The Long and the Short of It

summarising event sequences with serial episodes

Nikolaj Tatti & **Jilles Vreeken**

Question of the day

How can we discover

the key patterns from an event sequence?

Summarising Event Sequences

The ideal outcome of pattern mining

- patterns that show the structure of the data
- preferably a small set, without redundancy or noise

Frequent pattern mining does not achieve this

- pattern explosion → overly many, overly redundant results

We pursue the ideal for serial episodes

- we want a group of patterns that summarise the data well
- we take a **pattern set** mining approach

Event sequences

Alphabet Ω $\{\, a, b, c, d, \ldots \,\}$

Data D $a \;\; b \;\; d \;\; c \;\; a \;\; d \;\; b \;\; a \;\; a \;\; b \;\; c \;\; a \;\; d \;\; a \;\; b \;\; a \;\; b \;\; c$

one, or multiple sequences

$$\{\, a \;\; b \;\; d \;\; c \;\; a \;\; d \;\; b \;\; a \;\; a \;\; b \;\; c\,,$$
$$a \;\; b \;\; d \;\; c \;\; a \;\; d \;\; b\,,$$
$$a \;\; b \;\; d \;\; c \;\; a \;\; d \;\; b \;\; a \;\; a\,, \ldots \,\}$$

Event sequences

Alphabet Ω $\{ a, b, c, d, \ldots \}$

Data D $a\ b\ d\ c\ a\ d\ b\ a\ a\ b\ c\ a\ d\ a\ b\ a\ b\ c$

one, or multiple sequences

$\{ a\ b\ d\ c\ a\ d\ b\ a\ a\ b\ c,$
$a\ b\ d\ c\ a\ d\ b,$
$a\ b\ d\ c\ a\ d\ b\ a\ a, \ldots \}$

Patterns

serial episodes $a \rightarrow b$

'subsequences allowing gaps'

Event sequences

Alphabet Ω $\{ a, b, c, d, \ldots \}$

Data D $a\ b\ d\ c\ a\ d\ b\ a\ a\ b\ c\ a\ d\ a\ b\ a\ b\ c$

one, or multiple sequences

$\{ a\ b\ d\ c\ a\ d\ b\ a\ a\ b\ c,$
$a\ b\ d\ c\ a\ d\ b,$
$a\ b\ d\ c\ a\ d\ b\ a\ a, \ldots \}$

Patterns

serial episodes

$a \to b$

$d \to b \to c$

'subsequences allowing gaps'

Summarising Event Sequences

We want to find good summaries.

Three important questions

1. how do we **score** a pattern-based summary?
2. how do we **describe** a sequence given a pattern set?
3. how do we **find** good pattern sets?

Summarising Event Sequences

We want to find good summaries.

Three important questions

1. how do we **score** a pattern-based summary?
2. how do we **describe** a sequence given a pattern set?
3. how do we **find** good pattern sets?

Scoring a Summary

We want models that generalise the data

and hence, we need a score that

- rewards models that identify real structure, and
- punishes redundancy and noise

No off-the-shelf score available for serial episodes

- e.g. no well-founded priors
- we can, however, make these goals concrete by **MDL**

MDL

The Minimum Description Length (MDL) principle

given a set of models $\mathcal{M}$, the best model $M \in \mathcal{M}$
is that M that minimises

$$L(M) + L(D|M)$$

in which

$L(M)$ is the length, in bits, of the description of M

$L(D|M)$ is the length, in bits, of the description of the data when encoded using M

(see, e.g., Rissanen 1978, 1983, Grünwald, 2007)

MDL for Event Sequences

By MDL we define

the optimal set of serial episodes as the set that describes the data most succinctly

To use MDL, we need

- a lossless encoding for our models,
- a lossless encoding for the data given a model

(for itemsets, see Siebes et al, 2006, Vreeken et al 2011)

