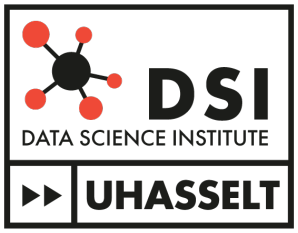
Temporal Pattern Discovery for Scalable and Interpretable Time Series Analysis

Len Feremans
Hasselt University
Data Science Institute



Some research questions

- Goal is to find and report interesting patterns towards end-users
  - What patterns are interesting?
  - How can we find interesting patterns efficiently?
  - How can we use patterns to make more accurate predictions?

Outline

1. **Pattern Mining**

2. **Representation**
   – Interpretable analysis of time series using patterns embeddings

3. **Motif discovery**
   – Finding motifs in time series

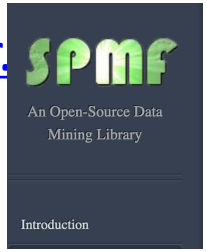
Pattern Mining

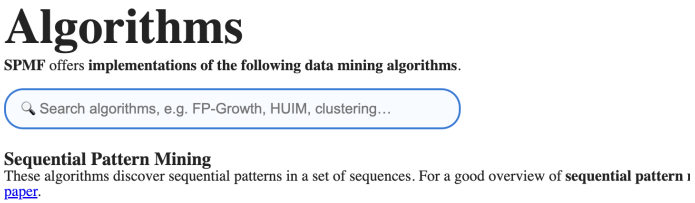
– What, why, how of pattern mining?

– Applications

Why pattern mining?

- **Exploratory data analysis:** Gems in every dataset waiting to be discovered
- **Extract interpretable features**
  - Intrinsic interpretable representations and models
  - Using rules as post-hoc explainable model
- **Human in the loop discovery**
  - Add constraints and domain knowledge
- **Large amount of** prior research
  - Hundreds of efficient algorithms
  - See https://www.philippe-fournier-viger.





What is Pattern?

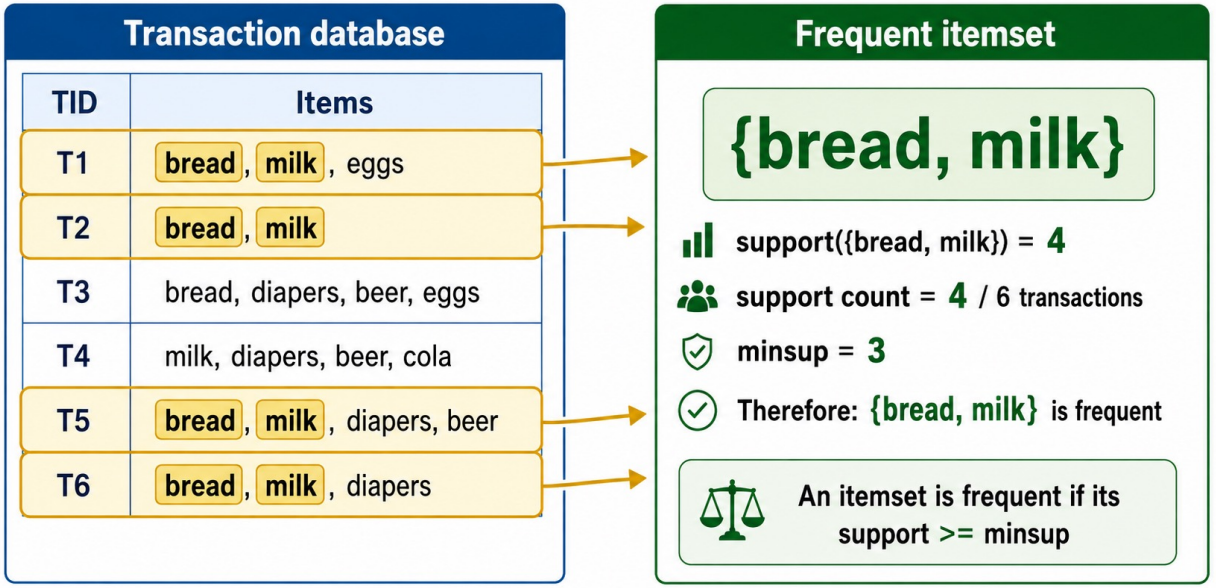
- An **itemset**: X={A,B}
  - Examples where both A and B co-occur
- A **sequential pattern**: X=[A,B]
  - Examples where A occurs before B
- A **rule**: X = A → B
  - If A occurs in an example, B also occurs with a certain likelihood
- A time series **motif**: X=(A,B)
  - A set of subsequences with high similarity

Properties of patterns

- **Large search space**
  - Singletons: {A},{B},… N
  - Pairs: {A,B}, {B,C}, {A,C} $\binom{N}{2}$
  - Triples: {A,B,C} …. $\binom{N}{3}$
  - Possible itemsets 2^N
- **Monotonicity**
  - If {A,B,C} occurs in **S** examples then any subset occurs at least **S** times
- **Constraints**
  - Limit on size, e.g. k=3
  - Limit on number of patterns, e.g. top-k=10.000
  - Gap constraints
- **Value of pattern**
  - Frequency
  - Confidence
  - Pos. or neg. for label
  - Cohesive

Applications

- **Frequent Itemset Mining** for Market-basket analysis [Agrawal, 1994]



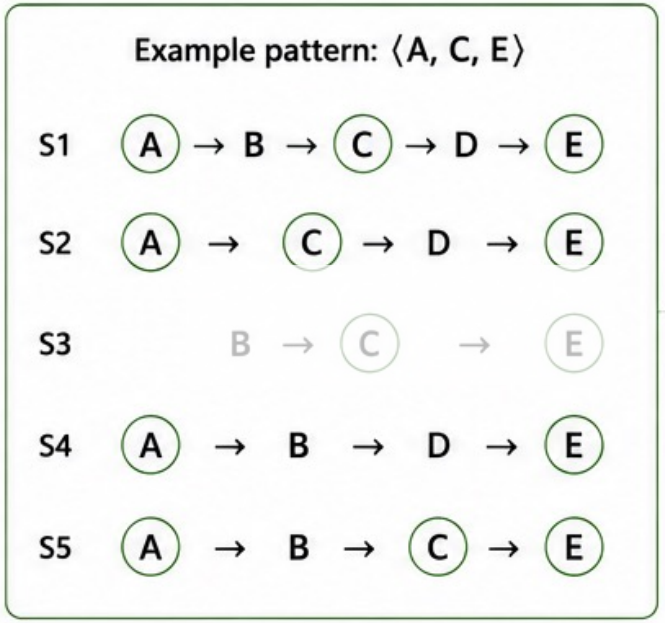
Applications

- **Sequential pattern mining**

1. Input: Sequence Database

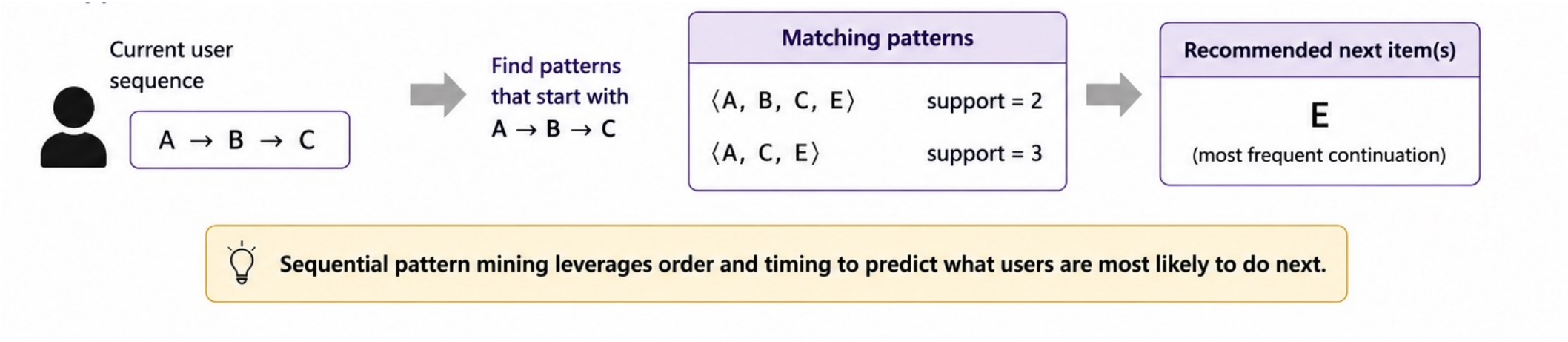
Session ID	Sequence of items (in time order)
S1	A → B → C → D → E
S2	A → C → D → E
S3	B → C → E
S4	A → B → D → E
S5	A → B → C → E

Items: A, B, C, D, E (e.g., songs, videos, products)



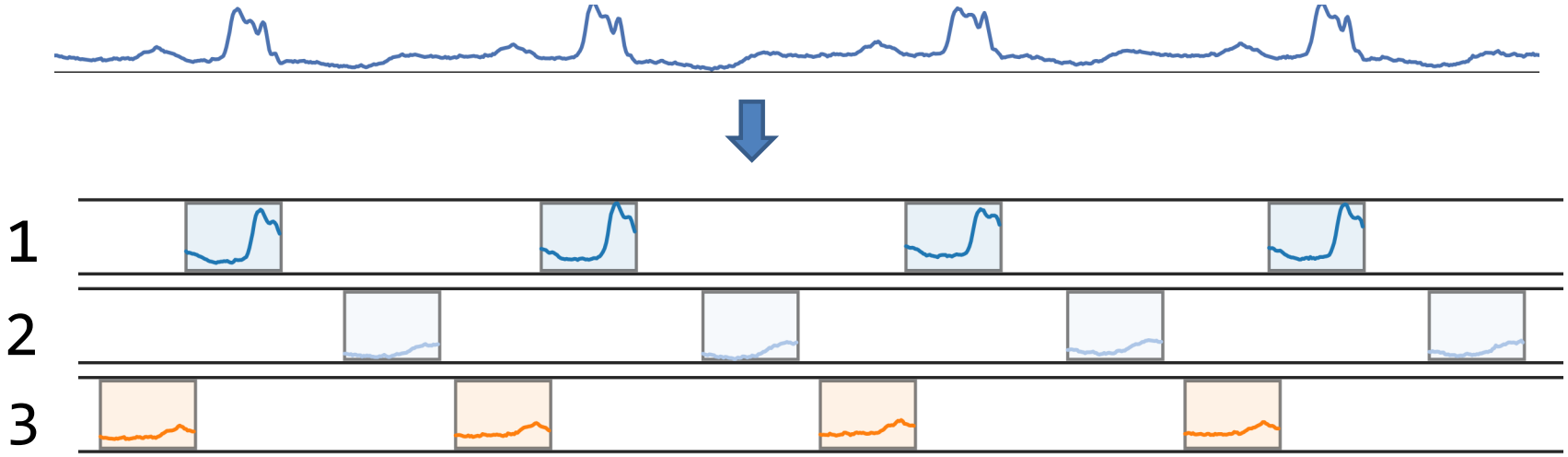
Applications

- Recommender systems



Applications

- Time series motifs
  - Repeating (non-overlapping) subsequences with high similarity



Applications

- Time series (hearth rate, blood pressure, oxygen)
- Events (alarms)

→ Rule mining → Insights → Smart alarms

(e)	$\{HF < 60, 28 < RF < 31\} \rightarrow \{intubation\}$	0.55	0.02
(f)	$\{74 < HF < 86, RF > 30\} \rightarrow \{intubation\}$	0.53	0.03
(g)	$\{ABP_d < 47, RF > 30\} \rightarrow \{intubation\}$	0.52	0.05
(h)	$\{ABP_d < 45, ABP_m < 64, RF > 31\} \rightarrow \{intubation\}$	0.51	0.02
(i)	$\{intubation, ABP_d < 45,\ 91 < S_pO_2 < 93\} \rightarrow \{ABP_m < 64\}$	0.51	0.02

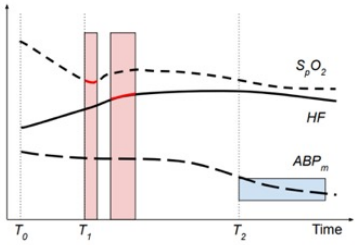


Fig. 1: Example of multivariate time series of a patient with alarms. S_pO_2 oxygen saturation, HF heart frequency, ABP_m arterial blood pressure.

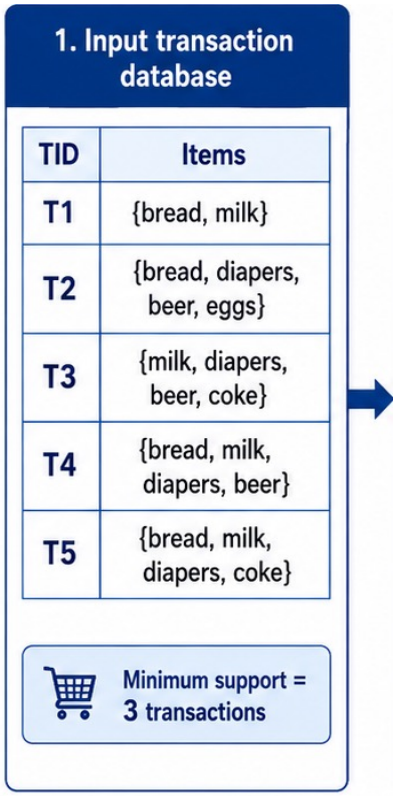
HF < 60 and
28 < RF < 32?

