



A Framework for On-Demand Classification of Evolving Data Streams

WENBO, ZOU
CPS Laboratory

Introduction

1.1 Background

- Previous work on stream classification simply treats it as a **one-pass mining** problem. This does not treat the classification problem in **the context of the underlying changes** which have occurred in the stream.
- A more temporally adaptive philosophy is desirable to improve the effectiveness of the underlying algorithms.

Introduction

1.2 Proposed Method

- Propose an **on-demand classification** process which can dynamically **select the appropriate window** of past training data to build the classifier.
- In such a case, a classification model which is constructed **using a smaller history** of data is likely to provide better accuracy.



Algorithm

2.1 Supervised Microcluster

- Data streams each consist of a set of multidimensional records $X_1 \dots X_k$
- Arriving at time stamps $T_1 \dots T_k$
- Each X_i is a multi-dimensional record $X_i = (x_i^1 \dots x_i^d)$.
- Each record X_i in the training data stream is associated with a class label C_j .



Algorithm

2.1 Supervised Microcluster

- Each class label is defined as the $(2*d+4)$ tuple :

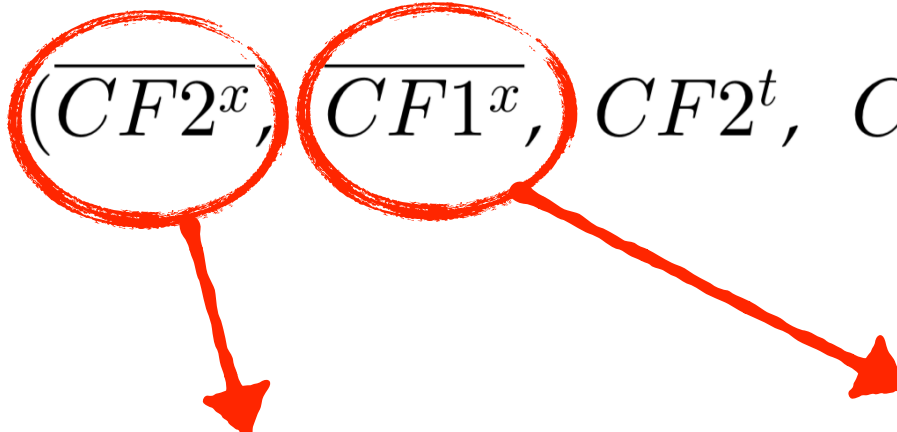
$$(\overline{CF2^x}, \overline{CF1^x}, CF2^t, CF1^t, n, class_id)$$



Algorithm

2.1 Supervised Microcluster

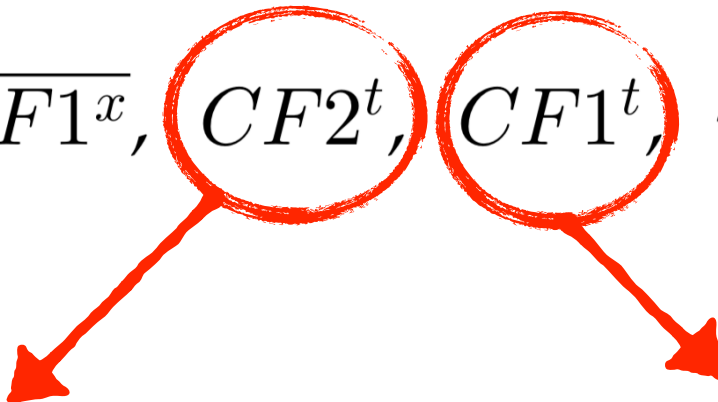
- Each class label is defined as the $(2*d+4)$ tuple :

$$(\overline{CF2^x}, \overline{CF1^x}, CF2^t, CF1^t, n, class_id)$$

$$\sum_{j=1}^n (x_{i_j}^p)^2 \qquad \sum_{j=1}^n x_{i_j}^p$$

Algorithm

2.1 Supervised Microcluster

- Each class label is defined as the $(2*d+4)$ tuple :

$$(\overline{CF2^x}, \overline{CF1^x}, CF2^t, CF1^t, n, class_id)$$


The sum of **the squares of the time stamps**

The sum of **the time stamps**

Algorithm

2.2 Time Frame

- The effectiveness of the classification model may be highly sensitive to the length of the horizon used for the training process
- This can be achieved by storing the behavior of the microclusters at different moments in time. These stored microcluster states are referred to as snapshots.



