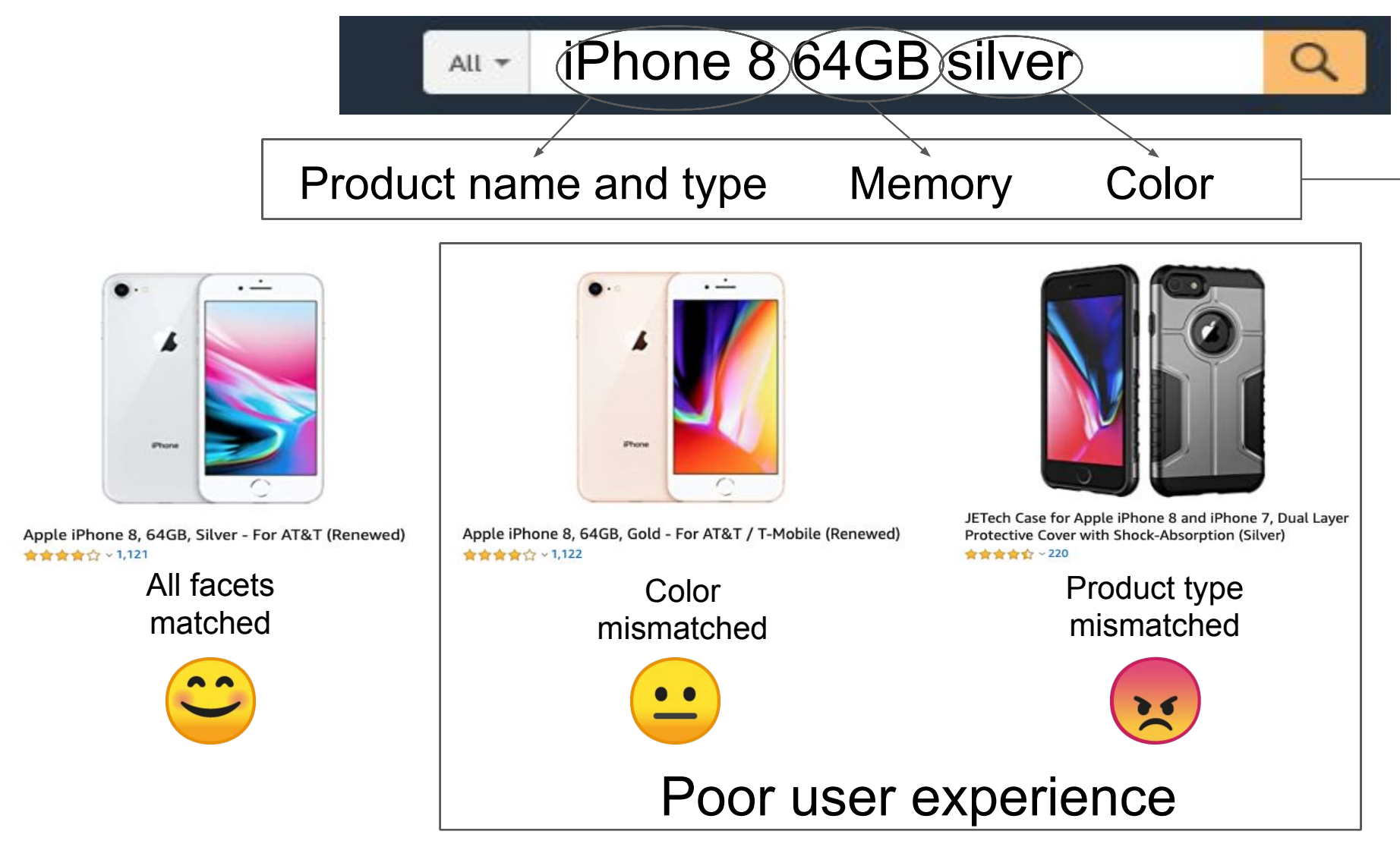
Identifying Facet Mismatches In Search Via Micrographs

Sriram Srinivasan⁺, Nikhil S Rao*, Karthik Subbian*, & Lise Getoor⁺
⁺ University of California, Santa Cruz
* Amazon LLC