55% chance of intubation

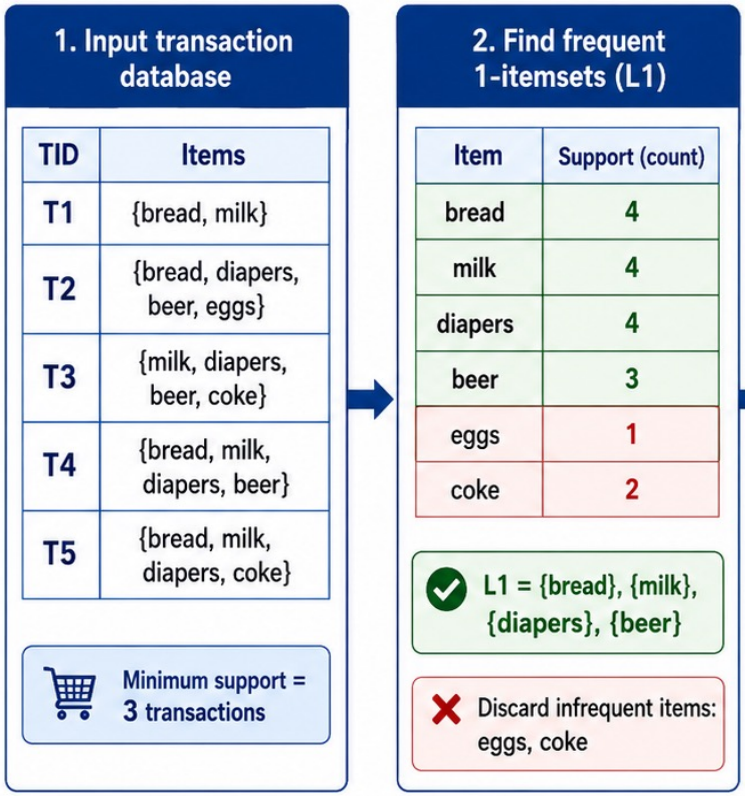
T_0 T_1 Time

Occurs in 0.2% of patients
(+- 15 patients) [Venturini, 2025]

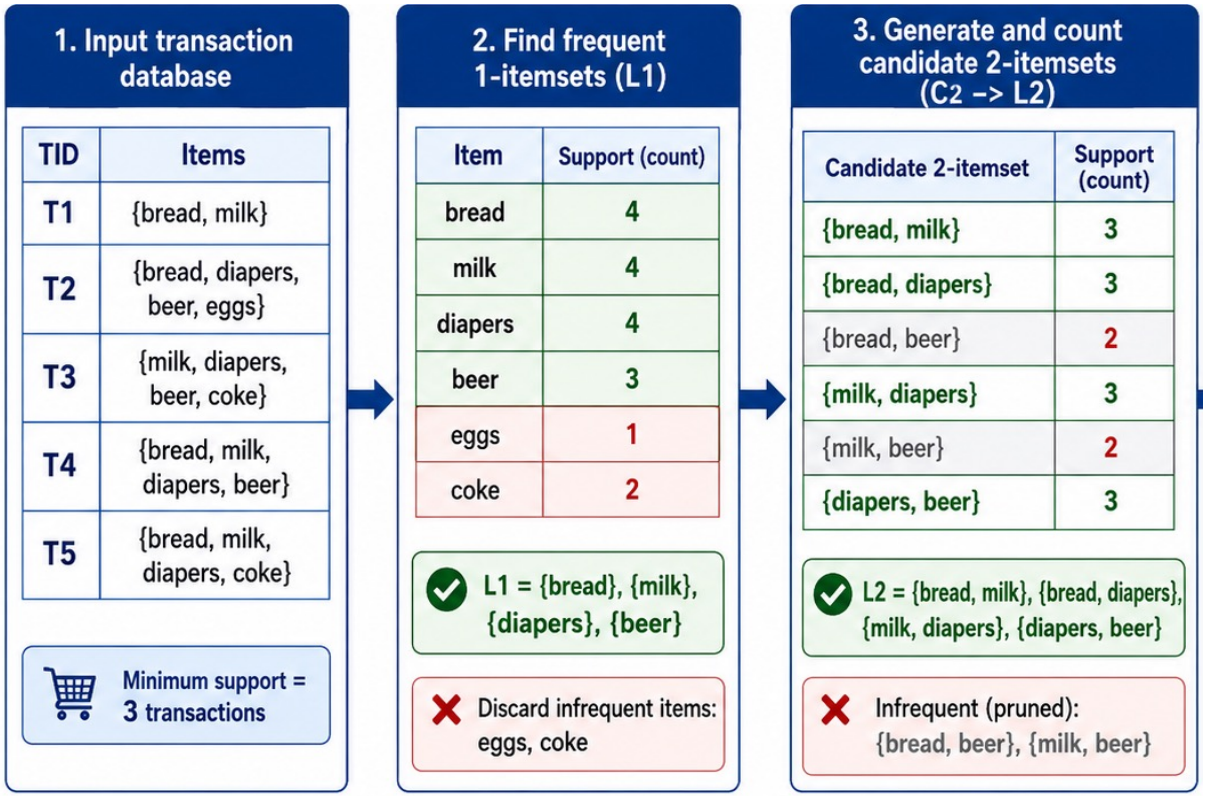
Apriori algorithm for frequent itemset mining



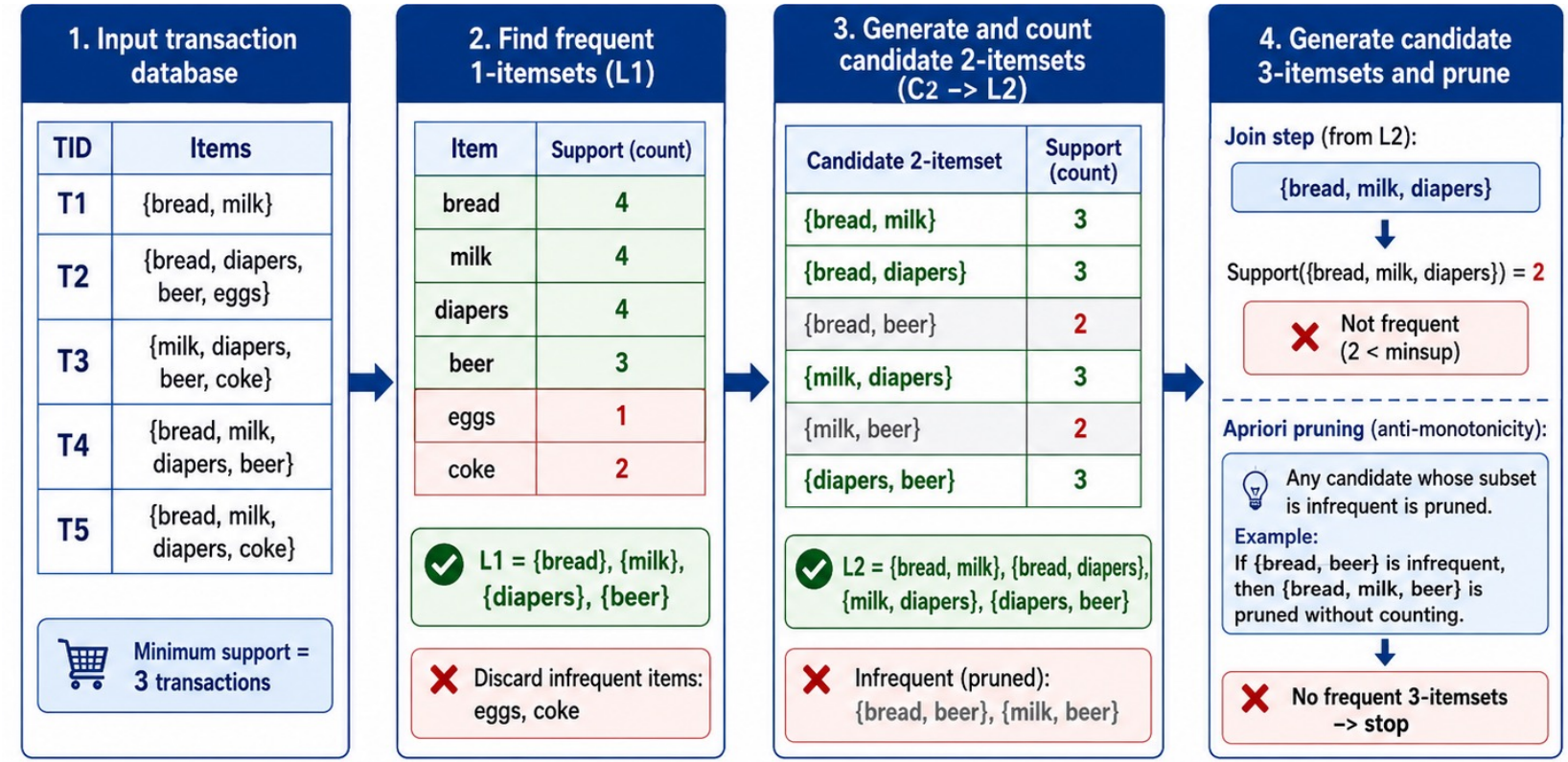
Apriori algorithm for frequent itemset mining



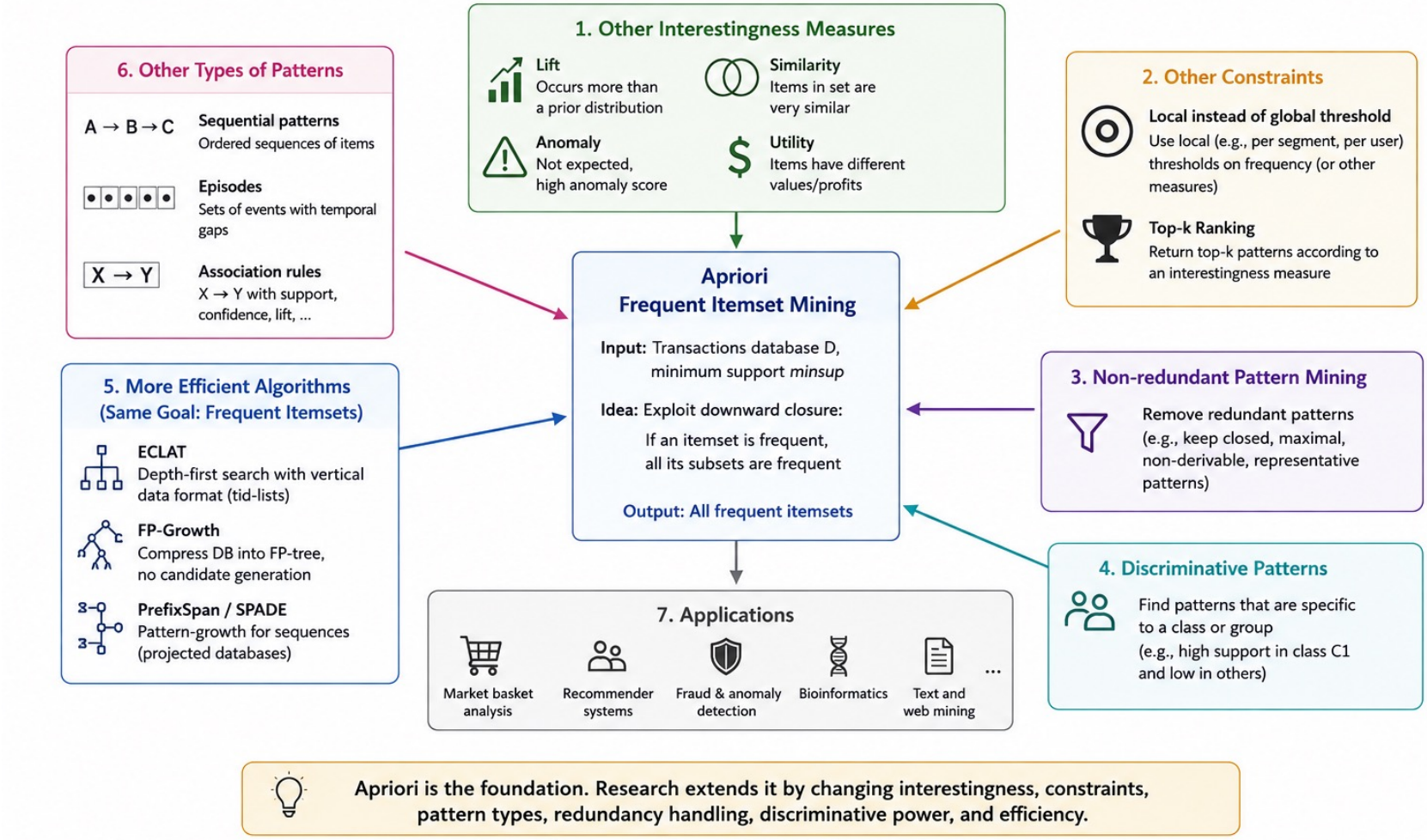
Apriori algorithm for frequent itemset mining



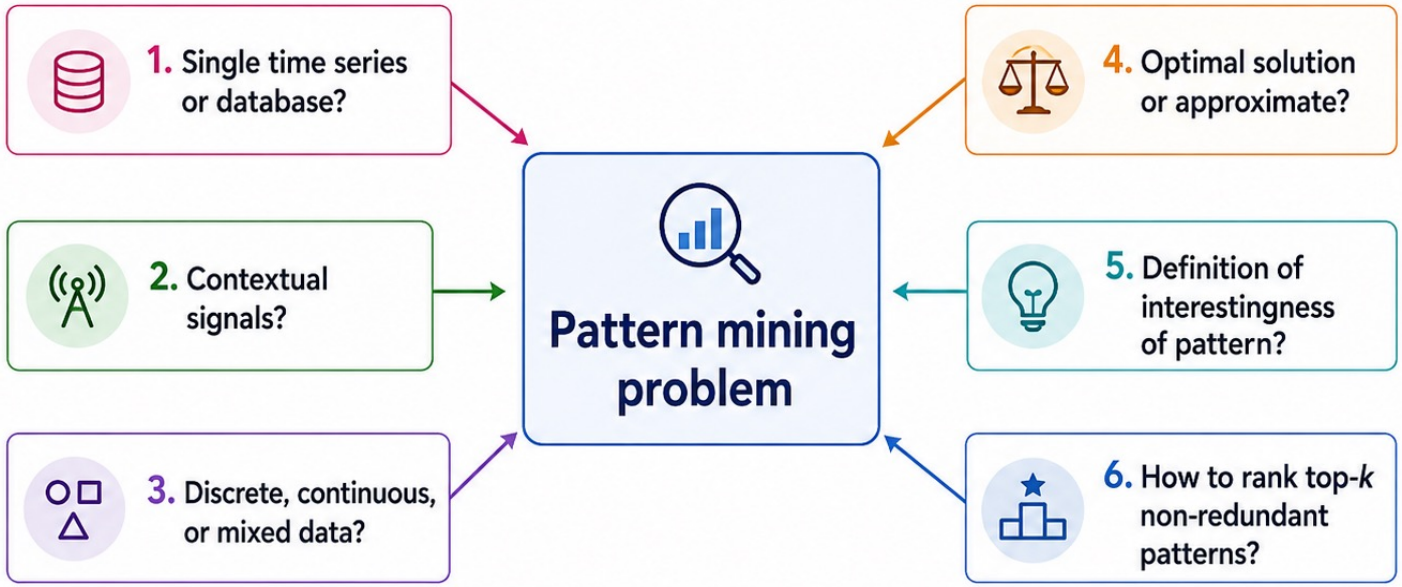
Apriori algorithm for frequent itemset mining



Extensions to Apriori



Bigger picture: pattern mining problems

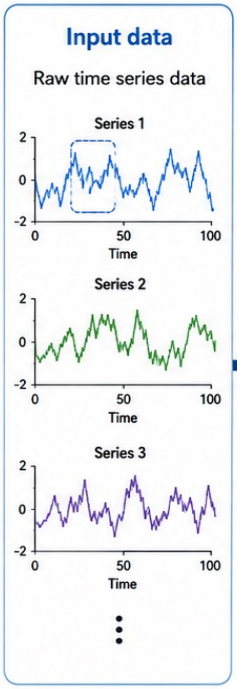


Many design choices determine the right pattern mining approach.

