AUTOMATED FEATURE CONSTRUCTION

Jesse Davis
KU Leuven
jesse.davis@cs.kuleuven.be

My Research Interests

I focus on two big areas:

1. Developing novel ML and DM algorithms
2. Applying them to real-world applications

Physical Activity





Anomaly Detection



Sports



Unifying theme: Analyzing heterogeneous data

Different Problems We Work On

- Detecting anomalies in time series data (resources, DNS traffic, airplanes, etc.)
- Predicting fatigue in runners
- Valuing actions in soccer
- Discovering tactics in sports (soccer, volleyball)
- Identifying whether a physical therapy exercise was done correctly or not
- Exploring the relationship between internal and external load in soccer players

Task changes: Challenges and approach do not

Challenge: Data Is Non-Standard

Soccer Match Event Data

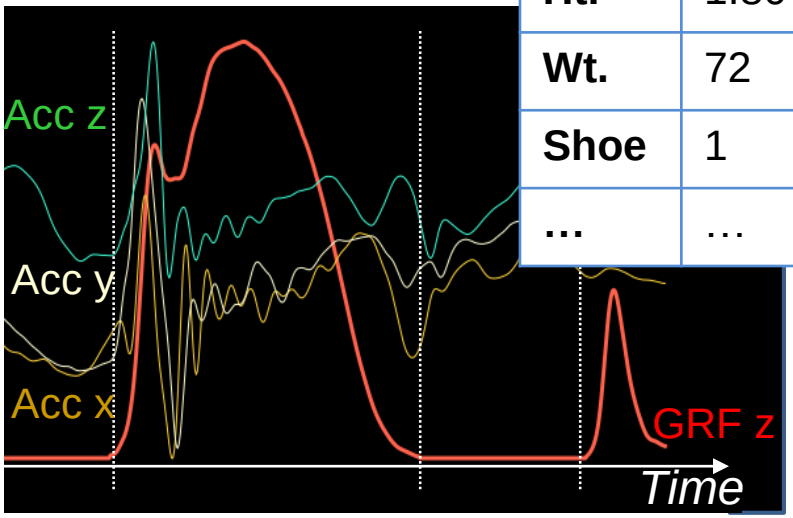
Everton v. Watford, 5/11/2017, Goodison Park, Attend: 39,572

Type	Pass
Start	(60,20)
End	(75,60)
Time	80.1
…	…

Type	Shot
Start	(75,60)
End	(100,60)
Time	80.2
…	…

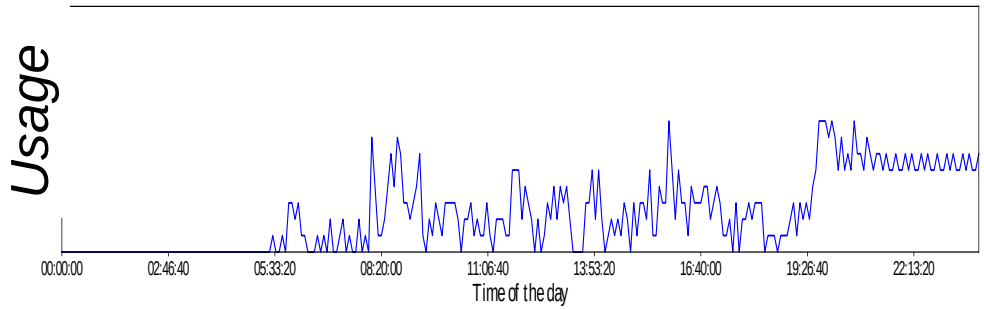
…

Running Data



Ht.	1.80
Wt.	72
Shoe	1
…	…

Resource Usage Data

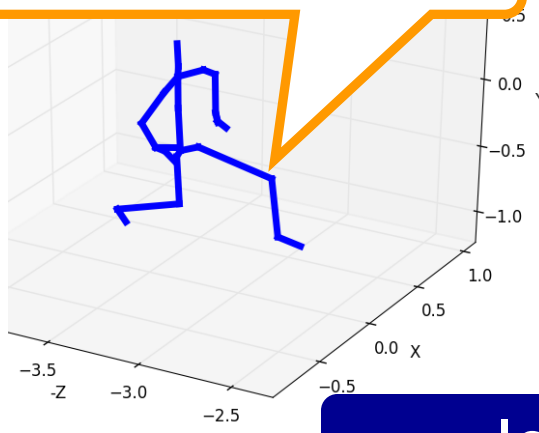


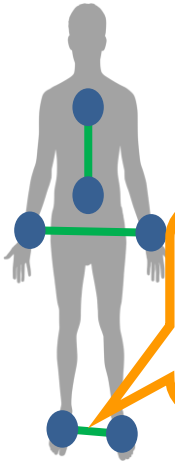
Day	Mon.
Holiday	No
Month	July
…	…

Challenge: Important to Incorporate Domain Knowledge

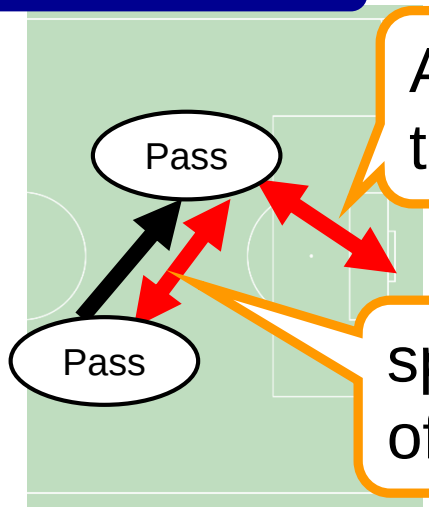
Defining relevant features

Angles are key





Symmetry relevant

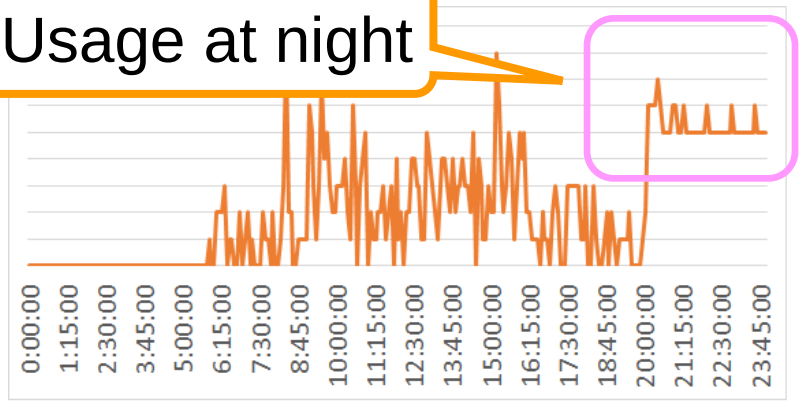


Angle to goal

speed of play

Identifying relevant contexts

Usage at night





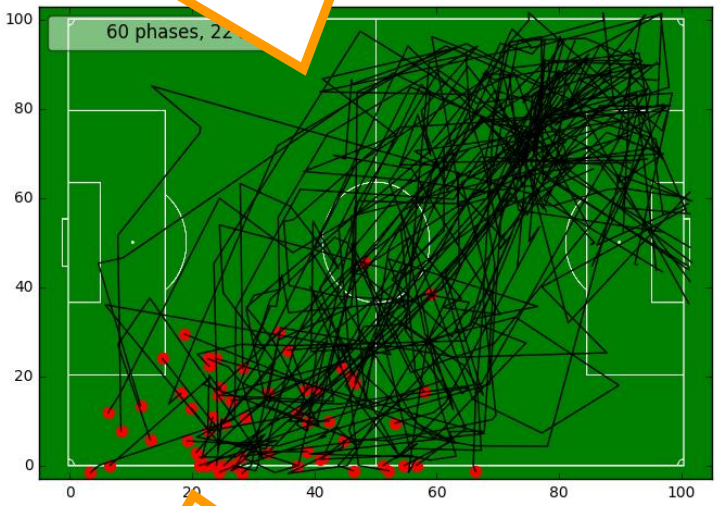
Game situation

Challenge: Relatively Small Amounts of Data with Limited Labels

What tactical advice was given?

How fatigued is he?

Anomaly?







~50 matches / year

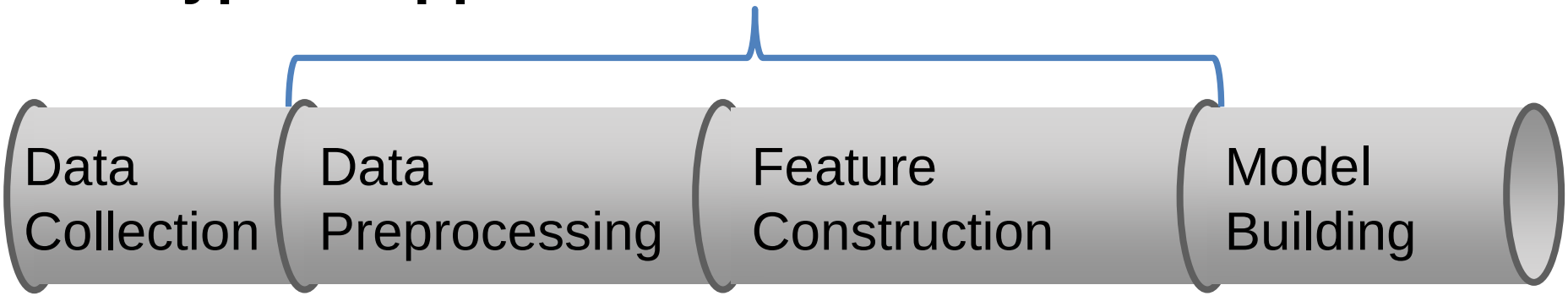
~100 runs / year

Few anomalies

The Standard Approach

Problem: This is very time intensive!

- **Goal:** Convert data to form amenable for analysis
- **Typical approach:** One-off, handcrafted solution



Goals of this talk is to argue that

1. These steps are crucial for performance
2. Cost a huge amount of time
3. Suitable for partial automation

Existing Classes of Techniques for Automated Feature Construction

- Four broad classes of techniques
  - Propositional data (e.g., Rendell'85, Khurana et al.'18, Horn et al.'19)
  - Deep learning
  - Propositionalization (e.g., Kramer et al.'01 Kanter et al.'15)
  - Time series techniques (e.g., Fulcher et al.'14, Christ et al.'18)

- Some drawbacks
  - Focus on one type of data
  - No support for "fancy" preprocessing/normalization
  - Limited support for incorporating knowledge

Outline

- Illustrative example for the need, challenges and concepts for feature construction
- Constructing features from relational data
- Constructing features from time series data
- Reflection and conclusion

Part I: Motivating Examples

Motivating Problem: Running

[Op De Beeck et al., KDD'18]

Context: Running is a popular form of exercise

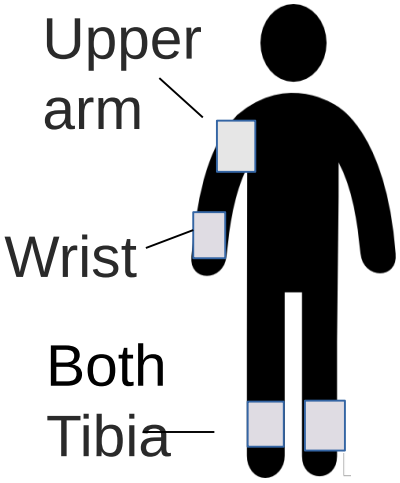


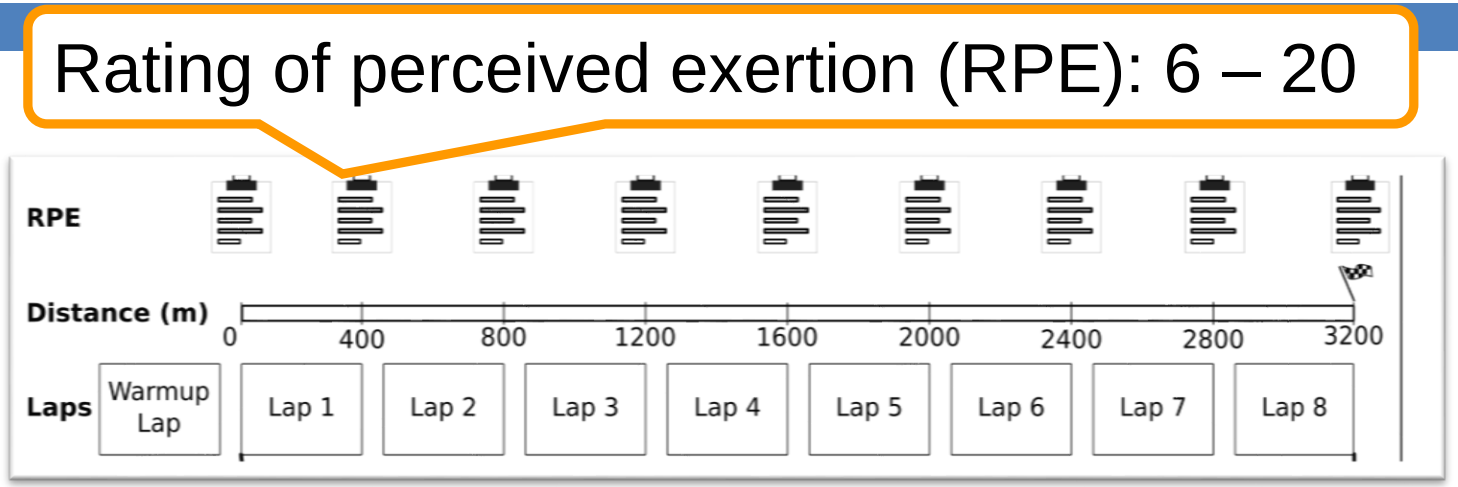


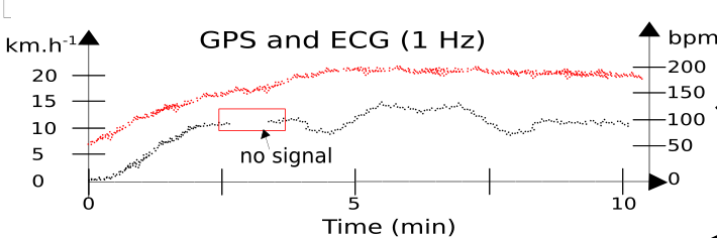
Problem: Up to 79% of runners suffer an injury in a year

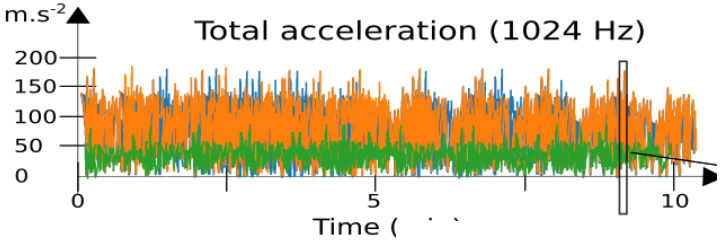
Question: Can applying ML to IMU data detect fatigue?