Facet Mismatch in Product Search



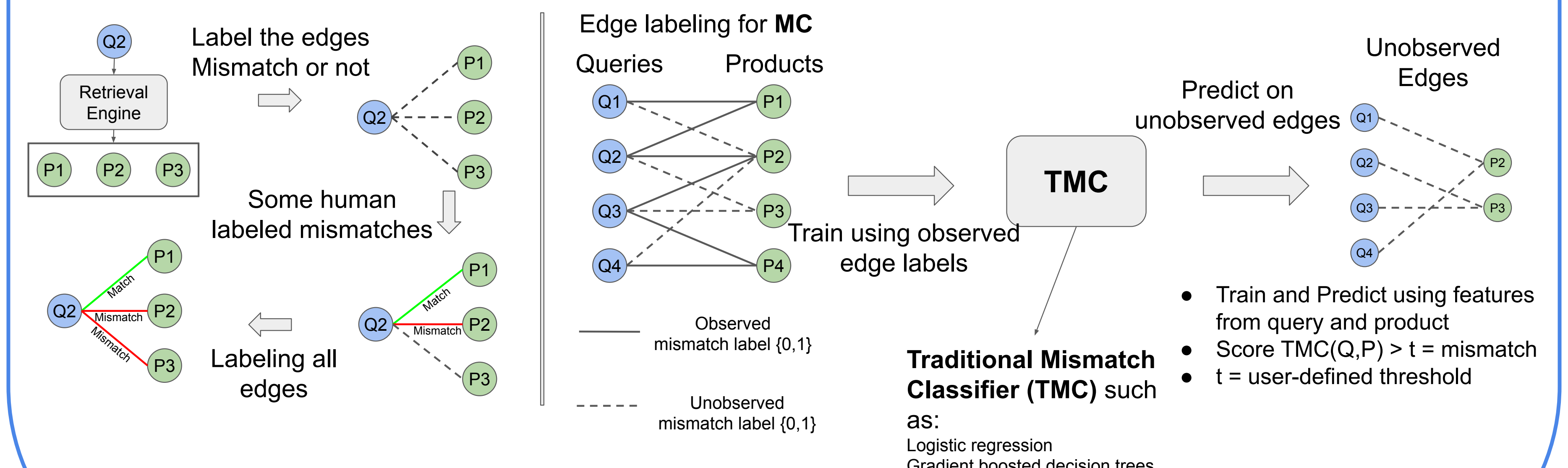
Why does it happen?

- Improper associations
- Lexical similarities
- etc..

In this work:

We identify facet mismatches
Refer to it as:
Mismatch Classification (MC)