Interpretable and scalable analysis of TS using **patterns embeddings**

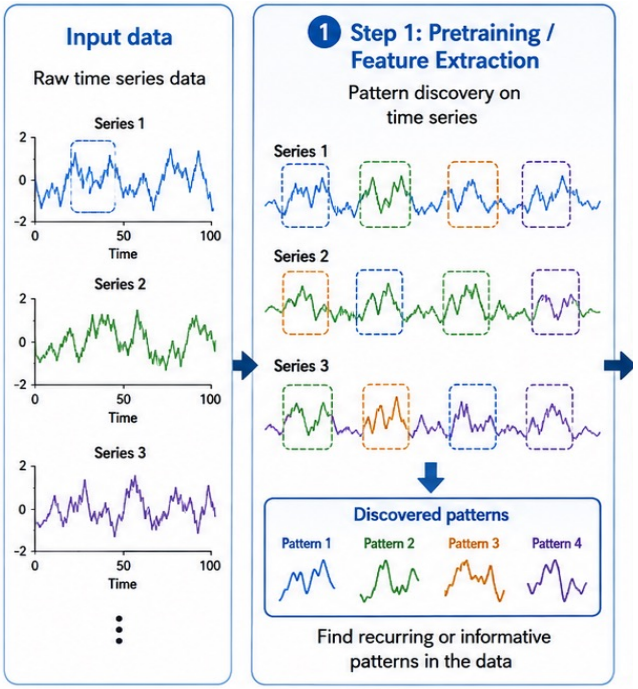
- **PBAD** [Feremans, 2019]**:**
  - **Anomaly detection** in mixed-type event sequences
- **PETSC** [Feremans, 2022]
  - **Time Series Classification**

What are pattern-based embeddings



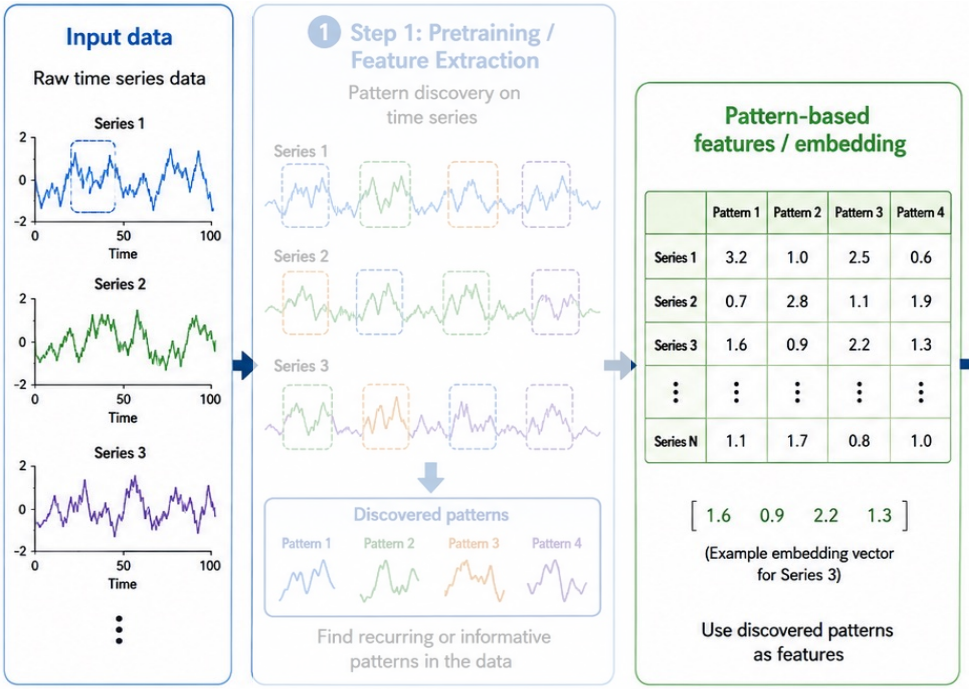
First discover patterns, then use them as features for prediction.

What are pattern-based embeddings



First discover patterns, then use them as features for prediction.

What are pattern-based embeddings

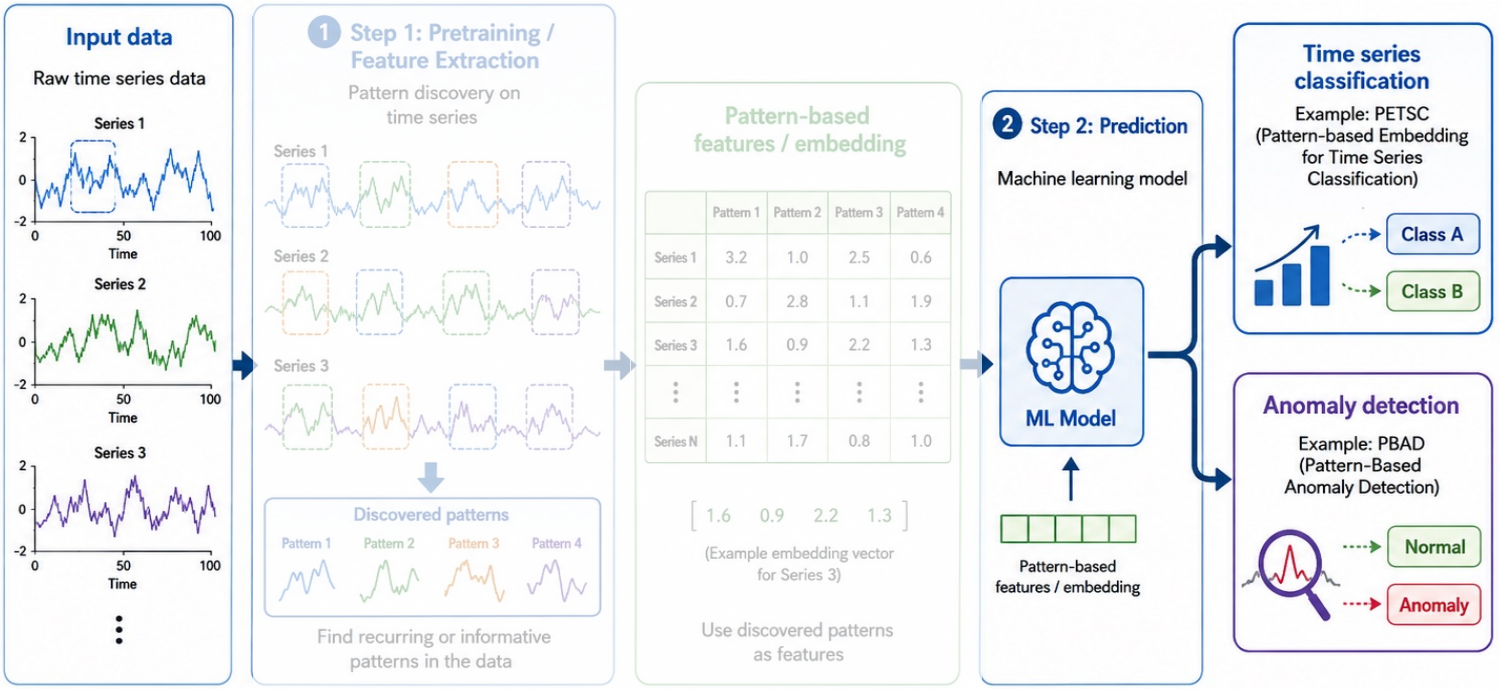


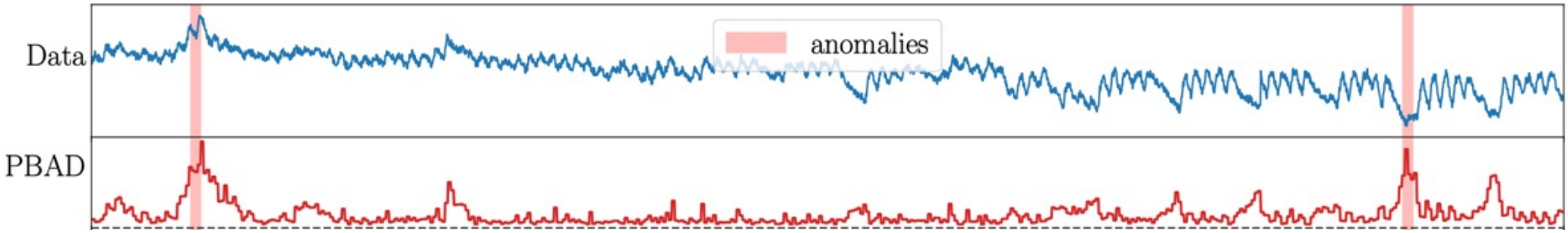
First discover patterns, then use them as features for prediction.

What are pattern-based embeddings

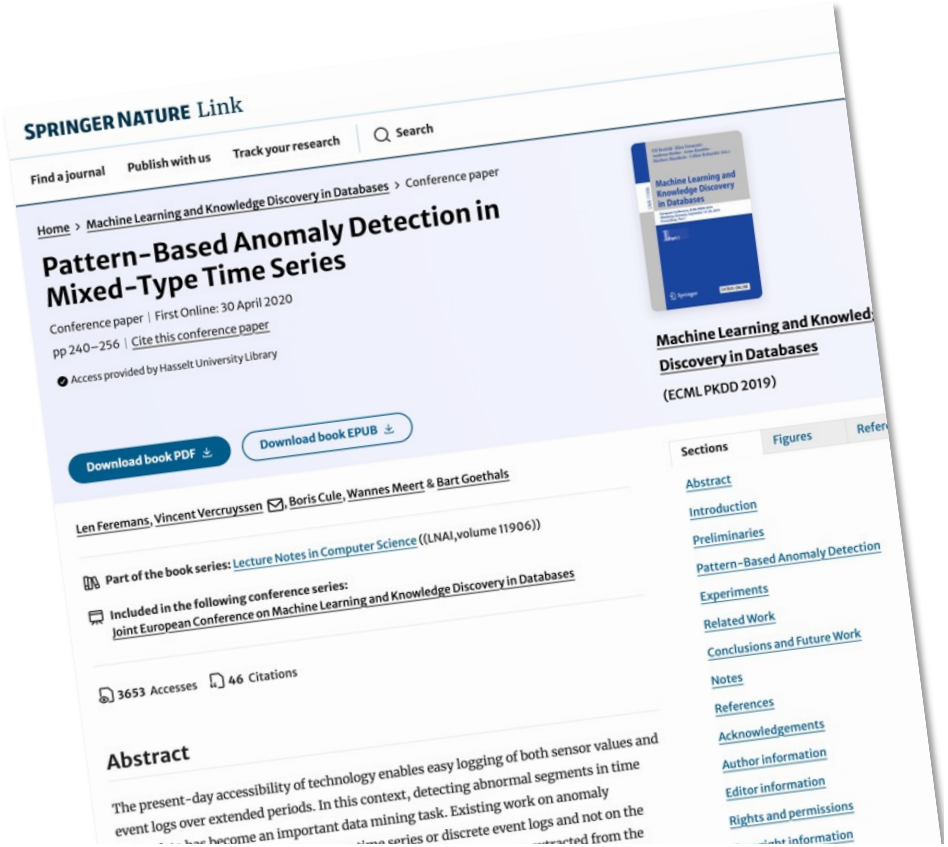
First discover patterns, then use them as features for prediction.