Algorithm

2.2 Time Frame

Geometric Time Frame

1. Assume that one unit of clock time is the smallest level of granularity.
2. Snapshots are classified into different frame numbers which can vary from 0 to a value no larger than $\log_2(T)$, where T is the maximum length of the stream.
3. Specifically, snapshots of frame number i are stored at clock times which are divisible by 2^i . For each frame number, at most max capacity snapshots are stored.



Algorithm

2.2 Time Frame

Geometric Time Frame

TABLE 1
A Geometric Time Window

| Frame no. | Snapshots (by clock time) |
|-----------|---------------------------|
| 0 | 69 67 65 |
| 1 | 70 66 62 |
| 2 | 68 60 52 |
| 3 | 56 40 24 |
| 4 | 48 16 |
| 5 | 64 32 |

Algorithm

2.2 Time Frame

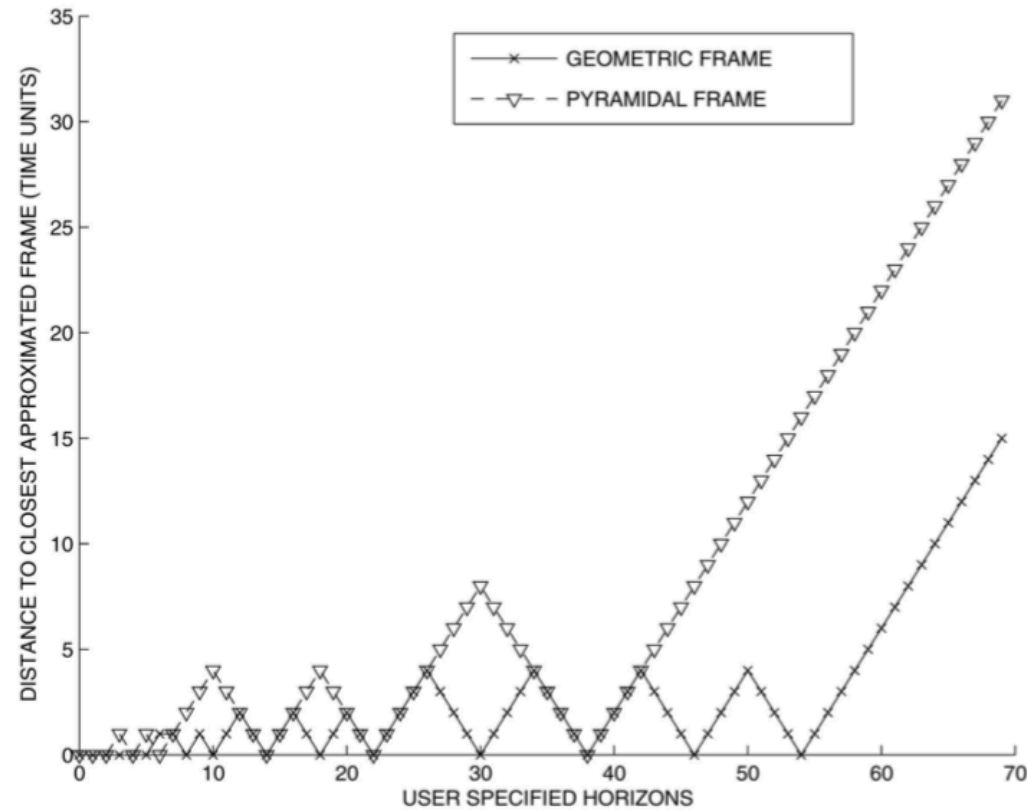
Geometric Time Frame

For any user-specified time window of h , **at least one stored snapshot** can be found within a factor of 2 of the specified horizon.



Algorithm

2.2 Time Frame



While the geometric and pyramidal time frame are conceptually similar, the geometric time frame has the advantage of **greater elegance** and **ease of efficient implementation without double counting**.

