = Aggregated dissimilarity count$_i$ * Entropy$_i$ = ADC$_i$ * Ent$_i$*

Aggregated dissimilarity count (*ADC*): $\mathcal{ADC}_i = \sum_{x\in\mathcal{L}_i} DC_i(x, y).$

$$DC_i(x, y) = \begin{cases} 0 & \text{if } x \text{ is unlabeled (i.e., } y = \phi) \\ |\mathcal{L}_i| - |\mathcal{L}_i(c)| & \text{if } x \text{ is labeled and its label } y=c \end{cases}$$
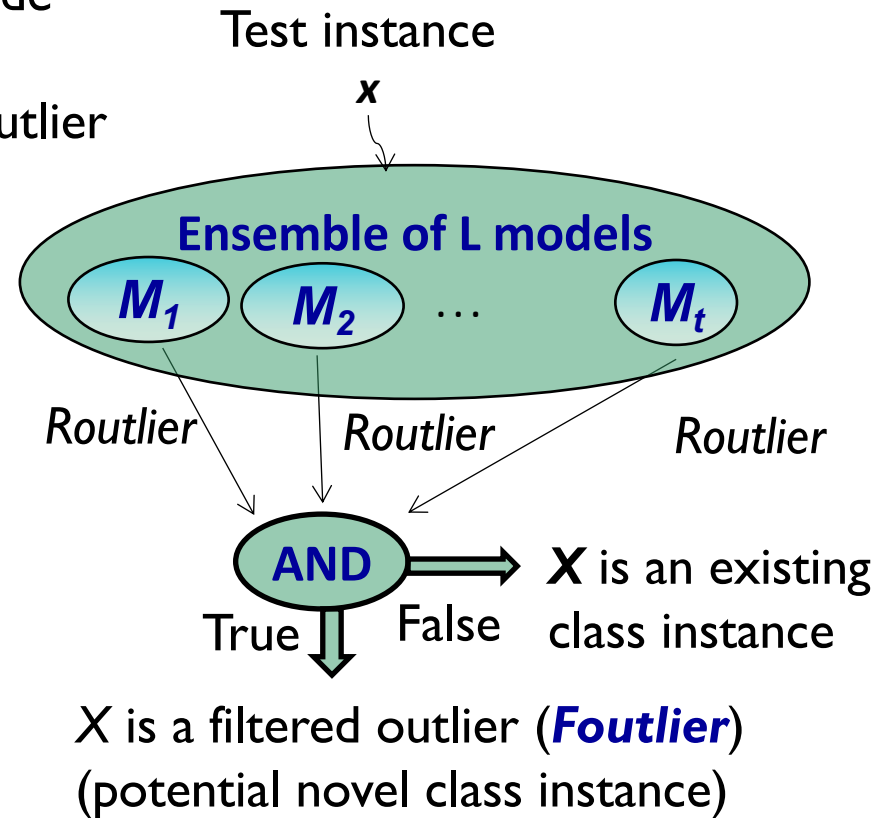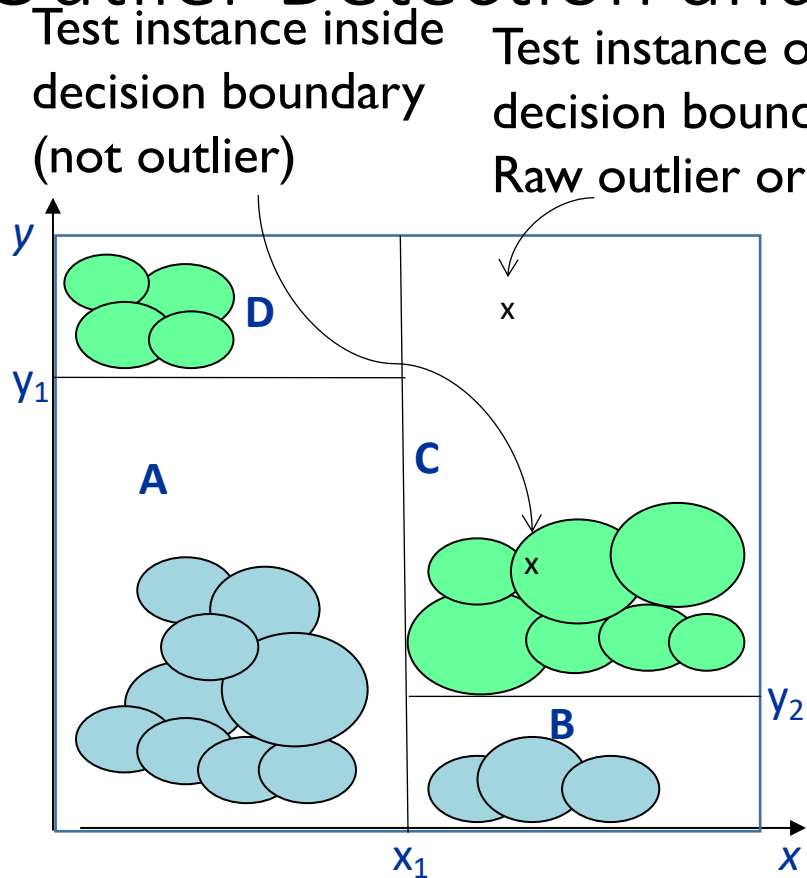
Entropy (*Ent*): $Ent_i = \sum_{c=1}^{C}(-p_c^i * log(p_c^i))$

The minimization problem is solved using the Expectation-Maximization (E-M) framework

[1] Mohammad M. Masud, Jing Gao, Latifur Khan, Jiawei Han, Bhavani M. Thuraisingham: A Practical Approach to Classify Evolving Data Streams: Training with Limited Amount of Labeled Data. ICDM 2008: 929-934
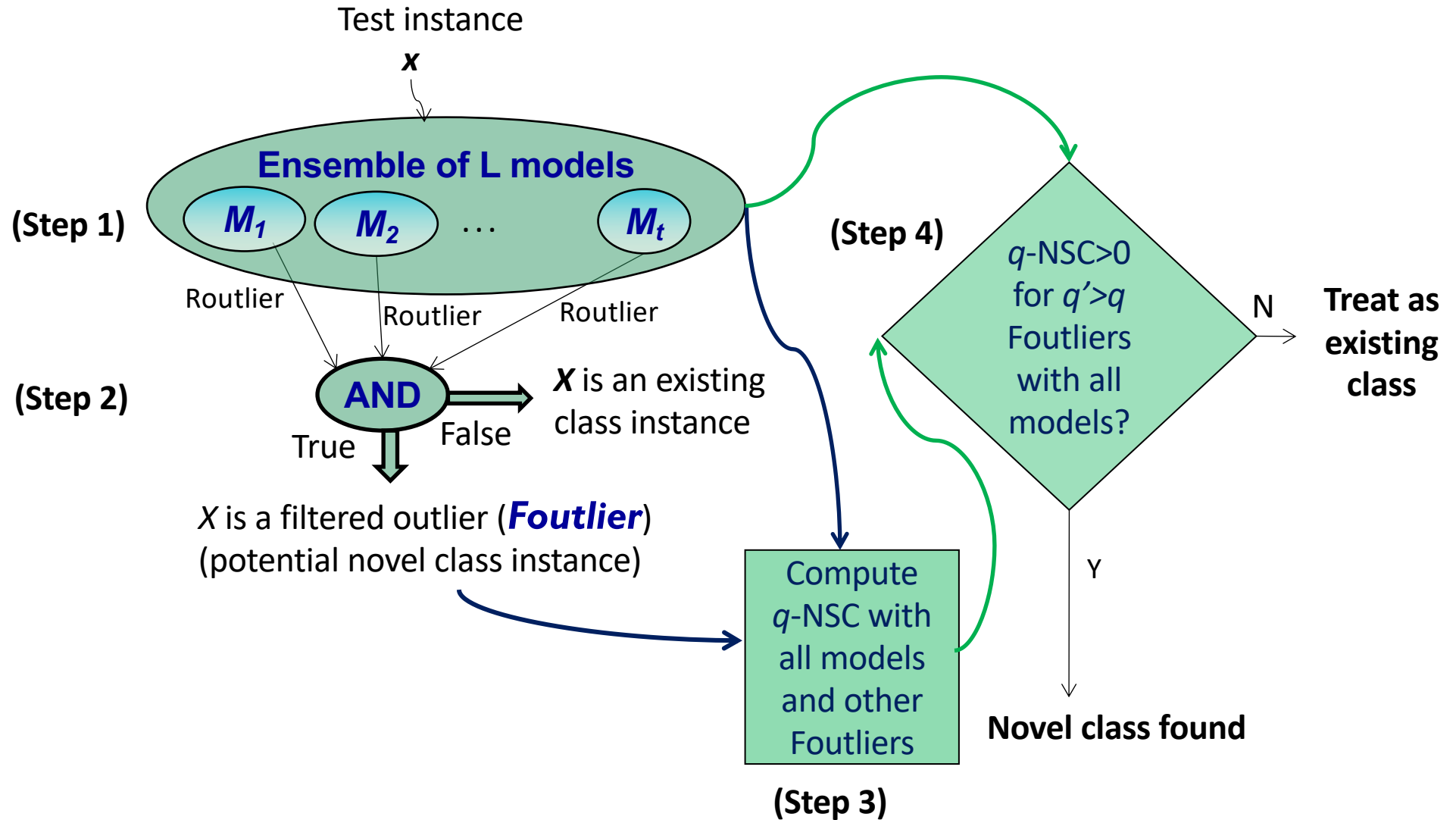[2] Mohammad M. Masud, Clay Woolam, Jing Gao, Latifur Khan, Jiawei Han, Kevin W. Hamlen, Nikunj C. Oza: Facing the reality of data stream classification: coping with scarcity of labeled data. Knowl. Inf. Syst. 33(1): 213-244 (2011)

Outlier Detection and Filtering

Test instance inside decision boundary (not outlier)

Test instance outside decision boundary
Raw outlier or Routlier

Test instance
x

Ensemble of L models

M_1 M_2 ... M_t

Routlier   Routlier   Routlier

AND ⟶ X is an existing class instance
True False

X is a filtered outlier (**Foutlier**)
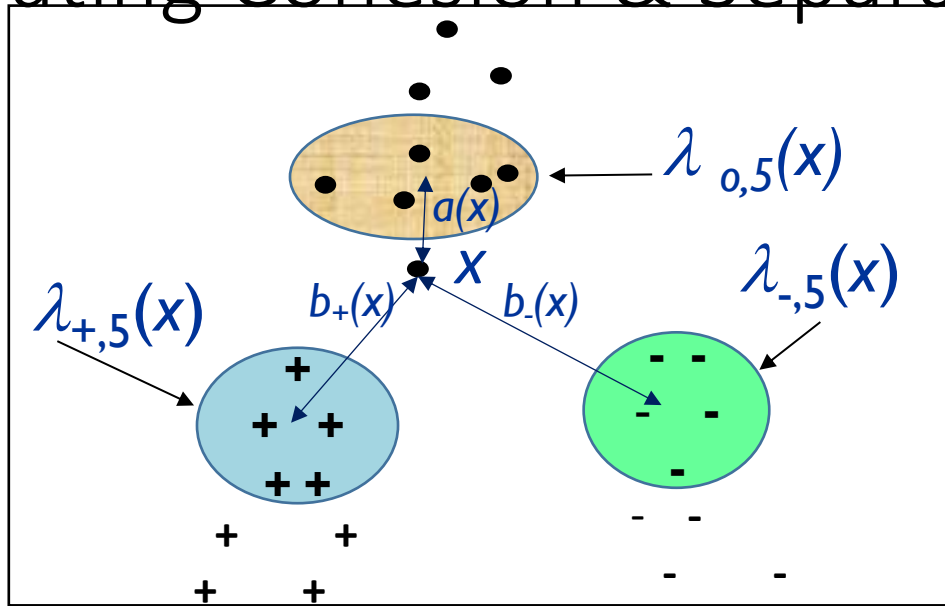(potential novel class instance)

Foutliers may appear as a result of novel class, concept-drift, or noise. Therefore, they are filtered to reduce noise as much as possible.

Novel Class Detection

Test instance
x

Ensemble of L models

(Step 1) M_1 M_2 $\ldots$ M_t

Routlier   Routlier   Routlier

(Step 2)   **AND** → *X* is an existing class instance

True   False

X is a filtered outlier (***Foutlier***)
(potential novel class instance)

Compute *q*-NSC with all models and other Foutliers

(Step 3)

(Step 4)

q-NSC>0 for *q'>q* Foutliers with all models?

N → **Treat as existing class**

Y

Novel class found

Computing Cohesion & Separation

$\lambda_{o,5}(x)$

$\lambda_{-,5}(x)$

$\lambda_{+,5}(x)$

$b_+(x)$ $b_-(x)$

$a(x)$ x

➤ $\lambda_c(x)$ is the
   *set of nearest neighbors
   of x belonging to class c*
➤ $\lambda_o(x)$ is the
   set of nearest Foutliers of x

- *a(x)* = mean distance from an *Foutlier x* to the instances in $\lambda_{o,q}(x)$
- $b_{min}(x)$ = minimum among all $b_c(x)$ (e.g. $b_+(x)$ in figure)
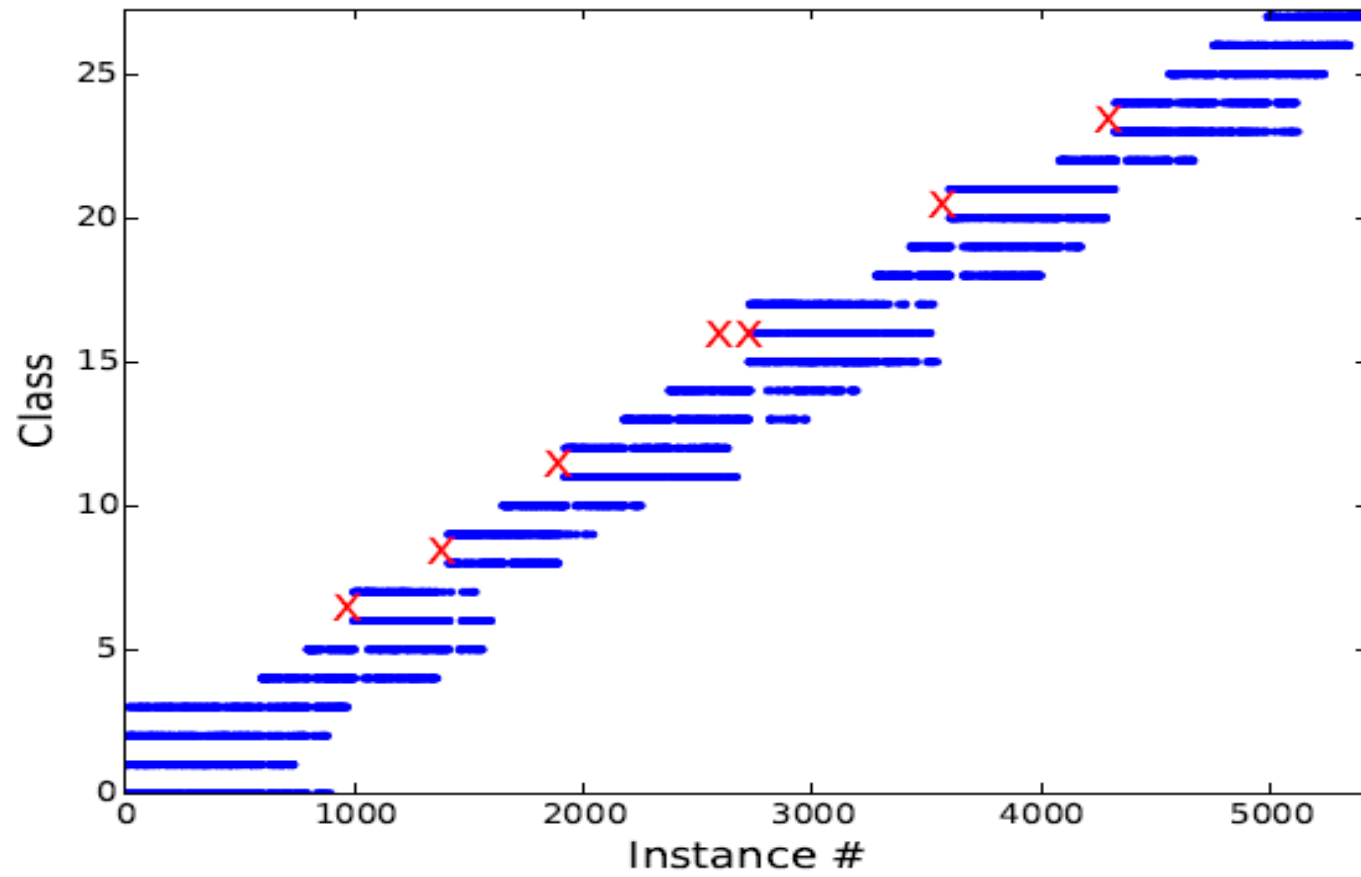- *q*-Neighborhood Silhouette Coefficient (*q*-NSC):

$$q\text{-}NSC(x) = \frac{(b_{min}(x) - a(x))}{\max(b_{min}(x), a(x))}$$

- If *q*-NSC(x) is positive, it means *x* is closer to *Foutliers* than any other class.