Fatigue Protocol Data







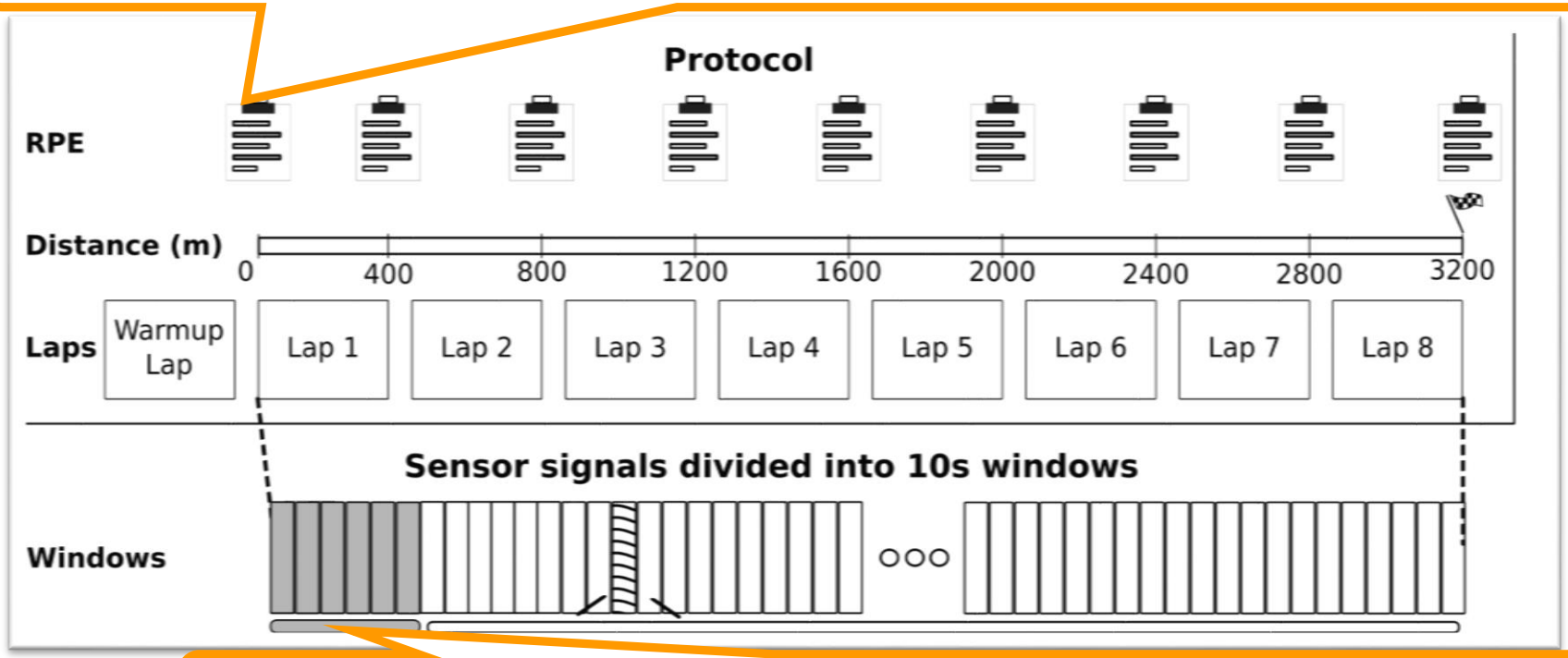


Three big challenges:

1. What preprocessing is needed?
2. What features should we consider?
3. What modeling granularity should consider?

Pre-processing: Normalizations Based on Domain Knowledge

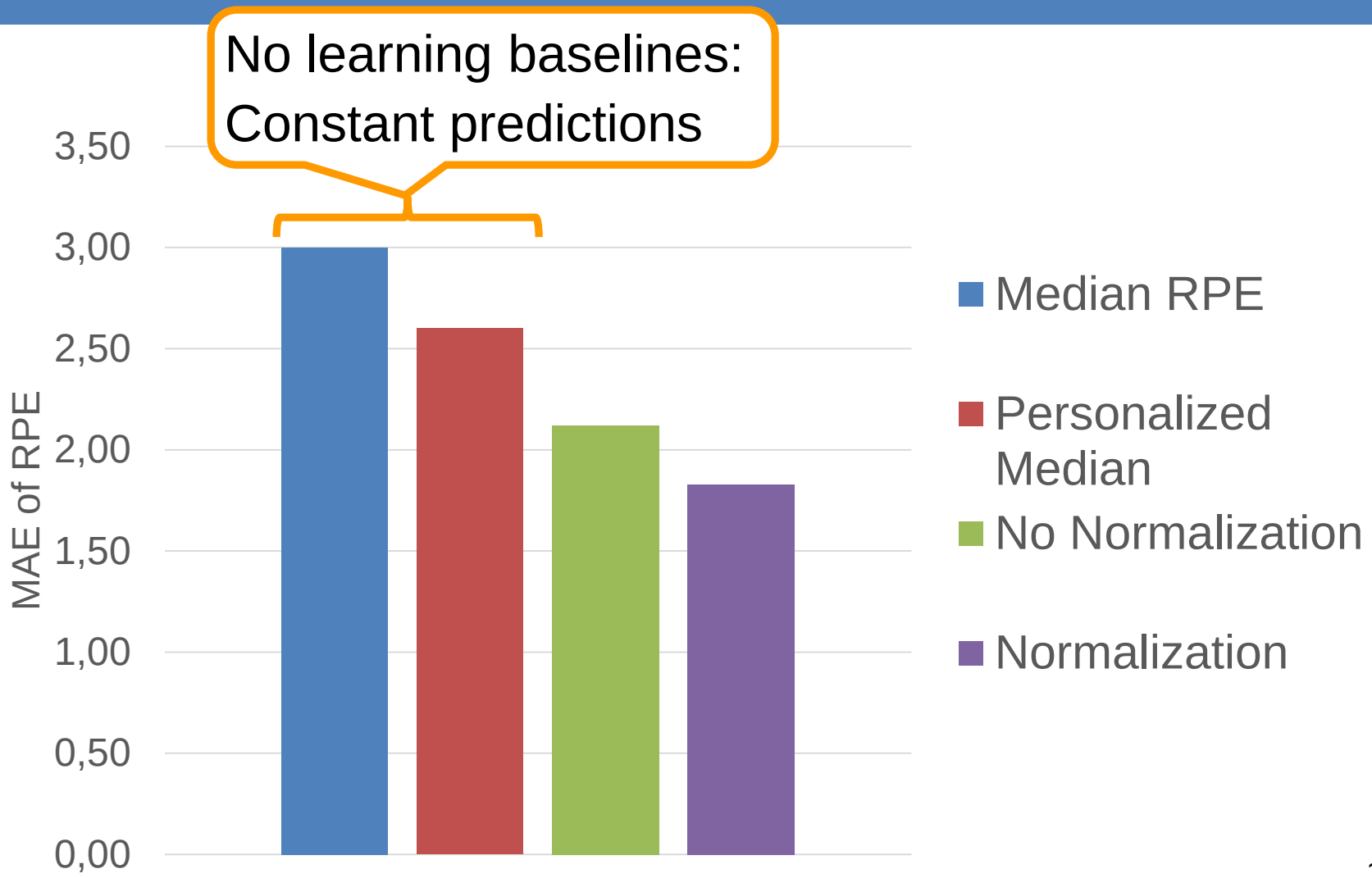
RPE evolution is trial-dependent: Normalize to first value



Normalize based on change from first windows

Domain Insight: Change in feature values over time is key

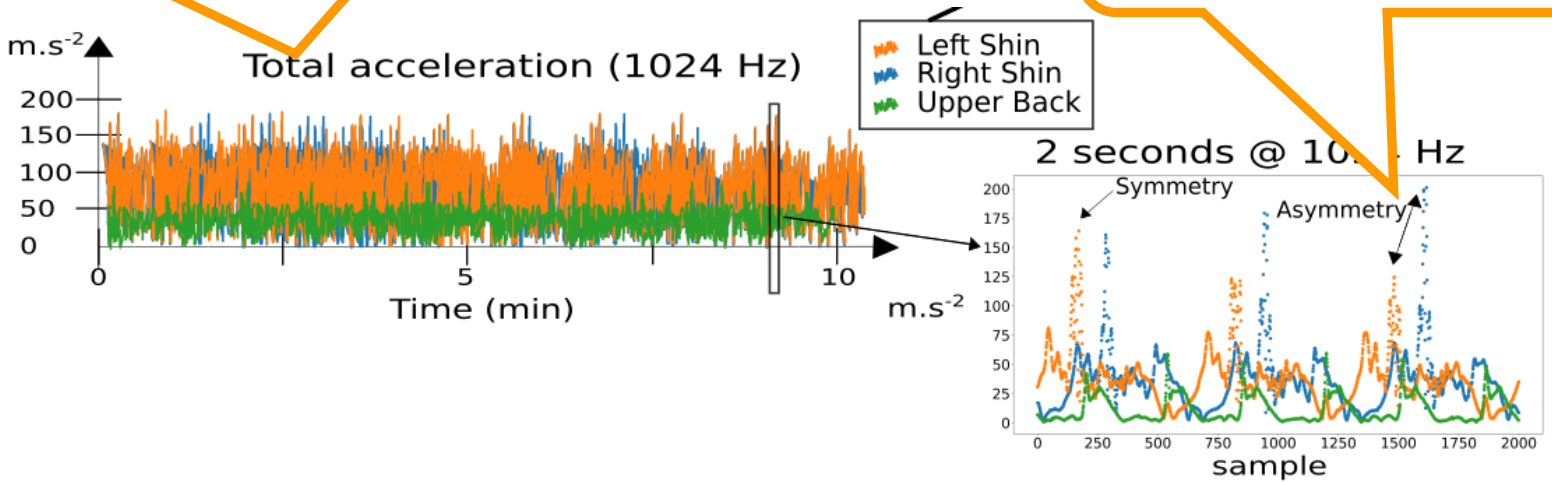
Effects of Feature Normalization for Gradient Boosted Trees