Algorithm

2.3 Maintenance of Supervised Microcluster

1. **Initial Microclusters** An offline clustering algorithm is applied to the disk resident points. **An equal number** of microclusters is created for each class by using a separate **k-means algorithm**
2. **Maximum Boundary** For the cluster feature vector of M_p , the **average deviation** of the other points from the **centroid of this cluster**. For a cluster with only one previous point, the maximum boundary is defined in a heuristic way



Algorithm

2.3 Maintenance of Supervised Microcluster

1. If the data point X_{i_k} lies **within this maximum boundary**, then it is **added** to the microcluster M_p
2. If the data point does **not lie within the maximum boundary** of the **nearest microcluster**, then a **new micro- cluster must be created** containing the data point X_{i_k} .



Algorithm

2.3 Maintenance of Supervised Microcluster

- In order to insert this new microcluster, the number of other clusters must be reduced by one. This can be achieved by either **deleting an old cluster** or **joining two microclusters which belong to the same class**.
- Using the **mean and standard deviation of the m last time stamps** which have arrived in the current microcluster.

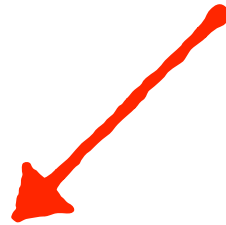
$$\sigma = \sqrt{CF2^t/n - (CF1^t/n)^2}.$$

- What's more if **the least relevant time stamp is below a user-defined threshold**, it **can be eliminated** for new arrivals id named room.
- When **all relevance stamps are larger than the user-defined threshold**, **two microclusters are merged** in order to create space for the new microcluster.

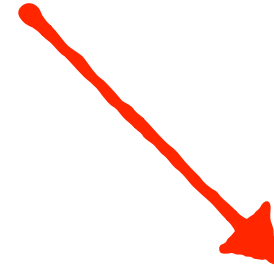
Algorithm

2.4 Classification on Demand

Arriving Data Streams



Maintenance
Model training



$\overline{X} \in \mathcal{Q}_{fit}$
For Horizon fitting

Algorithm

2.4 Classification on Demand

$$\overline{X} \in \mathcal{Q}_{fit}$$

For Horizon fitting

For each microcluster in the current set $S(t_c)$, we find the list of ids in each microcluster. For each of the lists of ids, we find the corresponding microclusters in $S(t_c - h')$ and subtract the CF vectors for the corresponding microclusters in $S(t_c - h')$. The resulting set of microclusters corresponds to the time horizon $(t_c - h, t_c)$.

—> $N(t_c, h)$.



Algorithm

2.4 Classification on Demand

- We find the **closest microcluster** in $N(t_c, h)$ to X .
- We determine the class label of this microcluster and **compare it to the true class label** of X . The accuracy over all the points in Q_{fit} is then determined. This provides the accuracy over that particular time horizon.



Algorithm

2.4 Classification on Demand

- For each given test instance X_t , the above described nearest- neighbor classification process is applied **using each h_i to H** .
- The **majority class** among these p class labels is reported as the relevant class.



Experiment

3.1 Accuracy Evaluation

TABLE 3
Default Parameter Settings

| Parameter | value | meaning |
|---------------------------|-----------------|--|
| <i>microcluster-ratio</i> | 5 | number_of_micro-clusters/number_of_natural_clusters |
| <i>max_capacity</i> | 32 | maximum number of snapshots for each frame number |
| <i>InitNumber</i> | 400 | number of initial points |
| <i>t</i> | 2 | maximum boundary factor |
| <i>H</i> | 8 | size of sliding window |
| δ | 512 | relevance stamp threshold |
| <i>p</i> | 1 | number of best time horizons used for classification |
| <i>m</i> | $0.32 \times n$ | number of last arrivals |