Summary of Datasets

Dataset	# Classes	# Dims	# Samples
Syn1	7	70	100,000
Syn2	7	70	100,000
Packets	28	6,000	5,600
SystemCalls	28	6,000	5,600
Fcover	7	54	150,000
IMDB	15	1000	15,000
PAMAP2	6	53	150,000

Packets and System calls datasets

Experimental Results

FPR%

Approach	Syn1	Syn2	Packets	SystemCalls	Fcover	PAMAP2	IMDB
Our approach	**2.01**	**2.01**	**1.01**	**1.01**	**1.82**	**1.01**	**0.01**
ECS-Miner	71.04	71.63	26.66	31.89	4.64	18.01	0.01
ECHO	14.90	15.21	0.01	6.11	2.83	4.97	0.01

FNR%

Approach	Syn1	Syn2	Packets	SystemCalls	Fcover	PAMAP2	IMDB
Our approach	**2.01**	**2.34**	**1.52**	**1.30**	**1.05**	**0.02**	**1.01**
ECS-Miner	2.03	17.43	25.09	23.64	4.38	0.05	29.14
ECHO	44.27	21.19	91.25	27.50	7.43	0.11	33.00

ERR%

Approach	Syn1	Syn2	Packets	SystemCalls	Fcover	PAMAP2	IMDB
Our approach	**2.02**	**2.06**	**1.46**	**1.26**	**1.97**	**1.58**	**2.46**
ECS-Miner	89.64	86.37	42.53	49.98	6.99	17.38	12.93
ECHO	31.19	27.49	75.52	30.38	5.09	5.37	9.78

Challenges: Fixed Chunk Size/ Decay Rate

Masud et al. [1], *Parker et al.* [2], *Aggarwal et al.* [3], *Klinkenberg*[4], *Cohen et al.* [5]
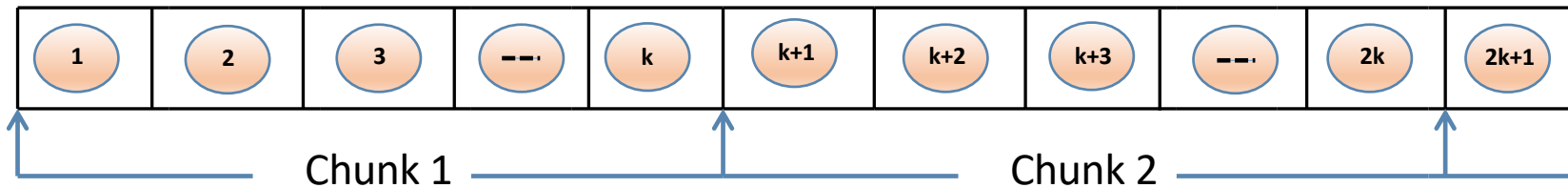
➤ Fixed chunk size
- requires *a priori* knowledge about the time-scale of change.
- delayed reaction if the chunk size is too large.
- unnecessary frequent training during stable period if chunk size is too small.

➤ Fixed decay rate
- assigns weight to data instances based on their age.
- decay constant must match the unknown rate of change.

[1] Mohammad M. Masud, Jing Gao, Latifur Khan, Jiawei Han, Bhavani M. Thuraisingham: Classification and Novel Class Detection in Concept-Drifting Data Streams under Time Constraints. IEEE Trans. Knowl. Data Eng. 23(6): 859-874 (2011)
[2] Brandon Shane Parker, Latifur Khan: Detecting and Tracking Concept Class Drift and Emergence in Non-Stationary Fast Data Streams. AAAI 2015: 2908-2913
[3] Charu C. Aggarwal, Philip S. Yu: On Classification of High-Cardinality Data Streams. SDM 2010: 802-813
[4] Ralf Klinkenberg: Learning drifting concepts: Example selection vs. example weighting. Intell. Data Anal. 8(3): 281-300 (2004)
[5] Edith Cohen, Martin J. Strauss: Maintaining time-decaying stream aggregates. J. Algorithms 59(1): 19-36 (2006)
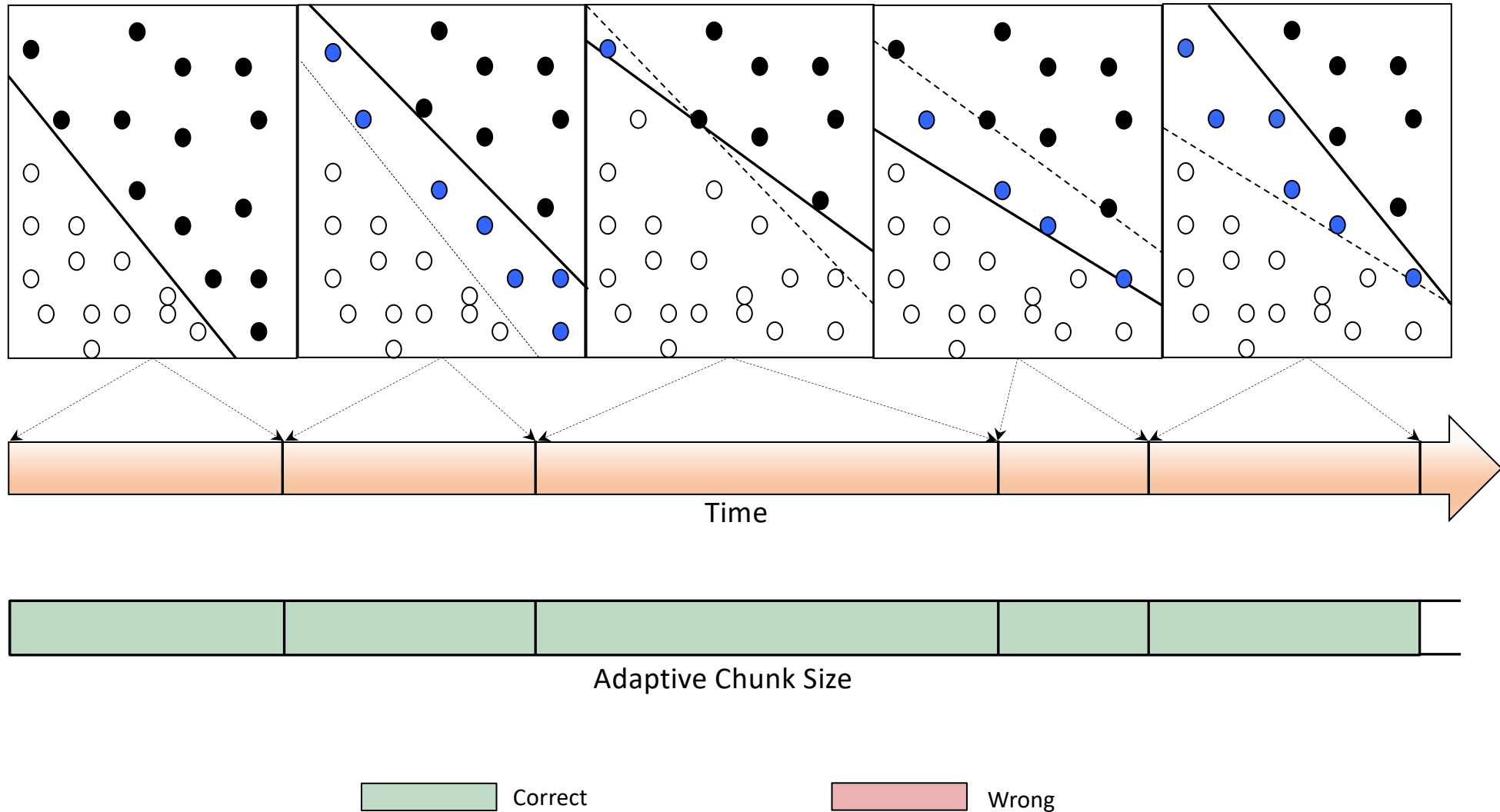
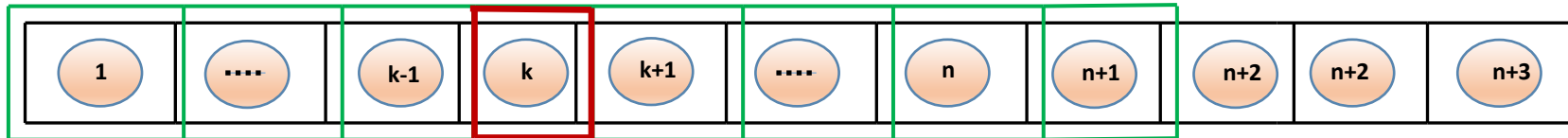# Challenges: Fixed Chunk Size

Concept Drifts

Chunk size too large – Delayed reaction

Chunk size too small – Performance issue

Correct   Wrong
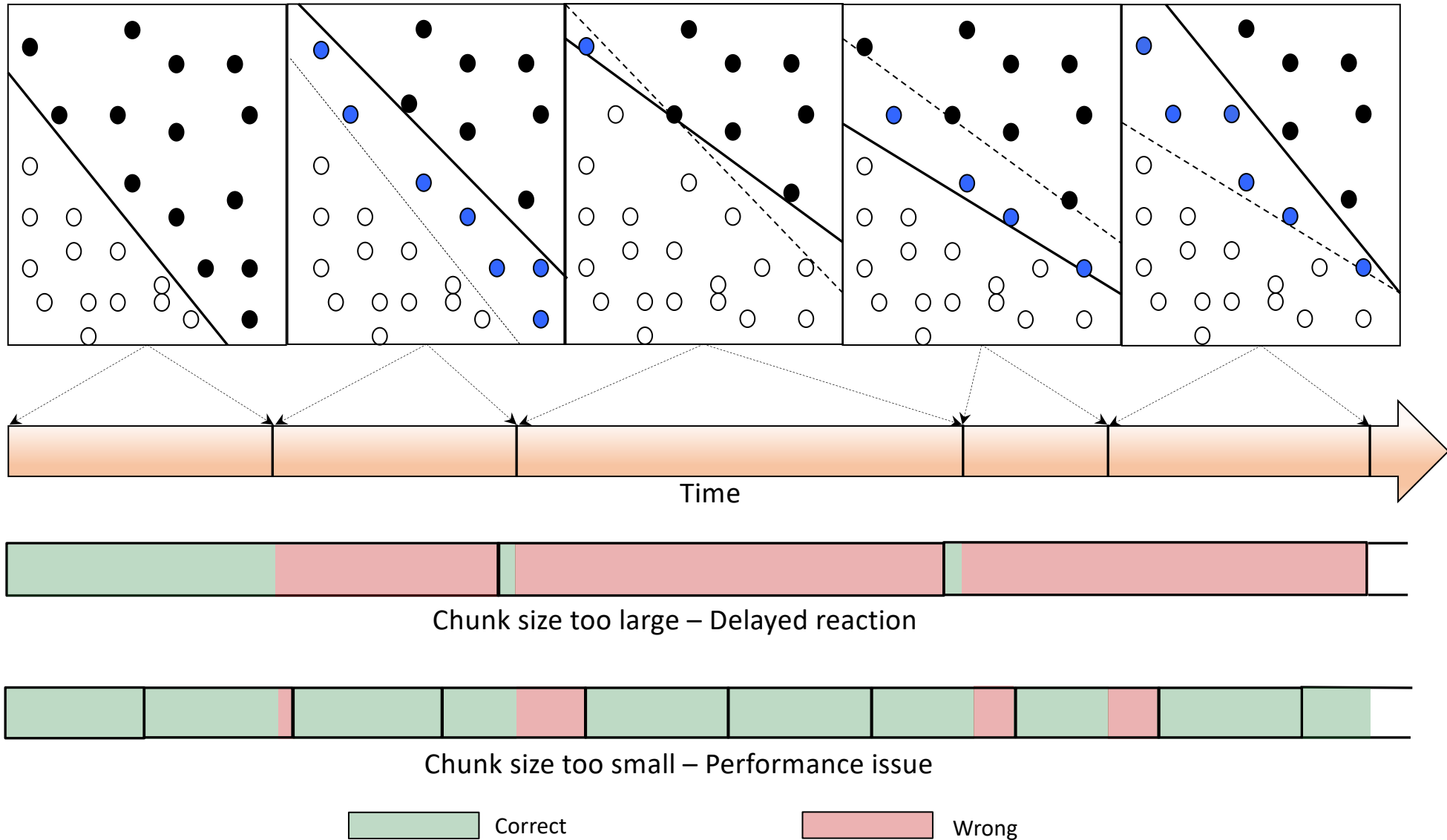
Solution: Adaptive Chunk Size

Concept Drifts

Adaptive Chunk - Sliding Window

Gamma et al. [1], *Bifet et al.* [2], *Harel et al.* [3]

> Existing dynamic sliding window techniques
- Monitor error rate of the classifier.
- Update classifier if starts to show bad performance.
- **fully supervised**, which is not feasible in case of real-world data streams.

[1] João Gama, Gladys Castillo: Learning with Local Drift Detection. ADMA 2006: 42-55
[2] Albert Bifet, Ricard Gavaldà: Learning from Time-Changing Data with Adaptive Windowing. SDM 2007: 443-448
[3] Maayan Harel, Shie Mannor, Ran El-Yaniv, Koby Crammer: Concept Drift Detection Through Resampling. ICML 2014: 1009-1017

Adaptive Chunk - Unsupervised
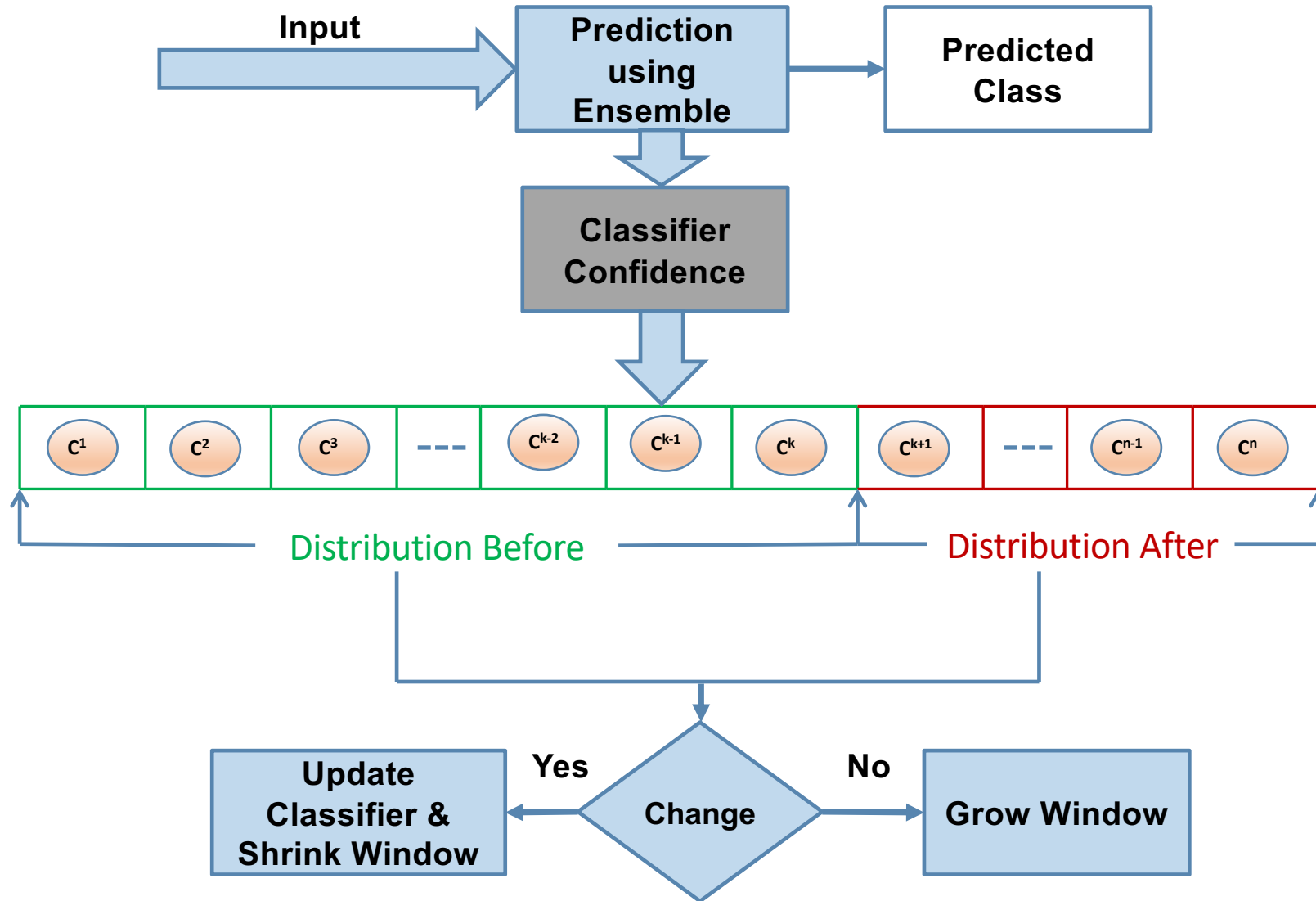
Haque et al. [1][2]

[1] Ahsanul Haque, Latifur Khan, Michael Baron, Bhavani M. Thuraisingham, Charu C. Aggarwal: Efficient handling of concept drift and concept evolution over Stream Data. ICDE 2016: 481-492.
[2] Ahsanul Haque, Latifur Khan, Michael Baron: SAND: Semi-Supervised Adaptive Novel Class Detection and Classification over Data Stream. AAAI 2016: 1652-1658.