Models

As models we use **code tables**

- dictionaries of patterns and associated codes

pattern	code	gap	non-gap
abc	p	?	!
da	q	?	!
a	a		
b	b		
c	c		
d	d		

Encoding Event Sequences

Data D: $a \ b \ d \ c \ a \ d \ b \ a \ a \ b \ c$

Encoding 1: using only singletons

C_p $\boxed{a}\boxed{b}\boxed{d}\boxed{c}\boxed{a}\boxed{d}\boxed{b}\boxed{a}\boxed{a}\boxed{b}\boxed{c}$

CT_1:

a	$\boxed{a}$
b	$\boxed{b}$
c	$\boxed{c}$
d	$\boxed{d}$

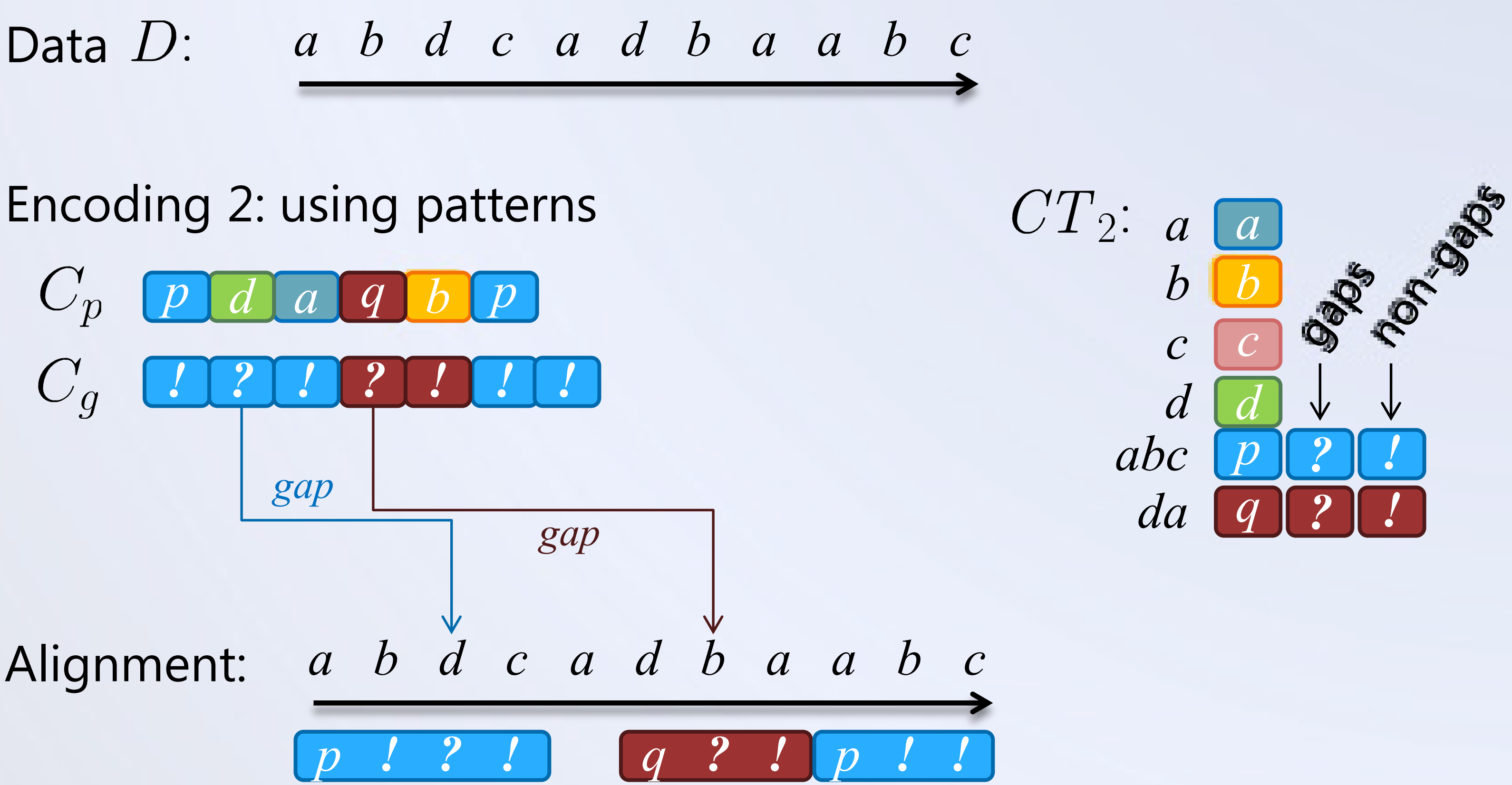
The length of the code $\boxed{X}$ for pattern X