Experiment

3.1 Accuracy Evaluation

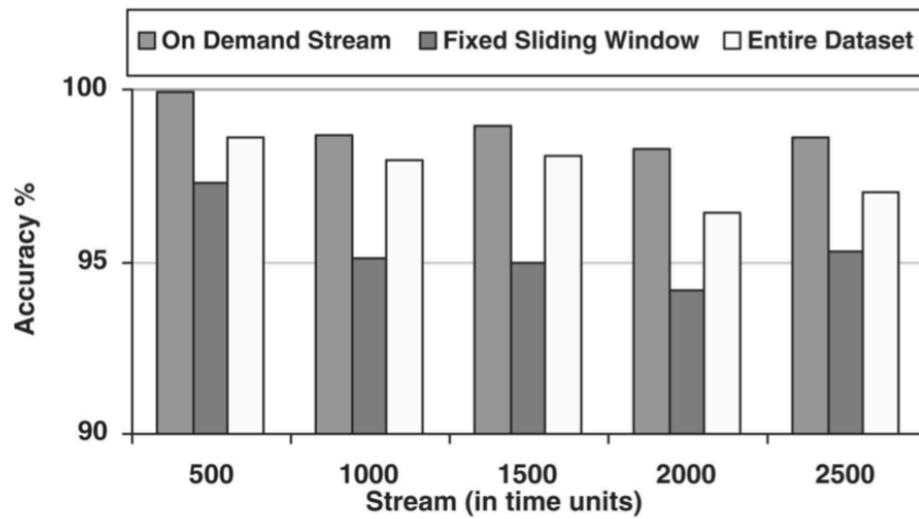


Fig. 2. Accuracy comparison (Network Intrusion data set, $stream_speed = 80$, $buffer_size = 1,600$, $k_{fit} = 80$, $InitNumber = 400$).

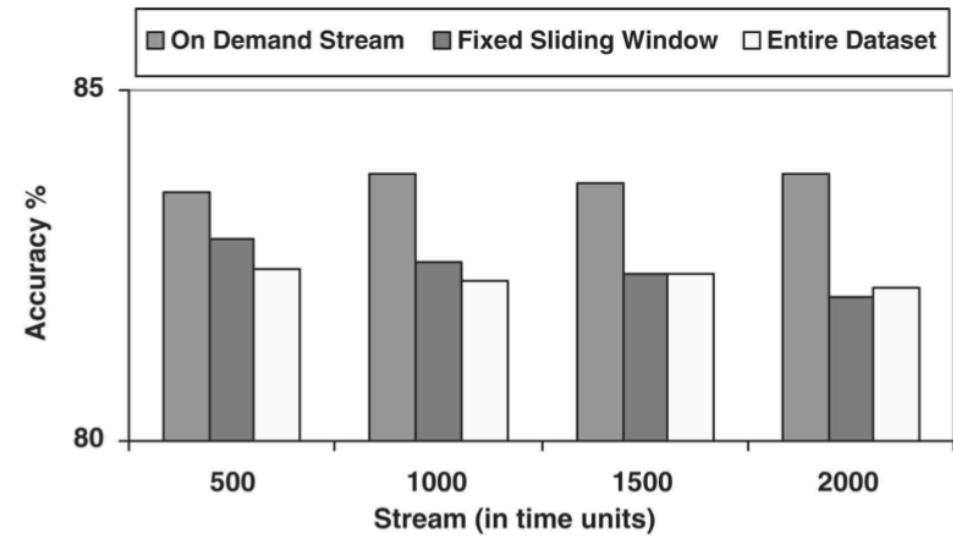


Fig. 6. Accuracy comparison (Synthetic data set B300kC5D20, $stream_speed = 100$, $buffer_size = 500$, $k_{fit} = 25$, and $InitNumber = 400$).