Feature Construction

Derived signal:
Combine X, Y and Z acceleration

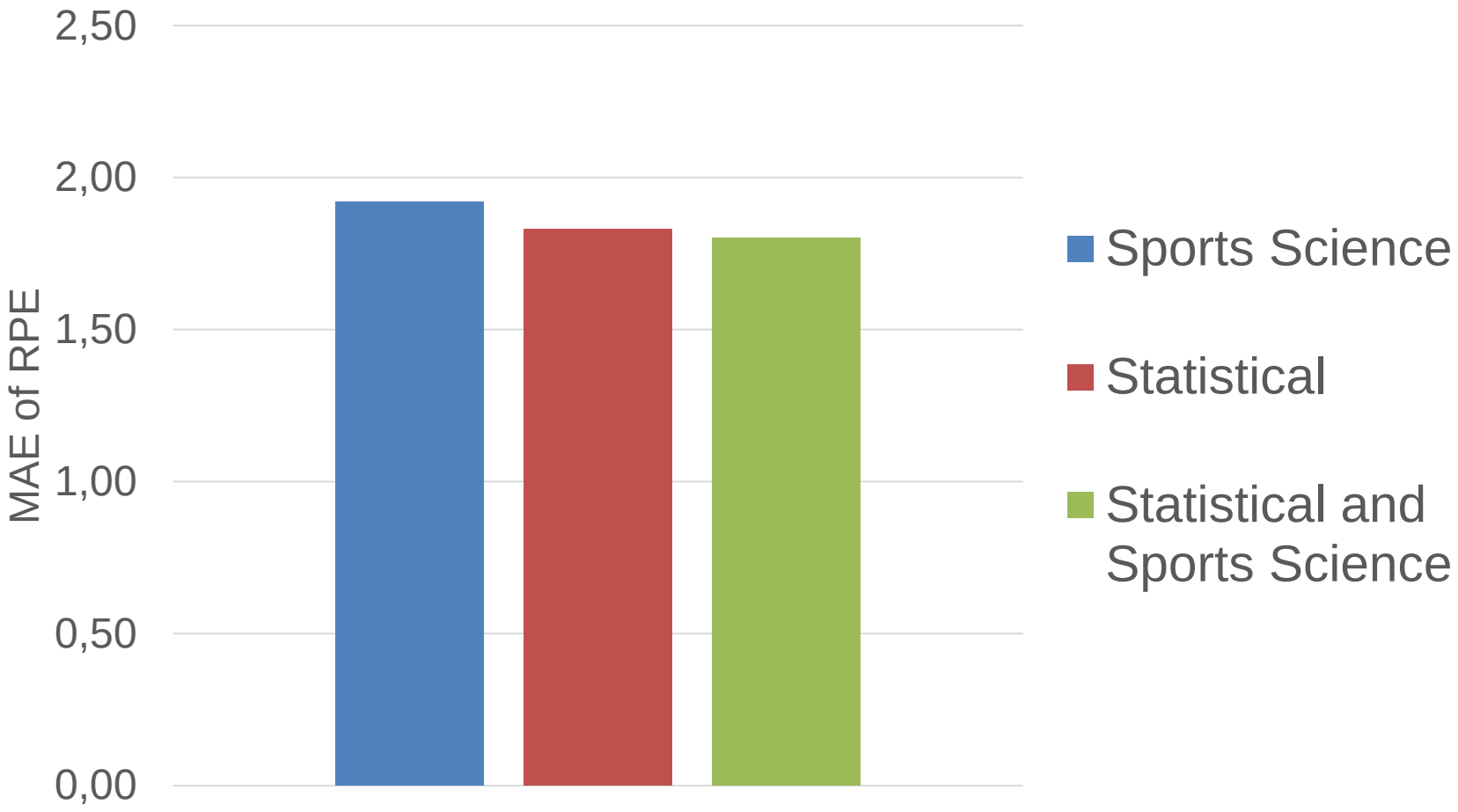
Combine two features from two sensors



Sport science features used in running and gait:

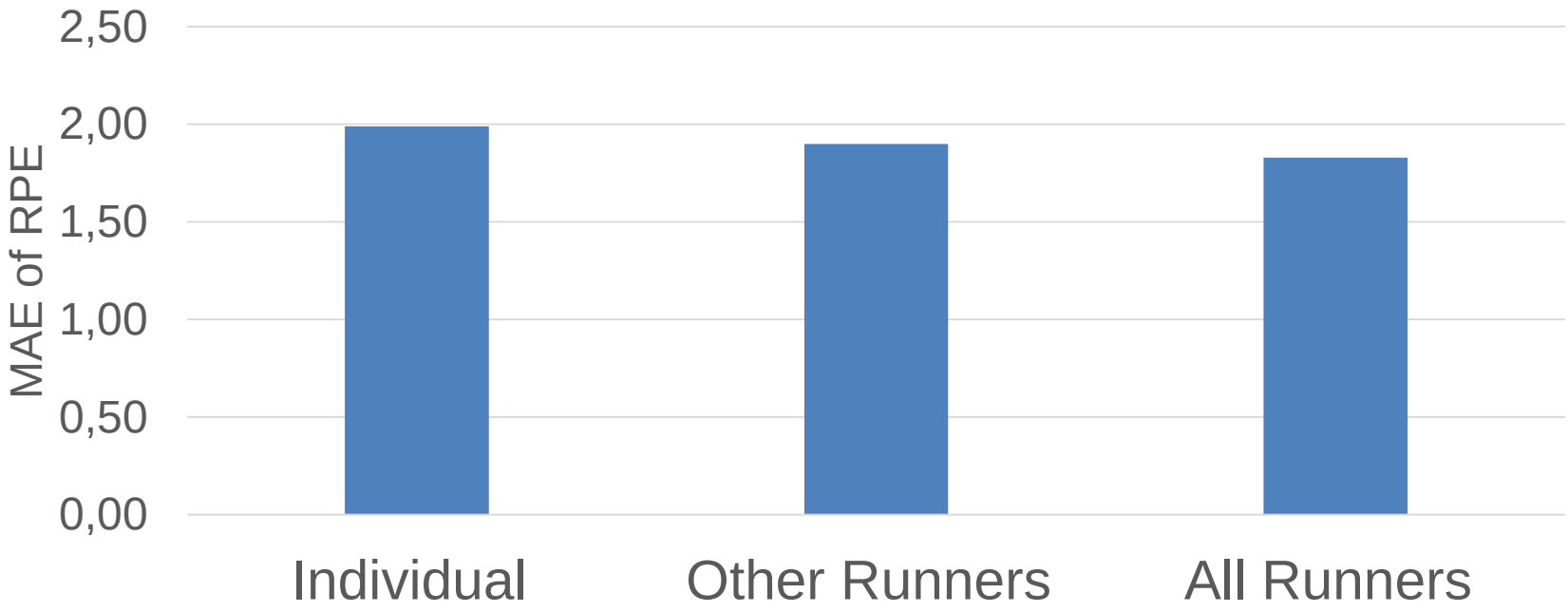
1. Sample entropy
2. Stride regularity

Effect of Feature Types for Gradient Boosted Trees

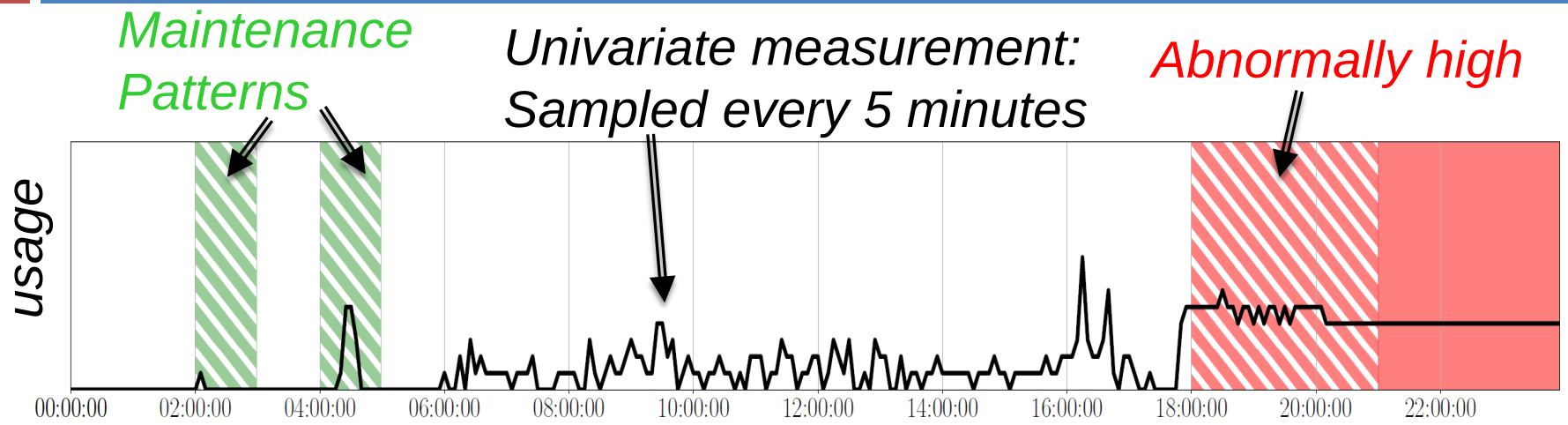


Modeling Granularity: Tension between Personalization and Data Availability

- **Individual model:** Requires labels for every runner
- **Other runners:** Trained without your data
- **All runners:** Use all data



Motivating Example: Resource Monitoring



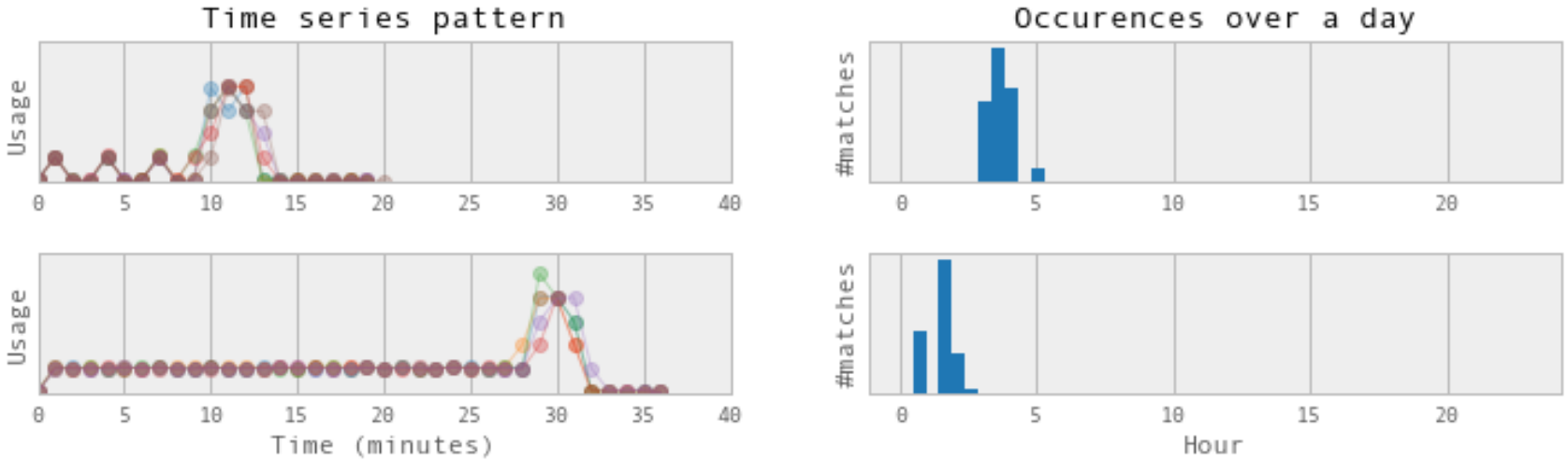
Context:

- Detect high periods of usage at retail stores
- Anomalous usage has real costs

Question: How can we automatically detect these types of anomalies in the data?

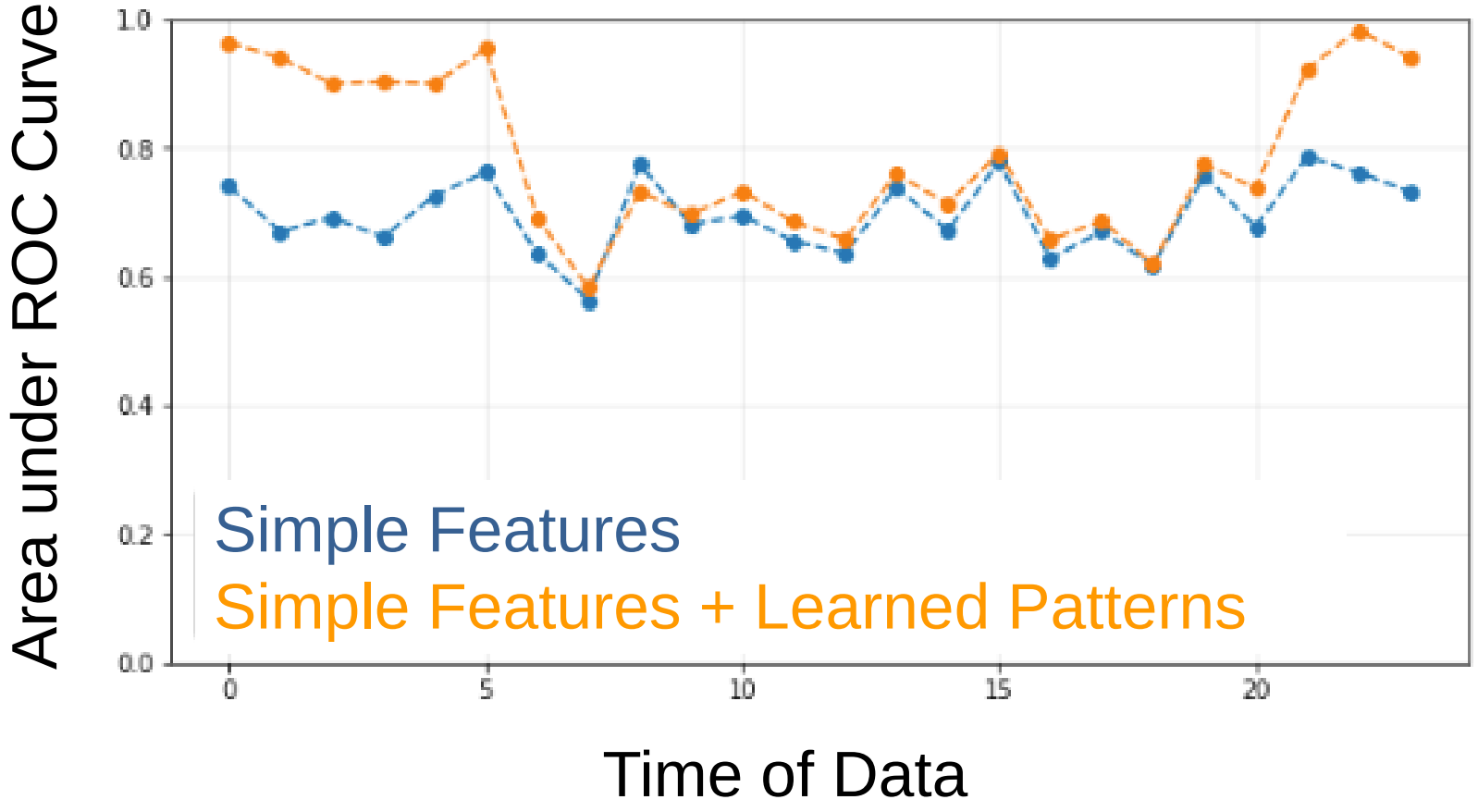
Learning Patterns

- Domain knowledge: Shape is important
- Partition and cluster data
- Feature is distance to prototype of clusters that
  - Have at least five elements
  - Histogram of occurences has low entropy



