Random Sampling over Data Streams for Sequential Pattern Mining

C. Raïssi et P. Poncelet

LIRMM, LGI2P/Ecole des Mines d'Alès

15 mars 2007

WDSA2007, Caserta, Italy.

Plan
Motivations
Preliminary Concepts
Related Work
Sequential Pattern Mining
Synopsis Construction
Sampling in static databases
Extending to Data Streams
Experimental Results
Conclusion and Summary

Motivations

- A new problem : data modeled as a potentially infinite flow of transactions
- Many recent real-world applications :
  1. Network traffic monitoring
  2. Trend analysis
  3. Sensor network data analysis

- Classical mining approaches are inefficient for this new problem
- In many cases, it may be acceptable to generate *approximate solutions* : synopsis structures ?

Definitions

- Let $\mathcal{D}$ be a database of customer transactions where each transaction T consists of :
  1. A customer-id, denoted by C_{id}
  2. A transaction time, denoted by *time*
  3. A set of items (called *itemset*) involved in the transaction, denoted by *it*

Example

Consider the following database $\mathcal{D}$ with $\mathcal{I} = \{a, b, c, d\}$:

C_1	T_1	a,b,c,d
	T_2	a,b
C_2	T_3	a,b
C_3	T_1	a,d
	T_4	c

Sequence

- Let $\mathcal{I} = \{i_1, i_2 ... i_m\}$ be a set of litterals called *items*.
- A sequence S is an ordered list of itemsets
- Sequence inclusions

Exemple

- $\mathcal{I} = \{a, b, c, d\}$
- $it_1 = (bcd)$, $it_2 = (ab)$
- $S = <(bcd)(ab)>$ (5-sequence)
- $<(bc)(a)> \preceq <(bcd)(ab)>$
- $<(a)(b)> \prec\succ <(b)(ab)>$

Definition (Support)

The support of a sequence S is defined as :

$$Support(S, \mathcal{D}) = \frac{|\{C \in \mathcal{D} | S \preceq C_{trans}\}|}{|\{C \in \mathcal{D}\}|}$$

Sequential Pattern mining

Extract all the frequent sequences S, i.e verifying :

$$Support(S, \mathcal{D}) \geq \sigma$$

with $0 \leq \sigma \leq 1$

Classical and incremental approaches

Classical approaches

1. Levelwise generate-and-prune :
   - SPADE : inverted database representation
   - SPAM : binary representation
2. Pattern-Growth :
   - PrefixSPAN : multiple database projection

Incremental approaches

Taking into account the dynamic evolution of a customer database ISE, ISM and IncSPAN (no deletion)

Remarks

1. Generation : Joint operations are known to be blocking operations [Babcock et al,2002]
2. There is more than 1 pass over $\mathcal{D}$ for all these algorithms however stream mining requires one-pass algorithms

Data streams approaches

- SMDS
- SPEED

Synopsis Construction

Requirements

- Broad Applicability
- One Pass Constraint
- Time and Space Efficiency
- Robustness
- Evolution sensitive

Techniques

1. Sampling Methods like Reservoir Sampling
2. Histograms
3. Wavelets
4. Sketches

Reservoir Sampling (Vitter 1985)

Main idea

An unbiased reservoir is maintained by probabilistic insertions and deletions

- Initialization : the first n points are directly added to the reservoir.
- When the $(t+1)^{th}$ point from the reservoir is received, it is added with a probability $\frac{n}{t+1}$ and replaces a random point in the reservoir.

Observations

- Insertion probabilities reduces with stream progression
- Unbiased reservoir maintained

Disadvantages

- The reservoir may not represent data stream evolutions
- Applications focusing on recent events from the data streams may get inaccurate results
- Smaller and smaller portions of the sample remains relevant with time

Biased Reservoir Sampling (Aggarwal 2006)

Main idea

- Use a temporal bias function to regulate the stream sample.
- This ensures that recent points from the data streams have higher probability to get inserted into the reservoir.

- Helps obtaining a biased and unbiased sample
- The bias is useful for applications focusing on representing the recent behaviour of the data streams

Observations

- An easy to use memory-less bias functions class is the *exponential bias functions* defined as :

$$f(r,t) = e^{-\lambda(t-r)}$$

  The parameter $\lambda \in [0,1]$ defines the bias rate
- The bias function is proportional to $p(r,t)$
- $p(r,t)$ is the probability that a point inserted at the instant r is still belonging to the reservoir when a point arrives at instant t
- In the special case of *exponential bias functions* the maximum reservoir requirement is bounded by $\frac{1}{\lambda}$ for small λ values

Challenges

- All classical mining algorithms have a strong hypothesis stating that a database can be loaded into main memory.
- What about real-world databases containing gigabytes of transactions ?
- Nowadays we can afford approximate solutions but can we assure bounds on the size of the samples given a desired accuracy ?

Sample size

- Error :

$$e(s, \mathcal{S_D}) = |Support(s, \mathcal{S_D}) - Support(s, \mathcal{D})|$$

X_i a random variable for the i^{th} customer with :

- $Pr[X_i = 1] = p_i$ if i^{th} customer supports the sequence s
- $Pr[X_i = 0] = 1 - p_i$, if not.