Experiment

3.1 Accuracy Evaluation

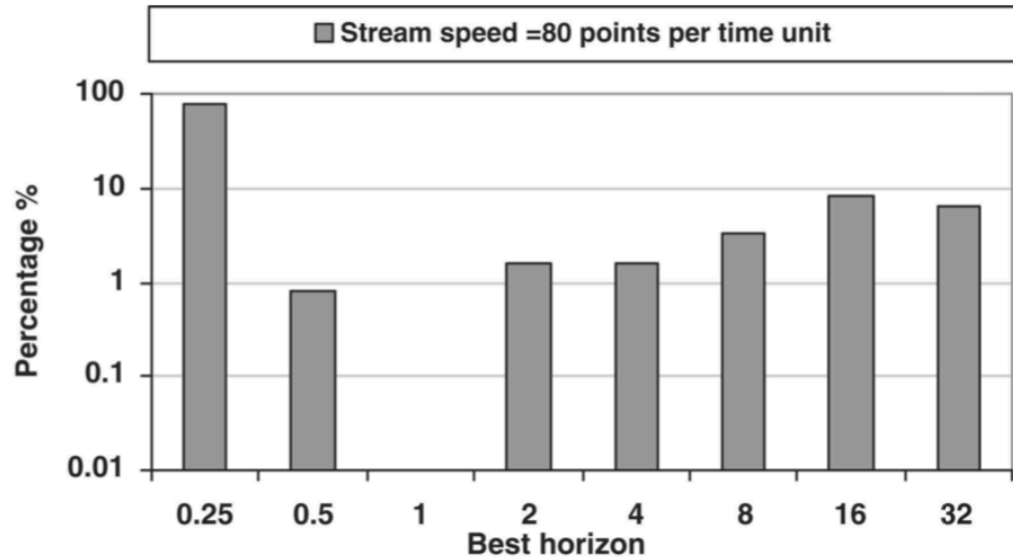


Fig. 3. Distribution of the (smallest) best horizon (Network Intrusion data set, Time units = 2,500, buffer_size = 1,600, $k_{fit} = 80$, $InitNumber = 400$).

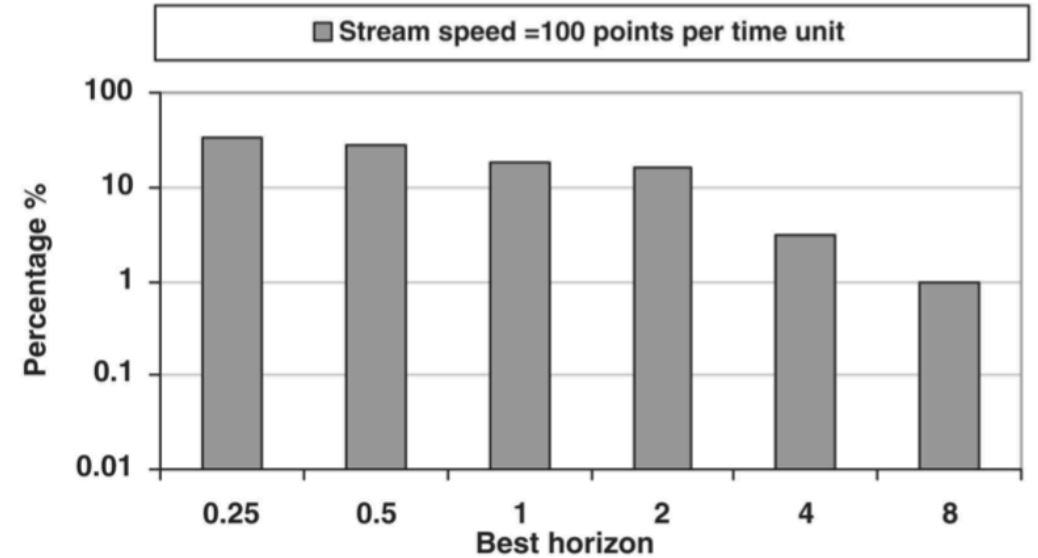


Fig. 7. Distribution of the (smallest) best horizon (Synthetic data set B300kC5D20, Time units = 2,000, buffer_size = 500, $k_{fit} = 25$, and $InitNumber = 400$).