What are pattern-based embeddings

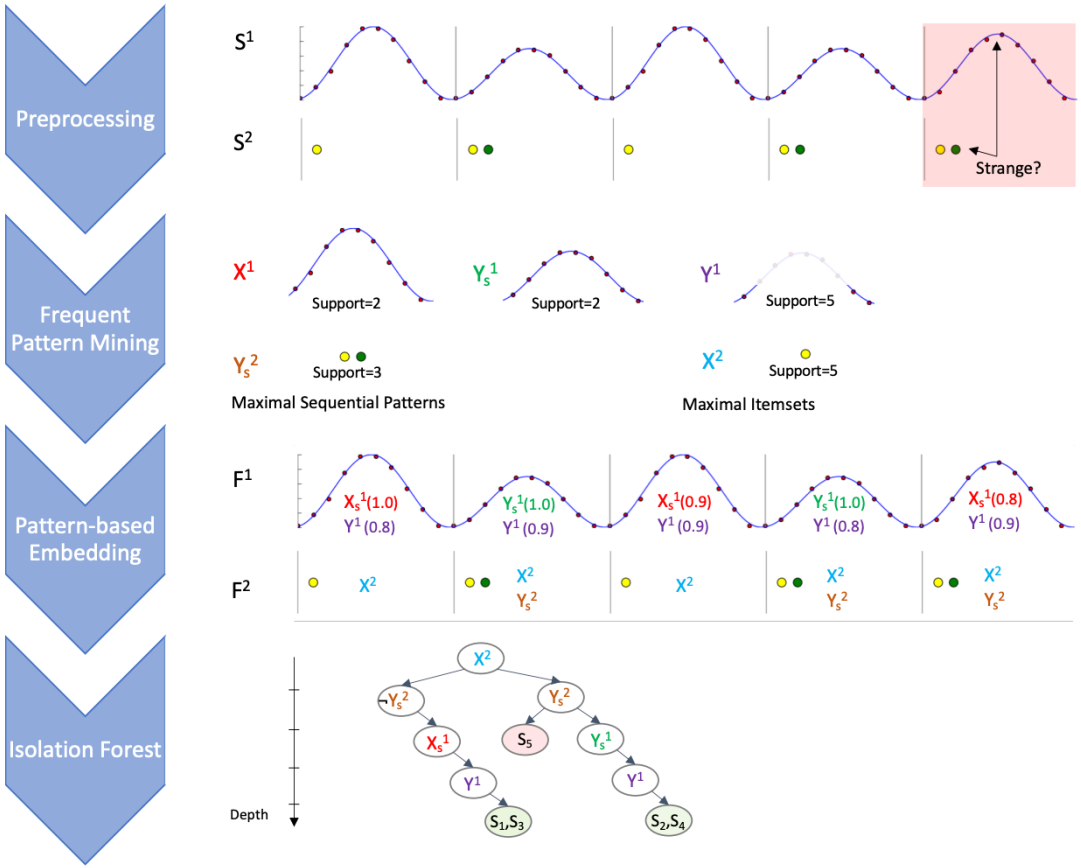




- If at window X_t, **few frequent patterns occur**, there is a high likelihood that there is an **anomaly** at X_t

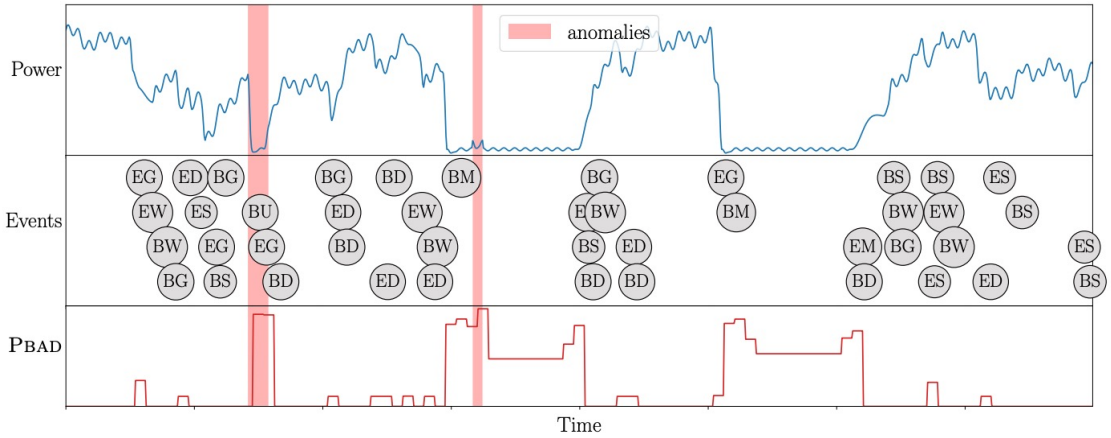


Pattern-based anomaly detection in mixed-type time series



Pattern-based anomaly detection in mixed-type time series

- Unlike compared baselines, PBAD jointly models continuous time series and event data. By combining patterns with **contextual features**, such as events, alarms or multivariate signals



- In our 2019 study, PBAD outperform time series anomaly detection techniques, including MatrixProfile, Pav, Mifpod and FPOF.

TIPM: Pattern mining and anomaly detection in multi-dimensional time series and event logs

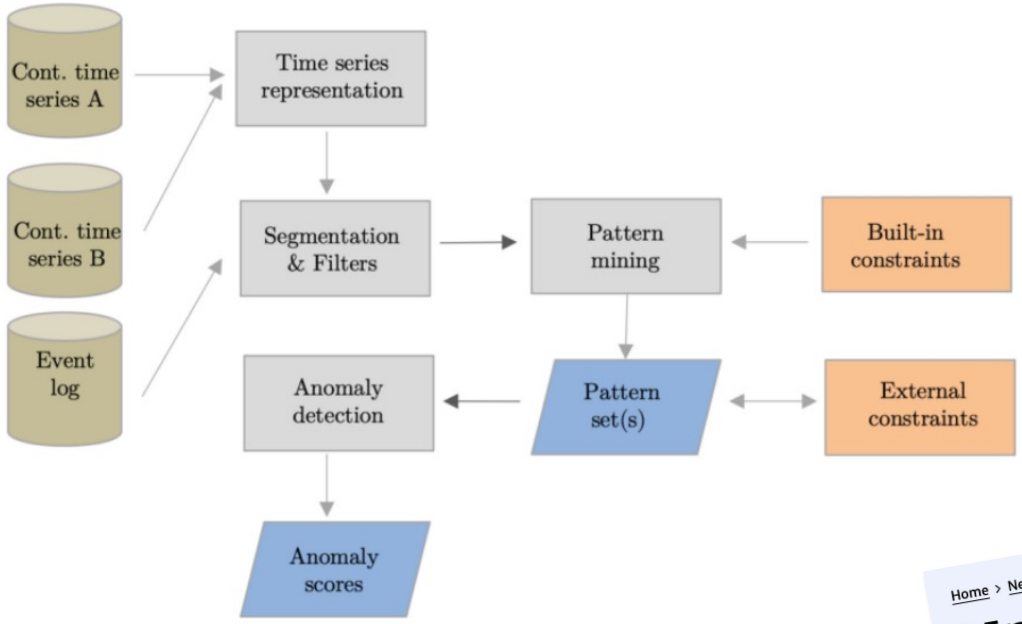
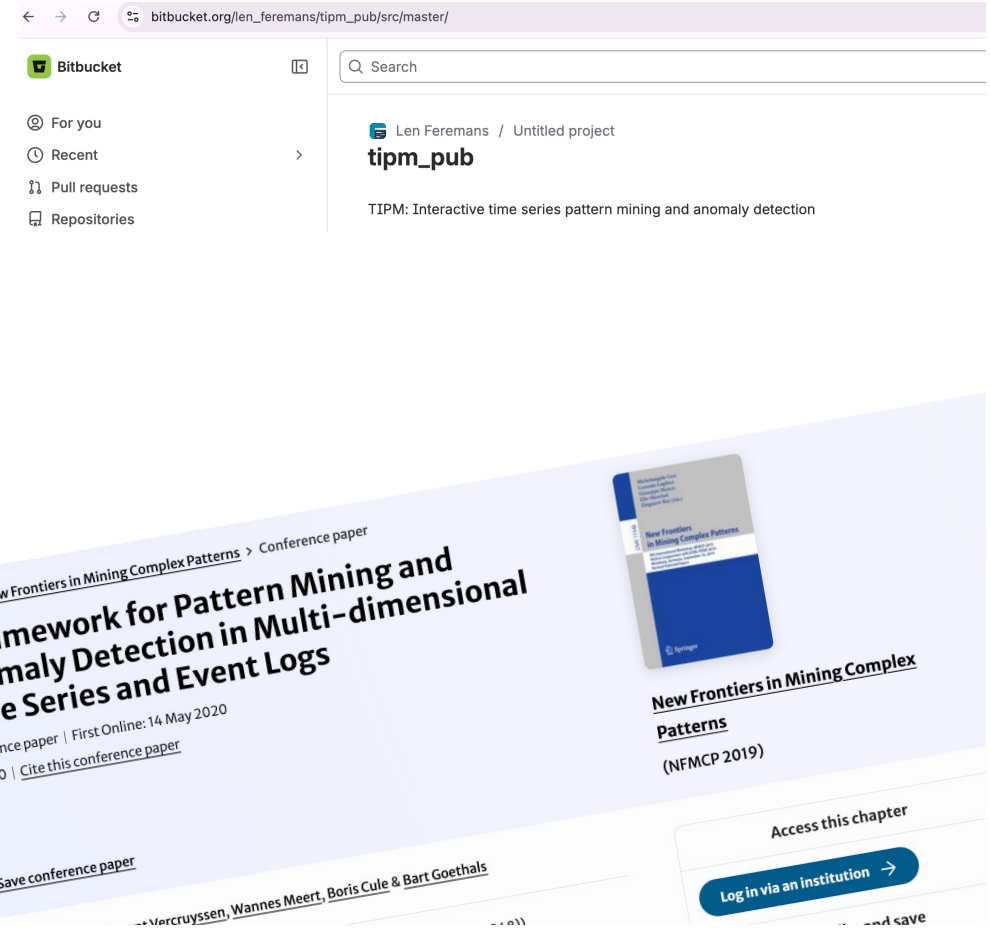
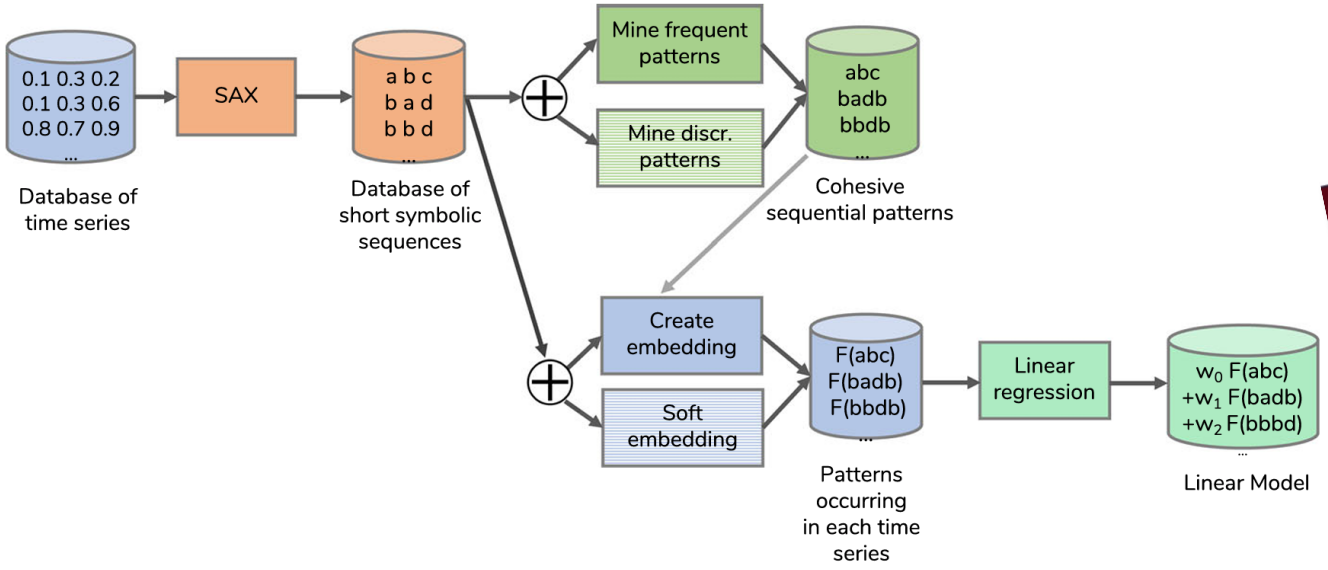


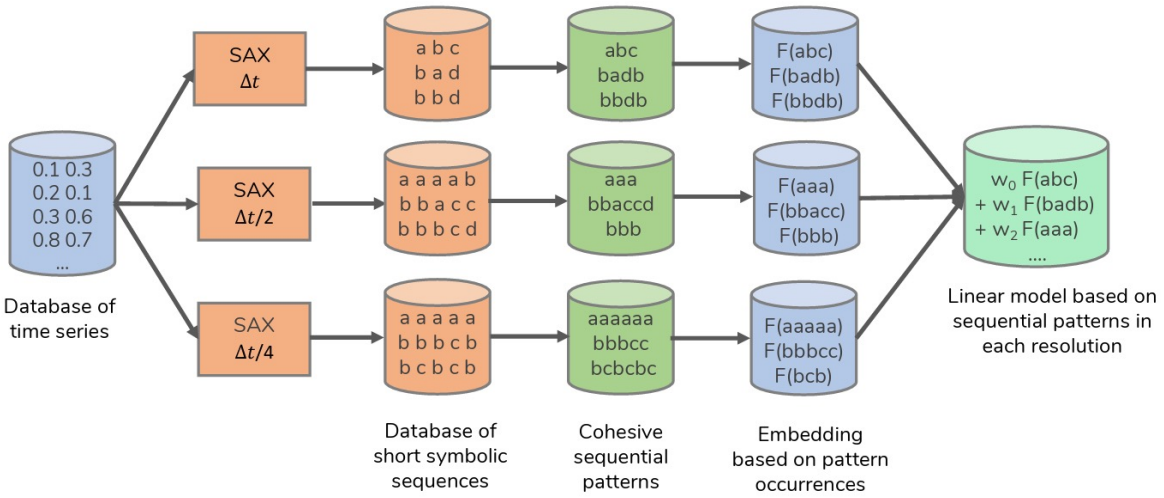
Fig. 2. Workflow of our framework.