Adaptive Chunk - Unsupervised
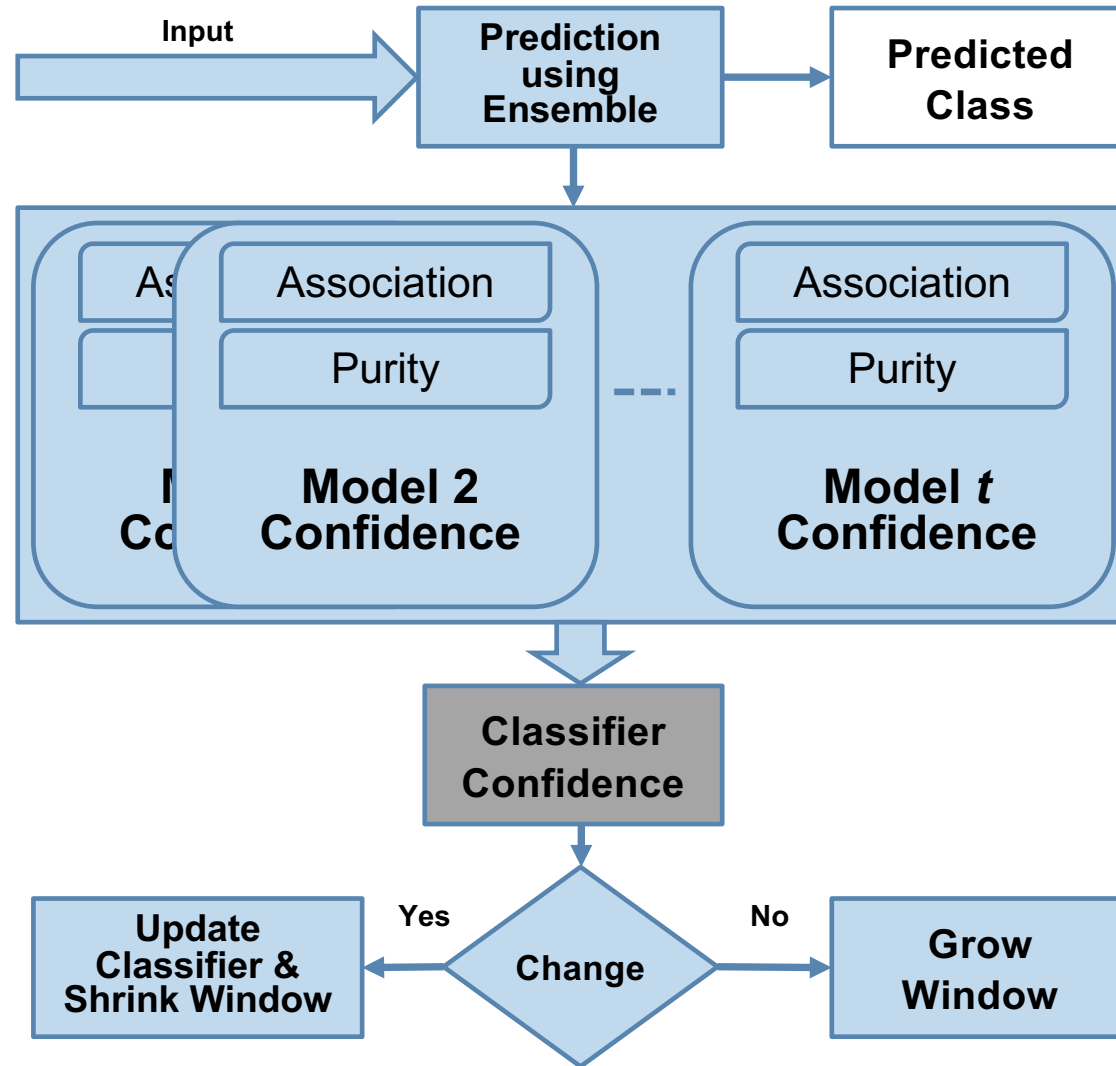
Haque et al. [1][2]

[1] Ahsanul Haque, Latifur Khan, Michael Baron, Bhavani M. Thuraisingham, Charu C. Aggarwal: Efficient handling of concept drift and concept evolution over Stream Data. ICDE 2016: 481-492
[2] Ahsanul Haque, Latifur Khan, Michael Baron: SAND: Semi-Supervised Adaptive Novel Class Detection and Classification over Data Stream. AAAI 2016: 1652-1658.

Confidence of a model

➢ For each testing instance x:

- Confidence for i^{th} model, $c_i^x = \boldsymbol{h_i^x . z_i}$

  - $\boldsymbol{h_i^x} = (a_i{}^x, p_i{}^x)$ is a vector of estimator values on test instance x.
  - $\boldsymbol{z_i}$ = vector containing weights of the estimators for i^{th} model.

➢ To estimate confidence of the entire ensemble, we take the average confidence of the models towards the predicted class.

Confidence Estimators

➢ Let h be the closest cluster from data instance x in model M_i, confidence of M_i in classifying instance x is calculated based on the following estimators:

> ➢ **Association:** $a_i^x = R_h - D_i(x)$, *where* R_h
> *is the radius of h and* $D_i(x)$ *is the distance of* x
> *from h.*

> ➢ Purity: $p_i^x = N_m / N_s$, where N_s is the number of labeled instances in h, and N_m is the number of instances from the majority class in h.
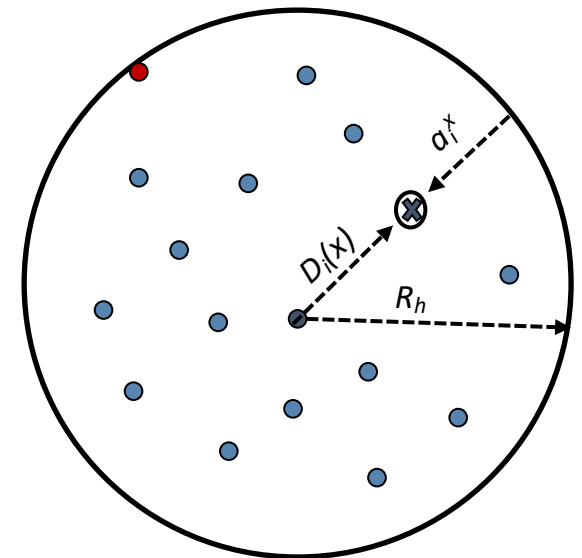
- $N_s = 15$, $N_m = 14$
- $p_i^x = 14/15$

How good are the estimators?

1. For each model M_i, training instance k, compute $\boldsymbol{h_i^k} = (a_i^k, p_i^k)$.

   ➤ a_i^k = association of model M_i on instance k.

   ➤ p_i^k = purity of model M_i on instance k.

Calculation of z_i^a from g_i and a_i.

2. Calculate g_i^k values.

   ➤ g_i^k = 1 if $\hat{y}_i^k = y_i^k$, and g_i^k = 0 if $\hat{y}_i^k \mathrel{!=} y_i^k$.

3. Calculate $\boldsymbol{z_i} = (z_i^a, z_i^p)$ which is a vector of correlation values.

   ➤ z_i^a = correlation between a_i and g_i

   ➤ z_i^p = correlation between p_i and g_i

4. $\boldsymbol{z_i}$ is scaled so that $z_i^a + z_i^p$ = 1.
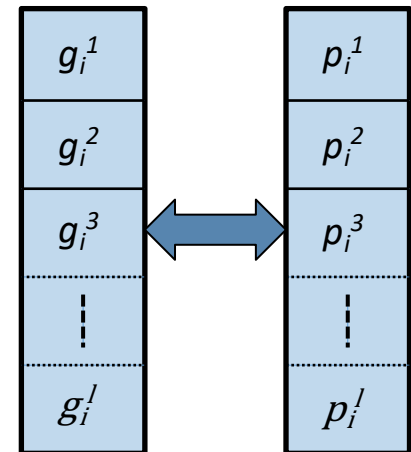
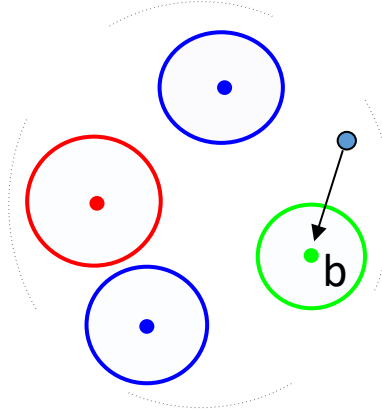Calculation of z_i^p from g_i and p_i.

Example: Confidence Calculation
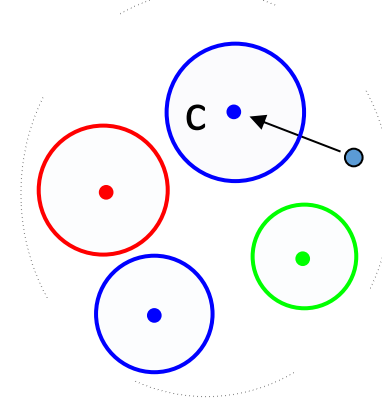
❖ An example with t = 3 (3 models) and C = 3 (3 classes)

Classification Model (M_1)   Model (M_2)   Model (M_3)

Test instance (x)
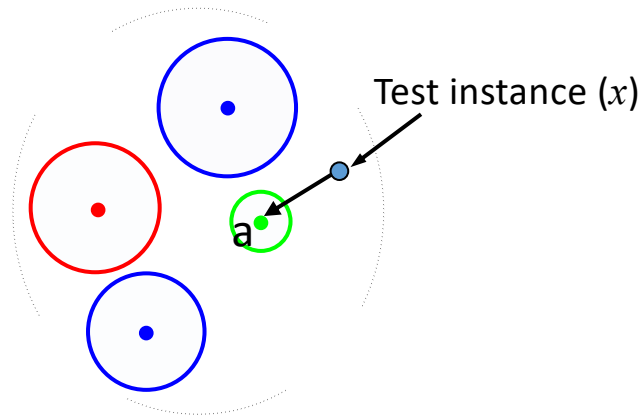
The nearest micro-cluster to x in M_1, M_2 and M_3 are a, b, c, respectively

Training:

Z_1 = (0.52, 0.48)

Z_2 = (0.41, 0.59)

Z_3 = (0.36, 0.64)

Testing

h_1^x = (0.85, 0.75), h_2^x = (0.78, 0.79), h_3^x = (0.25, 0.12)

c_1^x = 0.52 * 0.85 + 0.48 * 0.75 = 0.81
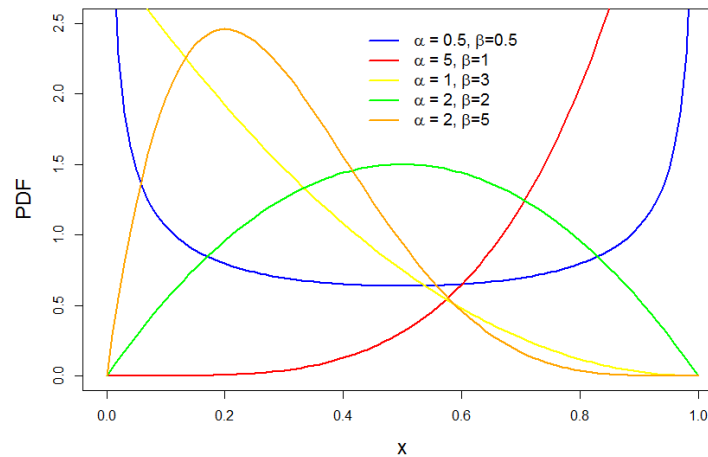
c_2^x = 0.41 * 0.78 + 0.59 * 0.79 = 0.79

c_3^x = 0.36 * 0.25 + 0.64 * 0.12 = 0.17

Ensemble Confidence = (0.81 + 0.79) / 3 = 0.53

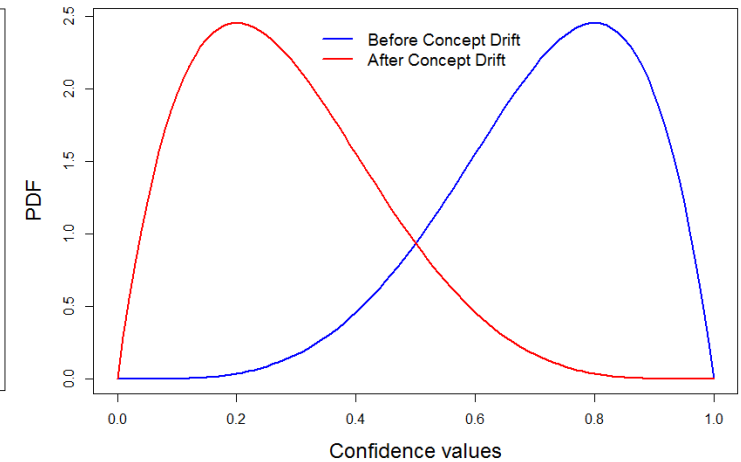# Confidence Value Distribution

$$f(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)},$$
where B(.) is Beta function

Beta Distribution

Change in Confidence Value Distribution

➢ Beta distribution has two parameters, i.e., α and β.
- distribution is symmetric if $\alpha = \beta$ and unimodal if $\alpha, \beta > 1$.
- approaches infinity at 0 if $\alpha < 1$.
- approaches infinity at 1 if $\beta < 1$.

Multistream Classification and Regression

Ahsanul Haque, Hemeng Tao, Swarup Chandra, Jie Liu, Latifur Khan:
A Framework for Multistream Regression With Direct Density Ratio Estimation. AAAI 2018

Bo Dong, Yifan Li, Yang Gao, Ahsanul Haque, Latifur Khan, Mohammad M. Masud:
Multistream regression with asynchronous concept drift detection. BigData 2017: 596-605

Ahsanul Haque, Zhuoyi Wang, Swarup Chandra, Bo Dong, Latifur Khan, Kevin W. Hamlen:
FUSION: An Online Method for Multistream Classification. CIKM 2017: 919-928

Ahsanul Haque, Swarup Chandra, Latifur Khan, Kevin W. Hamlen, Charu C. Aggarwal:
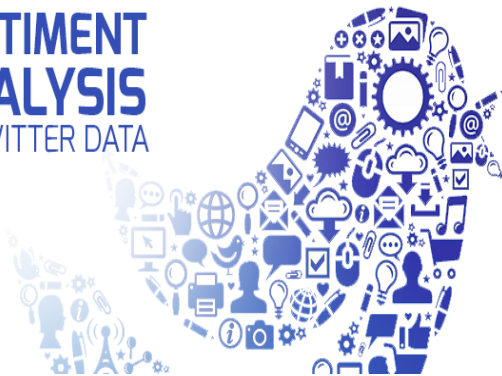**Efficient Multistream Classification Using Direct Density Ratio Estimation.** ICDE 2017

Bias Between Training and Test Dataset

- Classification:
  - Predict Sentiment Label of Twitter Users (sentiment, label).
- Regression:
  - Predict Flight Delay Time (carriers, airport location, departure time, arrival time, distance).
  - Predict Gas Price (time, weather condition, location).

Sentiment Analysis

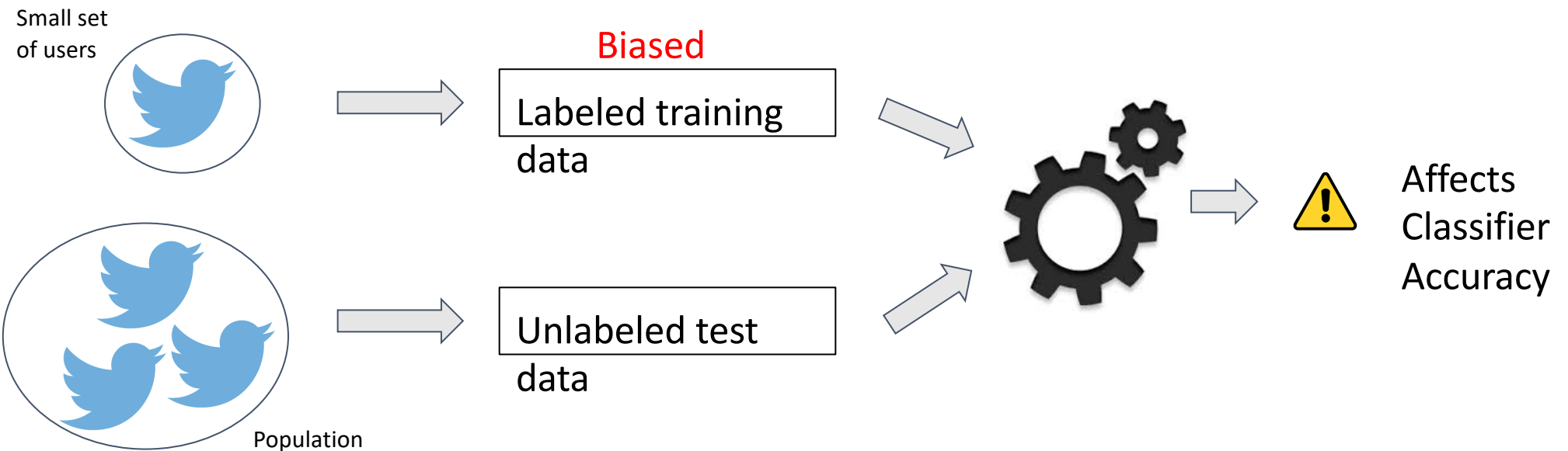

Flight Delay
Time

Gas
Price

Motivation for Biased Data
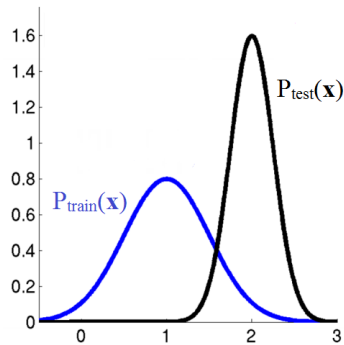

Example Scenario

Small set of users

Population

Biased

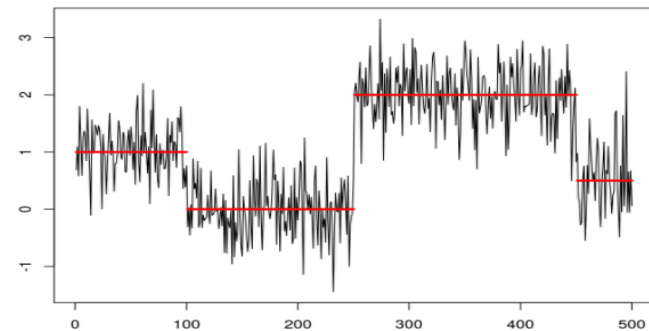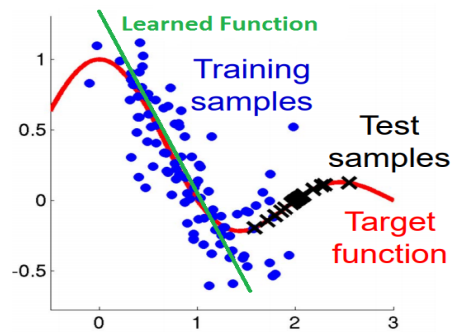Labeled training data