Experiment

3.2 Efficiency Test

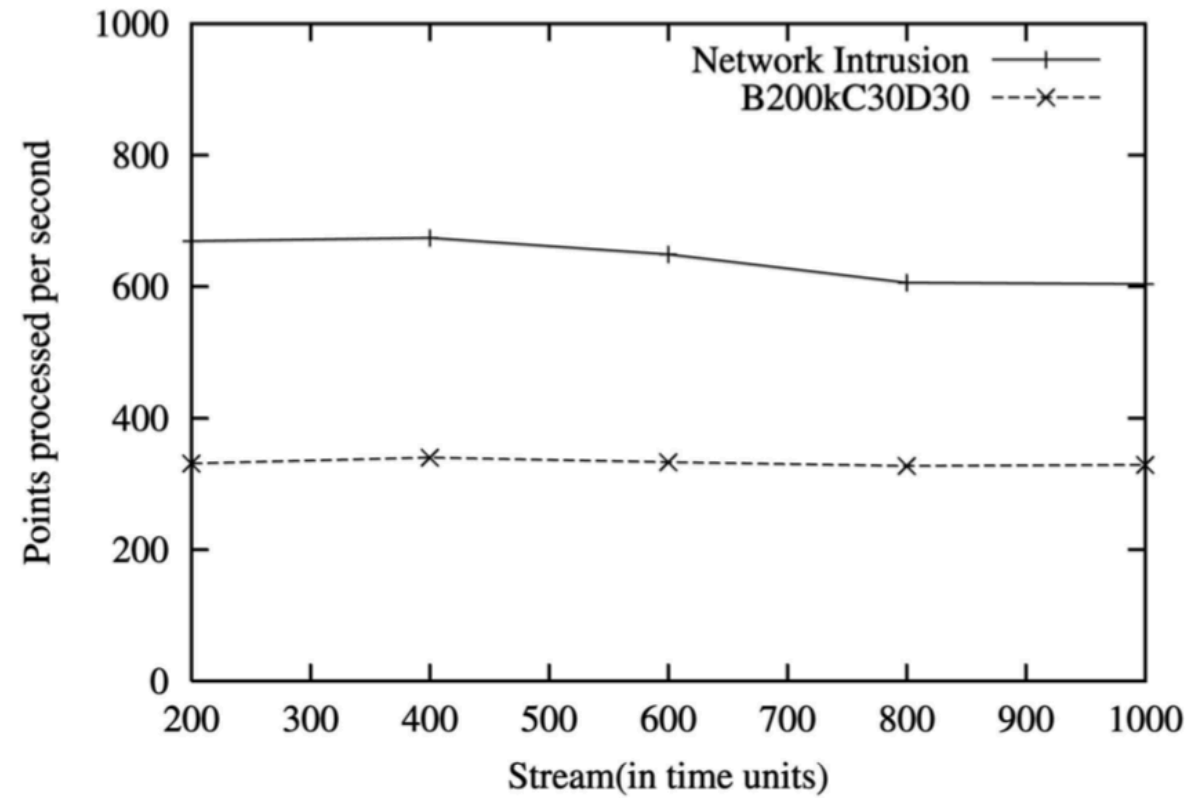


Fig. 10. Processing time.