Mismatch Classification as edge labeling

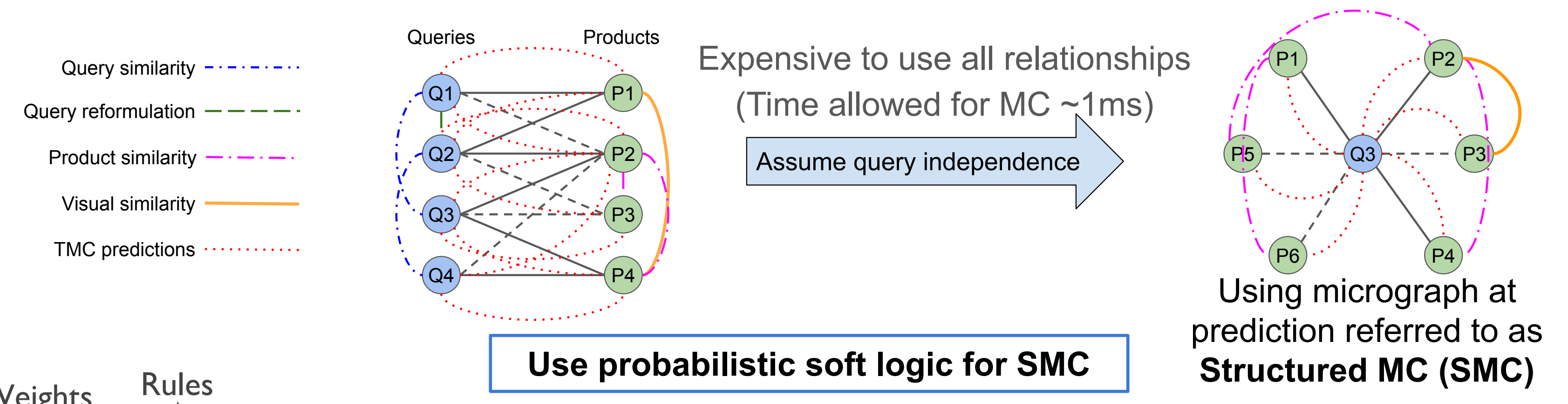

Q2
Retrieval Engine
P1
P2
P3
Label the edges
Mismatch or not
Some human labeled mismatches
Match
Mismatch
Labeling all edges
Edge labeling for MC
Queries
Products
Q1
Q3
Q4
P4
Observed mismatch label {0,1}
Unobserved mismatch label {0,1}
Train using observed edge labels
TMC
Predict on unobserved edges
Unobserved Edges

Traditional Mismatch Classifier (TMC) such as:
Logistic regression
Gradient boosted decision trees
Deep learning

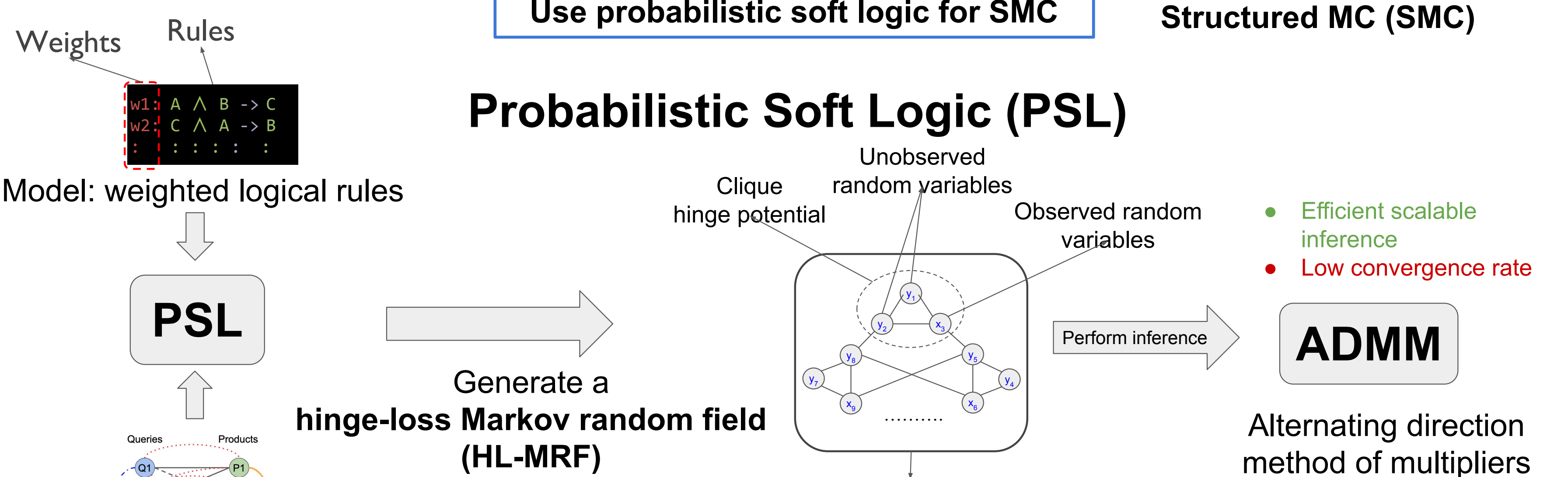
- Train and Predict using features from query and product
- Score TMC(Q,P) > t = mismatch
- t = user-defined threshold

- **Limitations of TMC:**
  - Insufficient labeled data
  - Ignores the rich relational structure such as similarities, hierarchy, etc.