Unlabeled test data

Affects Classifier Accuracy

Challenges

- Difference in training (source) and test (target) distribution.
- Covariate Shift: $P_{train}(y|\boldsymbol{x}) = P_{test}(y|\boldsymbol{x})$, but $P_{train}(\boldsymbol{x}) \neq P_{test}(\boldsymbol{x})$.
- Concept Drift: $P_{train}(y, \boldsymbol{x}) \neq P_{test}(y, \boldsymbol{x})$.
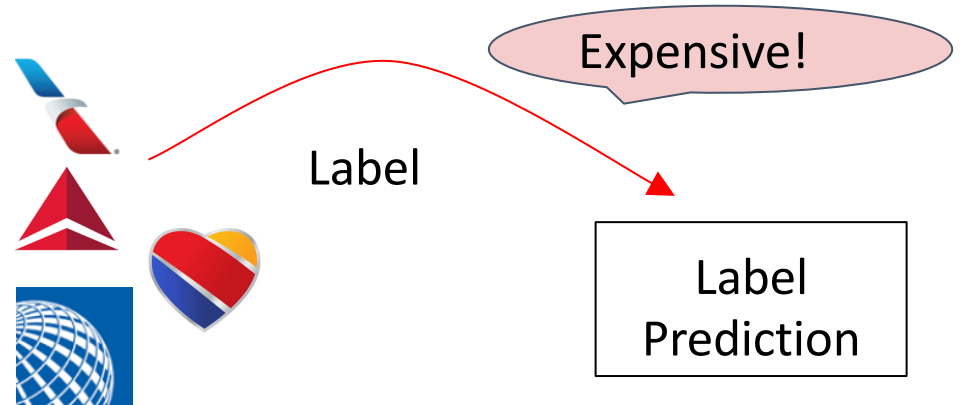

Covariate Shift

Concept Drift

❑ Masashi Sugiyama, Taiji Suzuki, Takafumi Kanamori: Density Ratio Estimation in Machine Learning. Cambridge University Press 2012, ISBN 978-0-521-19017-6, pp. I-XII, 1-329.

Motivation

- What if we do not find a good training set?
  - Varies locations may cause biased training data with respect to test data.
  - Sufficient labeled data is sparingly available in many real-world applications.

Label

Expensive!

Label Prediction

Motivation (Multistream Classification)

➢ Two types of data stream (independent).

Source Stream → Labeled Data → t → Create training data → t

Target Stream → Unlabeled Data → t

Unlabeled Data → $t+1$

→ Label Prediction

Motivation (Multistream Classification)

➢ Two types of data stream (independent).

Source Stream → Labeled Data → t → Create training data → t → (gears) → Label Prediction

Target Stream → Unlabeled Data → t → Create training data; t+1 → (gears)
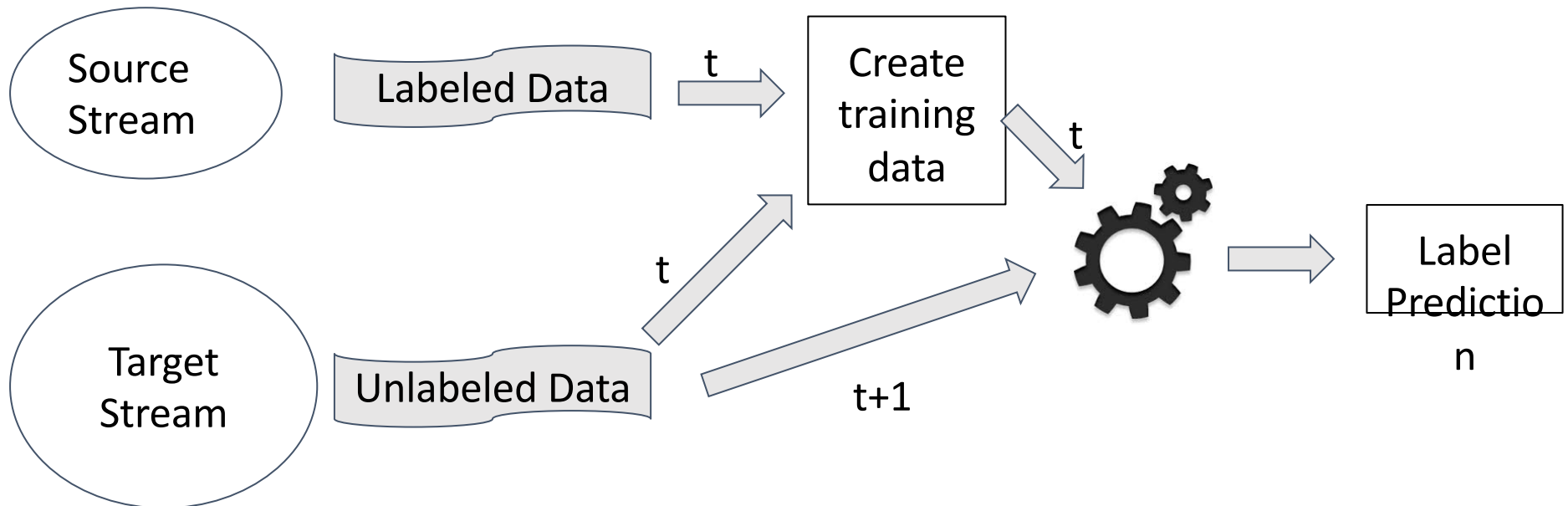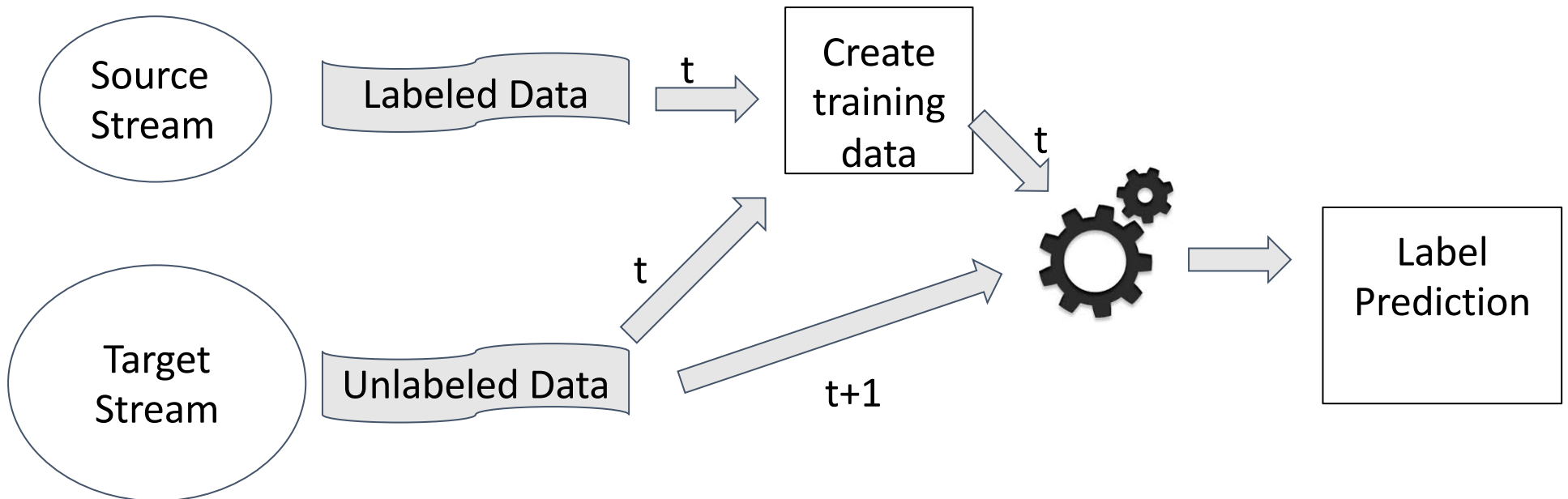
Challenges

➢ Asynchronous concept drift in source and target stream.
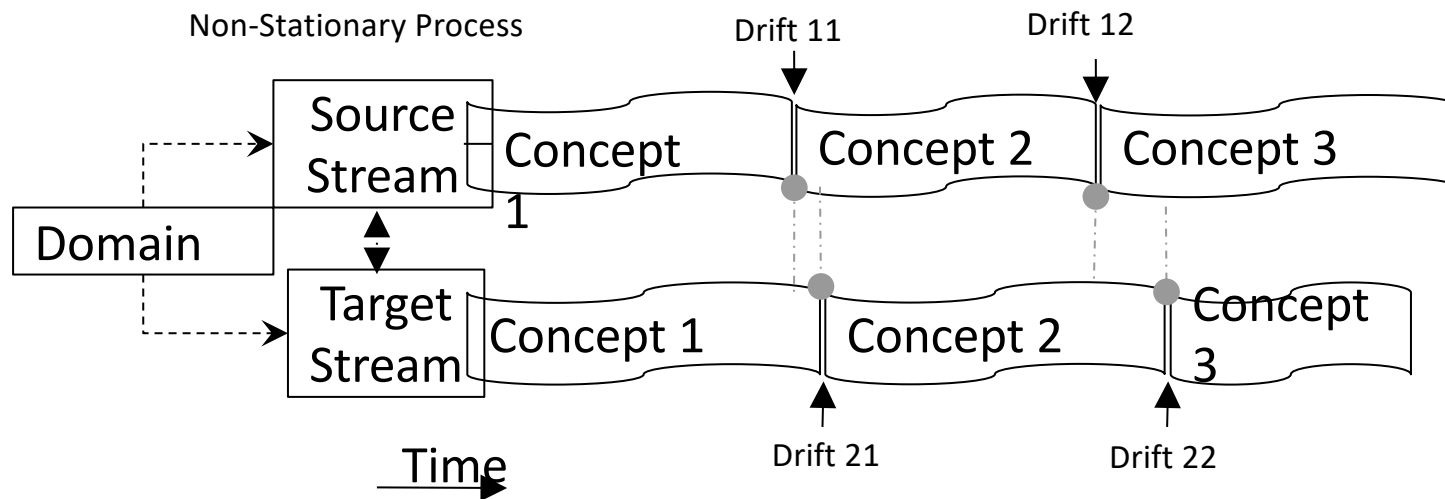
❑ Swarup Chandra, Ahsanul Haque, Latifur Khan, and Charu Aggarwal. An Adaptive Framework for Multistream Classification. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16). Indianapolis, Indiana, USA, 1181–1190.

Challenges

➢ Can the two streams be combined?
  ➢ Data distributions of two streams are different.
  ➢ Combining two streams together usually assumes that they follow the same distribution.
  ➢ Separate representation has advantages when multiple sources are present.

Related Work: Limitations

1. Deal with fixed size of data.
   - Kernel Mean Matching(KMM), KLIEP and unconstrained Least Square Importance Fitting(uLSIF).
   - Handling covariate shift for fixed-size source and target data.

2. Applied to single data stream.
   - Extended KLIEP deduce direct online density ratio estimation.

3. Change detection after receiving new instance in source or target stream.
   - Cubic time complexity.
   - Expensive!

- T. Kanamori, S. Hido, and M. Sugiyama, "A least-squares ap- proach to direct importance estimation," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1391–1445, 2009.
- Y. Kawahara and M. Sugiyama, "Sequential change-point detection based on direct density-ratio estimation," *Statistical Analysis and Data Mining*, vol. 5, no. 2, pp. 114–127, 2012.
- M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Advances in neural information processing systems*, 2008, pp. 1433–1440.

Kernel Mean Matching(KMM)

➢ Minimize mean distance between weighted training data distribution and test data distribution.

➢ Density ratio (Importance weight): $\beta(X) = \dfrac{P_{te}(X)}{P_{tr}(X)}$

> Requires complete training and test data to be in the memory

➢ Maximum Mean Discrepancy

$$\left\| E_{X \sim P_{tr}(X)}[\beta(X)\emptyset(X)] - E_{X \sim P_{te}(X)}[\emptyset(X)] \right\|$$

➢ Empirical $\hat{\beta} \approx \min_{\beta} \dfrac{1}{2}\beta^T K \beta - k^T \beta$ $\qquad subject$

$subject\ to\ \ \beta(X^i) \in [0, B], \forall\, i\, \in \{1 \dots n_{tr}\}$

$\left| \sum_{i=1}^{n_{tr}} \beta(X^i) - n_{tr} \right| \le n_{tr}\varepsilon$

$K^{ij} = h(X_{tr}^i, X_{tr}^j) \qquad k^i = \dfrac{n_{tr}}{n_{te}} \sum_{j=1}^{n_{te}} h(X_{tr}^i, X_{te}^j)$

➢ If $\beta(X) \to 1$, training data distribution and test data distribution are close.

➢ If $\beta(X) \to 0$ or $\beta(X) \to B$ training data distribution is very different from test data distribution, here B is the upper bound on the solution search space.
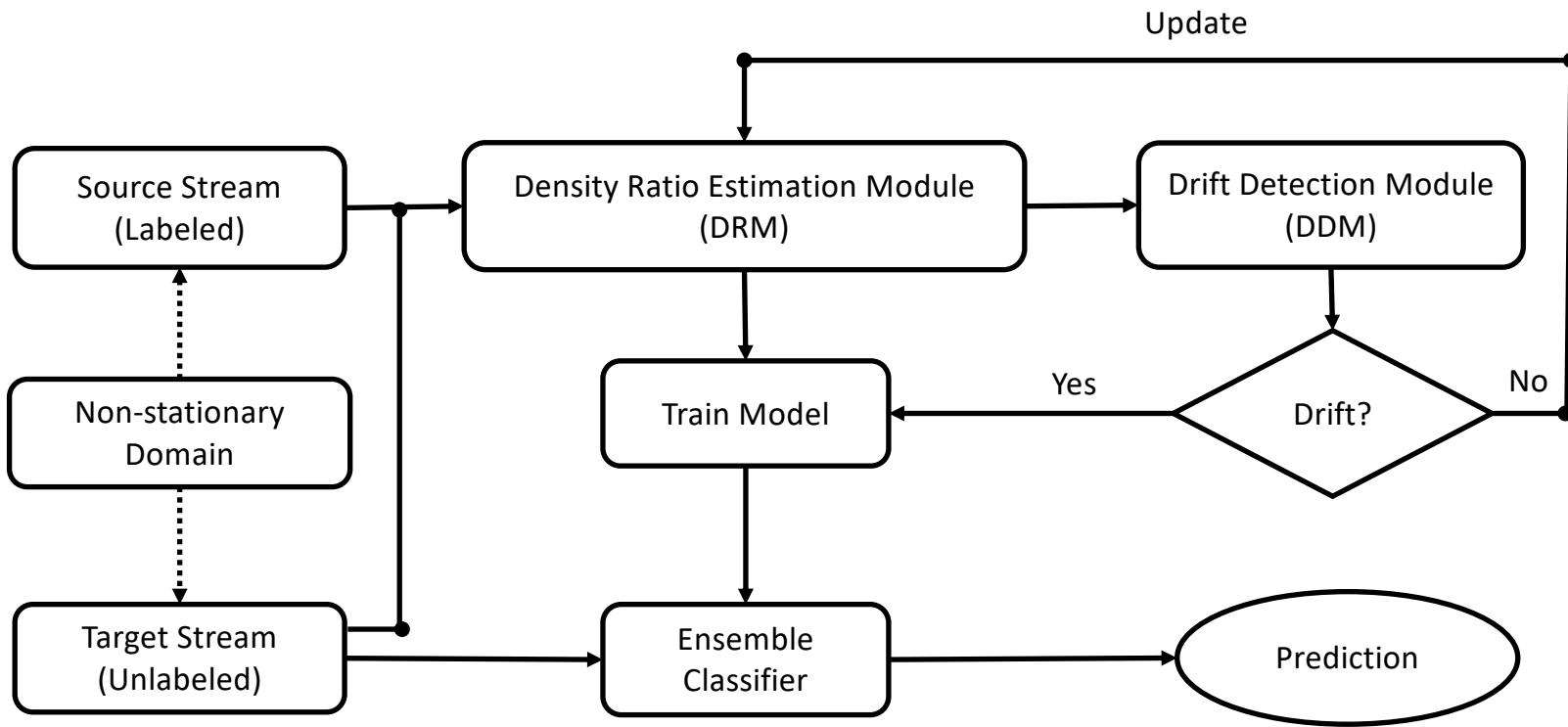
The Proposed Approach: FUSION

Figure: Overview of FUSION (eEfficient mUltiStream classification using direct densIty ratio estimatiON)

FUSION: Density Ratio Estimation Module

➤ Importance weight is estimated using a Gaussian Kernel Model.