https://github.com/wannesm/dtaidistance

Results: Anomaly Detection for Water Usage



Critical Reflection

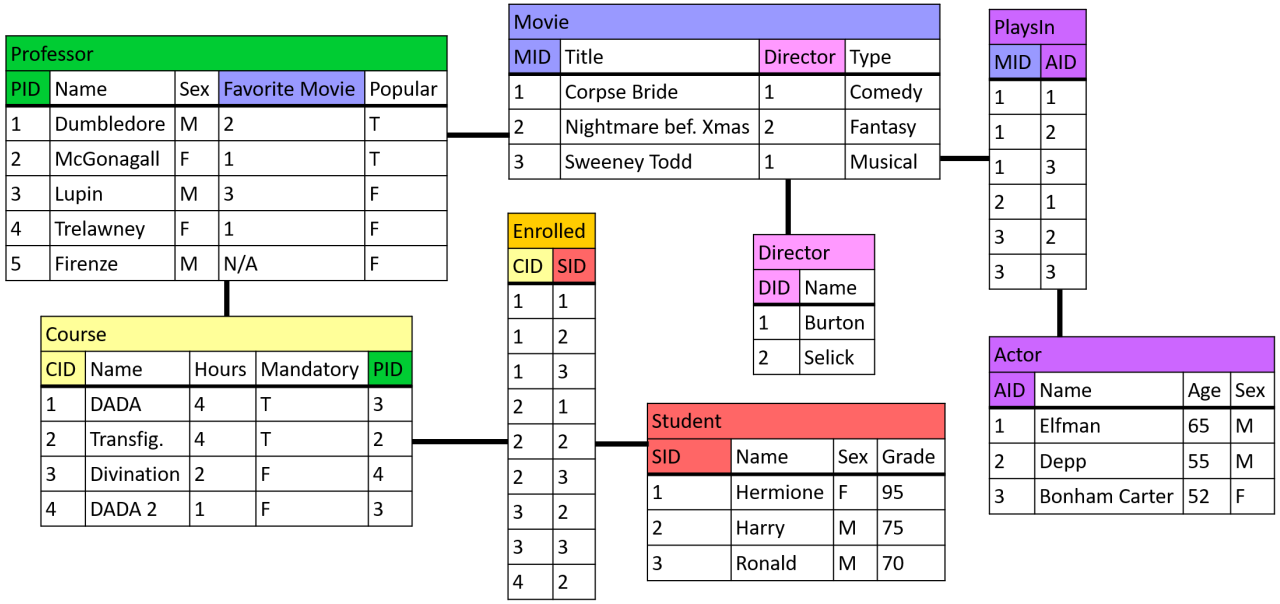
- **Observation 1:** Large performance gains from
  - Hand-crafting complex data manipulations
  - Encoding domain knowledge into problem
  - Hand-crafting features to propositionalize the data
- **Observation 2:** >90% time spent on coding above steps and data collection
- **Observation 3:** We repeatedly solve nearly identical issues in other domains from scratch

Our main lesson learned:

1. Automating these operations = huge win
2. Still need someway to incorporate knowledge

Part II: Constructing Features from Relational Data(bases)

Relational Data



Professor

PID	Name	Sex	Favorite Movie	Popular
1	Dumbledore	M	2	T
2	McGonagall	F	1	T
3	Lupin	M	3	F
4	Trelawney	F	1	F
5	Firenze	M	N/A	F

Movie

MID	Title	Director	Type
1	Corpse Bride	1	Comedy
2	Nightmare bef. Xmas	2	Fantasy
3	Sweeney Todd	1	Musical

PlaysIn

MID	AID
1	1
1	2
1	3
2	1
3	2
3	3

Director

DID	Name
1	Burton
2	Selick

Course

CID	Name	Hours	Mandatory	PID
1	DADA	4	T	3
2	Transfig.	4	T	2
3	Divination	2	F	4
4	DADA 2	1	F	3

Enrolled

CID	SID
1	1
1	2
1	3
2	1
2	2
2	3
3	2
3	3
4	2

Student

SID	Name	Sex	Grade
1	Hermione	F	95
2	Harry	M	75
3	Ronald	M	70

Actor

AID	Name	Age	Sex
1	Elfman	65	M
2	Depp	55	M
3	Bonham Carter	52	F

Patient	Abnormality	Date	Calc.	…	Mass Size	Loc	Cancer
P1	1	5/02	No		0.03	RU4	Y
P1	2	5/04	Yes		0.05	RU4	N
P1	3	5/04	No		0.04	LL3	N
…	…	…	…		…	…	…

Propositionalization

Given: Base Table

Do: Extend with newly constructed features

Abnormality	Patient	Date	Calcification Fine/Linear	…	Mass Size	Loc	Cancer
1	P1	5/02	Absent		3mm	RU4	No
2	P1	5/04	Present		5mm	RU4	Yes
3	P1	5/04	Absent		4mm	LL3	No
4	P2	6/00	Absent		2mm	RL2	No
…	…	…	…		…	…	…

Propositionalization

Given: Base Table

Do: Extend with newly constructed features

Abnormality	Patient	Date	Calcification Fine/Linear	…	Mass Size	Increase In Size	Loc	Cancer
1	P1	5/02	Absent		3mm	No	RU4	No
2	P1	5/04	Present		5mm	Yes	RU4	Yes
3	P1	5/04	Absent		4mm	No	LL3	No
4	P2	6/00	Absent		2mm	No	RL2	No
…	…	…	…		…	…	…	…

Three Dimensions of Propositionalization Systems

- Perspective: ILP-Based vs. Database Oriented
- Types of features constructed
- Static vs. Dynamic

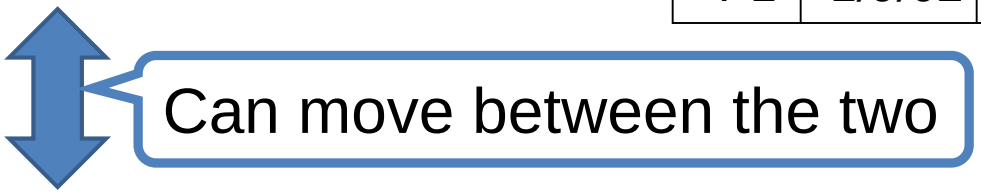
Perspective: Database vs. Logic

Patient

PID	Gender	Birthday
P1	M	3/22/63

Lab Tests

PID	Date	Lab Test	Result
P1	1/1/01	blood glucose	42
P1	1/9/02	blood glucose	45



patient(p1, m, 3/22/63)

lab_test(p1, 1/1/01, blood_glucose, 42)

lab_test(p1, 1/9/02, blood_glucose, 45)

Big Difference: Answers to queries
Database: Find ALL answers to a query
Logic programming: Find ONE answer to a query

Difference: How Features Are Defined

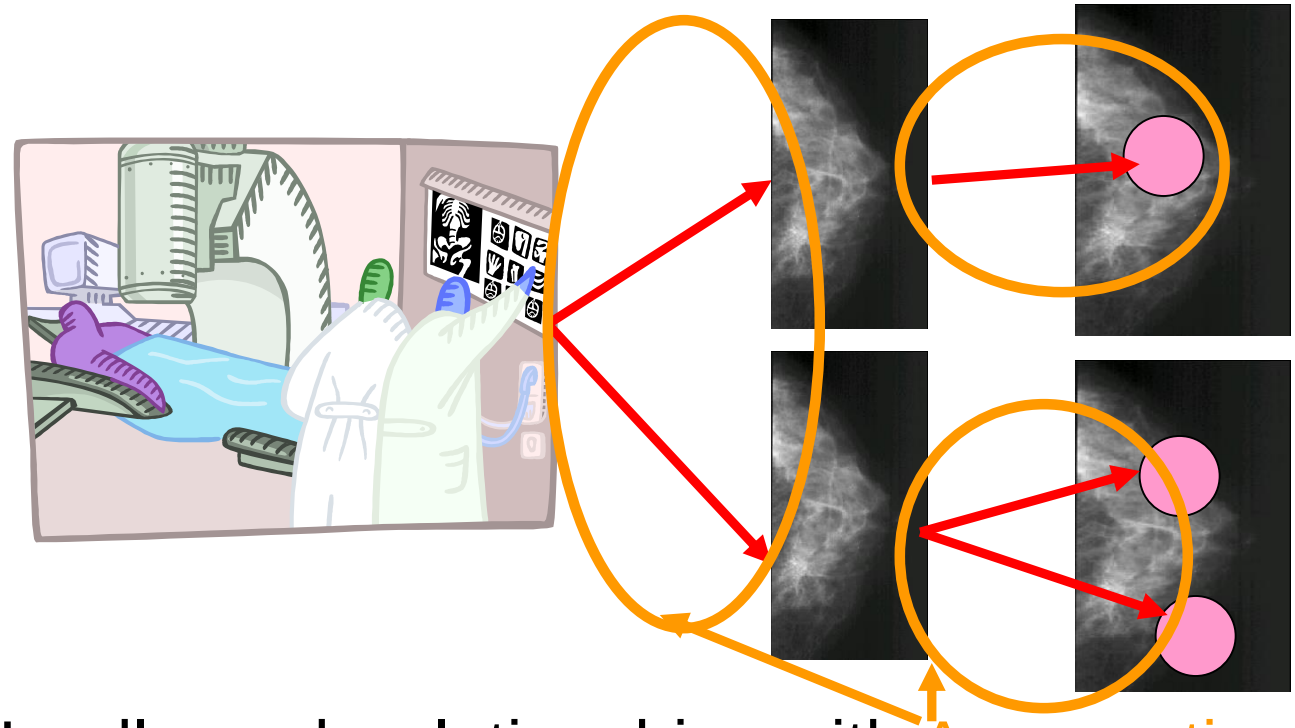
- Database: SQL queries

```
SELECT Student.SID
FROM Student
WHERE Student.SID in
      (SELECT S.SID
        FROM  Students as S, Enrolled as E, Courses as C
        WHERE S.SID = E.SID and
                    E.CID = C.CID and
                    C.difficulty = high)
```

- Logic: Clauses

```
intelligent(A) :-
      enrolled(A,C),
      difficulty(C,high).
```