Structured MC


Query similarity
Query reformulation
Product similarity
Visual similarity
TMC predictions
Queries
Products
Q1
Q2
Q3
Q4
P1
P2
P3
P4
Expensive to use all relationships
(Time allowed for MC ~1ms)
Assume query independence
Use probabilistic soft logic for SMC
Micrograph
P5
P6
Using micrograph at prediction referred to as Structured MC (SMC)

Probabilistic Soft Logic (PSL)


Weights
Rules
w1: A ∧ B -> C
w2: C ∧ A -> B
Model: weighted logical rules
PSL
Generate a hinge-loss Markov random field (HL-MRF)
Clique hinge potential
Unobserved random variables
Observed random variables
y1
y2
x3
y4
y5
x6
y7
y8
x9
Perform inference
Efficient scalable inference
Low convergence rate
ADMM
Alternating direction method of multipliers

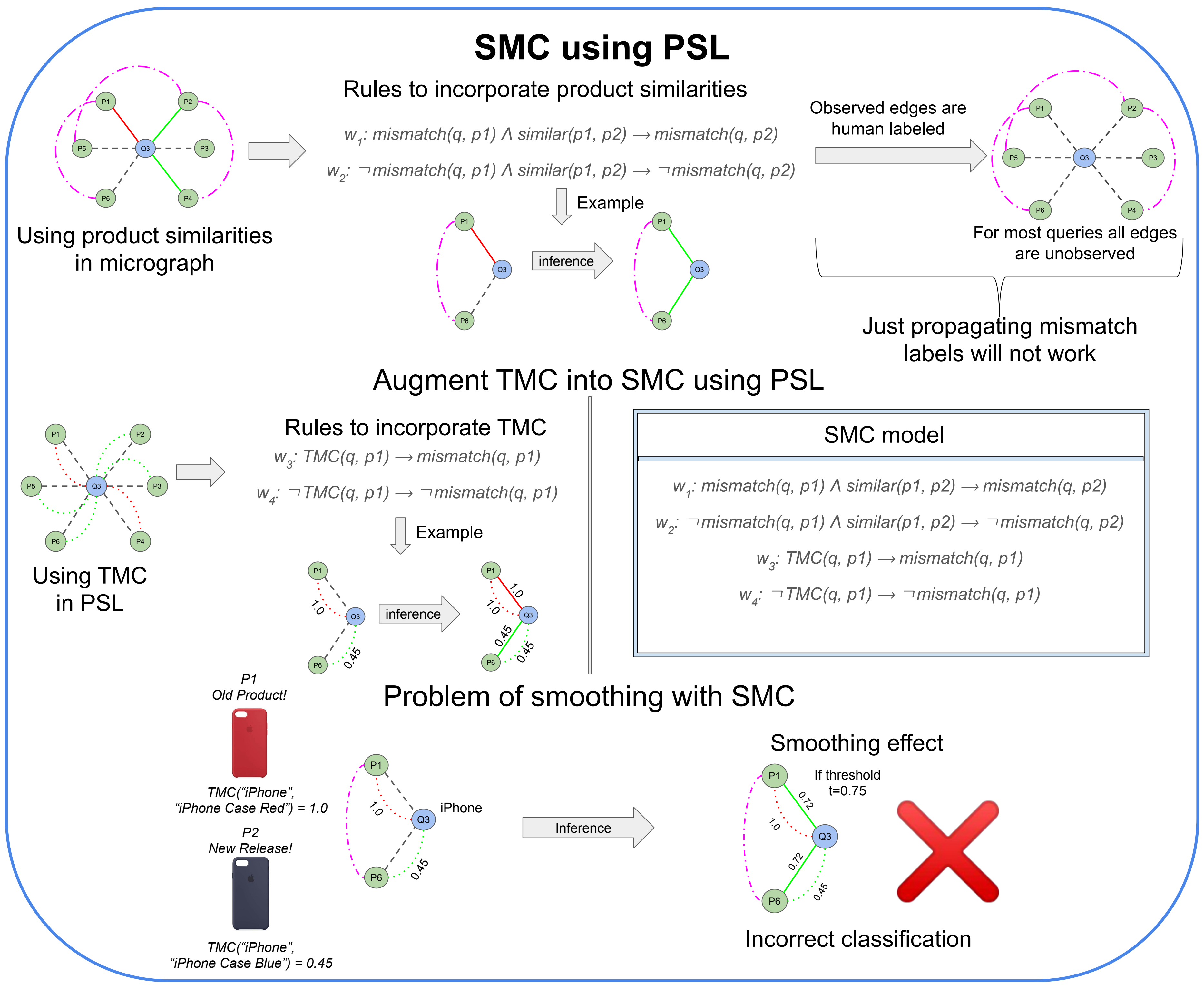
Clique potential: $\Phi_i(\mathbf{y},\mathbf{x}) = \max(l_i(\mathbf{y},\mathbf{x}), 0)^p$; where $p \in \{1,2\}$
Energy function: $f_w(y, x) = \sum_{i=1..m} w_i\Phi_i(\mathbf{y},\mathbf{x})$; where $w_i \in R^+$
Probability density: $P(\mathbf{y} \mid \mathbf{x}) = exp\{-f_w(\mathbf{y},\mathbf{x})\} / Z$
MAP inference: $argmax_y P(\mathbf{y} \mid \mathbf{x}) = argmin_y f_w(\mathbf{y},\mathbf{x})$