 ➤ $\hat{\beta}(\boldsymbol{x}) = \sum_{i=1}^{N_m} \alpha_i K_\sigma\left(\boldsymbol{x}, \boldsymbol{W}_T^{(i)}\right); K_\sigma(\boldsymbol{x}, \boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\sigma^2}}$

 ➤ Test sliding window instances are used as the Gaussian Kernel centers.

➤ Parameter estimation:

 ➤ Minimize **Kullback–Leibler** divergence from $P_T(\boldsymbol{x})$ to $\hat{\beta}(\boldsymbol{x})P_S(\boldsymbol{x})$.

 ➤ $\max_{\{\alpha_i\}_{i=1}^{N_m}} \left[\sum_{j=1}^{N_m} \log\left(\sum_{i=1}^{N_m} \alpha_i K_\sigma\left(\boldsymbol{W}_T^{(j)}, \boldsymbol{W}_T^{(i)}\right)\right)\right]$

> Gradient Ascent

 such that $\frac{1}{N_m}\sum_{j=1}^{N_m}\sum_{i=1}^{N_m} \alpha_i K_\sigma\left(\boldsymbol{W}_S^{(j)}, \boldsymbol{W}_T^{(i)}\right) = 1$, and $\alpha_1, \alpha_2, \ldots, \alpha_{N_m} \geq 0$.

➤ Online Update:

> Stochastic Gradient Ascent

 ➤ New source instance: satisfy constraints.

 ➤ New target instance: $\begin{cases} \hat{\alpha}_i \leftarrow (1-\eta\lambda)\hat{\alpha}_{i+1}; & i = 1, \ldots, N_m - 1 \\ \hat{\alpha}_i \leftarrow \frac{\eta}{\hat{\beta}\left(\boldsymbol{W}_T^{(N_m-1)}\right)}; & i = N_m \end{cases}$

η = Learning Factor
λ = Regularization Constant

❑ Yoshinobu Kawahara, Masashi Sugiyama: Sequential change-point detection based on direct density-ratio estimation. Statistical Analysis and Data Mining 5(2): 114-127 (2012).

FUSION: Ensemble Classifier

➢ Ensemble of L classification models.

➢ The models are trained using weighted source instances.

➢ $\hat{\beta}\left(\boldsymbol{W}_S^{(i)}\right) = \sum_{j=1}^{N_m} \alpha_j K_\sigma\left(\boldsymbol{W}_S^{(i)}, \boldsymbol{W}_T^{(j)}\right)$; i $= 1, \dots, N_m$.

➢ Any learning algorithm that incorporates importance weight of training instances can be used.

➢ We used SVM in our experiments.

➢ Ensemble Maintenance:

➢ First model is trained on the warm-up period instances.

➢ If a drift, i.e., **a significant difference between** $P_T(\boldsymbol{x})$ and $\hat{\beta}(\boldsymbol{x})P_S(\boldsymbol{x})$ **is detected:**

➢ The parameters for the Gaussian kernel model are estimated again.

➢ A new model is trained, and the ensemble is updated.

FUSION: Drift Detection Module

➤ $P_T(x)$ is estimated by $\hat{P}_T(x) = \hat{\beta}(x)P_S(x)$.

➤ Drift detection:

  ➤ Significant change between $P_T(x)$ and $\hat{\beta}(x)P_S(x)$.

  ➤ A change score is calculated:

  ➤ $S = \sum_{i=1}^{N_m} \ln \frac{P_T\left(w_T^{(i)}\right)}{\hat{\beta}_0 P_S\left(w_T^{(i)}\right)} = \sum_{i=1}^{N_m} \ln \frac{\hat{\beta}_t\left(w_T^{(i)}\right)}{\hat{\beta}_0\left(w_T^{(i)}\right)}$

  ➤ Where $\hat{\beta}_0$ and $\hat{\beta}_t$ are defined by initial $\{\alpha\}_{i=1}^{N_m}$ and at time t respectively.

  ➤ A drift is detected if $S \geq -\ln(\tau)$.

➤ Performance analysis:

  ➤ Data shift and data drift is addressed using the same Gaussian Kernel model.

  ➤ Time Complexity: $O\left(N_m^2 + f(N_m)\right)$.

  ➤ Space Complexity: $O(N_m^2)$.

Empirical Evaluation: Dataset

Dataset	# features	# classes	# instances
ForestCover [1]	54	7	150,000
KDD [3]	42	23	200,000
PAMAP [2]	52	19	150,000
Electricity [1]	8	2	45,311
SynRBF@002-1 [1]	50	5	100,000
SynRBF@002-2 [1]	70	7	100,000
SynRBF@003 [1]	70	7	100,000

Real World — ForestCover, KDD, PAMAP, Electricity

Synthetic — SynRBF@002-1, SynRBF@002-2, SynRBF@003

Divide dataset into Source (10%) and Target Stream (90%), with bias in source stream data selection according to: $e^{-|x-\bar{x}|^2}$

[1] Moa massive online analysis-real time analytics for data streams repository data sets. http://moa.cms.waikato.ac.nz/datasets/.

[2] Reiss, A., Stricker, D.: Introducing a new benchmarked dataset for activity monitoring. In: ISWC, [4] Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Philipp Kranen, Hardy Kremer, Timm Jansen, and Thomas Seidl, Moa: Massive online analysis, a framework for stream classification and clustering, Journal of Machine Learning Research, 2010, pp. 44-50.

[3] M. Lichman. 2013. UCI Machine Learning Repository. (2013). http://archive.ics.uci.edu/ml

Empirical Evaluation: Classification

Name of Data Set	FUSION		MSC [1]		AHT [2]		SVM	
	Accuracy	Time (Seconds)	Accuracy	Time (Seconds)	Accuracy	Time (Seconds)	Accuracy	Time (Seconds)
ForestCover	**85.10**	469.89	84.4	270.57	61.52	0.05	69.29	8.78
KDD	**97.30**	417.85	96.80	451.54	97.2	0.05	96.29	10.0
PAMAP	**99.80**	471.99	97.40	564.56	94.95	0.08	88.04	7.54
Electricity	**76.50**	238.33	74.60	601.08	75.02	0.02	73.37	0.09
SynRBF@0.002-1	**98.10**	415.22	93.60	533.33	85.58	0.07	86.29	8.51
SynRBF@0.002-2	**96.20**	561.86	69.80	232.34	83	0.13	44.13	7.72
SynRBF@0.003	**93.10**	591.18	58.30	194.49	80.11	0.12	41.28	8.75

[1] Swarup Chandra, Ahsanul Haque, Latifur Khan, and Charu Aggarwal. An Adaptive Framework for Multistream Classification. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16). Indianapolis, Indiana, USA, 1181–1190.
[2] Albert Bifet, Ricard Gavaldà. Adaptive Learning from evolving data streams - Advances in Intelligent Data Analysis VIII, 2009.

Multistream Classification for Cyber Threat Data with Heterogeneous Domain Adaptation*
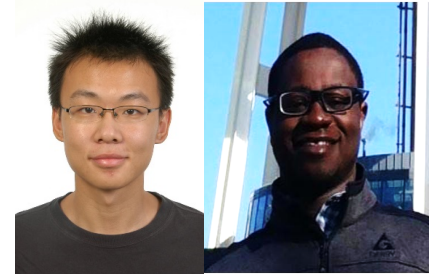
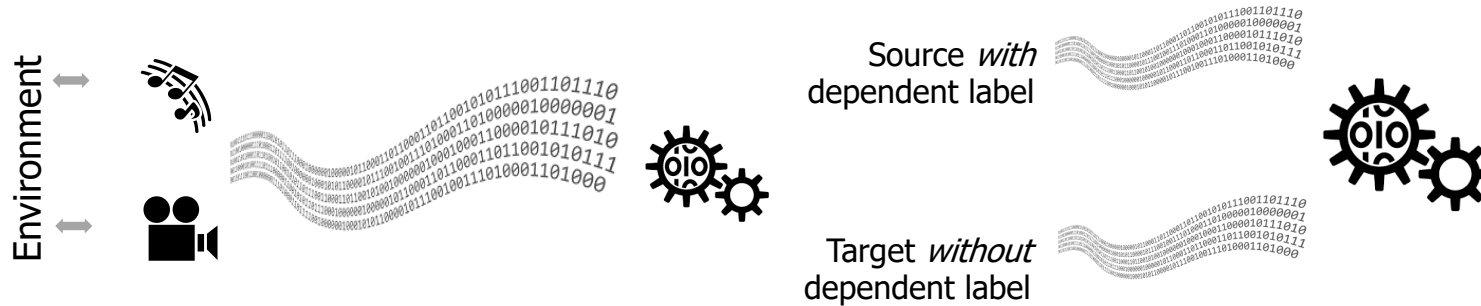Yi-Fan Li, Yang Gao, Gbadebo Ayoade, Hemeng Tao, Latifur Khan, Bhavani Thuraisingham

Big Data Mining and Analytics Lab

The University of Texas at Dallas

Motivations

Amazon Reviews
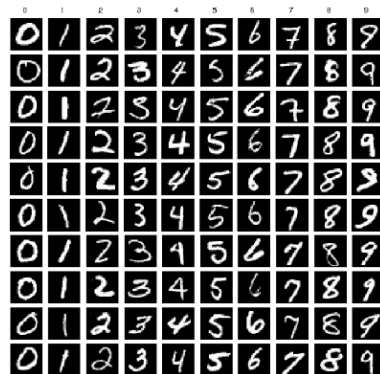

Environment

Source *with* dependent label

Target *without* dependent label

✓ Require model update
✓ Need truth labels

✓ Leverage available truth label
✓ Predict on target data

Image Recognition

Motivation

Challenges:

- **Heterogeneous Domains*;**

- **Concept Drift;**

Basic idea

To solve this problem, first we project the heterogeneous domains to a common latent feature space, while keeping the structure of data from source and target domain unchanged

Projection of source and target data to latent feature space

Classification using any classifiers
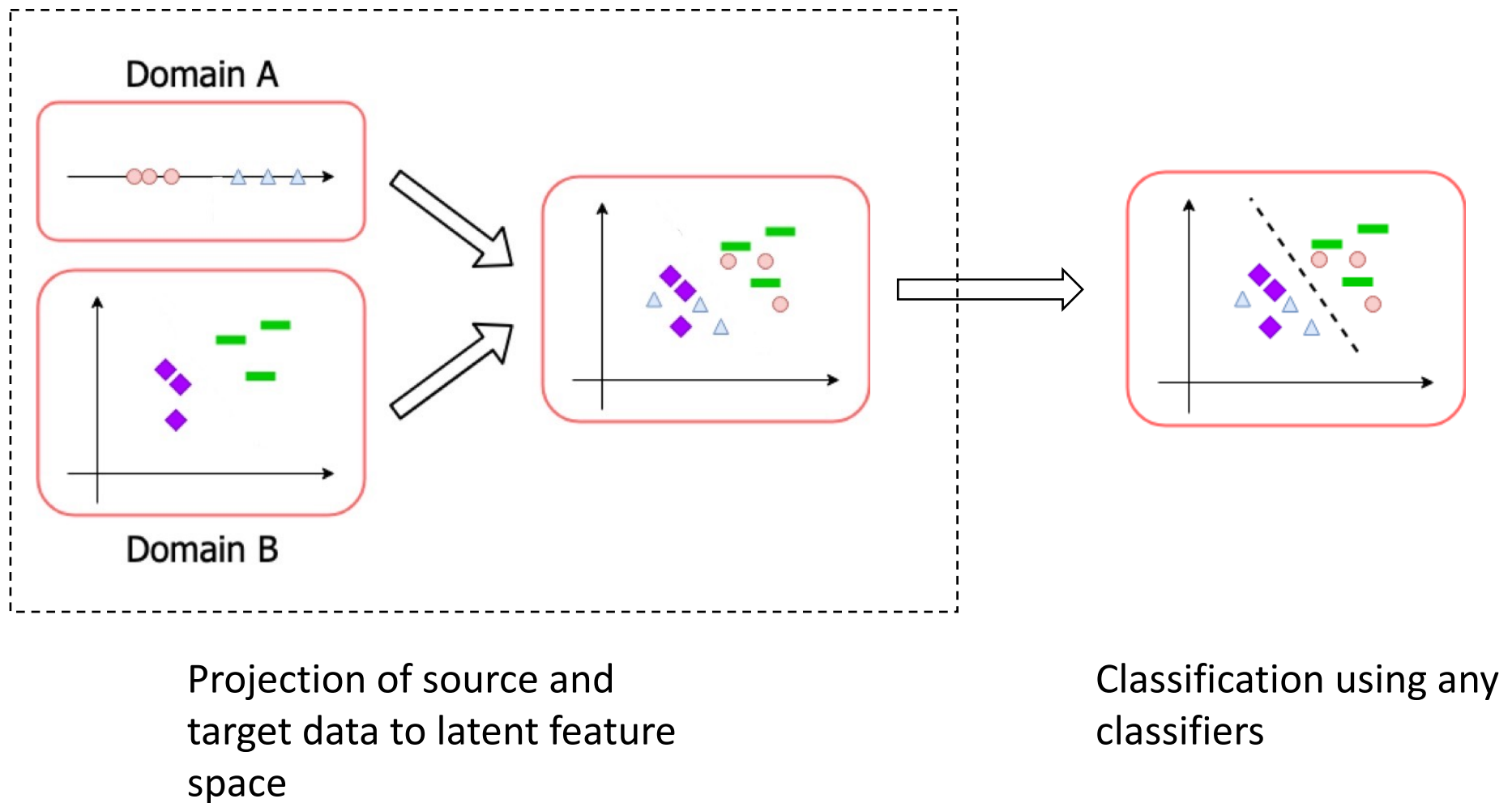
Proposed Method

Algorithm 1 MSDA Algorithm

Require: Labeled source stream S, Unlabeled target stream T, The size of sliding window N_m. Similarity parameter β.

Ensure: Labels predicted on T.

1: /* Initialization */
2: $B_s, B_t \leftarrow readData(S, T)$
3: /* DA for initial buffer */
4: $W_s, W_t \leftarrow genProjectFunction(B_s, B_t, \beta)$ → Initialization
5: $L_s, L_t \leftarrow genProjectMatrix(B_s, B_t, W_s, W_t)$
6: $M \leftarrow buildModel(L_s, Y_s)$
7: **while** S or T exists **do**
8:     $B_s, B_t \leftarrow readData(S, T)$
9:     /* Concept drift detection and correction */ → Drift Detection
10:     Call $ChageDetection$ /* Algorithm 2 */
11:     **if** $ChangeDetection = True$ **then**
12:         /* Update prediction model */
13:         /* DA for stream buffer */
14:         $W_s, W_t \leftarrow genProjectFunction(B_s, B_t, \beta)$ → Domain Adaptation
15:         $L_s, L_t \leftarrow genProjectMatrix(B_s, B_t, W_s, W_t)$
16:         $M \leftarrow buildModel(L_s, Y_s)$
17:     /* Generate predictions */
18:     $\hat{y}_t \leftarrow getPrediction(M, L_t)$ → Prediction

---

Loop, while exist

Proposed Method

Domain Adaptation:

Overall Loss: (Minimize Ws and Wt)

Minimize the loss during projection which help preserve structure of data between original space (source and target) and the latent space

Penalty term: minimize the loss between projected data, which aligns the distributions between data projected from source and target data (cross domain projection)

$$O = \min_{L_s, L_t} \ell(B_s, L_s) + \ell(B_t, L_t) + \beta \mathbf{D}(L_s, L_t)$$

Reconstruct Loss

Regularization

$$O = \|B_s - L_s W_s\|_F^2 + \|B_t - L_t W_t\|_F^2$$
$$+ \frac{1}{2}\beta \|B_s - L_t W_s\|_F^2 + \frac{1}{2}\beta \|B_t - L_s W_t\|_F^2$$

Projection: Use Ws and Wt to get Ls and Lt

$$\begin{cases} L_s^{(i)} = B_s^{(i)} W_s^+ & B_s^{(i)} \in B_s \\ L_t^{(j)} = B_t^{(j)} W_t^+ & B_t^{(j)} \in B_t \end{cases}$$

Where Ls and Lt are projected instances
Bs and Bt are source/target instances
Ws+ and Wt+ are inverse matrix of W

Security Domain Implementation

Data Collection System
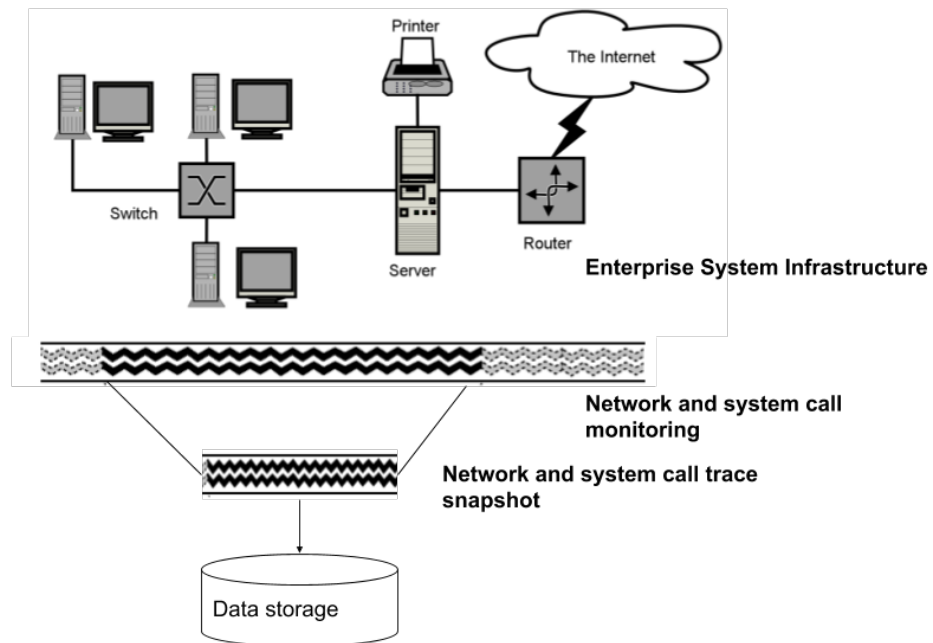
Security Domain Implementation

System Call Features

- Collection of OS events (e.g., open, read, select)
- Linux x86_64 systems: About 314 distinct system call events
- For example, a sequence of system call could be open, open, read, close

1-gram will be open, open, read, close
Bi-gram: {open,open},{open,read},{read,close}
Tri-gram: {open,open,read}, {open,read,close}, etc.