PETSC: Pattern-based embedding for time series classification



PETSC: Pattern-based embedding for time series classification

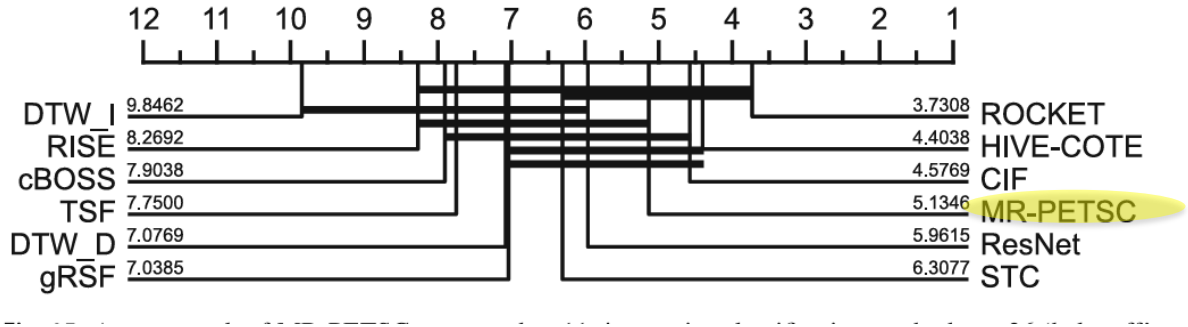


An ensemble combining **different resolutions** works best

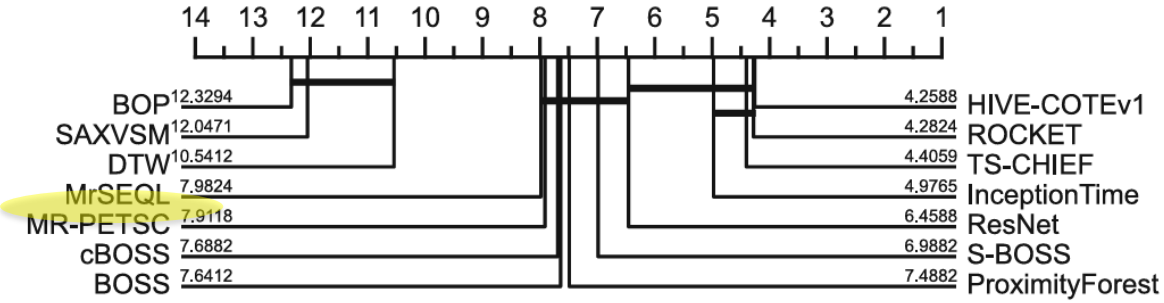
PETSC: Pattern-based embedding for time series classification

- Competitive accuracy compared to deep learning and ensemble learning (2022)

26 multivariate datasets



85 univariate datasets



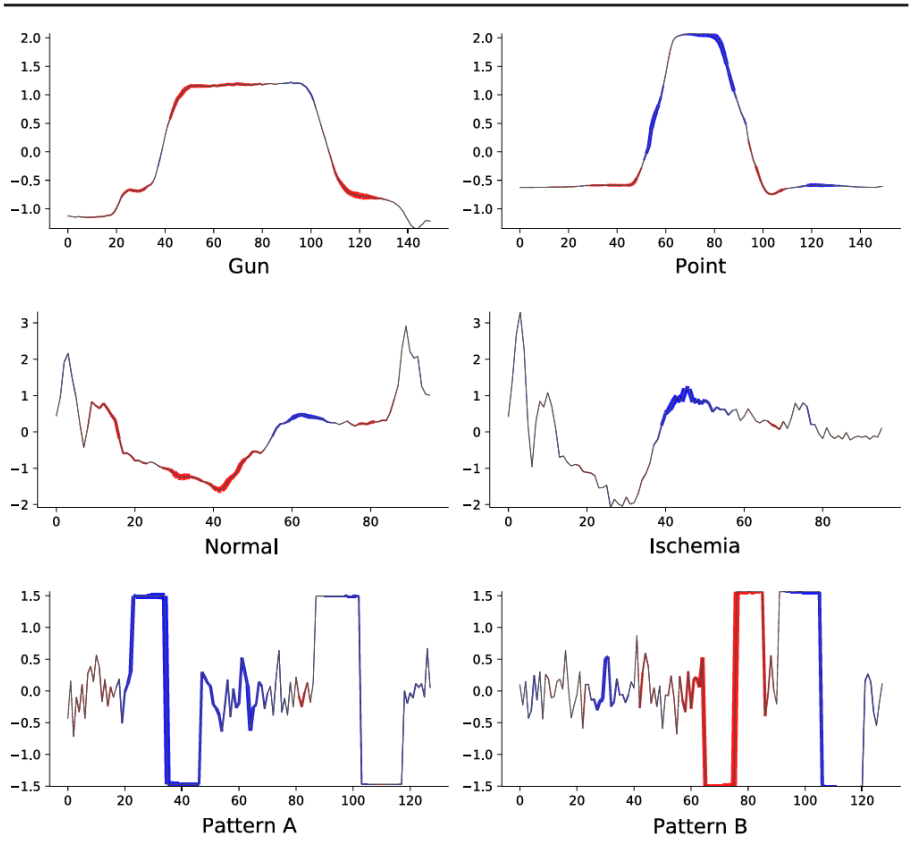
PETSC: Pattern-based embedding for time series classification

- Order-of-magnitude faster compared to most deep learning and ensemble methods

	ROCKET	MR- PETSC	CBOSS	MISTICL	MR- SEQL	INCEPTION	TIME PROXIMITY	FOREST	TS- CHIEF
Total time	**1.4 h**	2.7 h	19.5 h	20.2 h	23.7 h	6 d	11 d		11 d

Total time for **training and predictions on 85** univariate datasets

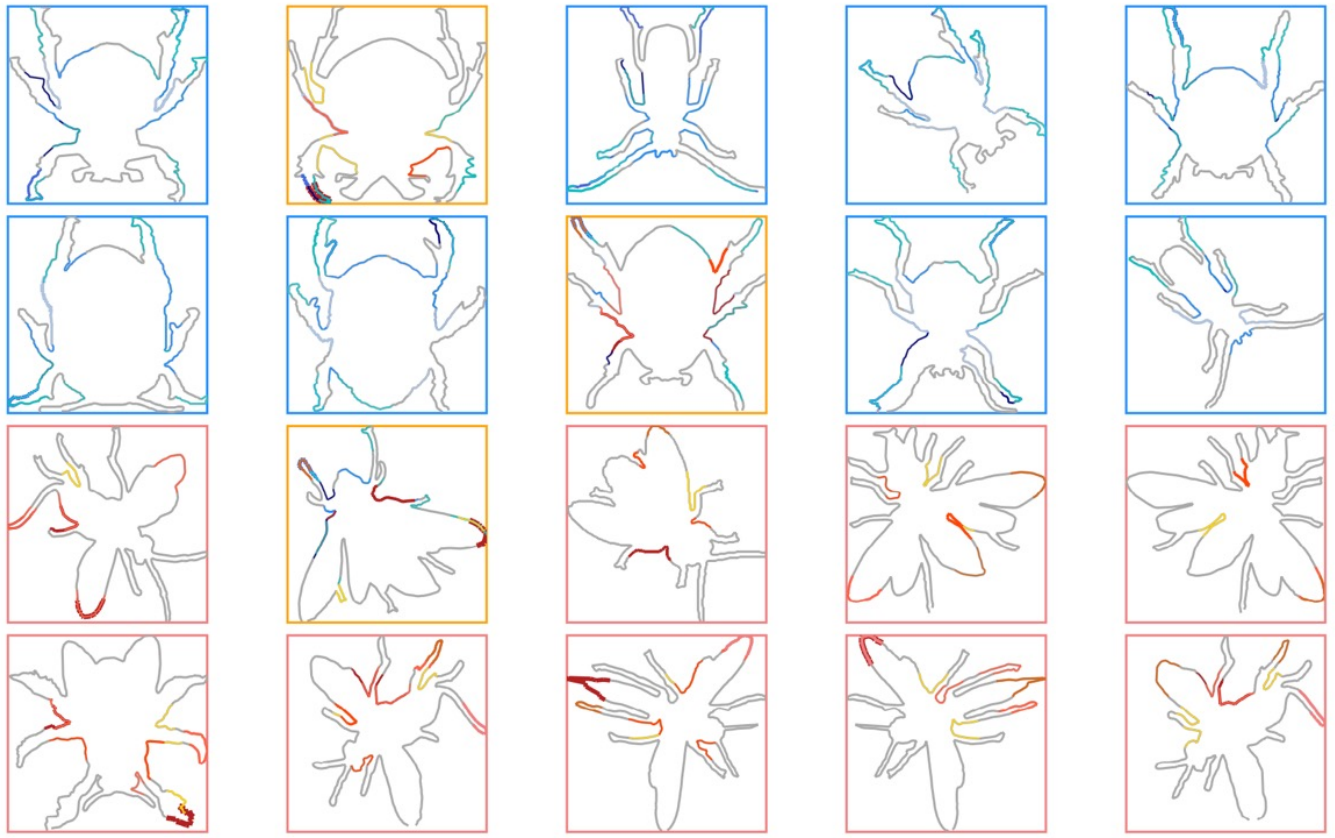
PETSC: Pattern-based embedding for time series classification



Interpretable decisions case-by-case!

Highlight part of time series typical of class A or B

PETSC: Pattern-based embedding for time series classification

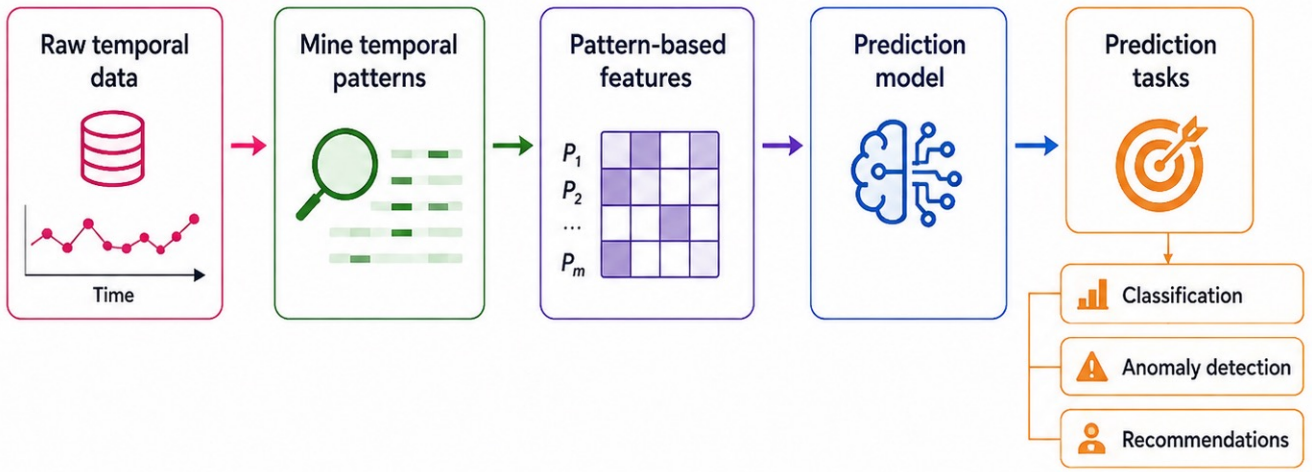


P-0: w=-0.27, sup(Beetle)=10
P-1: w=0.20, sup(Fly)=0
P-2: w=0.23, sup(Fly)=1
P-3: w=-0.61, sup(Beetle)=9
P-4: w=-0.01, sup(Beetle)=9
P-5: w=0.09, sup(Fly)=9
P-6: w=-0.01, sup(Beetle)=10
P-7: w=-0.05, sup(Beetle)=8
P-8: w=-0.60, sup(Beetle)=9
P-9: w=1.00, sup(Fly)=5
P-10: w=0.28, sup(Fly)=7
P-11: w=-0.14, sup(Beetle)=3
P-12: w=-0.11, sup(Beetle)=9
P-13: w=-0.32, sup(Beetle)=10
P-14: w=0.40, sup(Fly)=4
P-15: w=0.39, sup(Fly)=6
P-16: w=0.40, sup(Fly)=6
P-17: w=0.09, sup(Fly)=5
P-18: w=0.22, sup(Fly)=8
P-19: w=0.18, sup(Fly)=2

Interpretable model!

Show patterns typical for class A or B

Temporal patterns for prediction



Competitive accuracy

Efficient computation

Explanations linked to observed temporal patterns

Use temporal patterns as features for downstream prediction tasks.

Finding motifs in time series

- Find the **best** motifs approximate and exact

- Find the **best collection** of motifs using MotiPlus and MotiSet

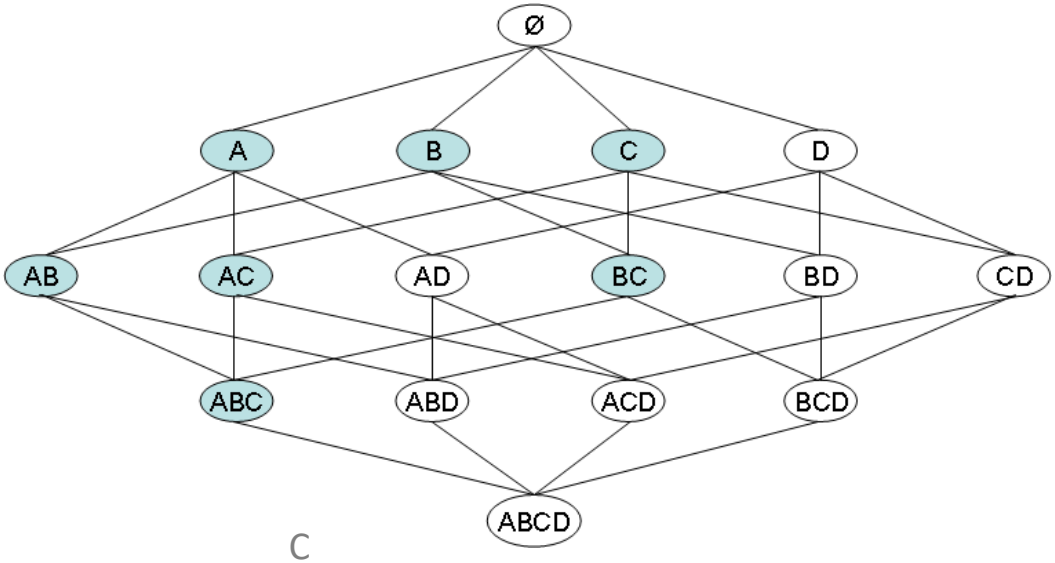
Motivation

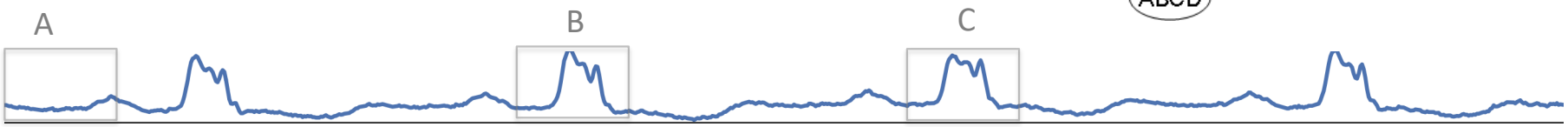
- **Existing motif discovery methods**

✗ Use **heuristics**
✗ Lower **accuracy** on top-k discovery
✗ **Sensitive parameters**
✓ Quadratic **runtime** performance.