Convex objective

Data

Bach et. al. JMLR 2017
https://psl.linqs.org

SMC using PSL


P1
P2
P3
P4
P5
P6
Q3
Using product similarities in micrograph
Example
inference
Observed edges are human labeled
For most queries all edges are unobserved
Just propagating mismatch labels will not work
Using TMC in PSL
1.0
0.45
P1
Old Product!
TMC("iPhone", "iPhone Case Red") = 1.0
P2
New Release!
TMC("iPhone", "iPhone Case Blue") = 0.45
iPhone
Inference
Smoothing effect
If threshold t=0.75
0.72
Incorrect classification

Rules to incorporate product similarities

w_1: mismatch(q, p1) ∧ similar(p1, p2) ⟶ mismatch(q, p2)
w_2: ¬mismatch(q, p1) ∧ similar(p1, p2) ⟶ ¬mismatch(q, p2)

Augment TMC into SMC using PSL

Rules to incorporate TMC

w_3: TMC(q, p1) ⟶ mismatch(q, p1)
w_4: ¬TMC(q, p1) ⟶ ¬mismatch(q, p1)

SMC model

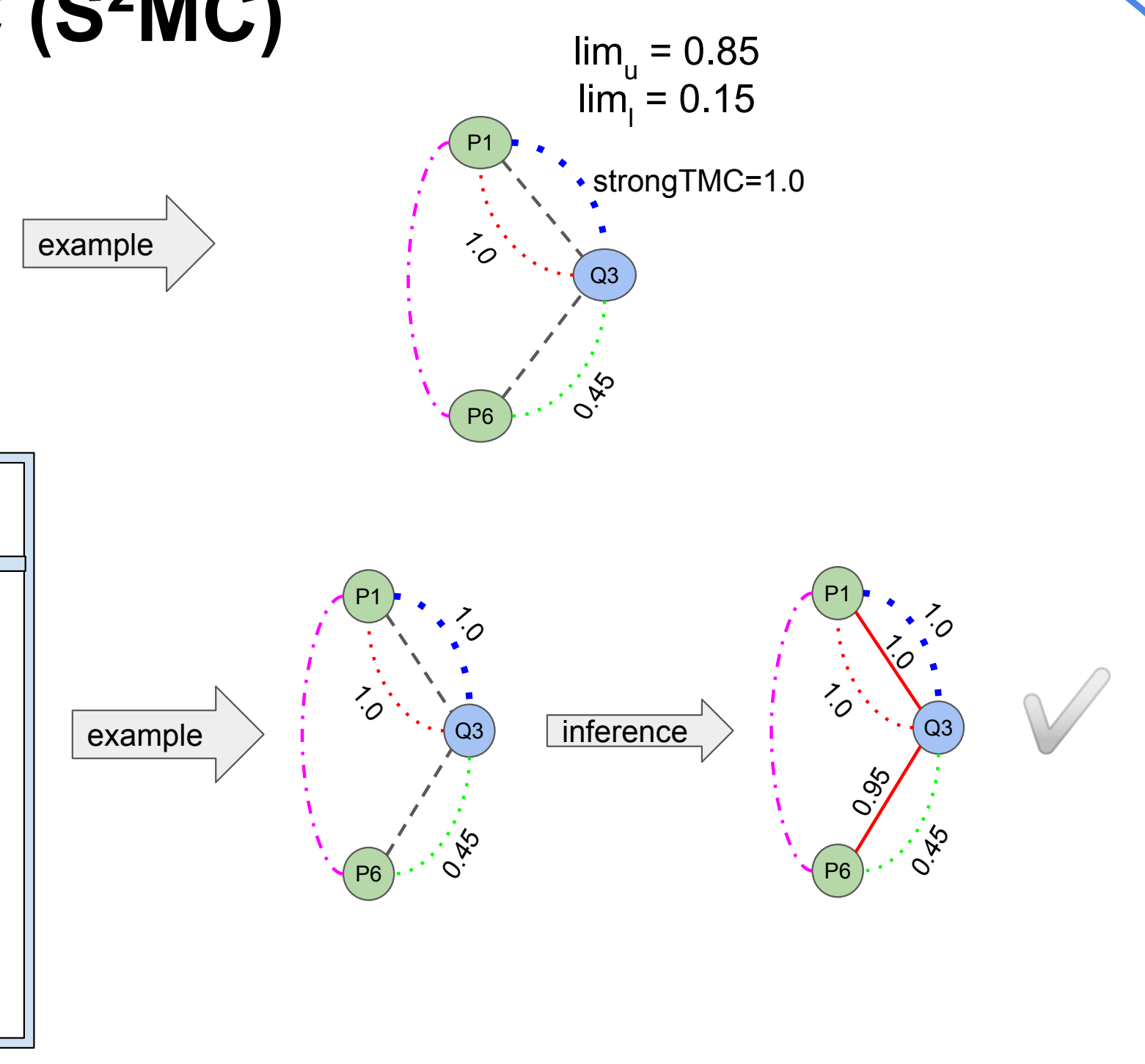
w_1: mismatch(q, p1) ∧ similar(p1, p2) ⟶ mismatch(q, p2)
w_2: ¬mismatch(q, p1) ∧ similar(p1, p2) ⟶ ¬mismatch(q, p2)
w_3: TMC(q, p1) ⟶ mismatch(q, p1)
w_4: ¬TMC(q, p1) ⟶ ¬mismatch(q, p1)

Problem of smoothing with SMC

Strong SMC (S^2MC)