Data Type	# of distinct features (seen in the dataset)	Num of instances
Win syscalls	28	10,870
Linux syscalls	127	10,801

Security Domain Implementation

Dark Web Features

- Determine words that are products such as malware, rootkits, credit card number
- Extracted Penn Part of Speech Tags and Stanford postagger system
- Generate feature vectors based on the corresponding part of speech tag. e.g., is word Noun, pronoun or verb?
- Word position in the blog or marketing post.

Darknet Forum	Num of features	Num of instances
Blackhat	38	32,414
Nulled	38	9,930
Darkode	67	100,000
Hack	67	100,000

Results

Datasets:
- Cyber Security: System call (Windows, Linux), Darkweb Forum (Blackhat, Darkode, Nulled, Hack)
- Non Cyber Security: Digits (USPS, MNIST), Amazon Review (Music, DVD, Video)
- Synthetic data: Syn01, Syn02
- Results are reported in terms of Error

Data type	Dataset	OTL	HeMap-S	HeMap-L	MSDA-S	MSDA-L
Cyber security	Win → Linux	28.53 ± 0.88	30.55 ± 0.86	29.08 ± 0.58	23.73 ± 0.62	**21.33 ± 1.62**
	BlackHat → Darkode	22.42 ± 0.85	24.70 ± 1.00	25.36 ± 1.00	22.12 ± 1.29	**20.82 ± 0.64**
	Nulled → Hack	28.86 ± 1.54	26.78 ± 0.95	29.26 ± 0.50	25.47 ± 1.06	**24.75 ± 1.36**
Non cyber security	USPS → MNIST	32.31 ± 0.94	38.19 ± 0.75	39.04 ± 0.68	**28.72 ± 0.56**	29.96 ± 0.64
	Music → DVD	33.54 ± 0.69	36.06 ± 0.51	35.85 ± 0.77	32.91 ± 0.49	**31.64 ± 0.57**
	Video → Music	38.01 ± 1.03	40.41 ± 0.65	39.72 ± 0.85	**31.76 ± 0.72**	32.02 ± 0.75
	Video → DVD	35.88 ± 0.88	40.28 ± 0.52	40.09 ± 0.61	**32.35 ± 0.49**	33.84 ± 0.65
Synthetic data	Syn01 → Syn02	37.53 ± 1.37	41.83 ± 0.83	42.12 ± 1.06	**33.50 ± 0.97**	35.47 ± 0.92

State-of-the-art Benchmark — Our Method

GCI: A Transfer Learning Approach for Detecting Cheats of Computer Game

Motivation

➢ **Detecting cheats:**

➢ Part of players deploy cheats while playing, either for profit or for fun.
➢ Cheating in massive multiple online games (MMOGs) adversely affect the game's popularity and reputation among its users.

➢ **Limited client-side information:**

➢ Detecting cheats is challenging mainly due to the limited client-side information.
➢ The cheating techniques are unknown and complex.
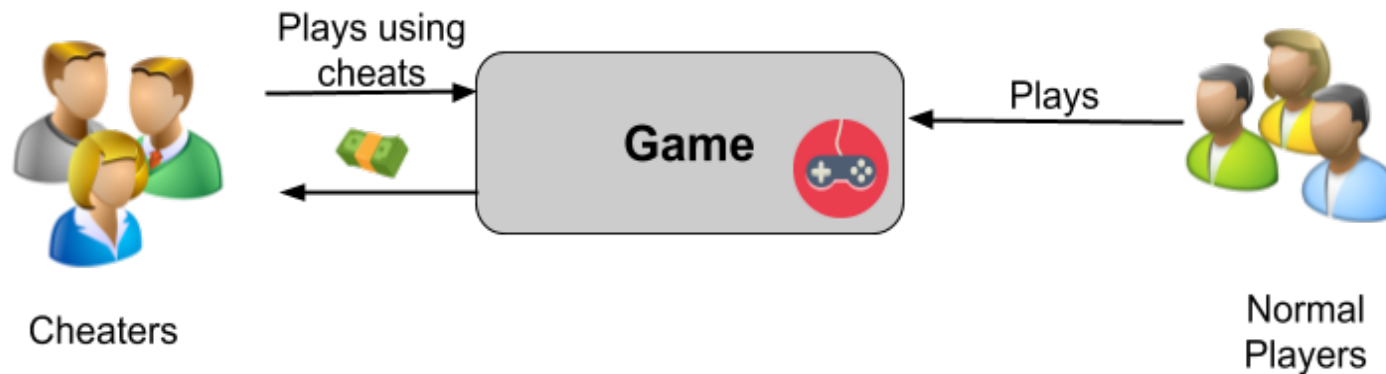
Challenges

➢ **Game dependent:**
  ➢ Most cheat detection methods analyze decrypted game dependent data.
  ➢ Using decrypted traffic involves game dependent variables.

➢ **Covariate shift:**
  ➢ The assumption of training set and test set having similar distribution may not hold.
  ➢ This may be due to sampling bias caused by label scarcity, inaccessible, and the cost of label procurement.

➢ **Limited labeled data:**
  ➢ Supervised learning models such as SVM, kNN, and neural network typically perform well when training and test datasets have similar distribution.
  ➢ Supervised learning mechanism not suitable for very limited training data.

➢ **Computational efficiency:**
  ➢ Current cheat detection methods mainly have delayed detection.
  ➢ A large delay in detection (e.g., using game logs etc.) may not be effective to act upon cheaters at the right time.

Challenges: Covariate Shift

➢ What if we do not find a good training set?
  ➢ Different sets of players may cause biased training data with respect to test data.

Example Scenario

Small set of players → Biased training data

Large set of players → test data

⚠ Affects Prediction Accuracy
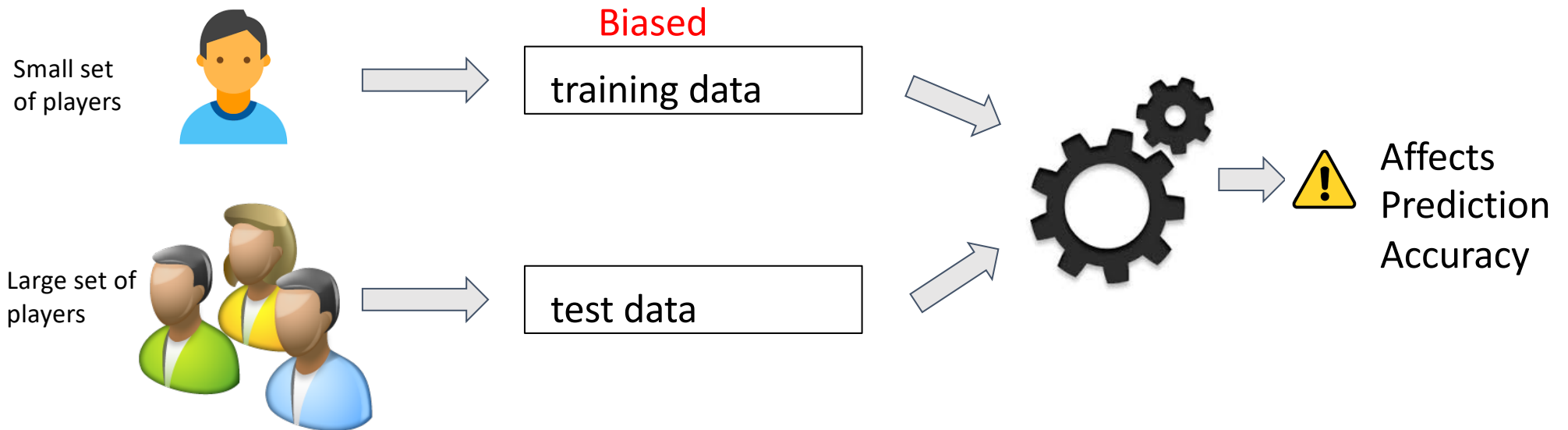
Contribution

➢ Game independent:

 ➢ We analyze the game traffic, which is encrypted and game independent.
 ➢ Detecting over encrypted traffic may act as a predecessor for further deep inspection that could be time consuming.
 ➢ It is easier to evaluate over encrypted traffic since most games are not open-source.

➢ Covariate shift:

 ➢ We utilize relative density ration to estimate importance weights associated with training data instances.
 ➢ We propose a expectation-maximization technique to automatically learn model parameters for relative density ratio estimation from available data.

Overview of GCI framework
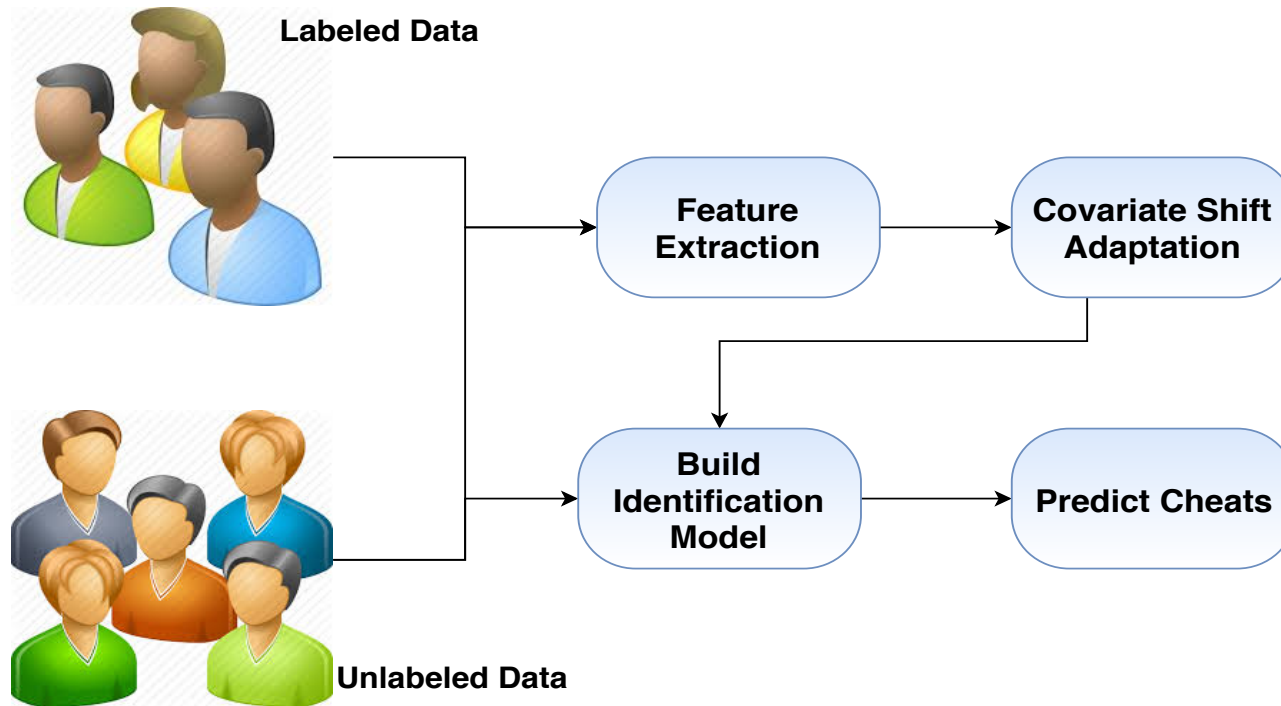
Figure 1: Overview of GCI Framework

Feature Extraction

➢ Feature extraction:

  ➢ Packets are encrypted.

  ➢ Extract features from packet headers.

➢ Some general features:

  ➢ Number of incoming packets.

  ➢ Number of outgoing packets.

  ➢ Sum of incoming packet sizes.

  ➢ Sum of outgoing packet sizes.

Feature Extraction

➤ BIND[1]:

  ➤ Uni-Burst:

    ➤ Size

    ➤ Time

    ➤ Direction

    ➤ Number of packets in the burst

  ➤ Bi-Burst:

    ➤ Size

    ➤ Time

    ➤ Number of packets in the burst

[1] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, "Adaptive encrypted traffic fingerprinting with bi-directional dependence," in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC '16. Los Angeles, California, USA, 2016, pp. 177–188.

Feature Extraction
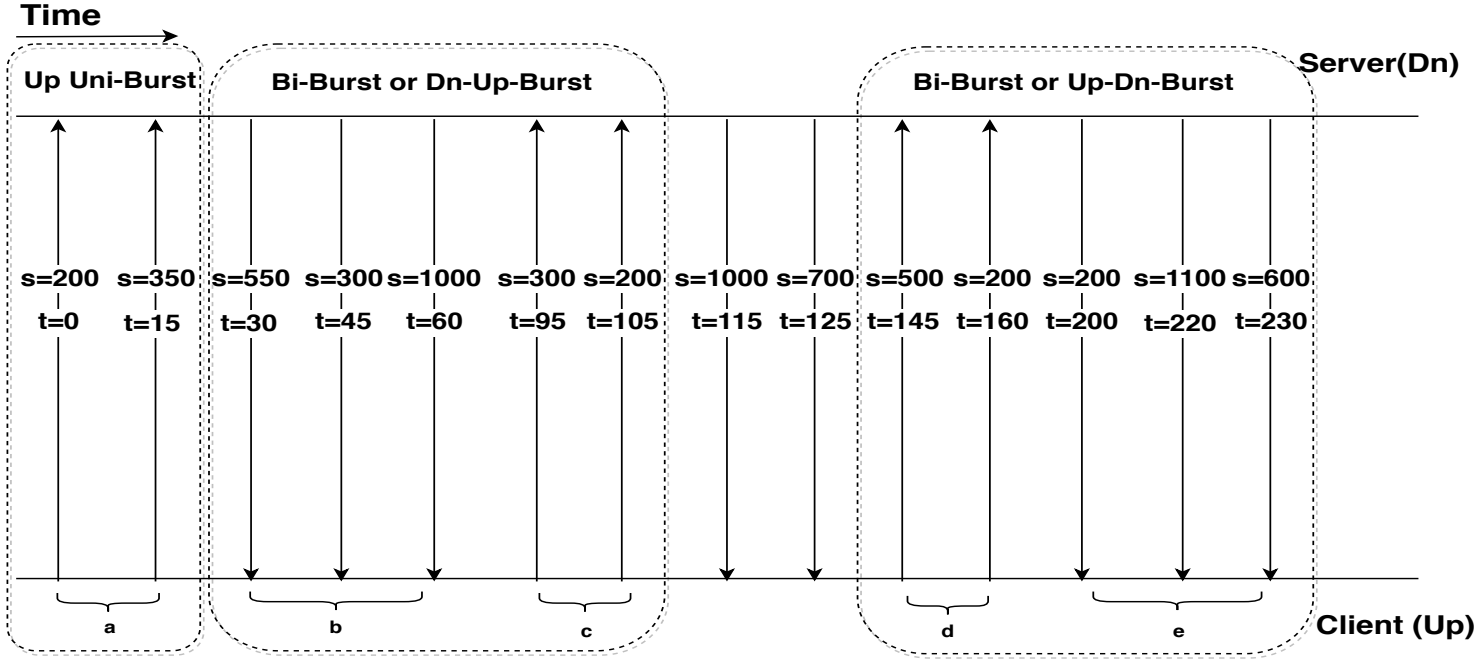
Figure 3: An example illustrating BIND[1] features

[1] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, "Adaptive encrypted traffic fingerprinting with bi-directional dependence," in *Proceed- ings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC '16. New York, NY, USA: ACM, 2016, pp. 177–188.

Covariate Shift Adaptation

➤ Relative density ratio[2] is defined as.

  ➤ $r_\alpha(\boldsymbol{x}) = \dfrac{P_{te}(\boldsymbol{x})}{\alpha P_{te}(\boldsymbol{x}) + (1-\alpha)P_{tr}(\boldsymbol{x})}$

➤ Relative density ratio is estimated using a Gaussian Kernel Model.

  ➤ $\hat{r}_\alpha(\boldsymbol{x}) = \sum_{i=1}^{N_{tr}} \theta_i K_\sigma\left(\boldsymbol{x}, B_{te}^{(i)}\right); K_\sigma(\boldsymbol{x}, \boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\sigma^2}}$

➤ Learn parameters:

  ➤ $loss = [\alpha - 1]_+ + [-\alpha]_+ + \lambda_1(\frac{1}{2}\boldsymbol{\theta}^T\widehat{\boldsymbol{H}}\boldsymbol{\theta} - \widehat{\boldsymbol{h}}^T\boldsymbol{\theta} + \frac{\lambda_2}{2}\boldsymbol{\theta}^T\boldsymbol{\theta})$

  ➤ Iteratively learning the parameters $\alpha, \boldsymbol{\theta}$ based on the loss we designed.

[2] M. Yamada, T. Suzuki, T. Kanamori, H. Hirotaka and M. Sugiyama, "Relative Density-Ratio Estimation for Robust Distribution Comparison," in *Advances in Neural Information Processing Systems 24*, 2011, pp. 594–602.