$$L(\boxed{X}) = -\log(p(\boxed{X})) = -\log\left(\frac{usg(X)}{\sum usg(Y)}\right)$$

The length of the code stream

$$L(C_p) = \sum_{X \in CT} usg(X) L(\boxed{X})$$

Encoding Event Sequences



Encoding Event Sequences

Data D: $a \quad b \quad d \quad c \quad a \quad d \quad b \quad a \quad a \quad b \quad c$

Encoding 2: using patterns

C_p $\boxed{p}\boxed{d}\boxed{a}\boxed{q}\boxed{b}\boxed{p}$

C_g $\boxed{!}\boxed{?}\boxed{!}\boxed{?}\boxed{!}\boxed{!}\boxed{!}$

CT_2:

		gaps	non-gaps
a	a		
b	b		
c	c		
d	d		
abc	p	$?$	$!$
da	q	$?$	$!$

The length of a gap code $\boxed{?}$ for pattern X

$$L(\boxed{?}) = -\log(p(\boxed{?} \mid \boxed{p}))$$

and analogue for non-gap codes $\boxed{!}$

Encoding Event Sequences

By which, the encoded size of D given CT and C is

$$L(D \mid CT) = L(C_p \mid CT) + L(C_g \mid CT)$$

...skipping the details of $L(CT \mid C)$...

Then, our goal is to minimise

$$L(CT, D) = L(CT \mid C) + L(D \mid CT)$$

Summarising Event Sequences

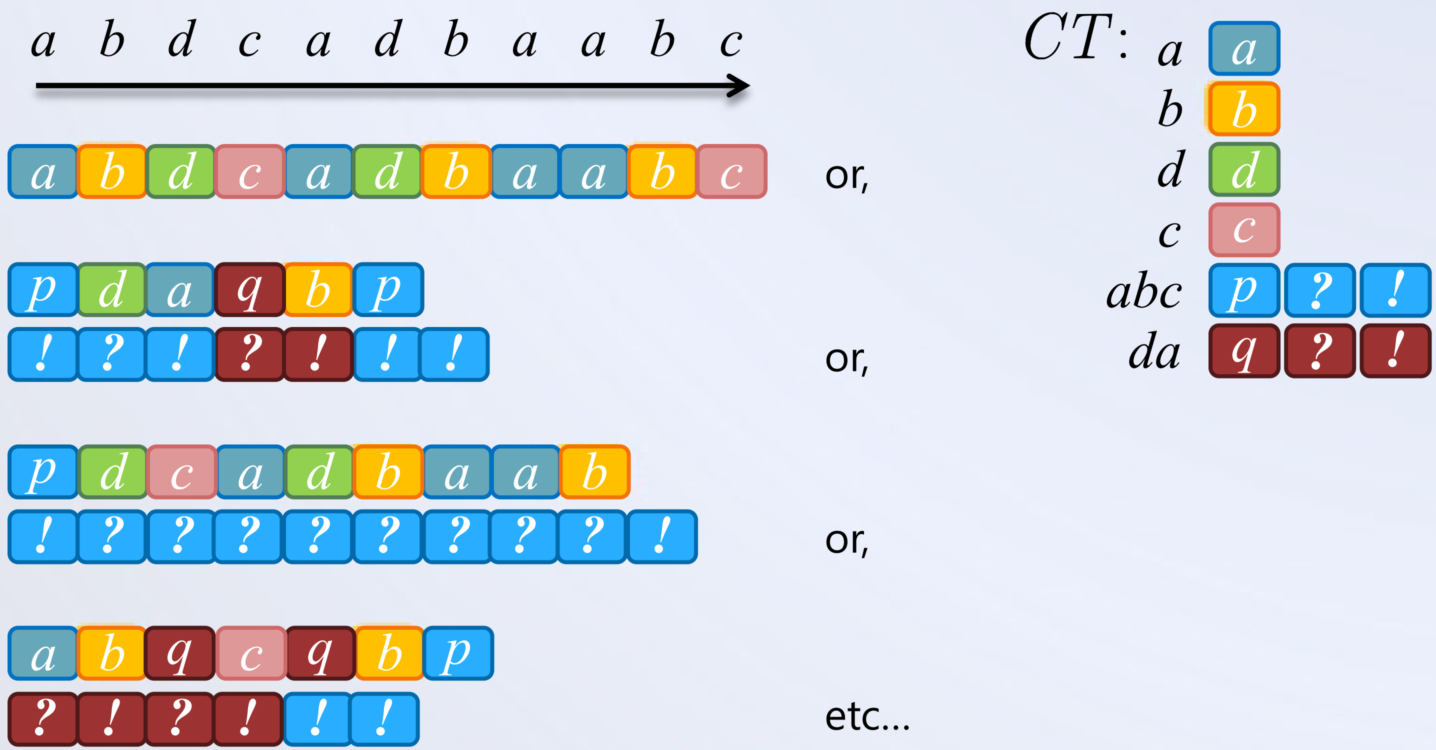
We want to find good summaries.