Introduce new edge
strongTMC = TMC iff TMC ≥ lim_u or TMC ≤ lim_l

User-defined threshold

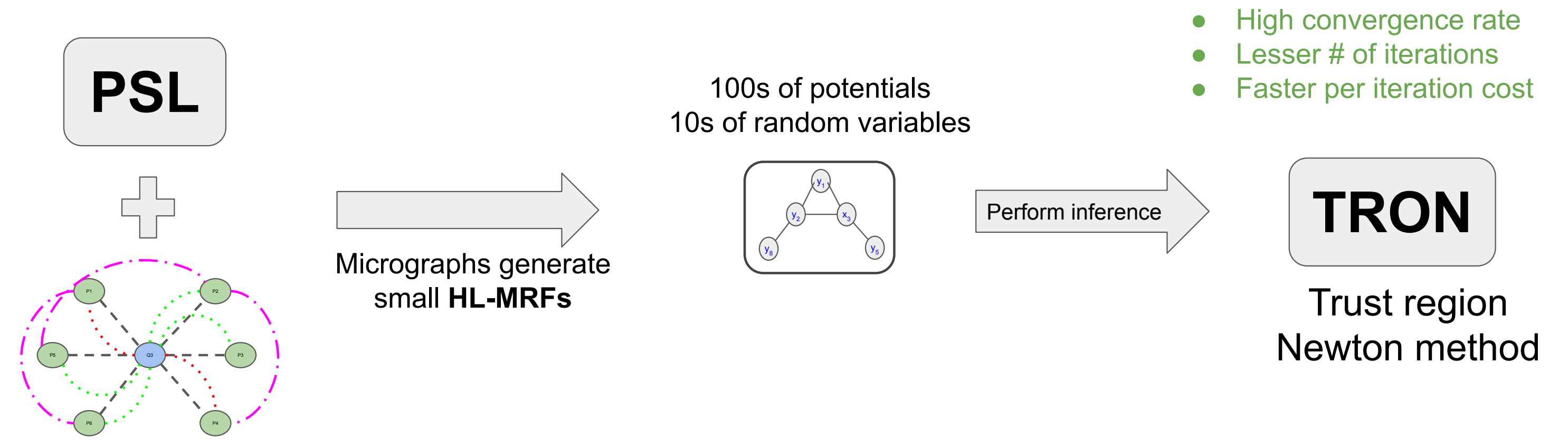


Full S^2MC model

w3: TMC(q, p1) ⟶ mismatch(q, p1)
w4: ¬TMC(q, p1) ⟶ ¬mismatch(q, p1)
w_6: strongTMC(q, p1) ∧ similar(p1, p2) ⟶ strongTMC(q, p2)
w_7: ¬strongTMC(q, p1) ∧ similar(p1, p2) ⟶ ¬strongTMC(q, p2)
w_8: strongTMC(q, p1) ⟶ mismatch(q, p1)
w_9: ¬strongTMC(q, p1) ⟶ ¬mismatch(q, p1)

Speedup S^2MC Using TRON in PSL

S^2MC not large scale problem



Fan et. al. JMLR 2008
https://www.csie.ntu.edu.tw/~cjlin/liblinear/