Motivation

- Motif mining
  - Variant of Apriori
- Motif set mining
  - Variant of Clustering





Motivation

- We want to **discover** the **top-k** most interesting motifs
  - Example: Lyrics and instrumental sections in music

MotiPlus and MotiSet: Discovering the Best Set of Motiflets in Time Series

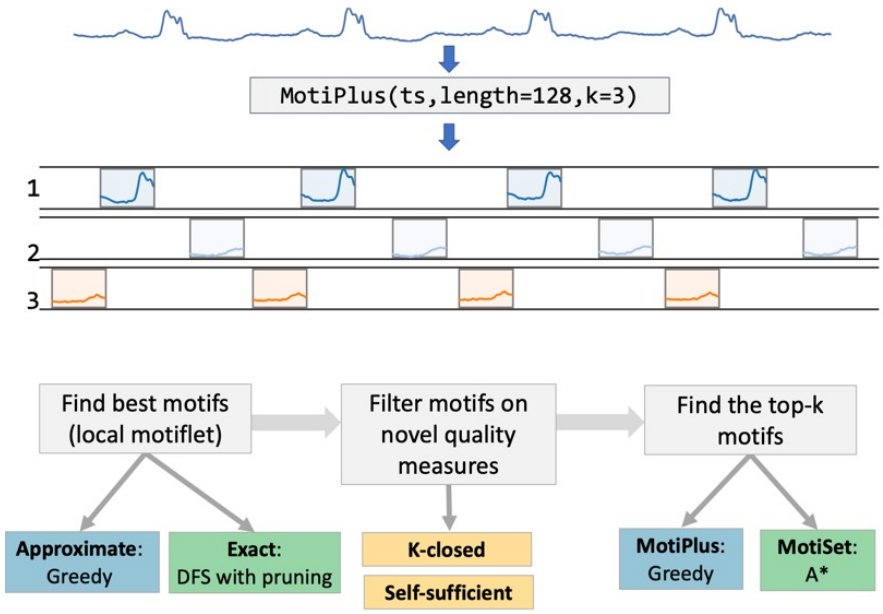
Len Feremans, Adrem Data Lab, University of Antwerp, Belgium
Patrick Schäfer, Humboldt-Universität zu Berlin, Germany
Wannes Meert, DTAI Research Group, KU Leuven, Belgium

bitbucket.org/len_feremans/kmotiflets

HUMBOLDT-UNIVERSITÄT ZU BERLIN

KU LEUVEN

Motif discovery: Find repeating subsequences with high similarity



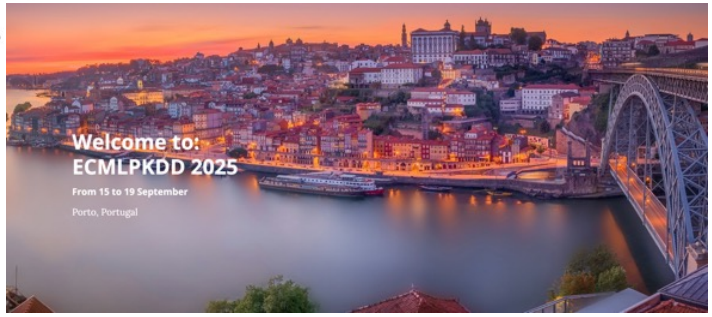
Existing approaches **have limitations**

1. **Existing motif discovery methods** such as GrammarViz, MMotif, Motiflets and LoCoMotif:

✗ Have lower **precision** and **recall** on the **TSMD-benchmark.**
✗ Performance varies widely with **sensitive parameters** such as the distance threshold.
✗ Uses **heuristics** for discovering motifs and collections of motifs.
✓ Have excellent **runtime** performance.

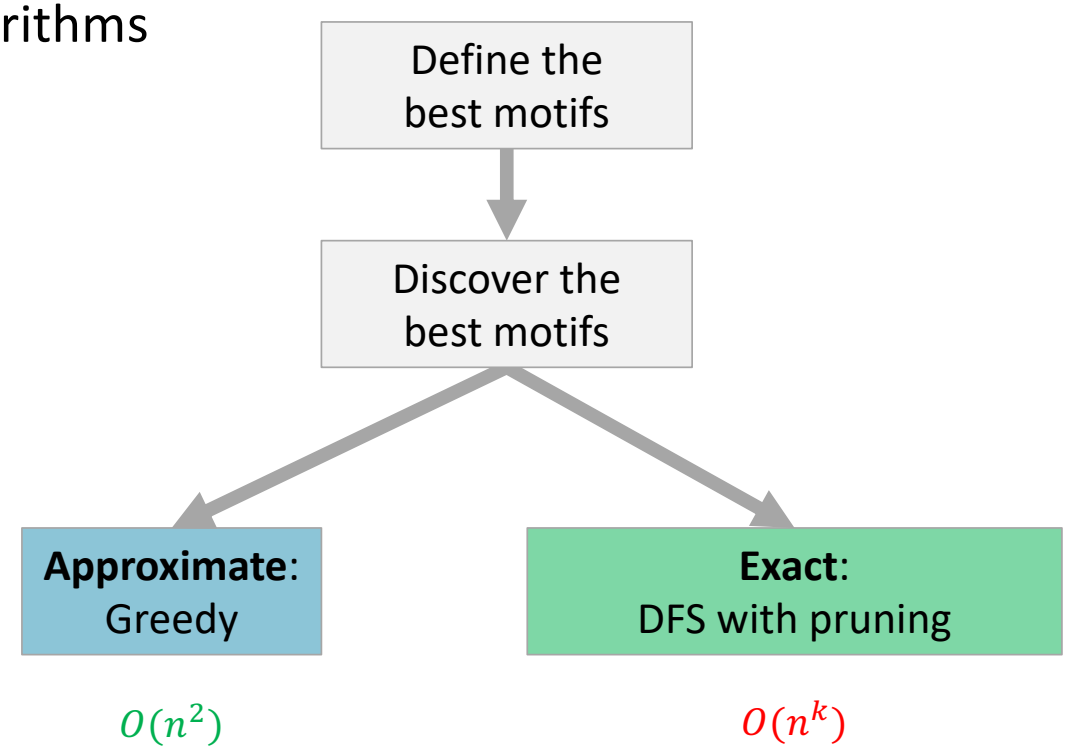
2. **MotiPlus** and **MotiSet** are novel algorithms that discover a set of motifs

✓ Have increased **precision** (+27%) and **recall** (+10%) on the **TSMD-benchmark.**
✓ Easy-to-tune **stable parameters**, i.e. maximal size of motif and motif collection.
✓ **A* search** and **exact motif search** guarantee **optimal** motifs and sets thereof.
✓ Greedy and anytime A* algorithms have excellent **runtime** performance.

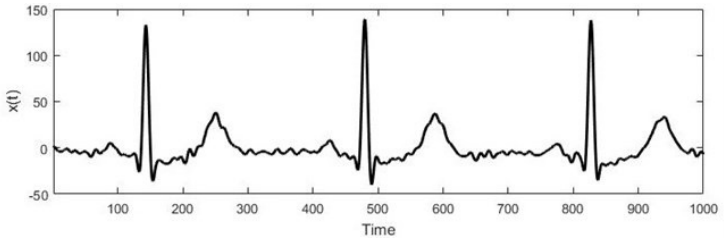


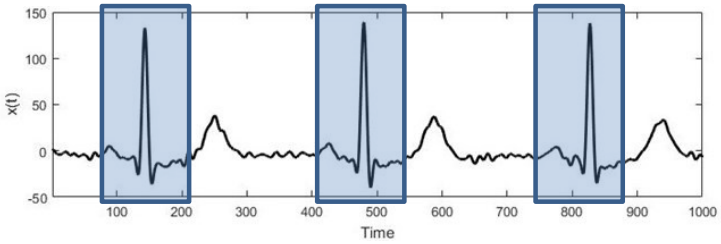
First contribution

- Novel definition of **the best motif** and **quality constraints**, i.e. **local motiflets**
- **Discovery** algorithms



Find k-**motiflets** [Schäfer and Leser, 2022]

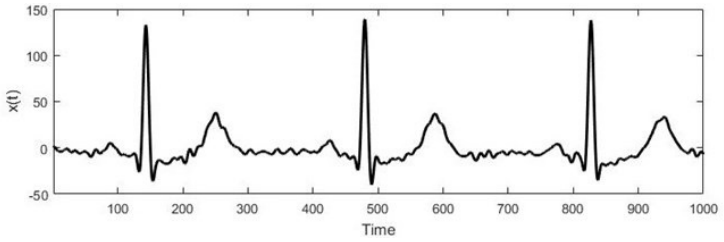


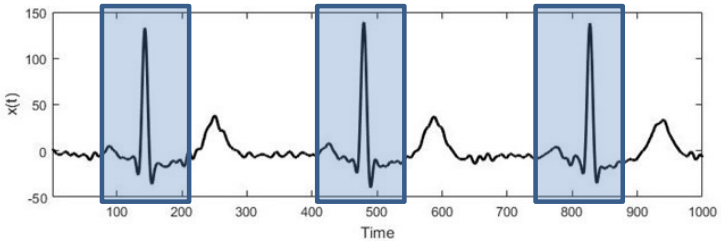


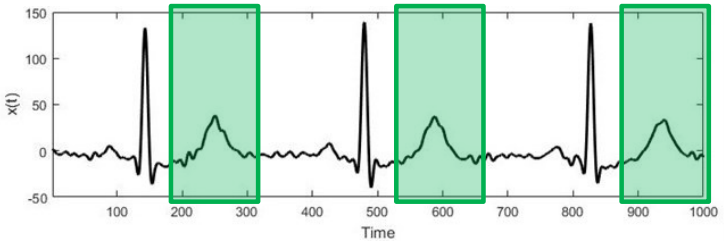
k-motiflet: Candidate motif with k non-overlapping subsequences with highest similarity*

*measured using maximal pairwise z-normalized euclidean distance

Find **local motiflets**







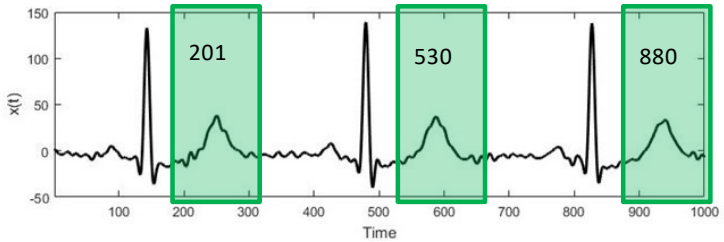
Local k-motiflet: **Rank on similarity** and not **redundant** to other local k-motiflets

Filter motifs on novel quality measures

K-closed

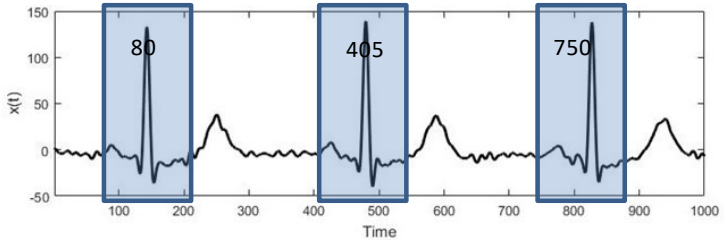
Self-sufficient [Webb, 2010]

Search for motiflets



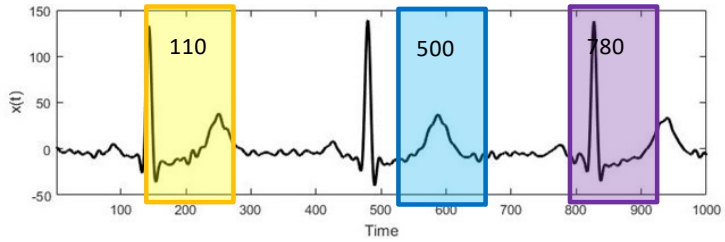
Distance: 0.5

Approximate search



Distance: 0.3

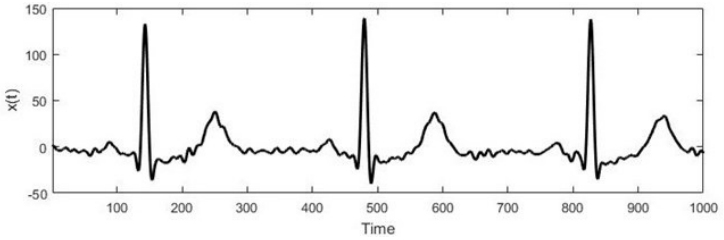
Exact search



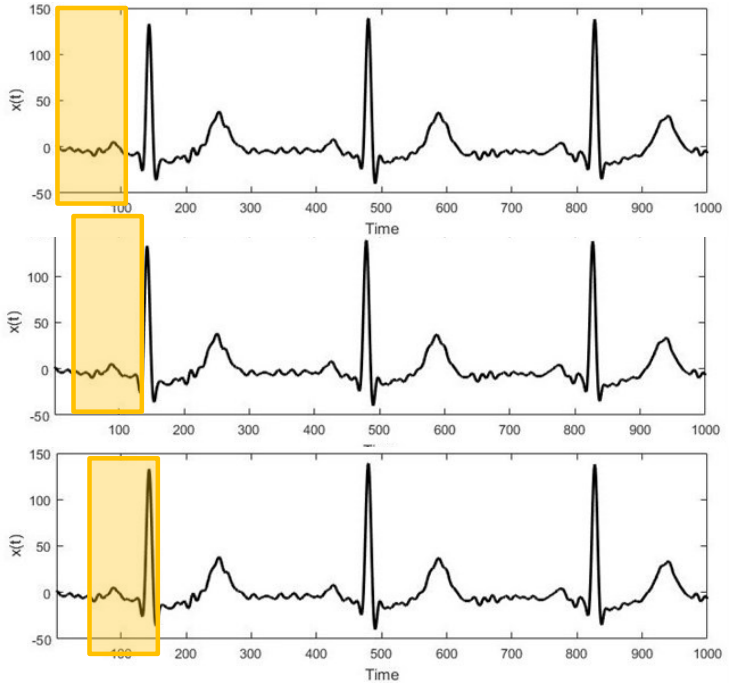
Distance: 5

Monotonicity: If motif {A,B} has a high distance, any superset of {A,B} will also have a high distance