Big Issue in Feature Construction: One-to-Many Relationships



Handle such relationships with Aggregation Functions

1. Existence
2. Simple aggregates
3. Complex aggregates

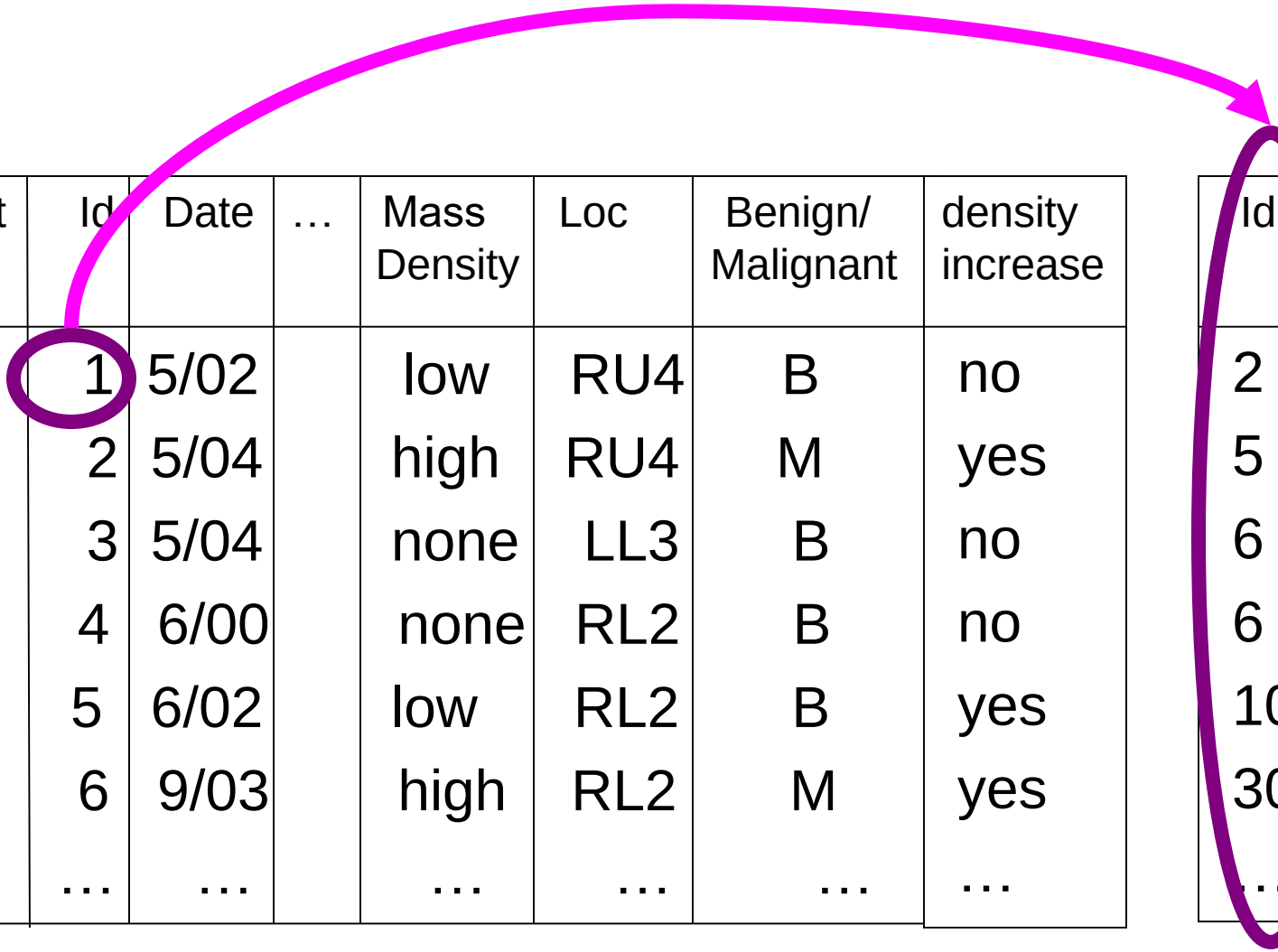
Existence Aggregation

density_increase(A,B) :- density(A,D1),
prior_mammogram_same_loc(A,B),
density(B,D2), D1 > D2.

Patient	Id	Date	…	Mass Density	Loc	Benign/ Malignant
P1	1	5/02		low	RU4	B
P1	2	5/04		high	RU4	M
P1	3	5/04		none	LL3	B
P2	4	6/00		none	RL2	B
P2	5	6/02		low	RL2	B
P2	6	9/03		high	RL2	M
…	…	…		…	…	…

Id	Id
2	1
5	4
6	5
6	5
10	9
30	28
…	…

Existence Aggregation

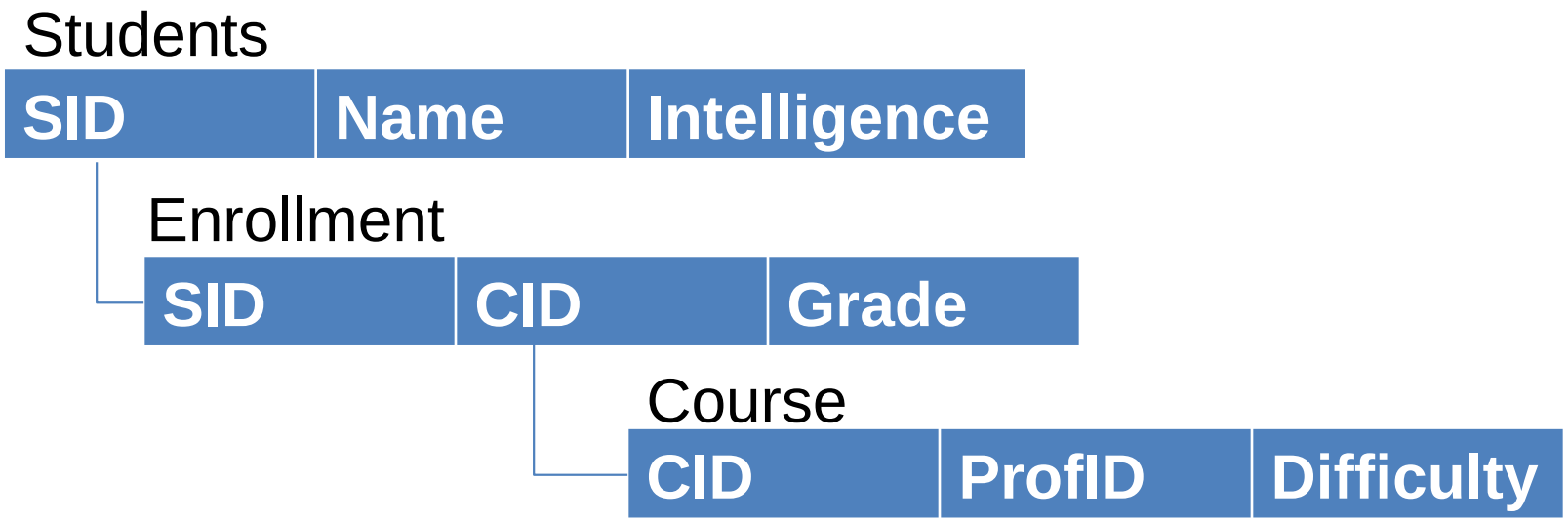


Patient	Id	Date	…	Mass Density	Loc	Benign/ Malignant	density increase
P1	1	5/02		low	RU4	B	no
P1	2	5/04		high	RU4	M	yes
P1	3	5/04		none	LL3	B	no
P2	4	6/00		none	RL2	B	no
P2	5	6/02		low	RL2	B	yes
P2	6	9/03		high	RL2	M	yes
…	…	…		…	…	…	…

Id	Id
2	1
5	4
6	5
6	5
10	9
30	28
…	…

Simple Aggregation

- Suppose goal is predict students intelligence



- Simple aggregate features : No conditioning
  - Average grade in enrolled classes
  - Max grade in enrolled classes
  - …

Example: Count Aggregation

density_increase(A,B) :- density(A,D1),
prior_mammogram_same_loc(A,B),
density(B,D2), D1 > D2.

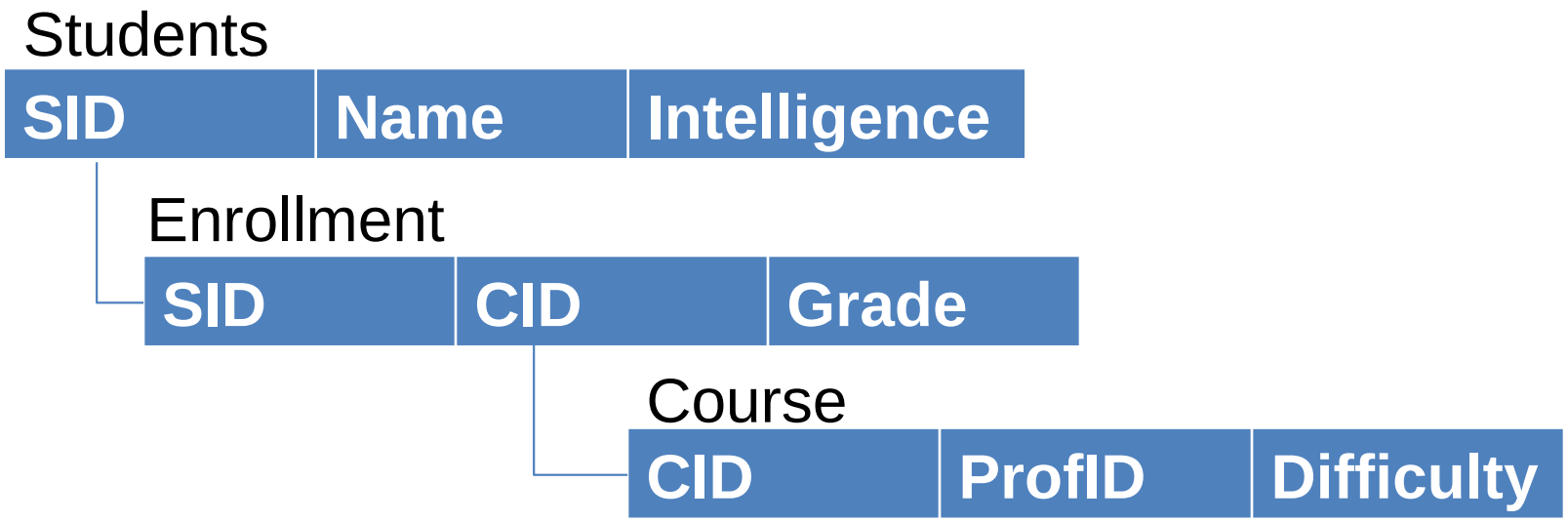


Patient	Id	Date	…	Mass Density	Loc	Benign/ Malignant
P1	1	5/02		low	RU4	B
P1	2	5/04		high	RU4	M
P1	3	5/04		none	LL3	B
P2	4	6/00		none	RL2	B
P2	5	6/02		low	RL2	B
P2	6	9/03		high	RL2	M
…	…	…		…	…	…

Id Count	Count
1	0
2	1
3	0
4	0
5	1
6	2
…	…

Complex Aggregation

- Suppose goal is predict students intelligence



- Complex aggregate features : Conditioning
  - Average grade in classes where Difficulty = High
  - Min grade in classes where Difficulty = High
  - …

Static vs. Dynamic

- Static approaches use a two-step process:
  - Convert the relational data to one table
  - Apply any propositional learner to the table

- Dynamic approaches interleave feature construction and model learning
  - Model guides search
  - Tied to specific model

Canonical Static ILP approach

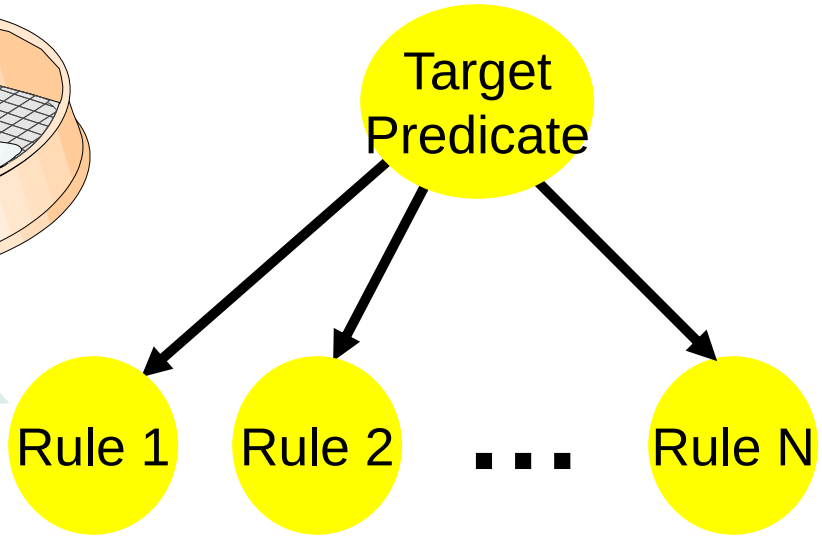
Step 1

Step 2

Step 3

ILP Engine





Learn

Select

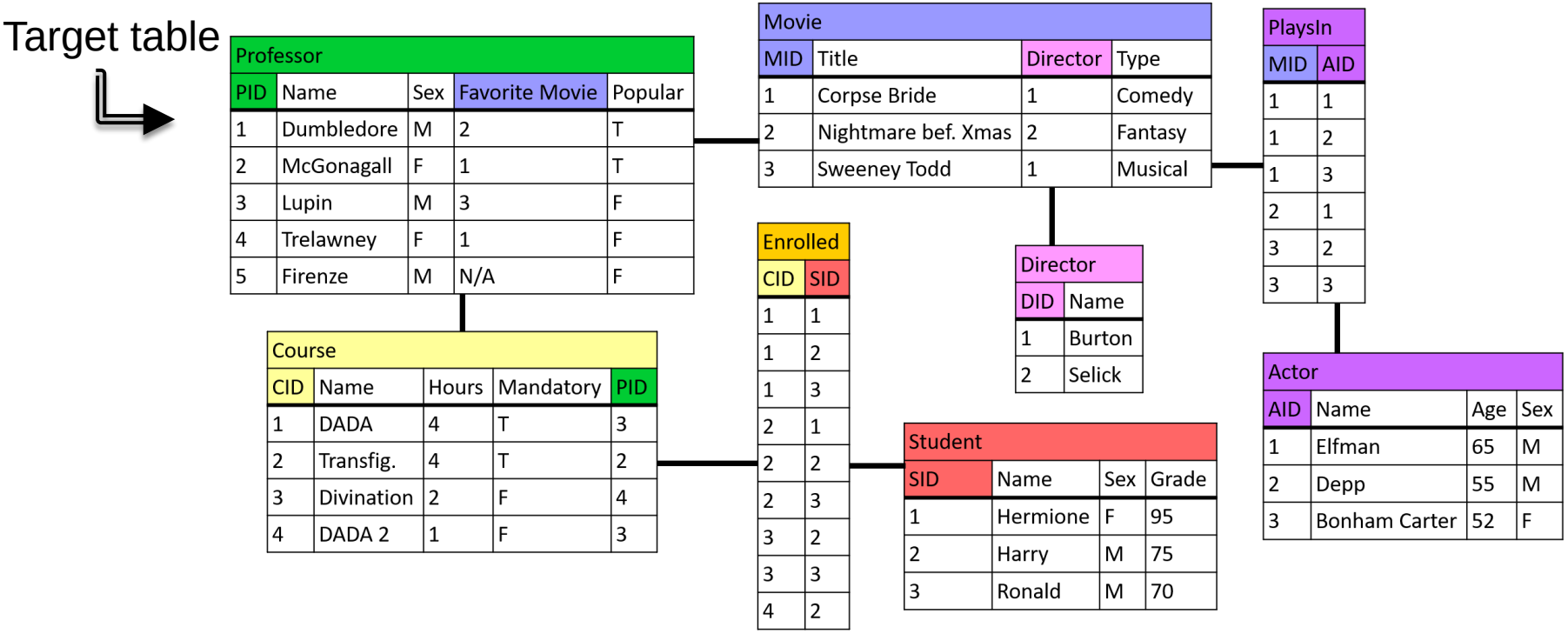
Build Model

Detailed Approach

- Step 1: Apply your favorite inductive logic programming system to generate rules
- Step 2: Optionally select a subset, e.g.,
  - Must cover at least 10 positive examples
  - Remove duplicates
  - 50 highest scoring |pos. covered| - |neg. covered|
- Step 3: Construct table s.t. each rule is a feature
  - 1 if example matches rule
  - 0 otherwise
- Step 4: Apply favorite propositional learner