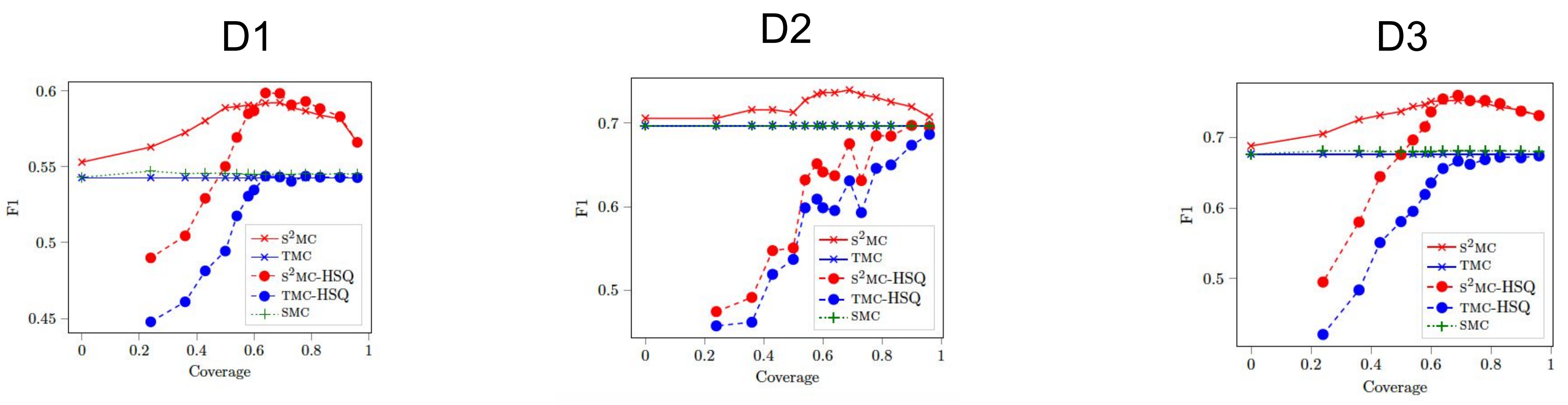
Empirical Evaluation

- We use three anonymized dataset from product search with query-product pair
- Take top 8 products for every query
- Task: identify **product type mismatch**
- ~200K labels generated by human annotators
- Use GBDT as TMC trained using labeled data
- Use threshold *t = 0.15* for classification
- Product similarity computed with title using word2vec
- Vary lim_u ∈ [0.15, 1] and lim_l ∈ [0, 0.15]
- High scoring queries **(HSQ)**: Queries with at least one product with strongTMC and one without
- Coverage = #HSQ/total