Search for motiflets



Sliding window



Motif discovery is **discrete**, there are N-window_length+1 candidate subsequence

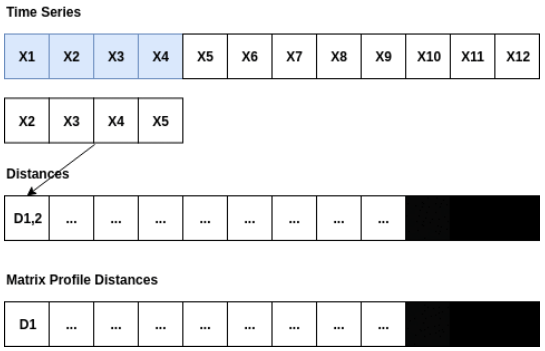
1

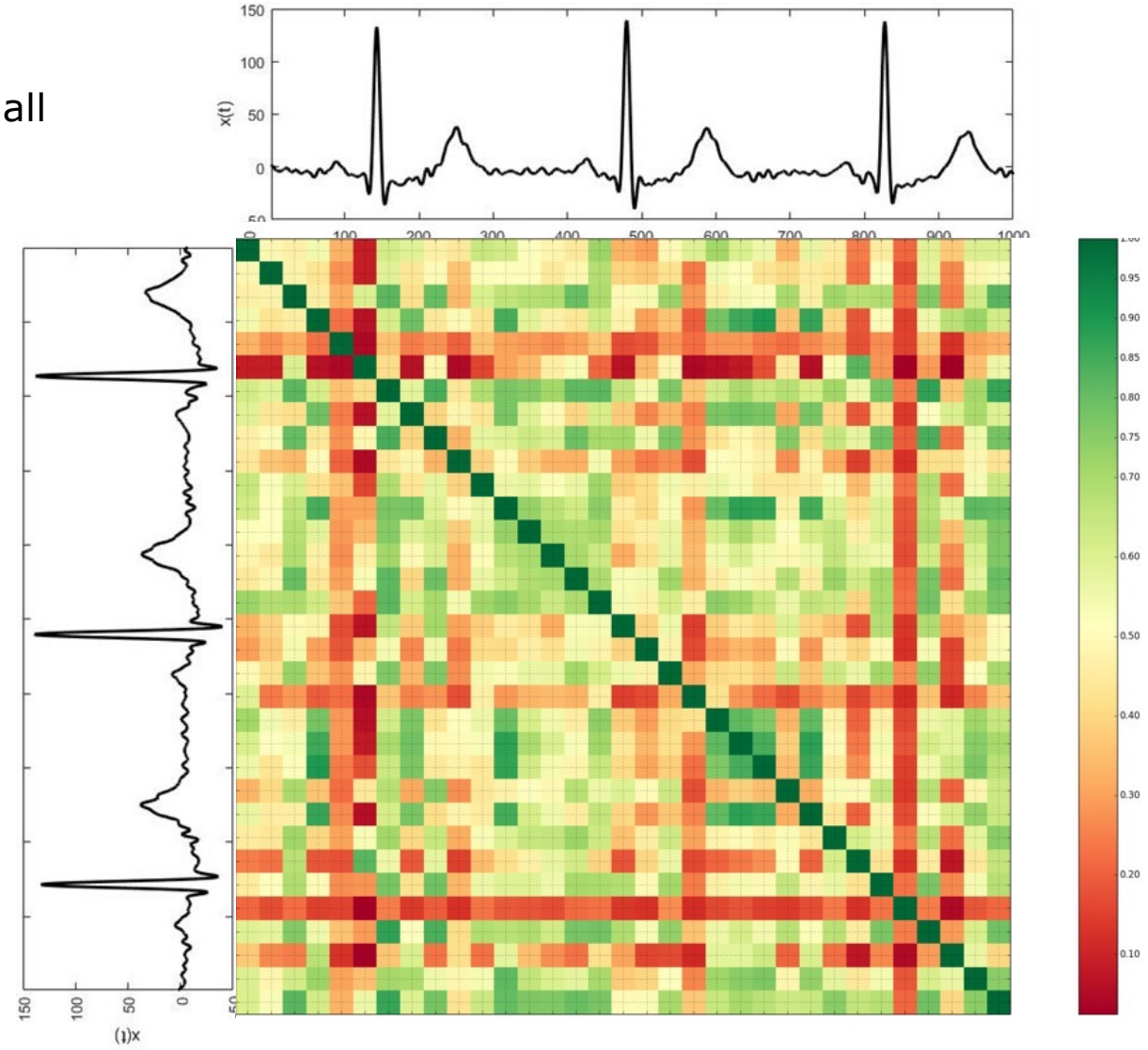
2

3

...

Next, we compute the **similarities** between all subsequence pairs



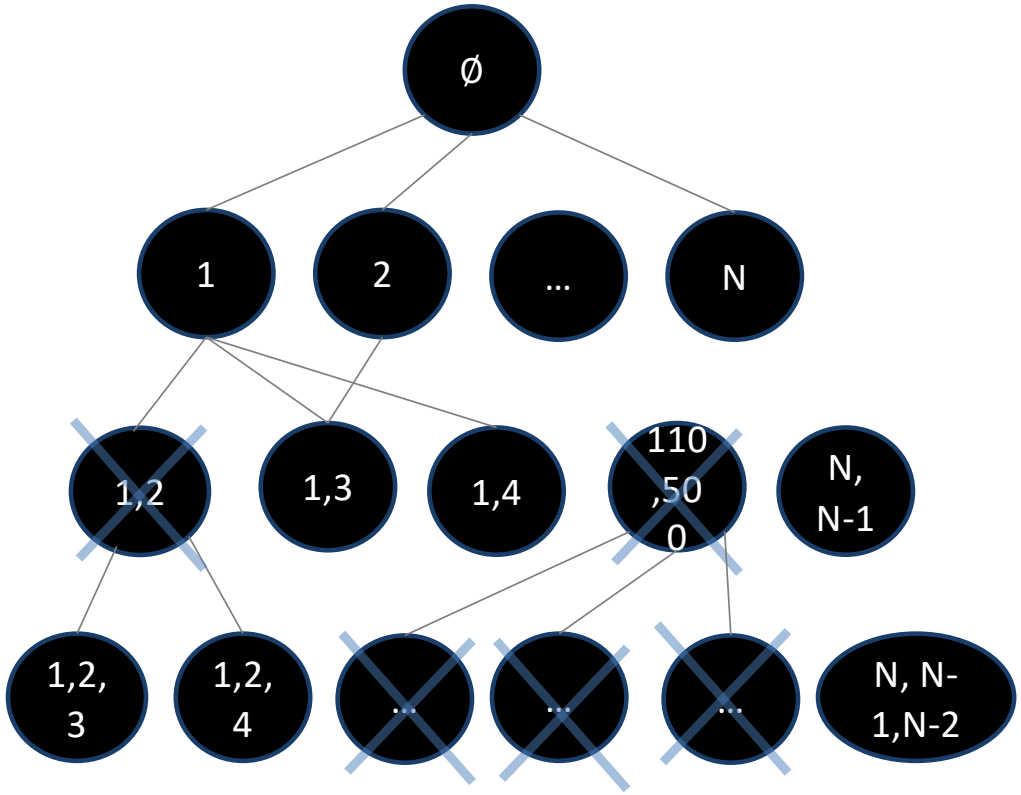


Optionally**, exact search** for motiflets

- **Depth-first-search** to find set of occurrences {**A,B,C**} having the lowest distance

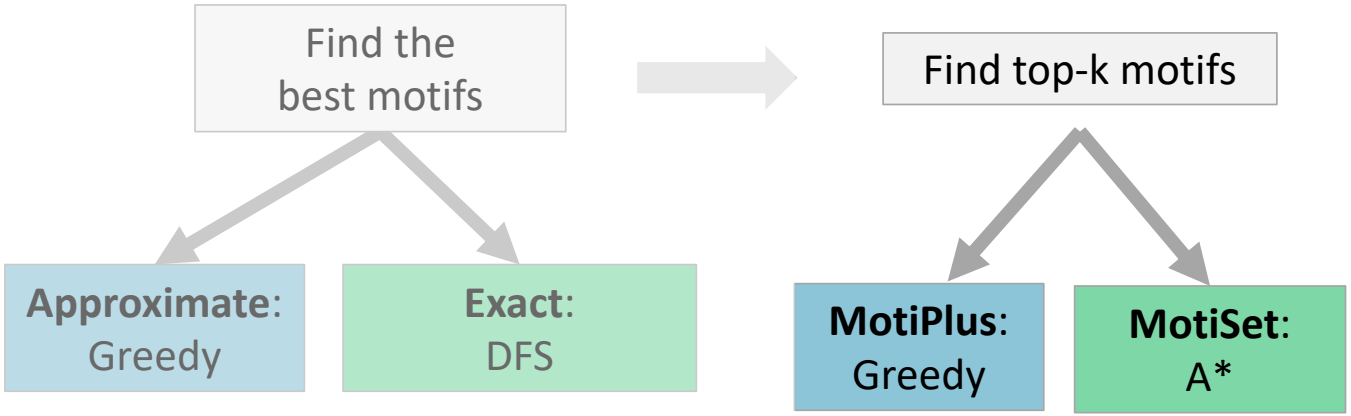
- **Pruning**
  - Initialize **upper bound on distance** using approximate solution
  - **Based** on **monotonicity** of distance
  - **Trivial overlap**

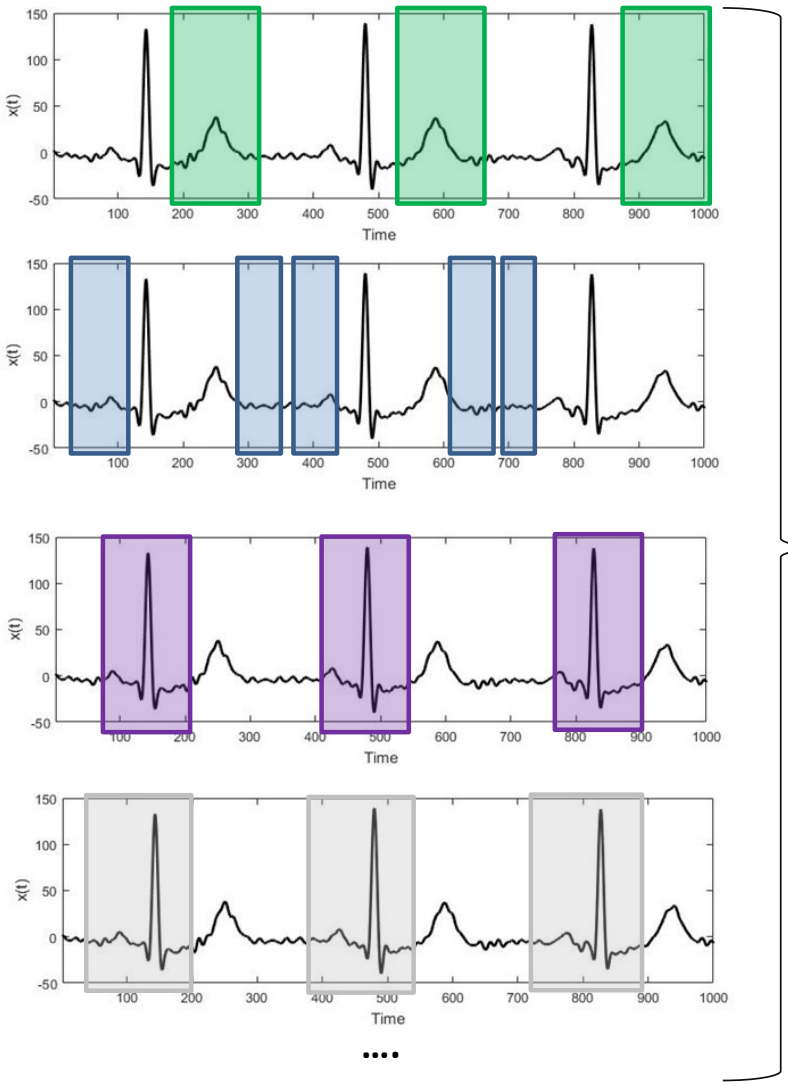
- **Results**
  - **Feasible** up to K=15
  - **Greedy search** is highly accurate



Second contribution

- Novel algorithms for **finding** the **best top-k collection of motifs,** greedy or **anytime**



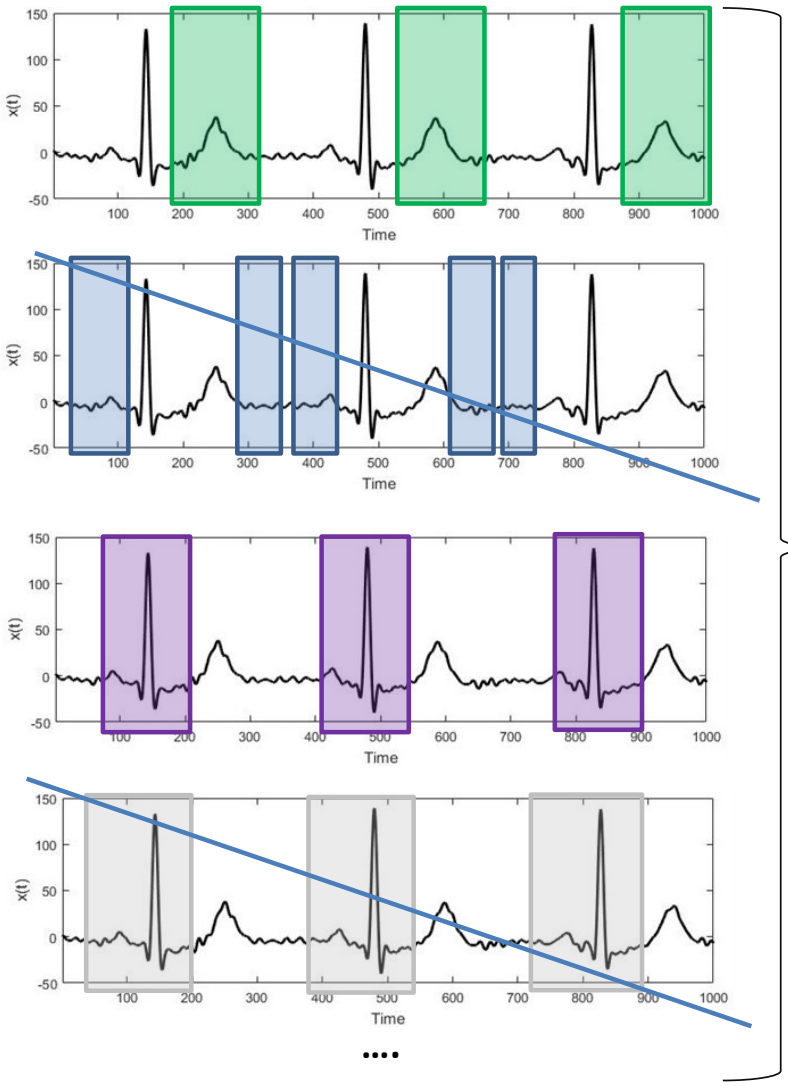


Top-K motifs?