Three important questions

1. how do we **score** a summary?
2. how do we **describe** a sequence given a pattern set?
3. how do we **find** good pattern sets?

How to Cover your String

There are many ways C to describe a sequence given a set of patterns. We are after the **optimum.**



How to Cover your String

There are many ways C to describe a sequence given a set of patterns. We are after the **optimum.**

1. if we fix the **cover**, we can obtain the optimal code lengths
2. if we fix the **code lengths**, we can obtain the optimal cover by dynamic programming

We alternate these steps until **convergence**

How to Cover your String

压缩

How to Cover your String

SQS

Summarising event seQuenceS

Summarising Event Sequences

We want to find good summaries.

Three important questions

1. how do we **score** a summary?
2. how do we **describe** a sequence given a pattern set?
3. how do we **find** good pattern sets?

Mining Code Tables

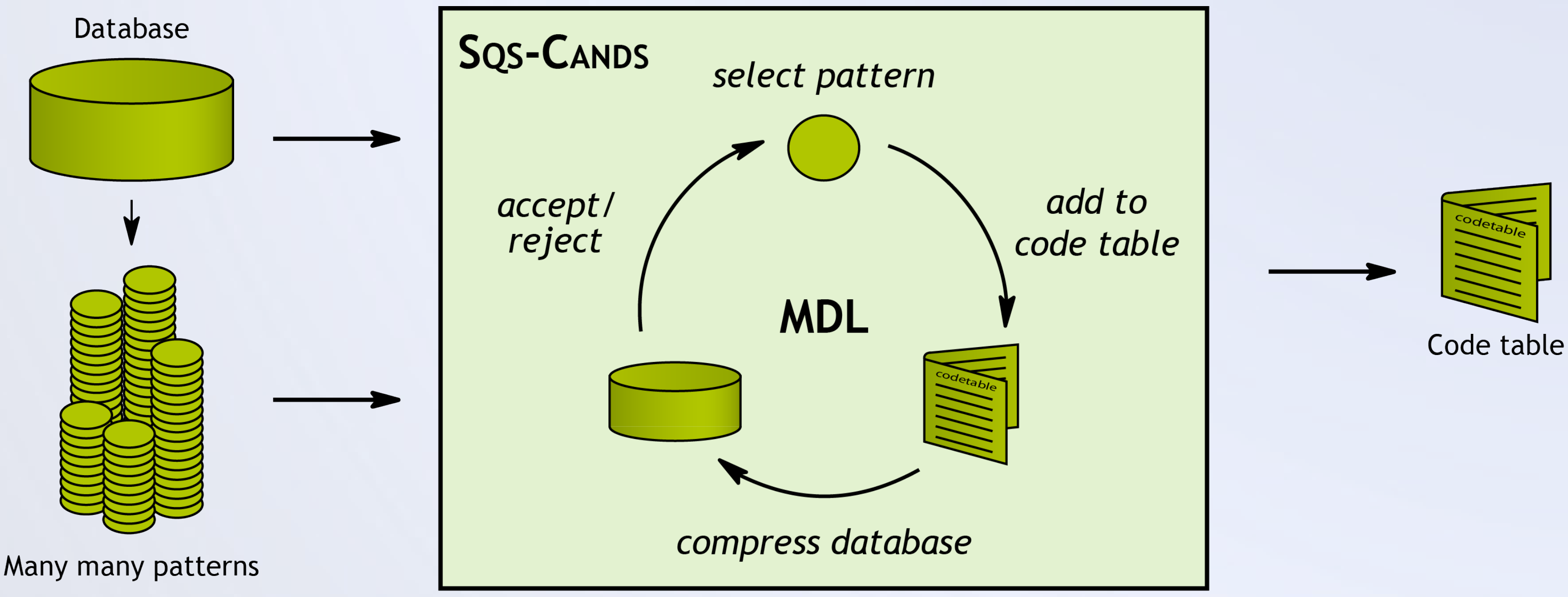
There are very many possible pattern sets.
We are after the **optimum**

However, the search space is huge, complex, and does **not** exhibit trivial structure

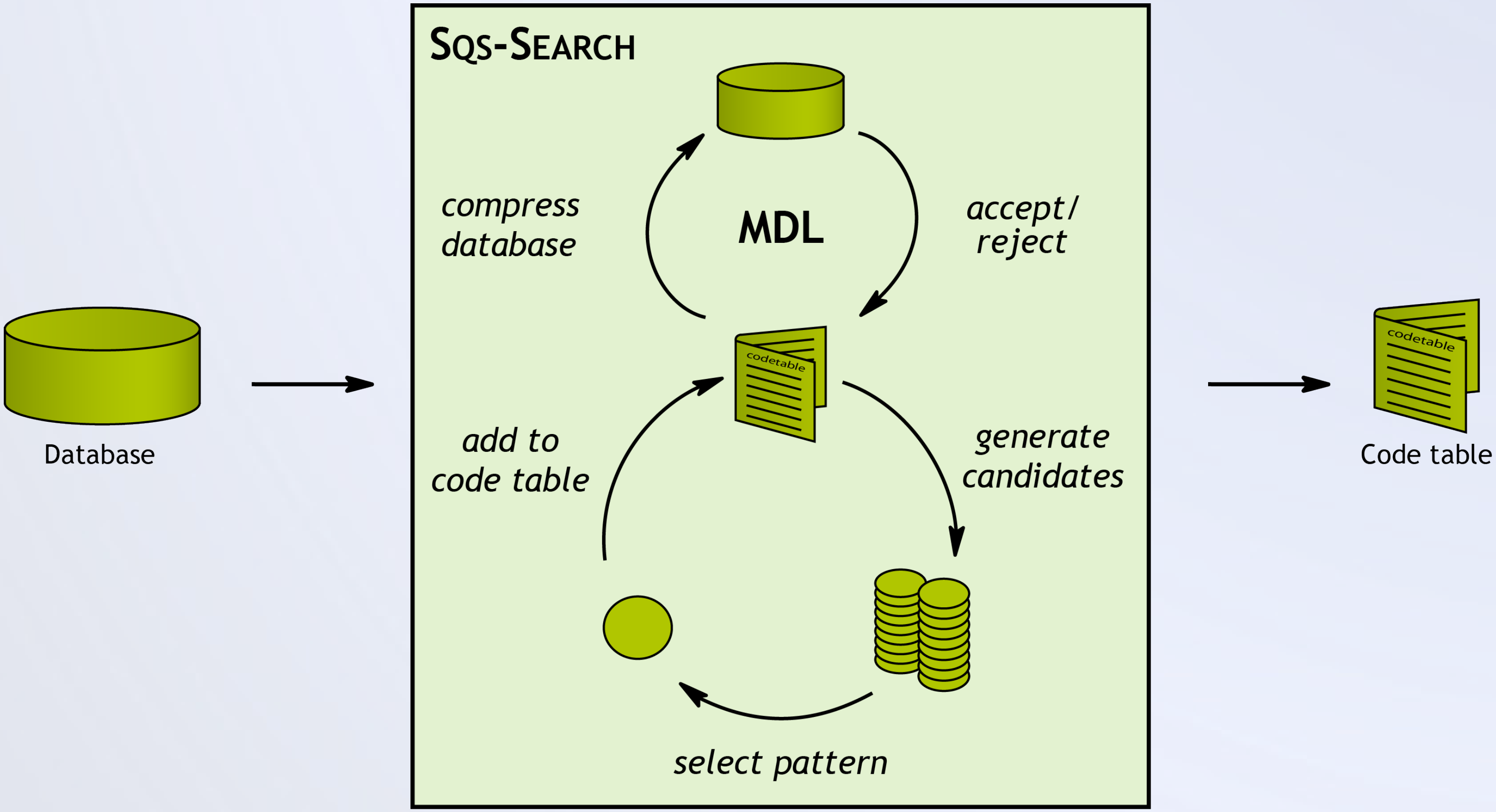
We propose two algorithms for mining code tables