OneBM: A Static Database Approach

[Lam et al., Slides from Jonas]



Big idea: Start at target table and iterative

1. Use foreign keys to access neighboring tables
2. Construct features based

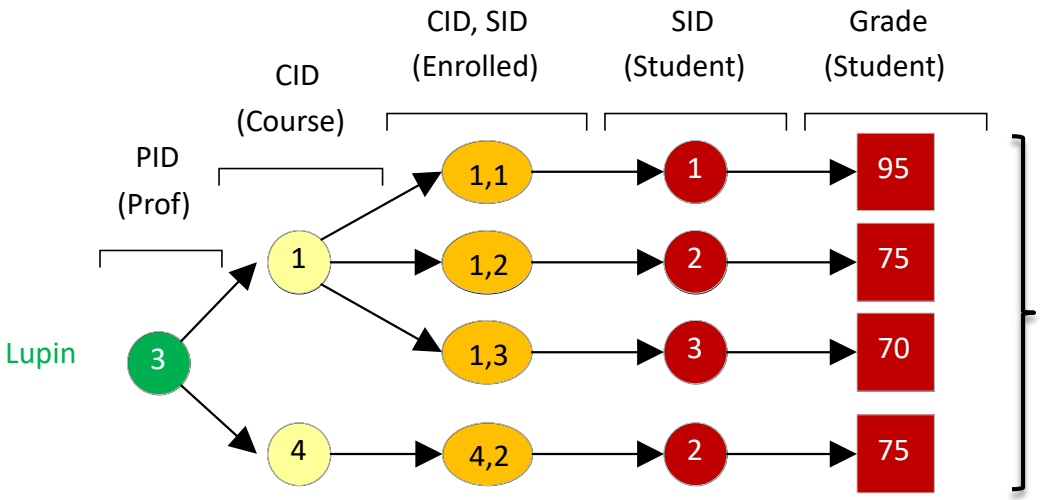
OneBM Approach

Equijoin on c1

Join path: $T_0 \xrightarrow{c_1} T_1 \xrightarrow{c_2} T_2 \xrightarrow{c_3} \cdots T_k$

Join-project path: $T_0 \xrightarrow{c_1} T_1 \xrightarrow{c_2} T_2 \xrightarrow{c_3} \cdots T_k \mapsto \mathrm{A}$

Join-project path: Professor $\xrightarrow{PID}$ Course $\xrightarrow{CID}$ Enrolled $\xrightarrow{SID}$ Student $\mapsto$ Grade

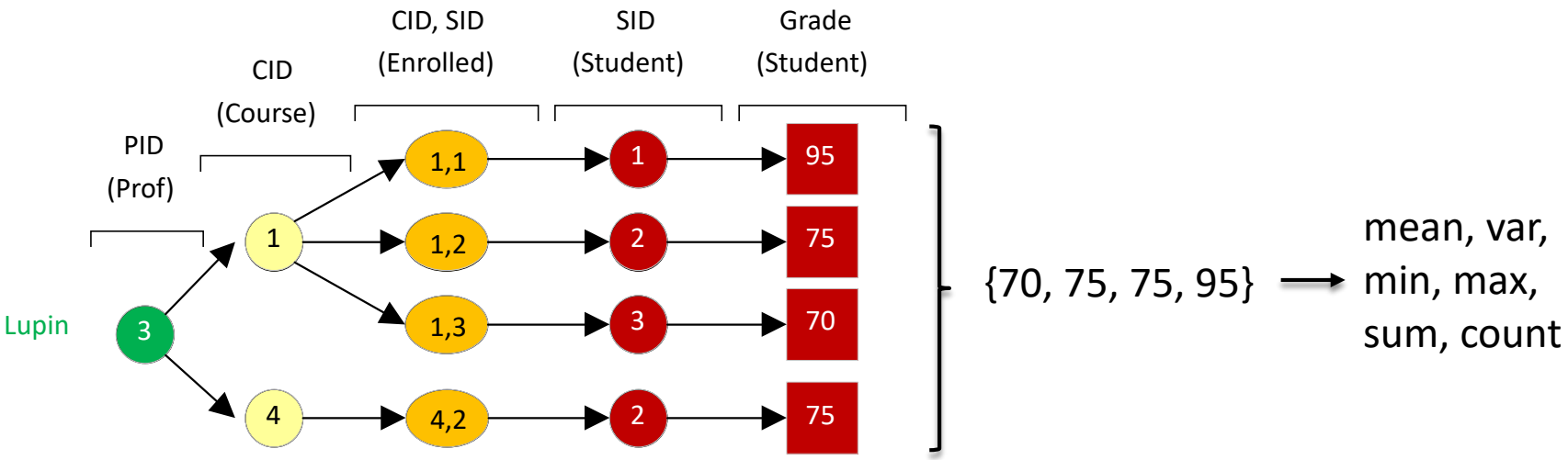


All grades given by each professor

{70, 75, 75, 95}

OneBM: Types of Features

- 1-to-many: Features depending on the type of A:
  - Categorical: count, count distinct
  - Numerical: mean, variance, min, max, sum, count
  - Other aggregations for other types
- 1-to-1: Simply use the value of A in the data



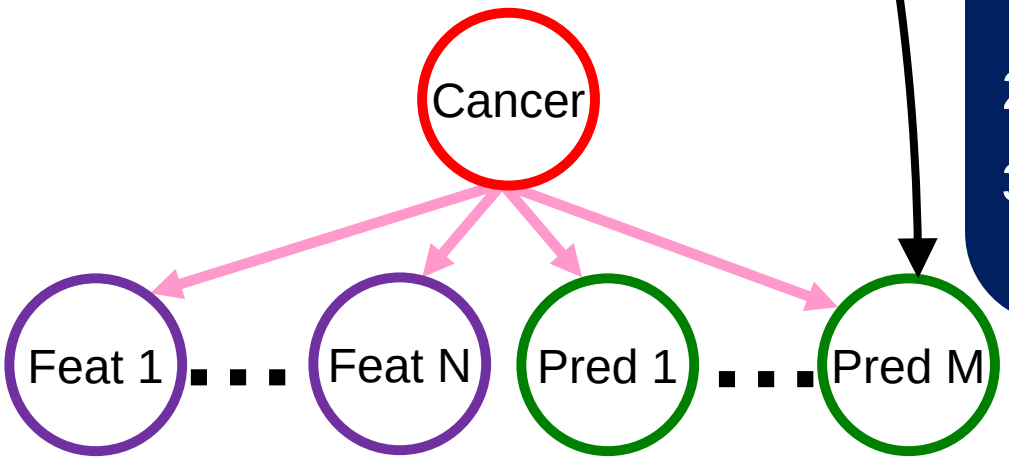
Overview of SAYU-VISTA

[Davis et al. ECML'05, Davis et al. IJCAI'07]

If-then rules capture implicitly defined features

$$\text{Prior}(id1,id2) \wedge \text{Size}(id1, s1) \wedge \text{Size}(id2,s2) \wedge s1 > s2 \Rightarrow \text{Cancer}(id1)$$

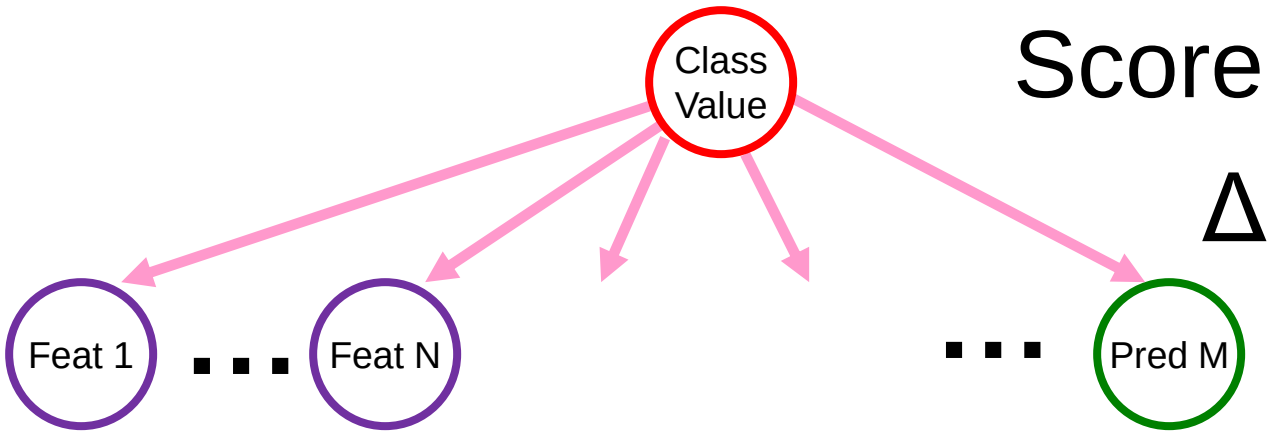
Rules become features in statistical model



SAYU-VISTA:

1) Integrates feature induction and model construction
2) Constructs implicit features
3) Produces human interpretable results

SAYU-VISTA Illustration

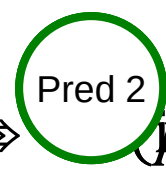


Score = 0.05

Δ = -0.03

$\text{Size}(id1, s1) \wedge \ldots \wedge$

$\text{Age}(id2, s2) \wedge \ldots \wedge \Rightarrow (id1, id2)$



Sample Learned Rule

[Burnside et al., AMIA'05]

Cancer(*id1*) **IF**

Birads_category(*id1*, B5) **AND**

MassPAO(*id1*, Present) **AND**

MassesDensity(*id1*, High) **AND**

HO_breastCA(*id1*, HxDCorLC) **AND**

In_same_mammogram(*id1*, *id2*) **AND**

Calc_pleomorphic(*id2*, NotPresent) **AND**

Calc_punctate(*id2*, NotPresent)