Which 2 do we want to keep?
Many small motifs or few large motifs?
We don't want **redundant** motifs…

Multiple **competing quality objectives**:

- High intra-similarity
- High diversity or low inter-similarity



Multiple **competing quality objectives**:

- High intra-similarity
- High diversity or low inter-similarity

1. **MotiPlus**: Greedy search
   - Time complexity: **$O(n^2)$**
2. **MotiSet**: **A* search**
   - Time complexity: **$O(n^m)$**
     where m is the total number of motifs

MotiPlus (Greedy Algorithm)

- Find best motif add it, then add next best motif that is not redundant, etc.
  - Efficient: **m** iterations

1. Compute distance matrix and find local motiflets
2. Ranks **local motiflets**:
   - lowest extent
   - Not overlapping in time domain
3. Apply **filters:**
   - **k-closed**
   - **self-sufficient**

Algorithm 1: MOTIPLUS: Discovering the top-m motifs

Input : A time series T, motif length l, number of motifs m, the maximum motif size k_{max}

Result: Set $\mathbf{S}$ with up to m motifs

```
1  procedure MotiPlus(T, l, m, k_max)
2      D ← calc_distance_matrix(T, l)
3      index ← k_motiflets_heap(T, D, l, k_max)
4      S ← {}
5      for i ← 1 to m do
6          S_i ← BestMotiflet(T, index, S, k_max)
7          if S_i = ∅ then
8              break
9          S ← S ∪ {S_i}
10     return S
11 procedure BestMotiflet(T, index, S, k_max)
12     C ← {}
13     for k ← 2 to k_max do
14         for S_k ∈ index[k]  sorted on extent do
15             if non-redundant(S_k, S) ∧ k-closed(S_k) ∧ self-sufficient(S_k) then
16                 C ← C ∪ {S_k}
17                 break
18             else
19                 index[k] ← index[k]\{S_k}
20     if C = {} then
21         return ∅
22     S ← argmax_{S_k ∈ C} elbow(k)
23     return S
```

MotiSet (A* Algorithm)

- **Find optimal set of motifs with minimal value**

$$\forall\ \mathbf{S}' \subseteq \mathcal{S} : f(\mathbf{S}) \leqslant f(\mathbf{S}')$$

- **Value** of **set of motifs**:

$$f(\mathbf{S}) = norm(smpd(\mathbf{S})) - \gamma \cdot norm(smod(\mathbf{S}))$$

  - High intra-motif similarity (**compactness**)

$$smpd(\mathbf{S}) = \frac{\sum_{k=1}^{m} \sum_{i=1}^{|S_k|} min\_dist\_in(T_{i,l}, S_k)^2}{\sum_{k=1}^{m} |S_k|}$$

  - Low inter-motif similarity (**diversity**)

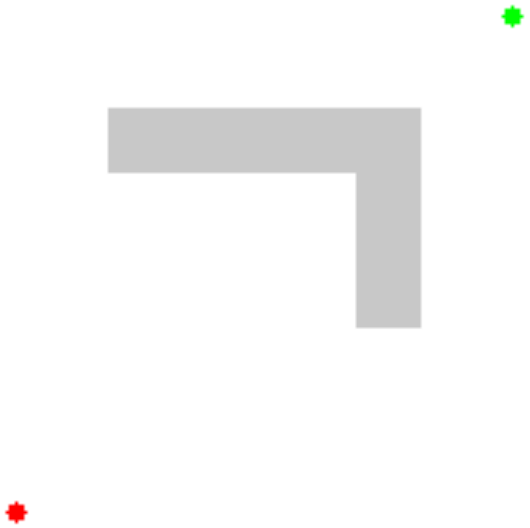
$$smod(\mathbf{S}) = \sum_{k=1}^{m} min\_dist\_out(S_k, T)^2$$

$$min\_dist\_in(T_{i,l}, S_k) = min(\{z\text{-}ED(T_{i,l}, T_{j,l}) \mid T_{j,l} \in S_k \wedge T_{i,l} \neq T_{j,l}\})$$

$$min\_dist\_out(S_k, T) = min(\{z\text{-}ED(T_{i,l}, T_{j,l}) \mid T_{i,l} \in S_k \wedge T_{j,l} \in T\})$$

- In practice we stop the **anytime** search early under a search budget (e.g. 10.000 iterations)

MotiSet (A* Algorithm)

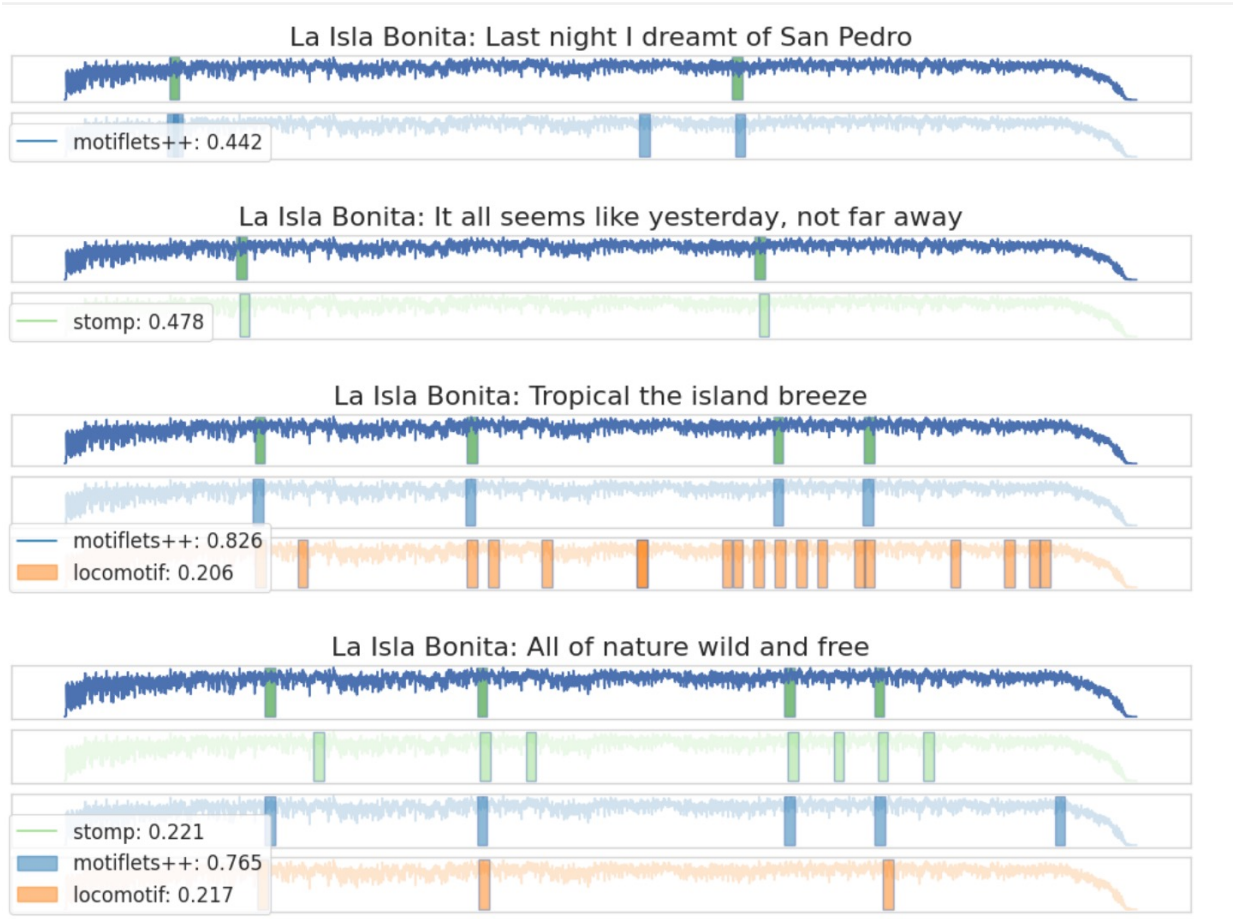


A* search [Hart, 1968]

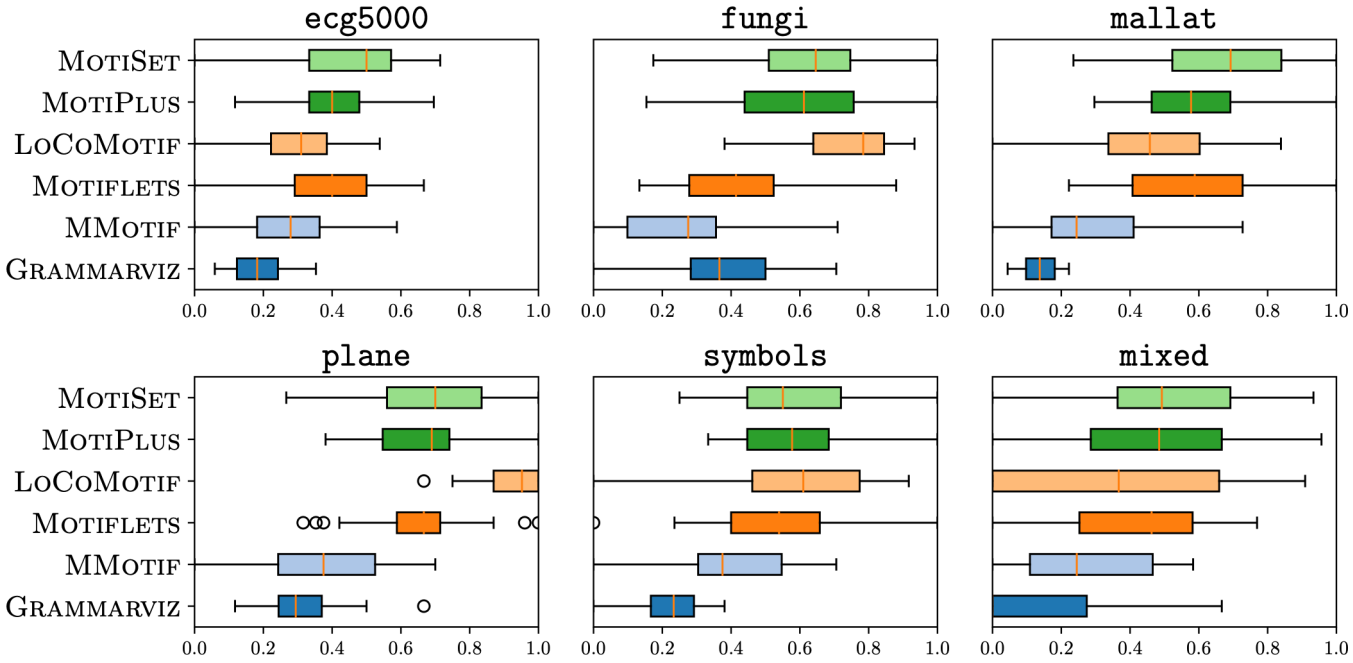
Results: Music Data

- **MotiPlus** uncovers **recurring** lyrics
- Higher **Jaccard similarity** than state-of-the-art methods
- More **similar** and **diverse motifs**, **better coverage**

- **Example:** La Isla Bonita by Madonna



Results: TSMD-benchmark [Van Wesenbeeck, 2026]



Results: TSMD-benchmark

Dataset	F1	Precision	Recall
GRAMMARVIZ	0.234 (+-0.097)	0.264 (+-0.196)	0.321 (+-0.088)
MMOTIF	0.314 (+-0.058)	0.400 (+-0.106)	0.323 (+-0.061)
MOTIFLETS	0.498 (+-0.104)	0.719 (+-0.125)	0.403 (+-0.094)
LOCOMOTIF	0.565 (+-0.232)	0.604 (+-0.268)	0.564 (+-0.195)
MOTIPLUS	0.552 (+-0.094)	0.543 (+-0.125)	**0.621** (+-0.087)
MOTISET	**0.586** (+-0.097)	**0.773** (+-0.068)	0.531 (+-0.108)

- **MotiSet** has highest F1 and precision. **MotiSet** is slower taking 30 min on average.
- **MotiPlus** has highest recall. Runtime **MotiPlus** is under 2 min.
- **LoCoMotif** highest **F1** on some datasets possibly due to its use of **Dynamic Time Warping.** Takes 10 min on average.

FRM Miner

- To discover motifs in a **database of time** series rather than one series
- Find motifs with varying lengths, varying size and levels of support
- Based on **sequential pattern mining** and post-processing to find highly similar motifs

Motif discovery: key take-aways

Benchmark gains on TSMD-benchmark

MotiSet: **+27%** precision

MotiPlus: **+10%** recall

Why this matters

- better motif quality
- more useful top-k results
- clear trade-off between accuracy and efficiency

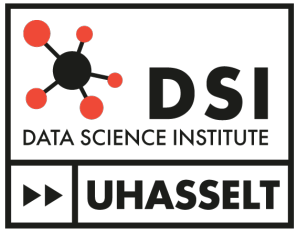
Optimality guarantees

Exact motif search guarantees optimal motifs

A* search guarantees optimal motif sets

Motif discovery benefits from better quality measures, efficient search, and exact optimization when needed.

Questions?



Len.feremans@uhasselt.be

Bitbucket.org/len_feremans

FWO grant 12B0V24N