Experiment

3.3 Sensitivity analysis

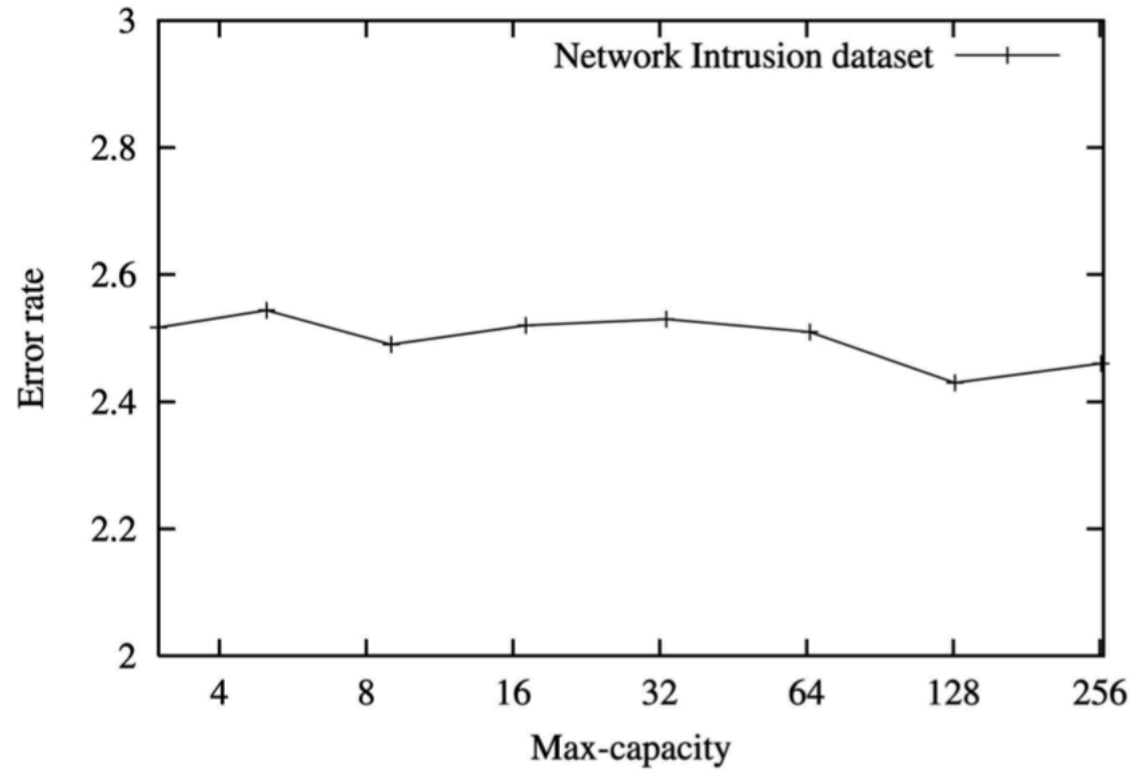


Fig. 13. Sensitivity analysis with Max_capacity.

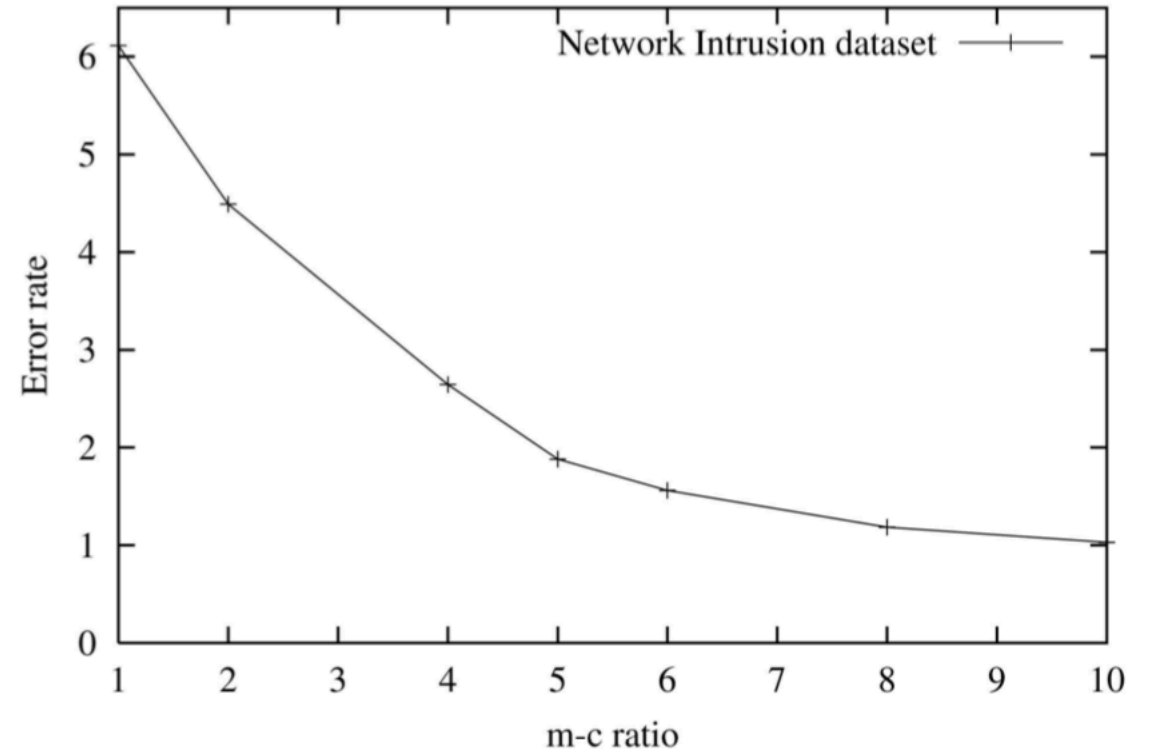


Fig. 14. Sensitivity analysis with MicroclusterRatio.



HANYANG UNIVERSITY

College of Computer Science& Engineering

Q & A

CHU, MUNBAK