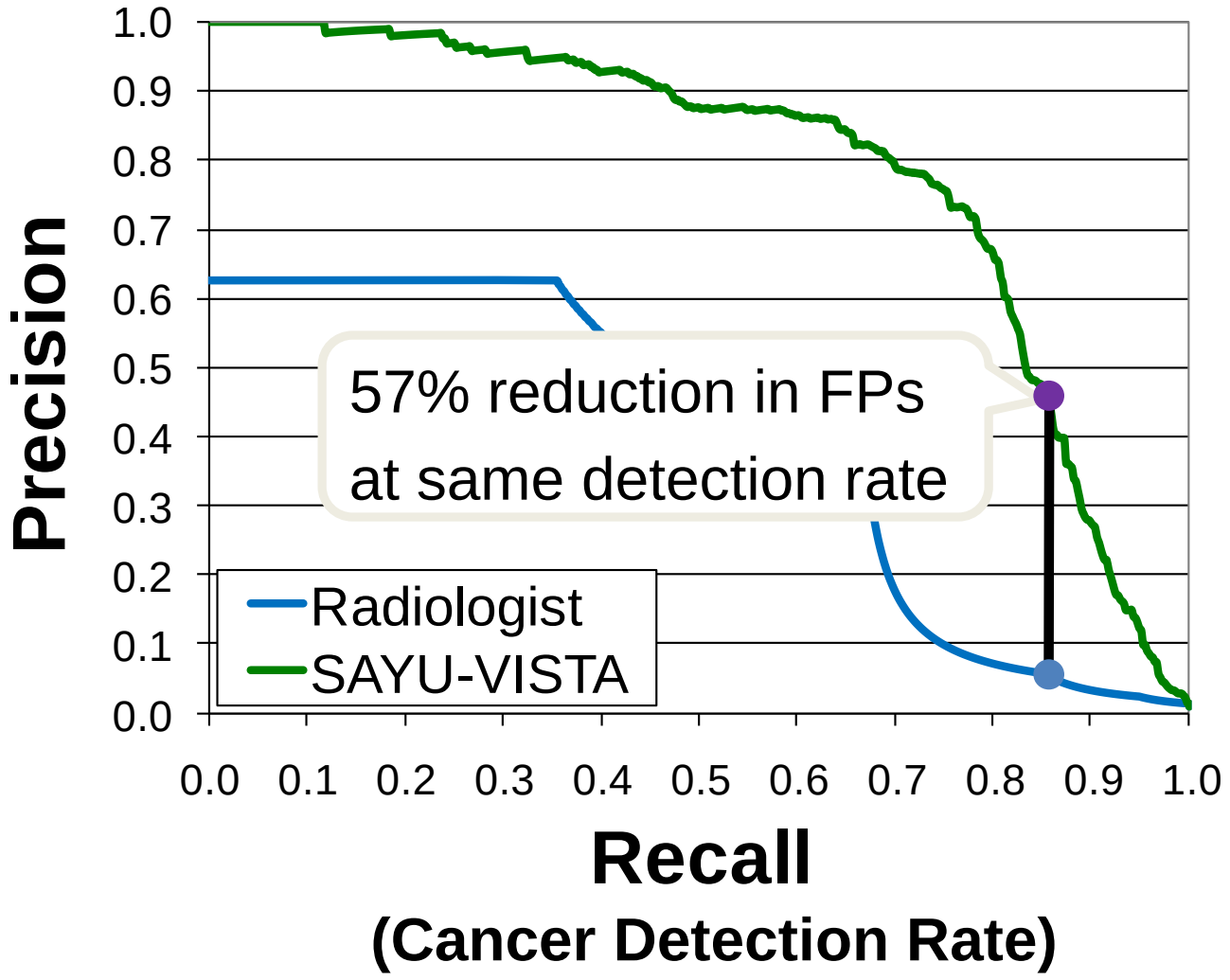
Empirical Evaluation

Task: Predict whether abnormality is malignant

- Radiologist description of a mammogram
- Matched to cancer registry to eliminate false negatives in data
- 500 positives and 62,000 negatives

Goal: Compare performance of model learned by SAYU-VISTA and radiologist

Results:
Precision-Recall Curve



Static vs. Dynamic: Pros and Cons

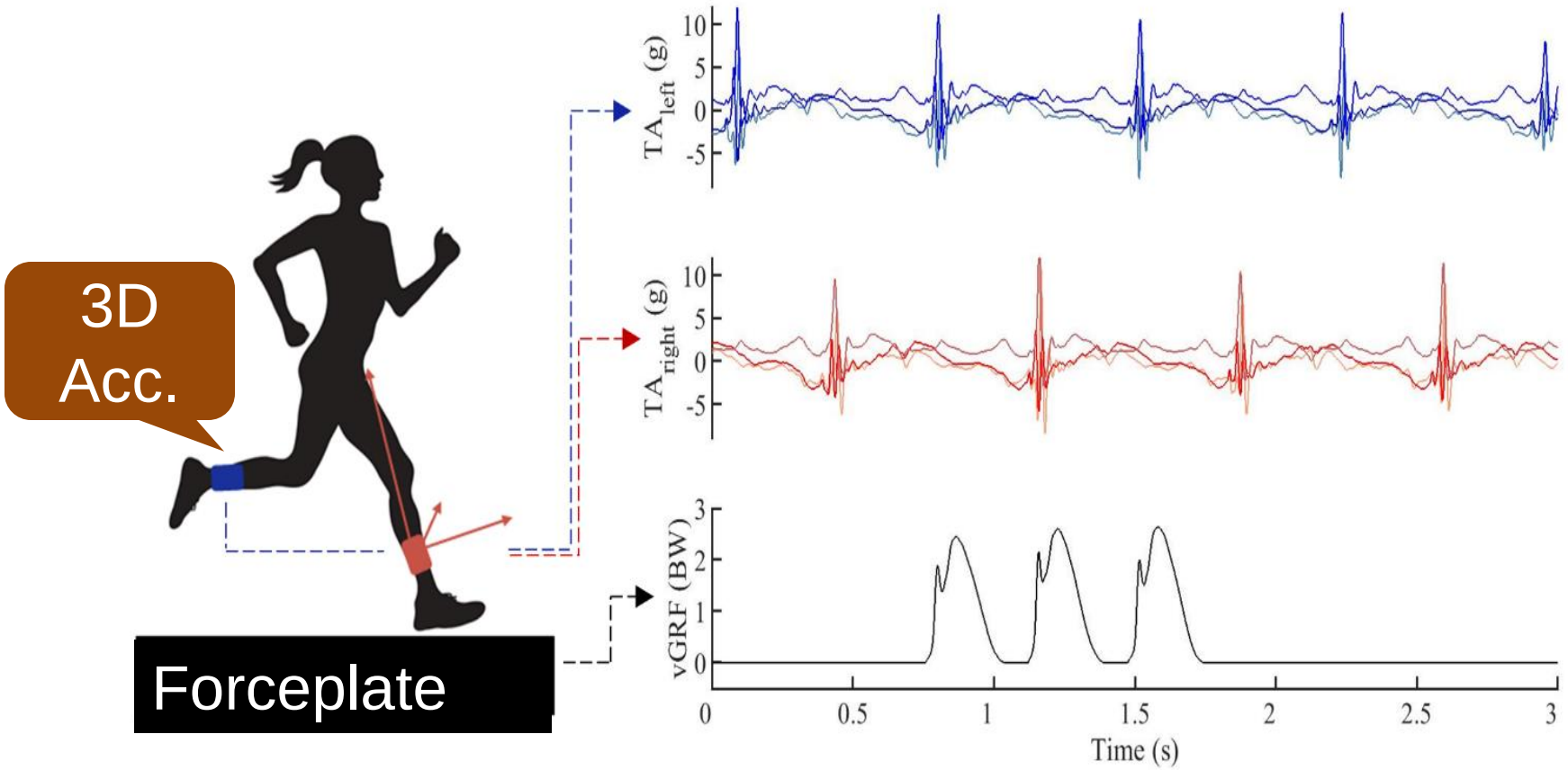
- Static
  - +Can apply any learner
  - +Can switch learning tasks
  - - Slow: Generate lots of features
- Dynamic
  - +Faster
  - +Focuses on features for useful for task at hand
  - - Potentially a more complicated search problem
  - - Constructed features tied to specific learner and task

An Incomplete List of Approaches

- Wordifcation
- nFOIL
- kFOIL
- SAYU
- LazyBum
- RSD
- LINUS
- MODL
- OneBM
- DeepFeature Synthesis
- Relf
- POLKA
- Relaggs
- DINUS
- …

Part III: Constructing Features from Time Series Data

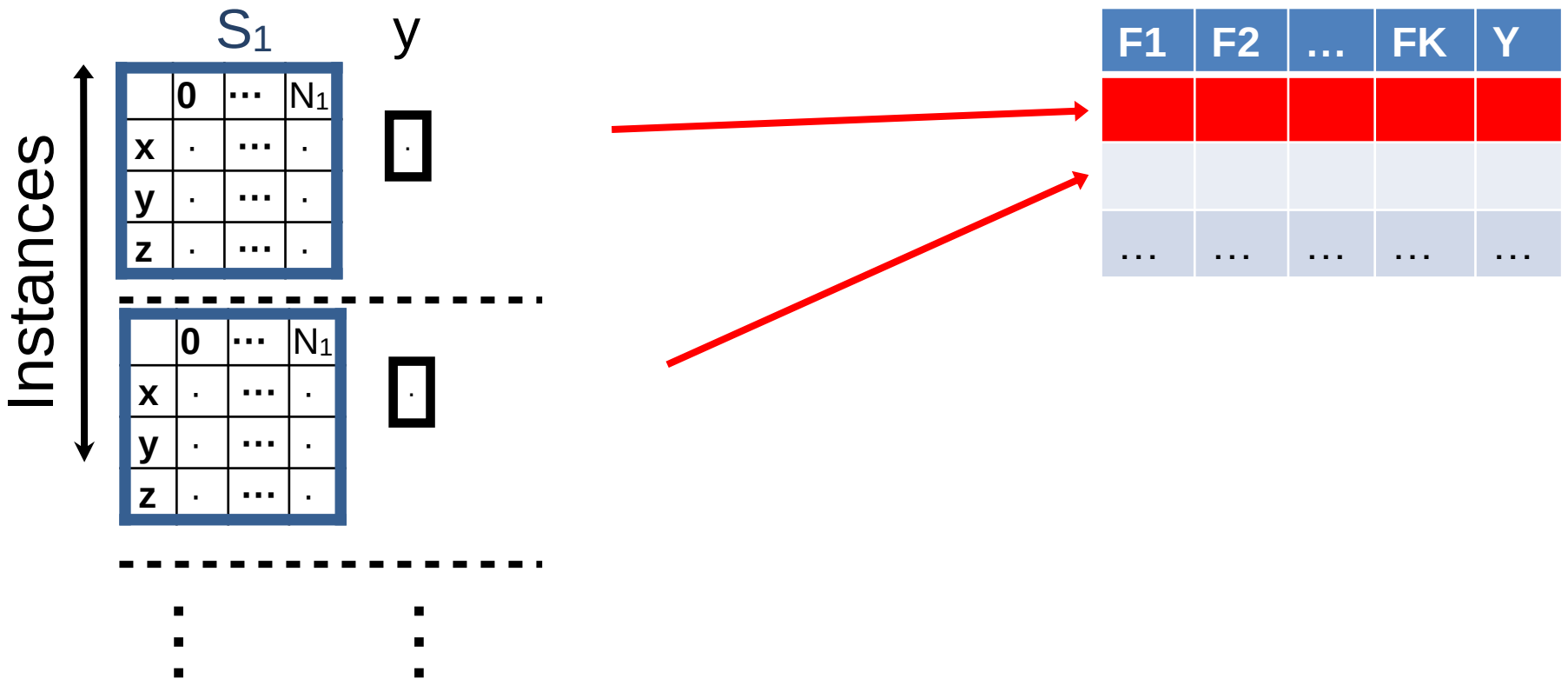
Example Time Series Data



Problem Setting 1

Given:
Uni or Multivariate Data

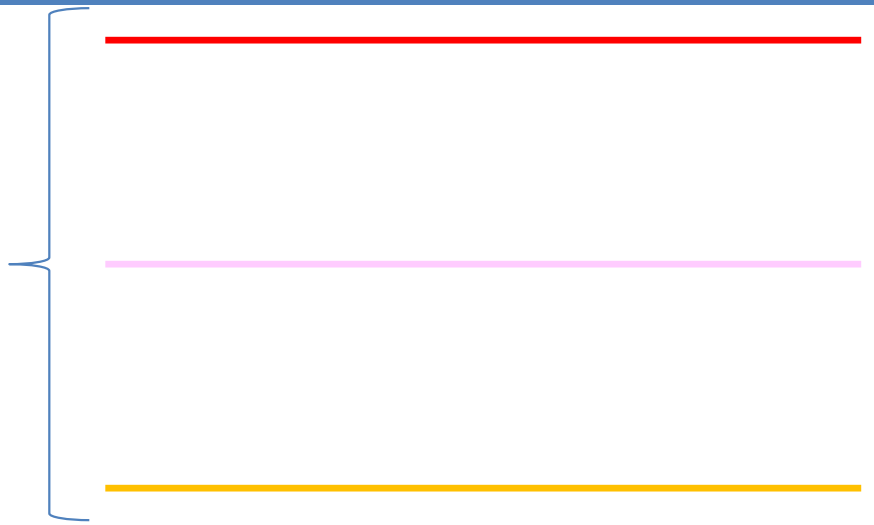
Do: Construct
Attribute-Value Table



tsFRESH: Each Series Treated Indepedently

Approach:

1. Extract features from each series
2. Optionally apply feature selection
3. Concatenate results



ID	F1	…	FK	F1	…	FK	F1	…	FK	Y
1										
…										
n	…	…	…							…

tsFresh Settings

- Types of features
  - Basic statistical: Min, Max, Median, StdDev,
  - Advanced stats.: Entropy, kurtosis, skewness, …
  - Time series: Peaks, autocorrelation, …
  - Functions: Linear trends, FFT, …
- Feature settings
  - Comprehensive: 63 “types” yielding 794 features
  - Minimal: 8 features
  - Efficient: Avoid expensive features

https://tsfresh.readthedocs.io/en/latest/text/feature_extraction_settings.html
https://tsfresh.readthedocs.io/en/latest/text/list_of_features.html