- **SQS-CANDS** filters ordered lists of pre-mined candidates
- **SQS-SEARCH** mines good code tables directly from data

SQS-CANDIDATES



SQS-SEARCH


Database
SQS-SEARCH
compress database
MDL
accept/ reject
codetable
add to code table
generate candidates
select pattern
codetable
Code table

Experiments

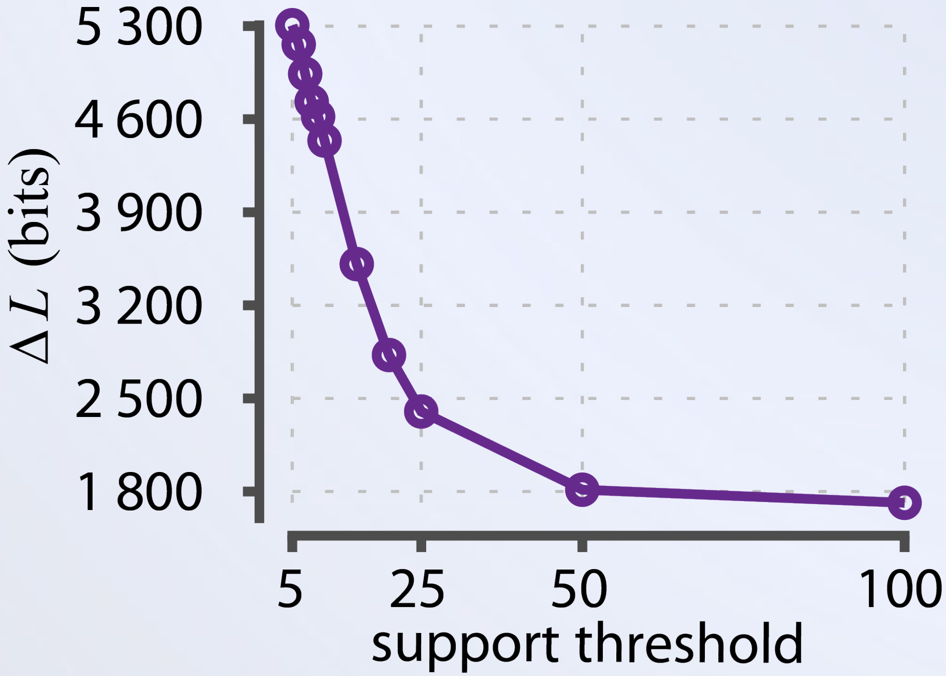
- **synthetic data** random ✓ no structure found
  HMM ✓ structure recovered
- **real data** text data for interpretation