Note

We are in presence of Poisson trials as the number t of trials in which the probability of success p_i varies from trial to trial.

Sample size

- The number of customers in the sample that supports the sequence s :

$$X(s, \mathcal{S_D}) = \sum_i X_i = Support(s, \mathcal{S_D}) \times |\mathcal{S_D}|$$

- The expected number of customers that support the sequence s in the sample is :

$$E[X(s, \mathcal{S_D})] = Support(s, \mathcal{D}) \times |\mathcal{S_D}|$$

Theorem

Given a sequence s then $Pr[e(s, \mathcal{S_D}) > \epsilon] \leq \delta$ iff the reservoir size is :

$$|\mathcal{S_D}| \geq \ln(\frac{2}{\delta})\frac{1}{2\epsilon^2}$$

Proof sketch

1. Start from $Pr[|Support(s, \mathcal{S_D}) - Support(s, \mathcal{D})| > \epsilon]$
2. introduce $X(s, \mathcal{S_D})$ and $E[X(s, \mathcal{S_D})]$
3. Use chernoff bounds to get the previous result

Observations

- We easily get an (ϵ, δ)-approximation
- Chernoff bound is not always very tight, but in this case it is acceptable
- We get samples of reasonable size with tolerable error :

ϵ	δ	$\mathcal{S_D}$
0.01	0.01	26492
0.01	0.001	38005
0.001	0.01	2649160

Extending to Data Streams : the challenges

- We would like to approximate sequences support by maintaining a dynamic sample
- We would like to have both biased and unbiased sample (user-defined granularity)
- Use biased reservoir approach but with respect to our (ϵ, δ)-approximation

Analysis

We are working on biased reservoir samples, the following corollary gives an upper bound on the bias rate :

Corollary

Given an error bound ϵ and a maximum probability δ that $e(s, \mathcal{S}_{\mathcal{D}}) > \epsilon$ we get an upper bound on the bias rate :

$$\lambda \leq \frac{2\epsilon^2}{ln(2/\delta)}$$

- Proof sketch
  - $|\mathcal{S}_{\mathcal{D}}| \leq \frac{1}{1-e^{-\lambda}}$
  - $|\mathcal{S}_{\mathcal{D}}| \leq \frac{1}{\lambda}$
  - replace in the theorem

Observations

- The bias rate depends of the accuracy we want
- the accuracy of our mining results is optimal when the reservoir is full
- The reservoir maintained is very small in term of space requirements

ϵ	δ	λ	$\mathcal{S}_{\mathcal{D}}$
0.01	0.01	0.0000377	26492
0.01	0.001	0.0000263127	38005
0.001	0.0001	0.0000201949	49518

Algorithm

1. Check if customer C_i is present in the reservoir
2. If no, throw a coin
   - if Success ($< \frac{q}{n}$) add the customer to the reservoir
   - Else replace with a random position in the reservoir
3. If present in the reservoir then add C_i itemset

Observations

We have to show that the replacement policy in the algorithm respects the exponential bias behaviour with $\lambda = \frac{1}{n}$

- Proof sketch
  - Probability that a customer is in the reservoir $\frac{1}{q}$
  - Probability to throw a customer is

$$(1 - \frac{1}{q})(\frac{q}{n})(\frac{1}{q}) = \frac{q-1}{qn}$$

  - If the customer is inserted at the time r and is still in the reservoir at time t, then it did not get ejected in $t - r$ iterations : $(1 - \frac{q-1}{qn})^{t-r}$
  - $(1 - \frac{q-1}{qn})^{t-r} = [(1 - \frac{q-1}{qn})^n]^{\frac{t-r}{n}}$
  - For large value of n, $(1 - \frac{q-1}{qn})^n$ is approximately equal to $\frac{1}{e}$

Experiments

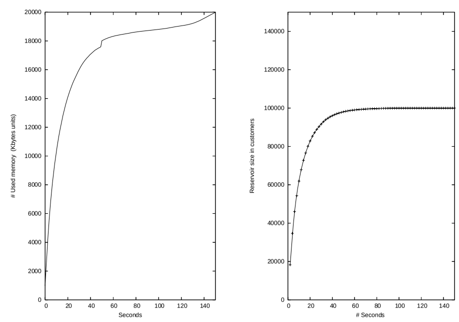

Fig.: Memory requirements for $\lambda = 0.00001$

Summary

- No sampling techniques for sequential patterns mining
- We introduced approximate approaches that work for mining on static databases and we extended it to data streams
- We get a biased sample, quality does not degrade with stream progression
- Extremely easy to implement and easy to maintain (small space requirements depending on bias rate defined by the user)

Plan
Motivations
Preliminary Concepts
Related Work
Sequential Pattern Mining
Synopsis Construction
Sampling in static databases
Extending to Data Streams
Experimental Results
Conclusion and Summary

Thank you for your attention