Deployment

Deploy GCI framework in game server-side:

- Since our mechanism is not game-specific, we can deploy cheat detection on the client-side as well.
- We plan to deploy our GCI framework in SGX[3] in game client-side for future work.
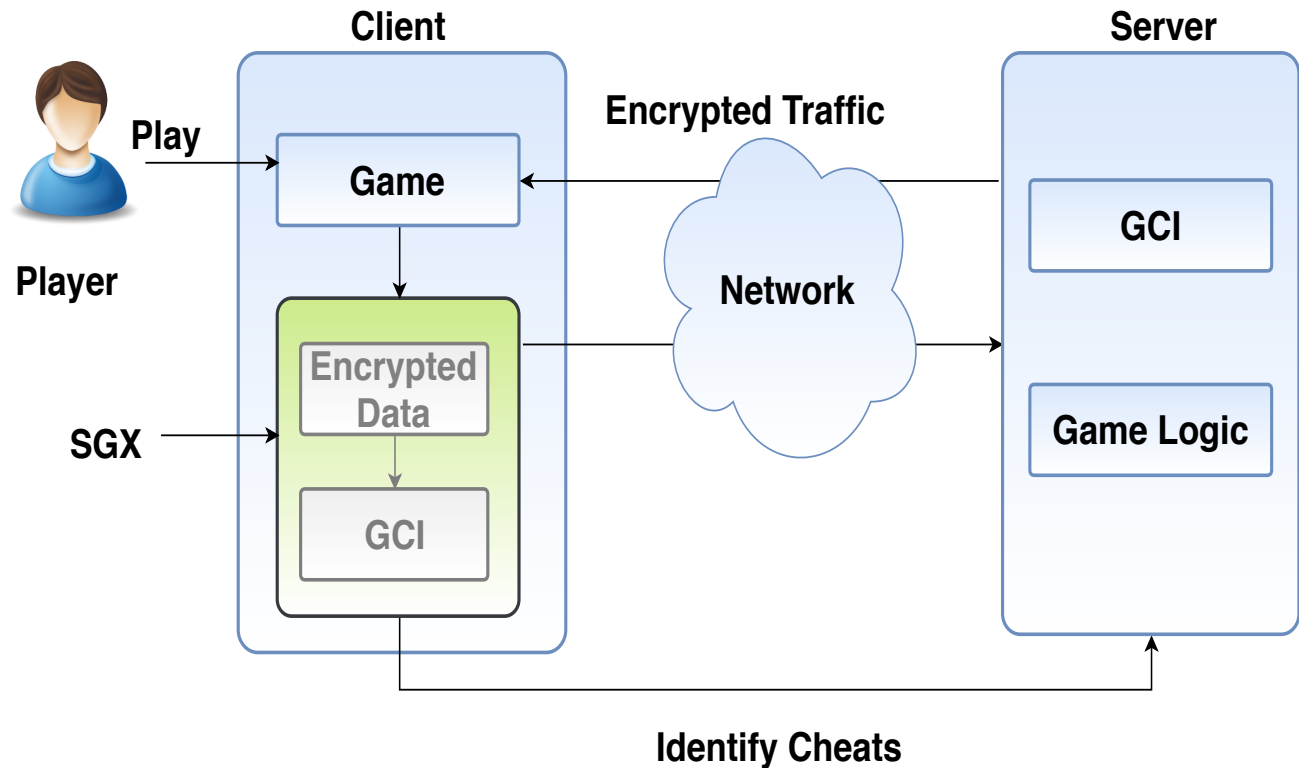
Figure 2: Overview of Our Architecture: GCI is either in server-side or client-side (future work)

[3] V. Costan and S. Devadas, "Intel sgx explained." *IACR Cryptology ePrint Archive*, vol. 2016, no. 086, pp. 1–118, 2016.

Empirical Evaluation: Data Collection
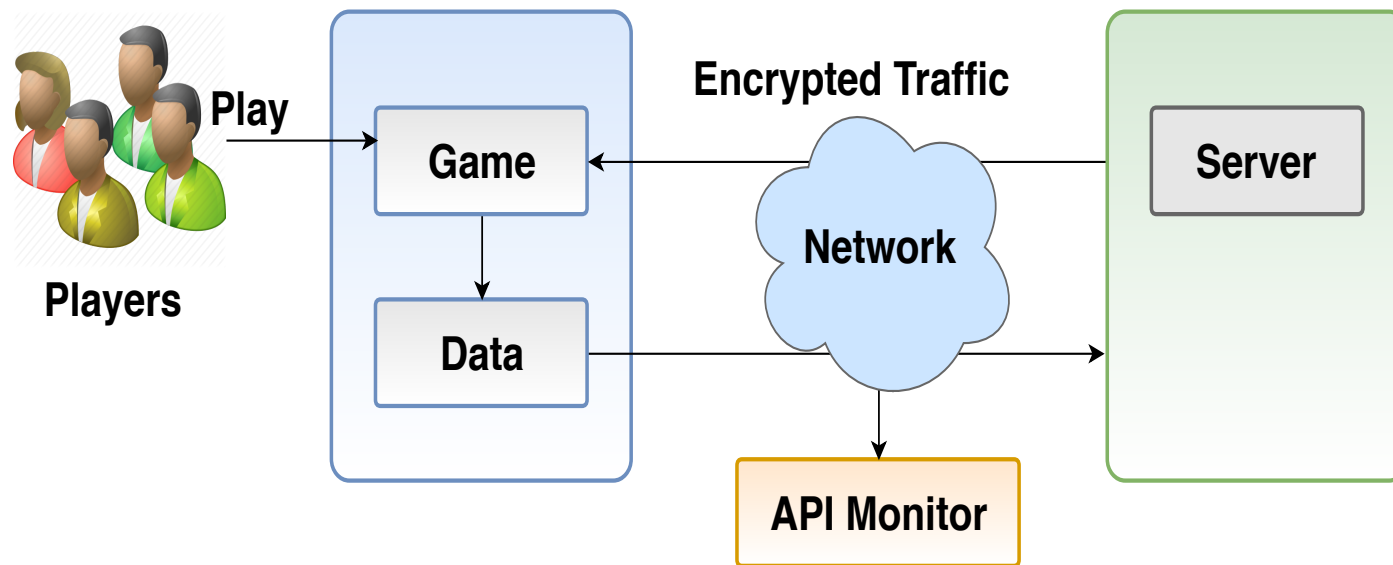
Figure 4: Overview of Data Collection Process

Empirical Evaluation: Data Sets

➢ Data sets:

➢ We collect game traffic with help of students from class CS 6301: Cyber Security Essentials of University of Texas at Dallas and Big Data Analytics and Management Lab.

➢ In total 20 students participate to collect data.

➢ Students install in their personal machines the game Counter-Strike 1.6 and the three selected cheat types downloaded from a diverse community of popular cheating sources[4][5].

➢ They connect to the server and play the game in both normal game mode as well as using the cheats applied to the game.

[4] https://www.gamespot.com/counter-strike/cheats/
[5] https://www.unknowncheats.me/forum/index.php
[6] http://www.rohitab.com/apimonitor

Empirical Evaluation: Counter-strike Cheats

➢ **Aim-bot:**
  - ➢ Enables automatic targeting the opponent while shooting.
  - ➢ This targeting works even if the opponent is too far away or behind walls.

➢ **Speed-hack:**
  - ➢ Enables speed increase in player's movement while playing the game.
  - ➢ A player can apply different variations of speeds and play the game.

➢ **Wall-hack:**
  - ➢ Makes the walls transparent for the player so that he or she can see the enemy through the walls.

Baseline Methods

Baseline Methods	Description
KMSVM	Equip KMM[7] with base classifier weighted SVM to build classification models.
KLISVM	Equip KLIEP[8] with base classifier weighted SVM to build classification models.
SVM	Multi class Support Vector Machine.

Proposed Method	Description
GCI	Equip revised RULSIF with base classifier weighted SVM to build classification models

[7] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Scholkopf, "Correcting sample selection bias by unlabeled data," in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, ser. NIPS'06. Cambridge, MA, USA: MIT Press, 2006, pp. 601–608.
[8] Y. Kawahara and M. Sugiyama, "Sequential change-point detection based on direct density-ratio estimation," *Stat. Anal. Data Min.*, vol. 5, no. 2, pp. 114–127, Apr. 2012

Empirical Evaluation: Experiment Settings

➢ **Feature extraction:**

    ➢ We first extract features follow[1].

➢ **Generate training and test data:**

    ➢ We select the game traffic data of a set of game players each time for training, then test the rest players.

    ➢ We randomly split the test data into two sets, one is combined with training set for building up the classification model, the other is for testing.

    ➢ We generate data in different 10 groups by selecting different fixed sized training set and run experiment by cross-validation.

[1] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, "Adaptive encrypted traffic fingerprinting with bi-directional dependence," in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC '16. Los Angeles, California, USA, 2016, pp. 177–188.

Empirical Evaluation: Experiment Settings

➢ **Multi class labels:**
  ➢ Aim-bot
  ➢ Speed-hack
  ➢ Wall-hack
  ➢ Normal (without cheats)

➢ **Binary class labels:**
  ➢ Cheats (aim-bot, speed-hack, wall-hack)
  ➢ Normal (without cheats)

[1] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, "Adaptive encrypted traffic fingerprinting with bi-directional dependence," in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC '16. Los Angeles, California, USA, 2016, pp. 177–188.

Empirical Evaluation: Performance
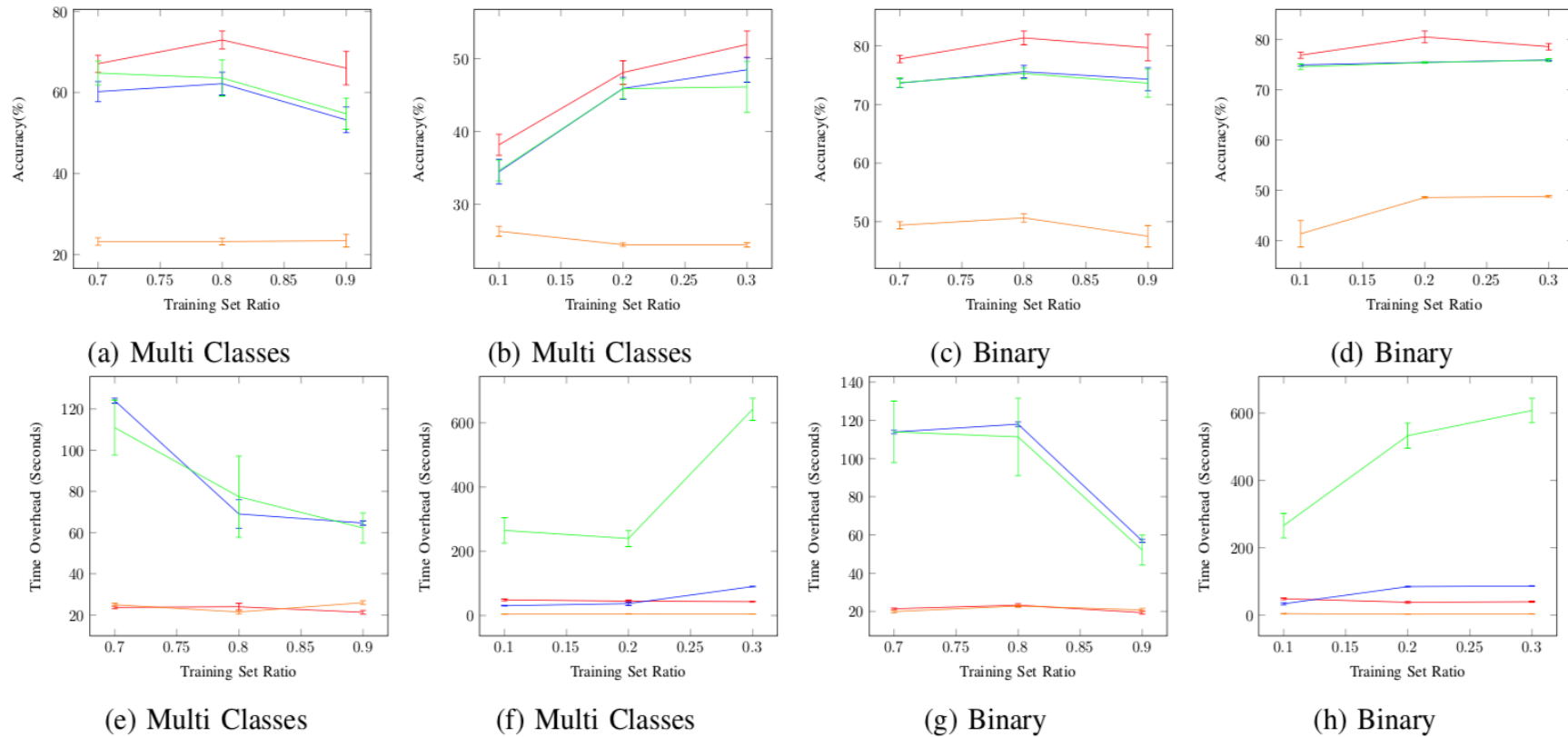
Figure 5: Performance of classification for all approaches. (—×— GCI; —— KMSVM; —— KLISVM; —— SVM).

Application: Encrypted Traffic Fingerprinting
Al-Naami et al. [1][2][3]

- Traffic Fingerprinting (TFP) is a Traffic Analysis (TA) attack that threatens web/app navigation privacy.

- TFP allows attackers to learn information about a website/app accessed by the user, by recognizing patterns in traffic.

- Examples: Website Fingerprinting
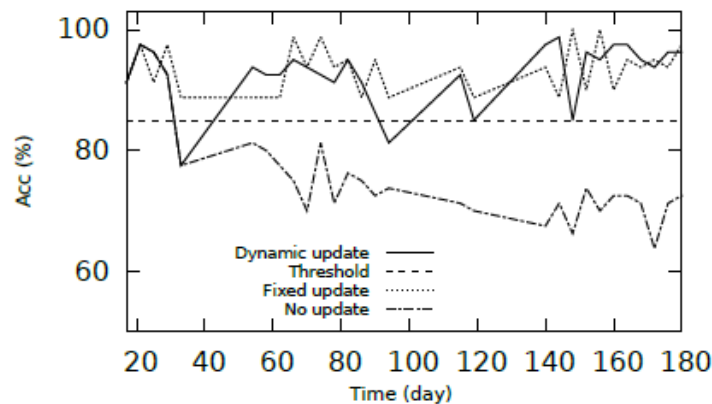
Figure 9: Adaptive Learning.
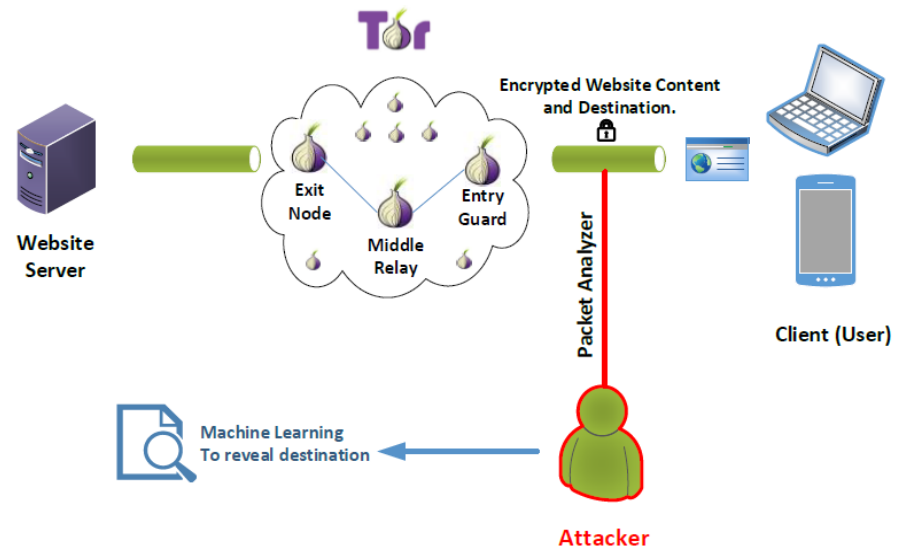
[1] K. Al-Naami, G. Ayoade, vA. Siddiqui, N. Ruozzi, L. Khan and B. Thuraisingham, "P2V: Effective Website Fingerprinting Using Vector Space Representations," Computational Intelligence, 2015 IEEE Symposium Series on, Cape Town, 2015, pp. 59-66.
[2] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham. 2016. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC '16), Los Angeles, CA.
[3] K. Al-Naami, L. Khan, *et al.*, "BiMorphing: A Bi-Directional Bursting Defense Against Website Fingerprinting Attacks," in *IEEE Transactions on Dependable and Secure Computing*, doi: 10.1109/TDSC.2019.2907240.