	$\lvert\Omega\rvert$	$\lvert D\rvert$	SQS-CANDS $\lvert F\rvert$	SQS-CANDS $\lvert P\rvert$	SQS-SEARCH $\lvert P\rvert$	ΔL
Addresses	5 295	56	15 506	138	155	5k
JMLR	3 846	788	40 879	563	580	30k
Moby Dick	10 277	1	22 559	215	231	10k

(implementation available at http://adrem.ua.ac.be/sqs)

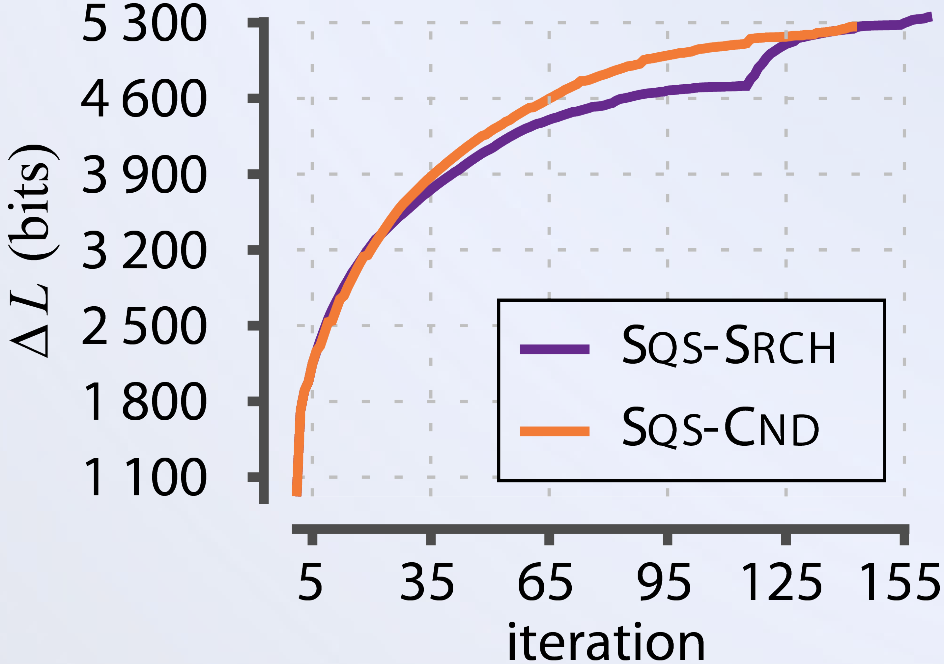
SQS-CANDIDATES

Compression improves with richer candidate sets
i.e. lower support thresholds



Optimising our Score

Both strategies show good convergence
- SQS-SEARCH dips due to batch-wise search



Selected Results

JMLR

support vector machine
machine learning
state [of the] art
data set
Bayesian network

Pres. Addresses

unit[ed] state[s]
public econ. expenditur
take oath
equal right
exercis power

(for Sqs-Search)

Conclusions

Mining informative sets of patterns

- is an important aspect of exploratory data mining

SQS approximates the ideal for serial episodes

- SQS-CANDS filters a pre-mined candidate list
- SQS-SEARCH mines good code tables directly from data

Future work includes

- richer data and pattern types
- applying SQS in real-world settings

(implementation available at http://adrem.ua.ac.be/sqs)

Thank you!

Mining informative sets of patterns

- is an important aspect of exploratory data mining

SQS approximates the ideal for serial episodes

- SQS-CANDS filters a pre-mined candidate list
- SQS-SEARCH mines good code tables directly from data

Future work includes

- richer data and pattern types
- applying SQS in real-world settings

(implementation available at http://adrem.ua.ac.be/sqs)