Zips: Mining Compressing Sequential Patterns in Streams

Hoang Thanh Lam, Toon Calders, Jie Yang
Fabian Moerchen and Dmitriy Fradkin
Agenda

- The problem
- Pattern explosion issue in frequent pattern mining
- Mining compressing patterns: the state of the art solution to the pattern explosion issue
- Mining Compressing patterns in streams
- Scalability
- Take away messages
Agenda

- The Problem
  - Pattern explosion issue in frequent pattern mining
  - Mining compressing patterns: a solution to the pattern explosion issue
  - Mining Compressing patterns in streams
- Experiments
- Take away messages
The Problem

• Given a stream of sequences, e.g. tweet stream, at any given time point return the most important sequential patterns in the stream
• The set of patterns must be non-redundant and meaningful.
Patterns on Tweets Stream

social media

thi weekend thi week thi morn upd at blog
good luck give away iphon app
complaint custom servic rep pleas respond
good morn favorit youtub video

blog post
youtub video

check thi video

Zips algorithm
Agenda

- The Problem
- Pattern explosion issue in frequent pattern mining
  - Mining compressing patterns: a solution to the pattern explosion issue
  - Mining Compressing patterns in streams
- Scalability
- Take away messages
Pattern explosion issue

• Exponential number of frequent patterns, causing redundancy issues
  – Reason: if a set is frequent, all of its subsets are also frequent

• The top most frequent patterns are usually trivial or meaningless.
  – Reason: frequent patterns are combinations of frequent items but unrelated to each other
# Pattern explosion issue

The most frequent closed sequential patterns mined from 7000 abstracts of articles from the Journal of Machine Learning Research (JMLR)

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Support</th>
<th>Pattern</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithm algorithm</td>
<td>0.376</td>
<td>method method</td>
<td>0.250</td>
</tr>
<tr>
<td>learn learn</td>
<td>0.362</td>
<td>algorithm result</td>
<td>0.247</td>
</tr>
<tr>
<td>learn algorithm</td>
<td>0.356</td>
<td>Data set</td>
<td>0.244</td>
</tr>
<tr>
<td>algorithm learn</td>
<td>0.288</td>
<td>learn learn learn</td>
<td>0.241</td>
</tr>
<tr>
<td>data data</td>
<td>0.284</td>
<td>learn problem</td>
<td>0.239</td>
</tr>
<tr>
<td>learn data</td>
<td>0.263</td>
<td>learn method</td>
<td>0.229</td>
</tr>
<tr>
<td>model model</td>
<td>0.260</td>
<td>algorithm data</td>
<td>0.229</td>
</tr>
<tr>
<td>problem problem</td>
<td>0.258</td>
<td>learn set</td>
<td>0.228</td>
</tr>
<tr>
<td>learn result</td>
<td>0.255</td>
<td>problem learn</td>
<td>0.227</td>
</tr>
<tr>
<td>problem algorithm</td>
<td>0.251</td>
<td>algorithm algorithm algorithm</td>
<td>0.222</td>
</tr>
</tbody>
</table>
Agenda

- The Problem
- Pattern explosion issue in frequent pattern mining
- Mining compressing patterns: a solution to the pattern explosion issue
- Mining Compressing patterns in streams
- Scalability
- Take away messages
Mining compressing sequential patterns

• The key idea is based on the Minimum Description Length Principle (MDL): the model that describes the data in the shortest way is the best model.
Pattern mining using MDL (Krimp algorithm Siebes et al. SDM 2006)

- Model: the set of patterns $M$.
- Encoding: compress the data $D$ with the help of model $M$
- Data description length:
  \[ L_M(D) = L(M) + L(D|M) \]
- Find the model $M^*$ that minimizes the data description length:
  \[ M^* = \arg\min_M L_M(D) \]
Pattern mining using MDL, how it works?

• Build a dictionary (a set of patterns)
• Encode the data given the dictionary (replace occurrences of patterns in the data by pointers to the dictionary)
• Pointers are represented by binary codewords. Shorter codewords are assigned to pointers with more usage.
Pattern mining using MDL, how it works?