WFP Diagram – Tor
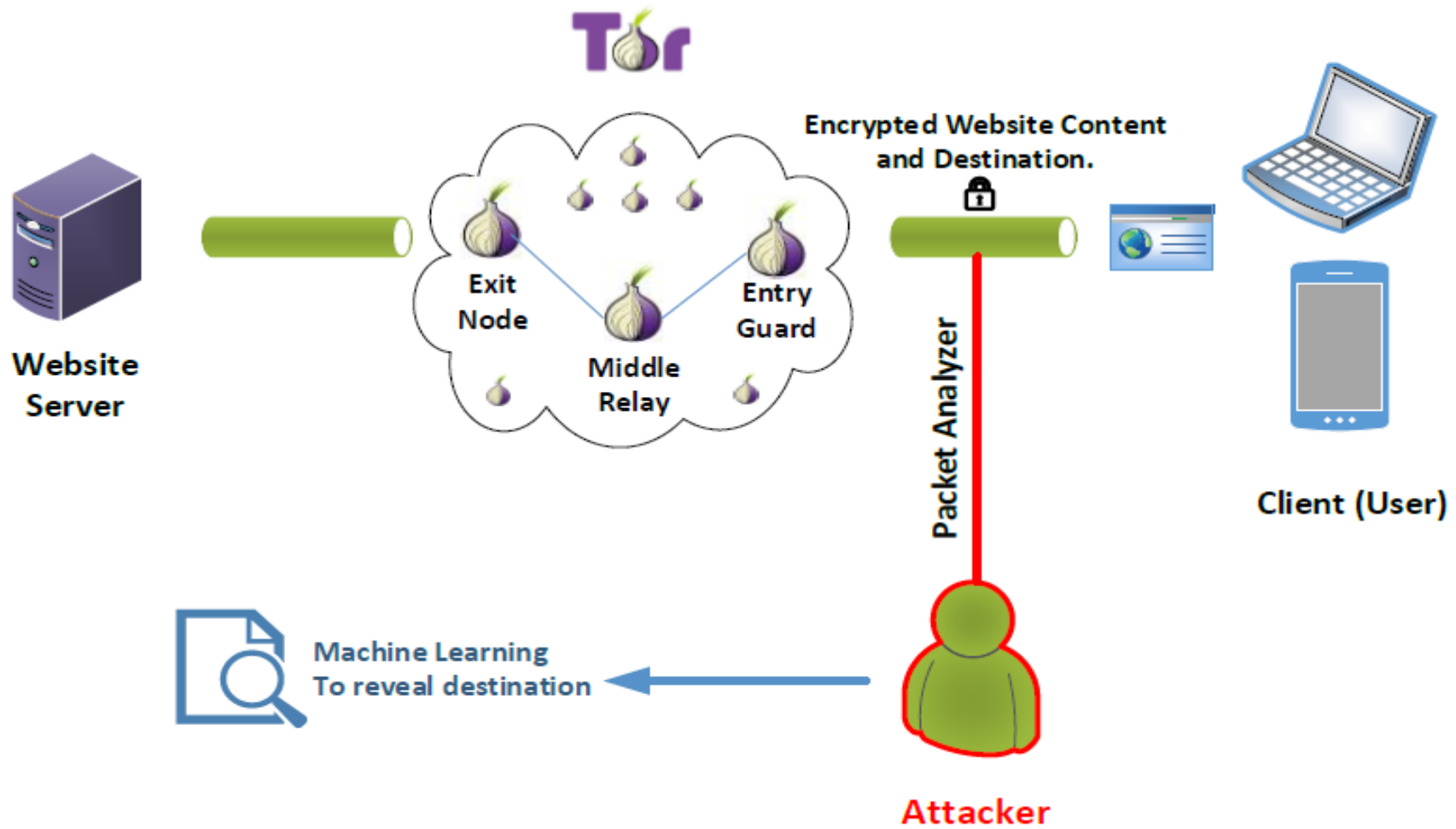
Application: Web Site Fingerprinting

- The Goal is to identify the websites

- Can harm certain individuals
  - Journalists
  - Activists
  - Bloggers

- Can also help identify threats
  - Bad people

K. Al-Naami, L. Khan *et al.*, "BiMorphing: A Bi-Directional Bursting Defense Against Website Fingerprinting Attacks," in *IEEE Transactions on Dependable and Secure Computing*, doi: 10.1109/TDSC.2019.2907240.
K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham. 2016. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC '16), Los Angeles, CA.

Challenge

- Had data not been encrypted,
  - No challenge → use NLP

- However,
  - data is encrypted

- All we can see is just:
 - packet size in bytes
  - packet time

Contributions

- A novel multi-domain coarse-feature extraction approach   (*BIND*) (fingerprinting with BI-directioNal Dependence) over encrypted data
  - considers the relationship among sequences of packets in opposite directions

- Across multiple domains
  - HTTPS
  - Tor

    Website Fingerprinting

- Closed-world and open-world settings
- The approach is more immune and resilient to known defenses

Terms

- **Tx : Uplink**
- **Rx: Downlink**
- **Basic Unit:**
  - **Tx Packet**
  - **Rx Packet**
- **A burst is a sequence of consecutive packet flows in the same direction**
  - **Tx Burst**
  - **Rx Burst**
- **Bi-Burst**
  - **Tx-Rx**
  - **Rx-Tx**
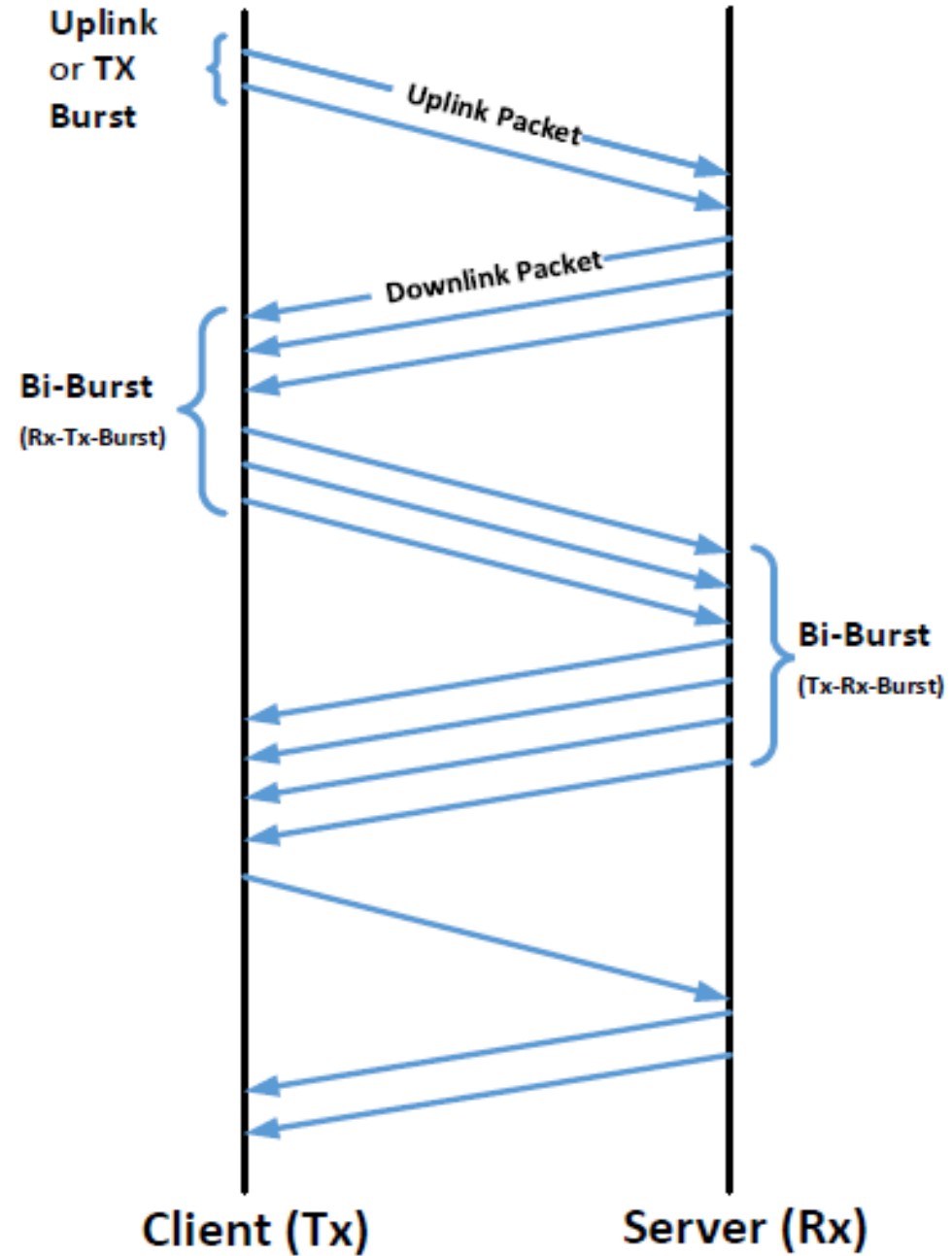
Bi-Direction Bursting (BIND)

- The main concern is how to extract meaningful features for website/app identification

Category	Features
Packet (Tx/Rx)	Packet length
Uni-Burst (Tx/Rx)	Uni-Burst size
	Uni-Burst time *
	Uni-Burst count
Bi-Burst (Tx-Rx/Rx-Tx)	Bi-Burst size *
	Bi-Burst time *
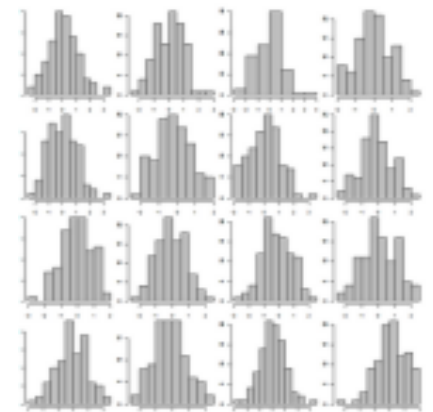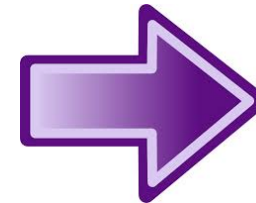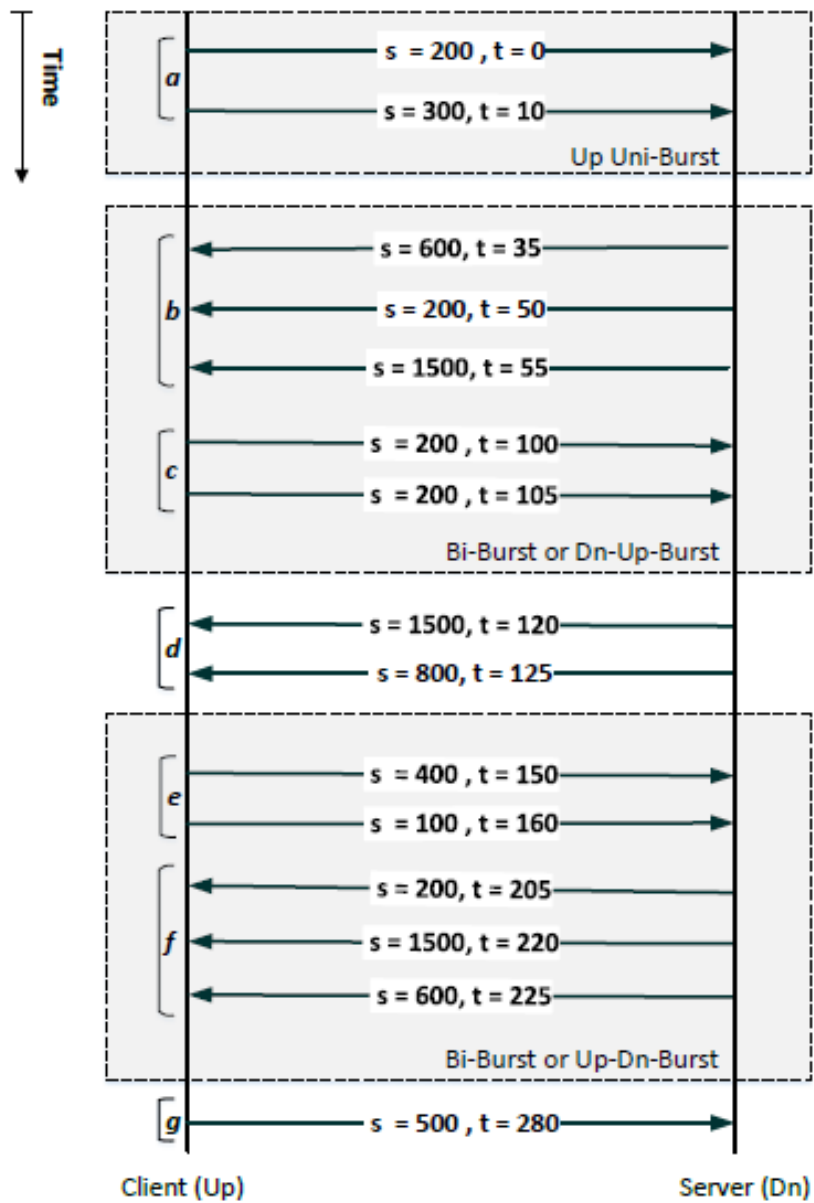
Figure 2: An example illustrating BIND Features.

Closed-world vs Open-world

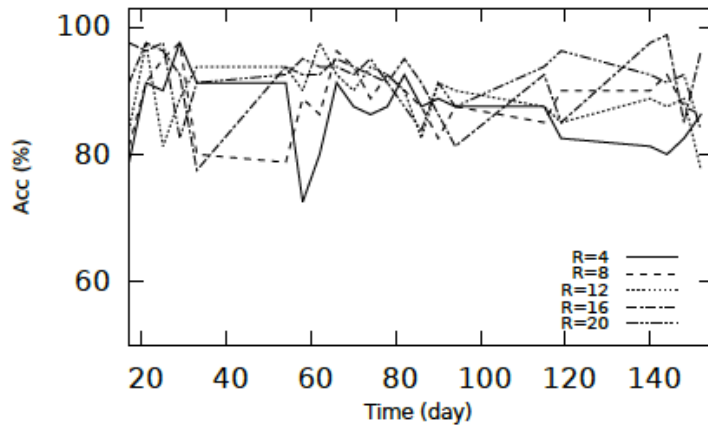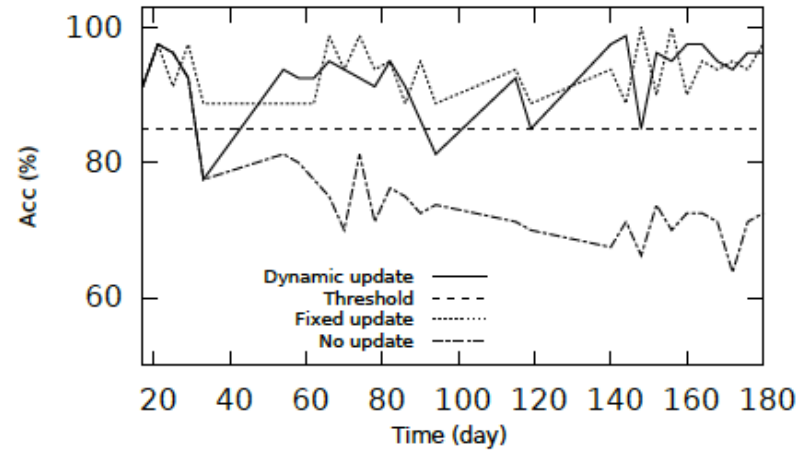| Item	Closed-world	Open-world
Set	Finite set of websites	- Monitored - Non-Monitored
Classification	Multi-class	Binary
Goal	Predict website/app	Predict if a Monitored or non-Monitored website/app
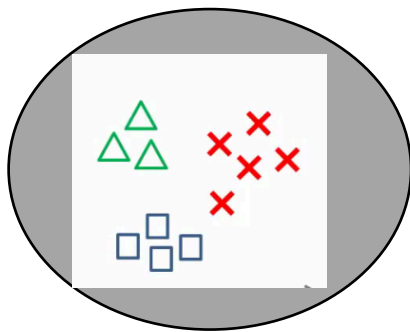
Closed-world
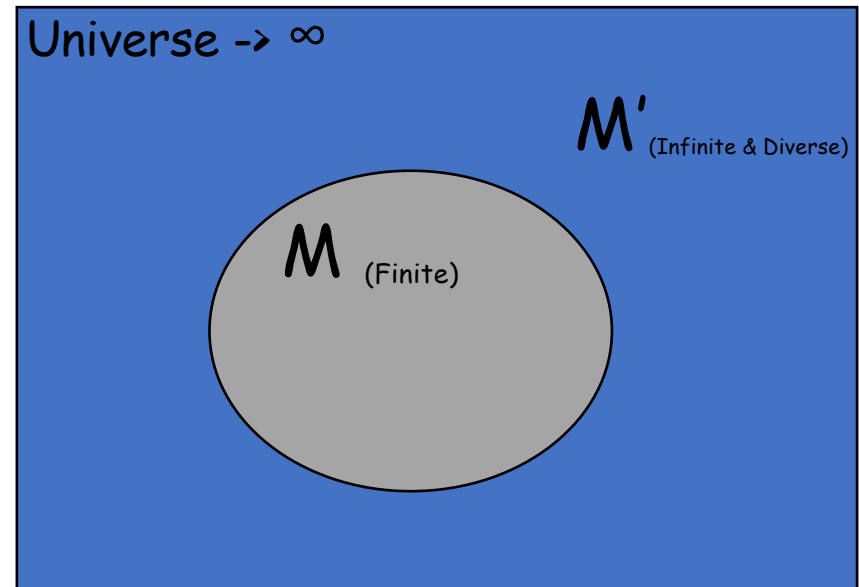
Open-world

Experiments: Website Fingerprinting -- Adaptive Learning

R	4	8	12	16	20
Average accuracy (%)	86.6	89.3	89.9	**92.6**	91.7
Number of updates	10	7	5	4	2

Open Source Tools

- Website Fingerprinting: https://github.com/khaled-alnaami/NetworkTrafficFingerprinting

- Multi-stream Regression/Classification:
  https://github.com/ahaque-utd/MSR
  https://github.com/ahaque-utd/FUSION

- Stream Classification:

  https://github.com/ahaque-utd/ECHO
  https://github.com/ahaque-utd/SAND