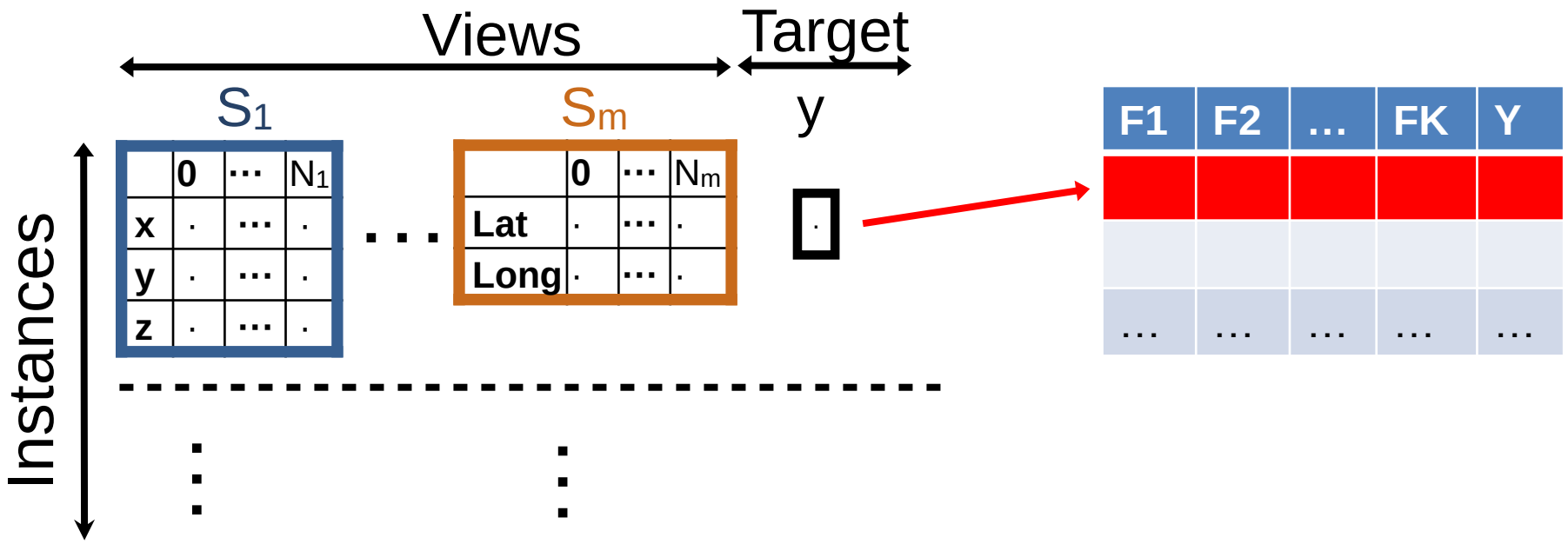
Desiderata for Time Series Feature Constructors

- Support a variety of feature types
  - Standard statistics
  - Patterns
- Combine information
  - Fuse signals to derive new ones
  - Fuse attributes to derive new ones
- Non-standard normalizations...ideally
- Efficient…ideally

Problem Setting 2

Given:
Multi-view Data

Do: Construct
Attribute-Value Table

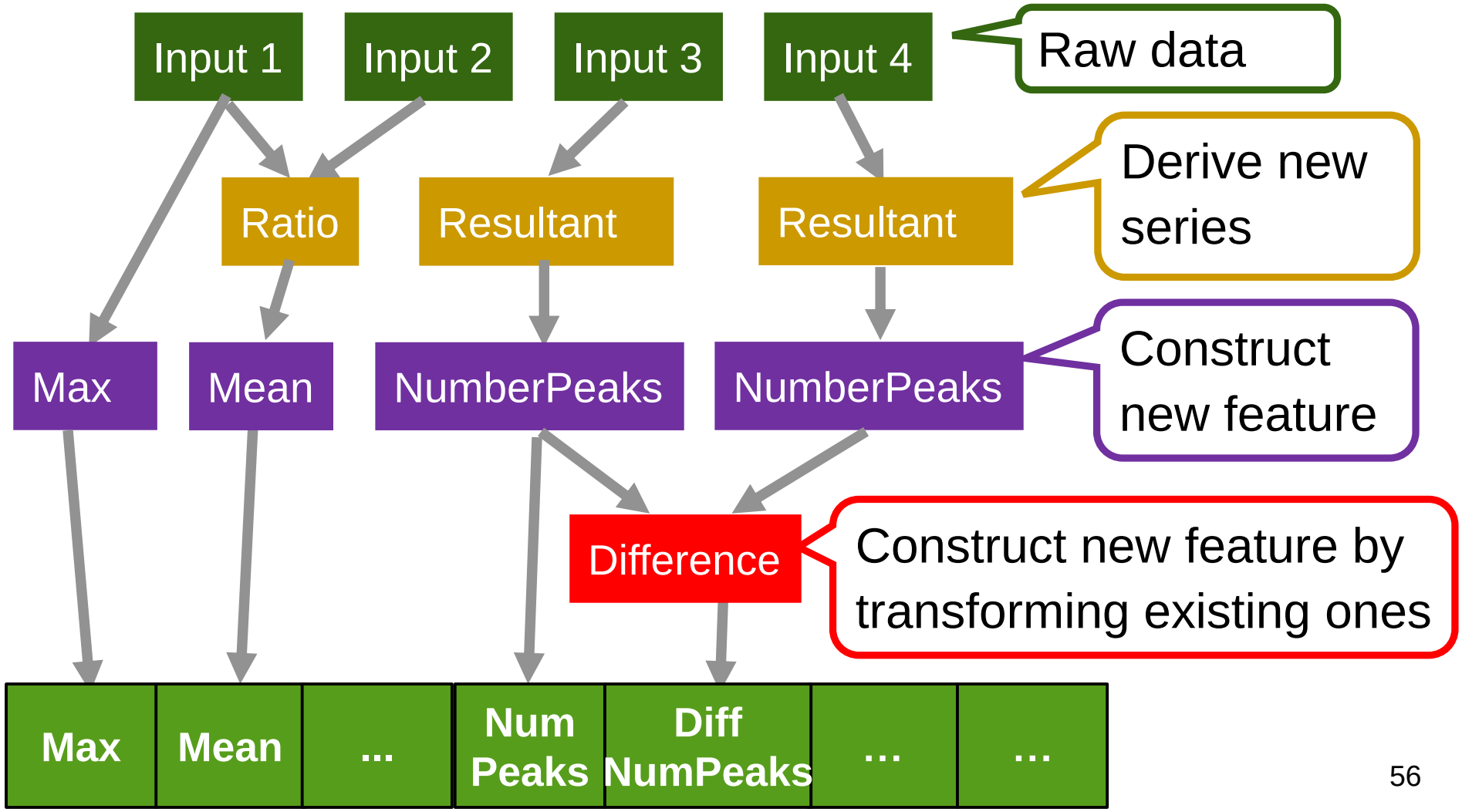


Main challenge: Too many operations to try them all

Approach: TSFuse

- **Insight 1:** Operations transform the data
  - Add new “signal” (or “view”)
  - Construct feature from signal
- **Insight 2:** Each transformation enables new ones
  - E.g., Derive new feature by comparing previously constructed features
- **Approach intuition:**
  - Represent feature construction process as a directed, acyclic computation graph
  - Search through the space of possible graphs

Representation: Computation Graph



Transformations

- **Series-to-series:** Derive new series (~5)
  - Resultant (e.g., acceleration)
  - Angles (e.g., movement data)
  - Etc.
- **Series-to-attributes:** Construct feature (~ 60)
  - Statistical features (min, max, average, etc.)
  - Patterns (e.g., clusters)
  - Etc.
- **Attribute-to-attribute:** Fuse features (~5)
  - E.g., Difference, Inverse, etc.

Searching for a Computation Graph

Start with an empty computation graph

Repeat

- **Phase 1:** Add new series
  - Add series-to-series transformers to graph
  - Remove transformers that construct redundant series
- **Phase 2:** Construct attributes
  - Add series-to-attribute transformers
  - Add attribute-to-attribute transformers

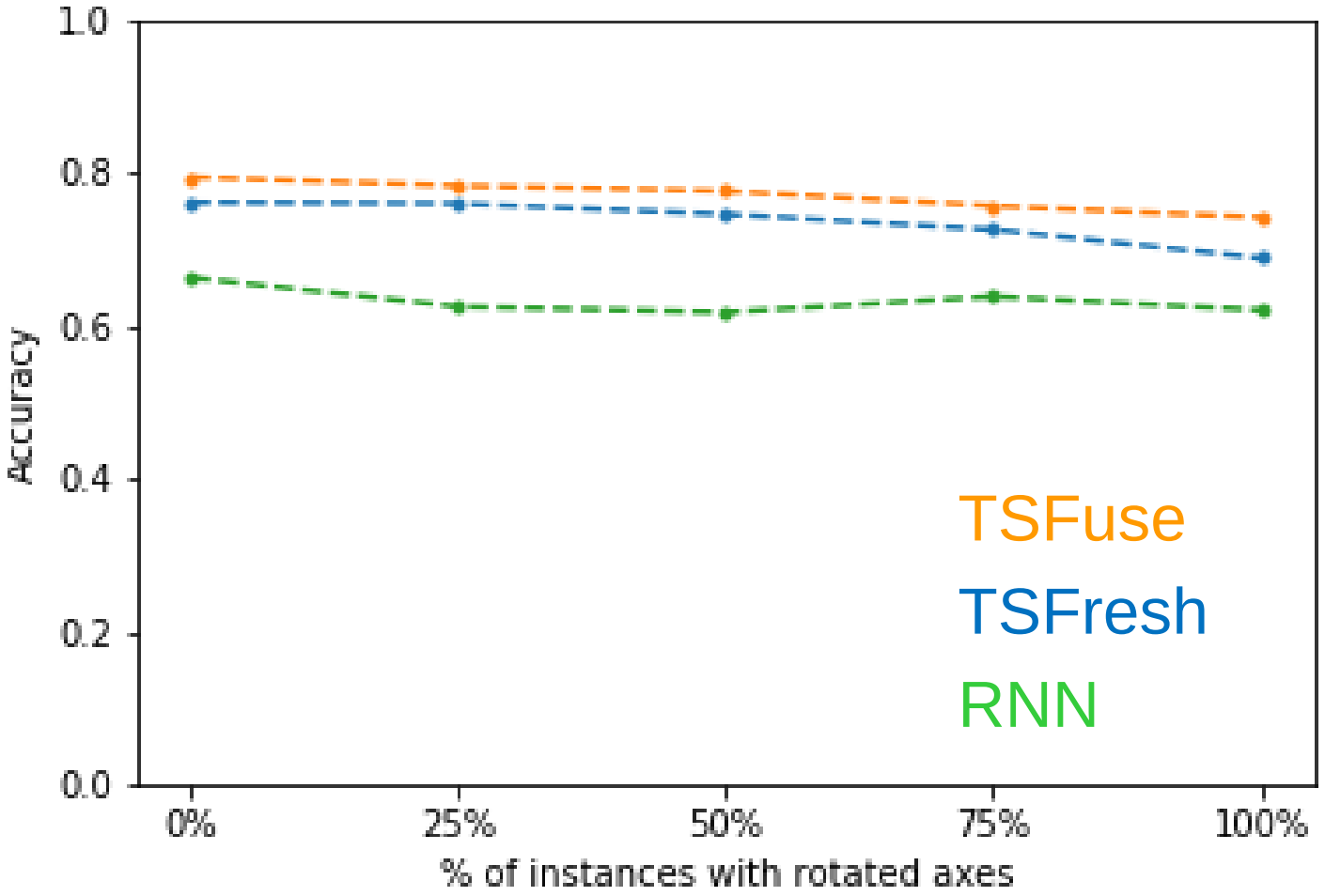
Limiting the Search Space

- **Redundancy check:** Evaluate zero lag cross-correlation between all pairs of series
- **Compatibility relation:** Domain-knowledge encodes which data sources can be combined
  - Meta-data about the sensors (location, type,…)
  - E.g., Only compare sensor on opposite sides of the body
- **Max-depth parameter:** Limit the number of chained transformers (e.g., series-to-series)

Use Case: Activity Recognition

- **Given:** 3D accelerometer data from a hip-mounted mobile phone
- **Predict:** Person's activity (walking, ascending stairs, descending stairs, cycling, jogging)
- Setup: Simulate different attachments by rotating axes
- Approaches compared:
  - TSFuse + GBT
  - TSFresh + GBT (Time series features, but no fusion)
  - RNN (LSTM)

Results Activity Recognition



We Are Not There Yet: User Still Must Do Many Complex Operations

- Window / partition time series and define relevant target labels
- Apply preprocessing operations
  - Filtering
  - Normalization
- Encode domain knowledge

Part IV: Conclusions

Feature Construction for Tabular Data

- Types of features
  - Discrete features: Logical formulas using and, or, negation to combine base features
  - Real-valued: Transform one base feature using inverse, sin, squared, …
  - Real-valued: Combine multiple base features using sum, difference, division, …
- Search
  - Brute force enumeration
  - Some work on RL-based approaches

Summary

- Feature construction = search + selection
- Search: Breadth first, depth-first, RL
- Selection:
  - Static: (1) Construct all features in search and (2) Select those that are relevant
  - Dynamic: Use model to guide selection of and search for features

Take Away: Preprocessing and Feature Construction Are Reoccurring Problems

- Real data is never in the right format
- Finding the right representation
  - Takes the vast majority of the time
  - Requires ways to encode domain knowledge
- (Partially) automating = huge time savings
  - E.g., Sports science PhD students spend ~50% on these tasks
  - We could easily make them twice as efficient

Take Away: Real-World Data ≠ UCI Data

- Data is often small: Purely data-driven approaches will struggle
  - 100 trainings each year per player
  - 50 games each year per player
- Data lacks labels
- Real-world problems often yield significant new technical and research challenges

Questions?