Loadstar: Load Shedding in Data Stream Mining

Yun Chi¹, Haixun Wang², Philip S. Yu²
¹Department of Computer Science, UCLA
²IBM Thomas J. Watson Research Center

Introduction

- Data stream systems
  - Data from embedded sensors
  - Financial and retailer data
  - Network traffic data
- Resources are limited
  - CPU cycles
  - Bandwidth
  - Memory
Load Shedding—Which to Drop?

- Load shedding
  - Dropping certain amount of loads
- Which to drop?
  - Randomly
  - Intelligently

Load Shedding—An Example of Temperature Sensors

Case 1

Case 2
Our Main Contributions

- A Novel Quality of Decision (QoD) measure
  - Discriminant functions
  - Predicted feature distribution
- A feature prediction model based on
  - Markov-chains
  - Real-time parameters update
- Loadstar
Quality of Decision — Discriminant Functions

![Discriminant Functions Diagram]

Quality of Decision — Based on Overall Risk

- Feature distribution in the next time unit
  \[ \bar{X} \sim p(\bar{x}) \]

- At a point \( x \), the conditional risk for \( ci \)
  \[ R(c_i \mid \bar{x}) = \sum_{j=1}^{k} \sigma(c_i \mid c_j) P(c_j \mid \bar{x}) \]

- The expected risk
  \[ E_{\bar{x}}[R(c_i \mid \bar{x})] = \int R(c_i \mid \bar{x}) p(\bar{x}) d\bar{x} \]

- The decision based on expected risk
  \[ \delta_2: \quad k = \arg \min_i E_{\bar{x}}[R(c_i \mid \bar{x})] \]
Quality of Decision — Based on Overall Risk

The Bayesian risk:

\[
E_x[R(c^* \mid \bar{x})] = \int_x R(c^* \mid \bar{x}) p(\bar{x}) d\bar{x}
\]

The Quality of Decision (QoD):

\[
Q_2 = 1 - \left( E_x[R(c_k \mid \bar{x})] - E_x[R(c^* \mid \bar{x})] \right)
= 1 - \int_x \left[ P(c^* \mid \bar{x}) - P(c_k \mid \bar{x}) \right] p(\bar{x}) d\bar{x}
\]

0 Q_2 1, the higher the Q_2, the more confident we are.

Q_2=1 if and only if c_k is the minimum-risk decision at all region of the feature space.
Feature Prediction

- Feature distribution: $\bar{x} \sim p(\bar{x})$
- Take advantage of temporal locality
  - Stock price data
  - Consecutive snapshots from satellites
- Discrete-time Markov-chains
- Real-time parameter updating

Parameter Updating

- MLE: $\hat{p}_y = \frac{n_y}{\sum_i n_{ik}}$
- EM?
- Approximation: $\hat{p}_y = \frac{n_y}{\sum_i n_{ik}}$
The Loadstar Algorithm

Algorithm Loadstar($N', C$)

inputs: data from $N'$ streams, and system capacity $C$;
outputs: decisions ($\delta_1, \ldots, \delta_N$);
static variables: feature distributions $p(x)$'s, Markov-chains MC's,
COR flags ($f_1, \ldots, f_N$);
1: update $p(x)$ for each feature $x$ using its MC;
2: compute decisions ($\delta_1, \ldots, \delta_N$)
   and QoDs ($q_1, \ldots, q_N$) using $p(x)$'s;
3: select $C$ streams from $N'$ data streams
   based on ($q_1, \ldots, q_N$) and ($f_1, \ldots, f_N$);
4: for each selected stream $i$ do
5:   observe the feature value for stream $i$;
6:   revise $\delta_i$ for stream $i$;
7:   revise $p_i(x)$ for stream $i$;
8:   if stream $i$ has had load in the
   previous time unit then
9:   update MC's for stream $i$;
10:  $f_i$ ← false;
11:  else $f_i$ ← true;
12: return ($\delta_1, \ldots, \delta_N$);

Demonstration

- Penalty of Load Shedding
- Resource Adaptability
- Data Streams with Concept-drifts
Penalty of Load Shedding—Performance

- Naive Algorithm
- Loadstar
- Loadstar

Penalty of Load Shedding—Resources Assigned to the Volatile Streams

- Naive Algorithm
- Loadstar
Resource Adaptive Load Shedding

Data Streams with Concept-drifts

- Use two Markov-chains:
  \[ P_A = \begin{bmatrix} .91 & .03 & .03 & .03 \\ .03 & .91 & .03 & .03 \\ .03 & .03 & .91 & .03 \\ .03 & .03 & .03 & .91 \end{bmatrix}, \quad P_B = \begin{bmatrix} .25 & .25 & .25 & .25 \\ .25 & .25 & .25 & .25 \\ .25 & .25 & .25 & .25 \\ .25 & .25 & .25 & .25 \end{bmatrix} \]

- First use \( P_A \); at time 1,000 switch to \( P_B \); at time 3,000 switch back to \( P_A \);

- Report the total Kullback-Leibler divergence:
  \[ \sum_j d(P, \hat{P}_j) = \sum_i \sum_j P_{ij} \log \left( \frac{P_{ij}}{\hat{P}_{ij}} \right) \]
Data Streams with Concept-drifts

Future Directions

- Certain time units reserved for learning
- Networks of dependent data streams
- Data mining as an intermediate block
- Control the communication rates