### Table 1: Codeword lengths

<table>
<thead>
<tr>
<th>word</th>
<th>Codeword $C(w)$</th>
<th>usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>b</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>c</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>d</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>abc</td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

**Encoded sequence**

```
abc   d   abc   e   abc
```

Codeword length is proportional to $-\log(\text{sum/usage})$, i.e.
Shorter codeword is assigned to more frequently used patterns.
Pattern mining using MDL, how hard?

• Finding an optimal dictionary and an optimal encoding given a dictionary is NP-hard (Lam et al. SDM 2012, SADM 2013)

• The state of the art approaches are based on greedy algorithm: grow the dictionary greedily (step by step add to the dictionary the next pattern that results in the most compression benefit).
It solved the pattern explosion issues

<table>
<thead>
<tr>
<th>Method</th>
<th>Patterns</th>
</tr>
</thead>
</table>
| SQS (Vreeken et al. KDD 2012) | support vector machine  
state art  
data set  
bayesian network  
larg scale  
nearest neighbor  
decis tree  
neural network  
cross valid  
featur select  
graphic model  
real world  
high dimension  
mutable inform  
sampl size  
learn algorithm  
princip compon analysi  
logist regress  
model select |
| GOKRIMP (Lam et al. SDM 2012, SADM Journal 2013) | support vector machine  
real world  
machin learn  
data set  
bayesian network  
state art  
high dimension  
reproduc hilbert space  
larg scale  
independ compon analysi  
neural network  
experiment result  
sampl size  
supervis learn  
support vector  
well known  
special case  
solv problem  
signific improv  
object function |
| Zips (This work)  | support vector machin  
data set  
real world  
learn algorithm  
state art  
featur select  
machine learn  
bayesian network  
model select  
optim problem  
high dimension  
paper propose  
graphic model  
larg scale  
result show  
cross valid  
decis tree  
neutral network  
well known  
hilbert space |

Meaningful patterns are extracted by the MDL based pattern mining approaches (SQS by Vreeken et al. KDD 2012, GoKrimp by Lam et al. SDM 2011 and Zips-this work) from the JMLR dataset
Agenda

- The Problem
- Pattern explosion issue in frequent pattern mining
- Mining compressing patterns: a solution to the pattern explosion issue
- Mining Compressing patterns in streams
- Scalability
- Take away messages
Data streams

• Infinite stream of sequences, for example, tweets stream, search engine query log, machine message log etc.
Compressing sequential patterns mining in Data streams

• Challenges:
  1. Single pass constraint
  2. Memory constraint
  3. High speed updates
Our Algorithm

S1  S2  S3  Data stream  S_t

encoding

Greedy Encoder

Dictionary

Update
Key technical contributions

• A new encoding called reference encoding for streams, usage is approximately updated in a single pass through data

• Space saving algorithm to keep the size of the dictionary always below a predefined threshold
Reference encoding

- Instead of using pointers to dictionary, use references to the most recently encoded instance of the pattern.
Reference encoding

• Benefits of using reference encoding:
  1. Efficient dictionary update: no need to recalculate usages for all the words in the dictionary per update
  2. Theoretical guarantee: in average, the codeword length of references provably converges to the codeword length assigned based on usage with assumption that the encoded instances of patterns are independent.
Space saving algorithm

• Dictionary extension: add to the dictionary a new candidate pattern by appending the current pattern with the following item.

• Dictionary is full: replace the pattern with the least overestimated usage by the new candidate patterns.
Agenda

- The Problem
- Pattern explosion issue in frequent pattern mining
- Mining compressing patterns: a solution to the pattern explosion issue
- Mining Compressing patterns in streams
- Scalability
- Take away messages
Scalability

- Zips scales linearly with the size of the tweet stream
- GoKrimp and SQS scale quadratically
Agenda

- The Problem
- Pattern explosion issue in frequent pattern mining
- Mining compressing patterns: a solution to the pattern explosion issue
- Mining Compressing patterns in streams
- Scalability
- Take away messages
Take away messages

• We solved the mining non-redundant sequential patterns problem in streams
• The quality of the patterns extracted by our solution is similar to patterns extracted by the state of the art algorithms.
• Our solution scales linearly with the size of data while the state of the art algorithms do not
Pattern Visualization on Stream

- We used wordcloud tool to visualize patterns extracted from sliding windows. (demo)
- Wordcloud shows important patterns in each snapshot but doesn’t show how importance score of patterns change overtime.
- Need a better visualization tool for stream!