Dataset	Queries	Products
D1	1194	7790
D2	149	866
D3	591	1959

lim_L	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10	0.11	0.12	0.13	0.14	0.15
lim_U	0.94	0.88	0.82	0.76	0.70	0.64	0.58	0.52	0.47	0.40	0.35	0.30	0.25	0.20	0.15
Coverage	0.96	0.90	0.83	0.78	0.73	0.69	0.64	0.60	0.58	0.54	0.50	0.43	0.36	0.24	0.00

Performance



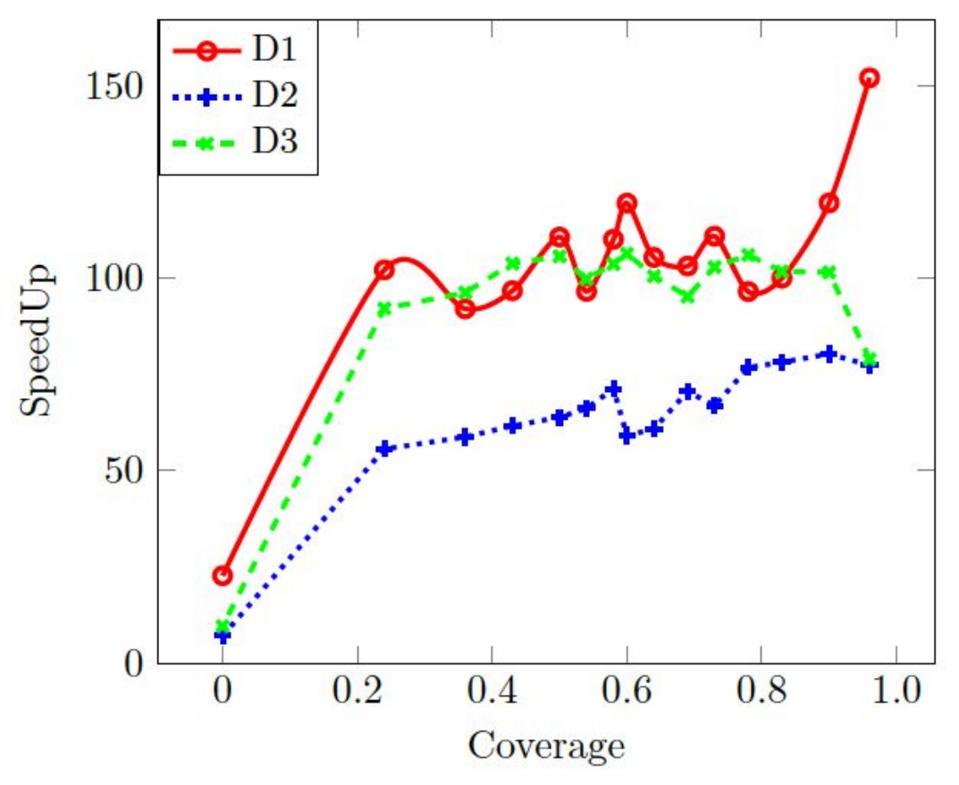
Ideal performance obtained using lim_l= 0.7 and lim_u= 0.58 and coverage = 64%

Speedup using TRON

- Time taken per-query to perform S^2MC using ADMM ~20ms
- Time taken per-query to perform S^2MC using TRON <1ms

SpeedUp = Time using ADMM / Time using TRON

Runtime computed for S^2MC using TRON from liblinear package and custom C++ implementation of ADMM for PSL



Conclusion

- Introduced improving search through mismatch classification
- Show relational structure improves traditional approaches
- Introduced micrographs to perform efficient classification at runtime
- Using PSL how micrographs can be incorporated effectively
- How TRON can be used to further speed up inference
- Empirical results on real datasets to show how micrographs improve MC

Future work

- How can we incorporate full relational graph?
- Include contextual information to determine important facets?