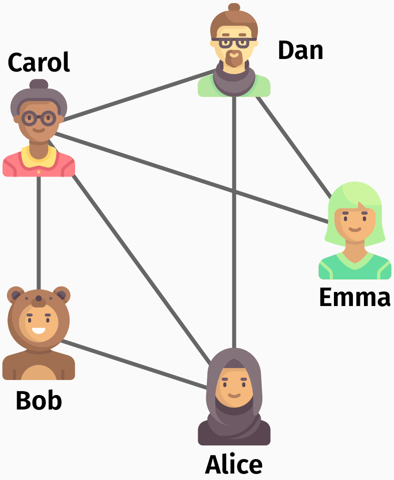
On the Parameterized Complexity of Learning First-Order Logic

Steffen van Bergerem[1] Martin Grohe[1] Martin Ritzert[2]

PODS 2022

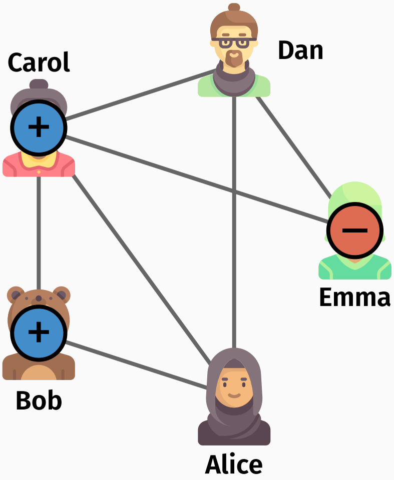
[1]RWTH Aachen University
[2]Aarhus University

Learning from Examples on Relational Structures



Icons made by Freepik from Flaticon

Learning from Examples on Relational Structures

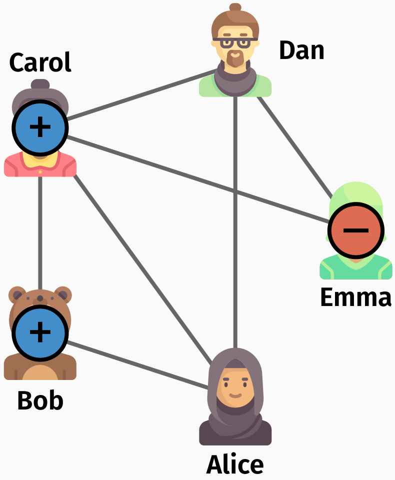


1

Positive examples
Bob
Carol

Negative examples
Emma

Learning from Examples on Relational Structures



1

Positive examples

Bob

Carol

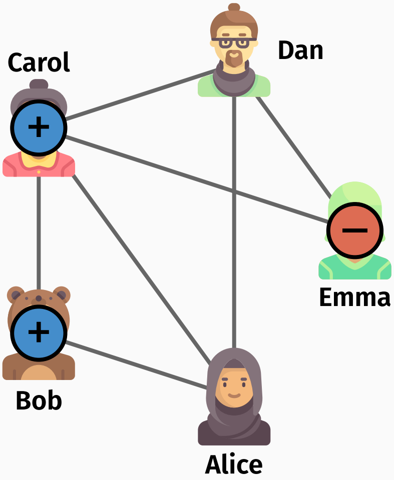
Negative examples

Emma

Possible hypothesis

Alice’s friends

Learning from Examples on Relational Structures



1

Positive examples

Bob

Carol

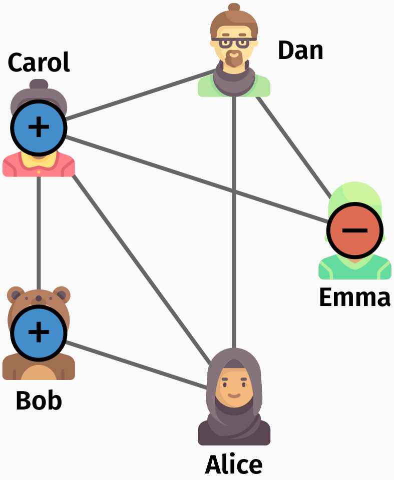
Negative examples

Emma

Possible hypothesis

Alice's friends

$$\varphi(x) = F(x, Alice)$$



1

Positive examples

Bob

Carol

Negative examples

Emma

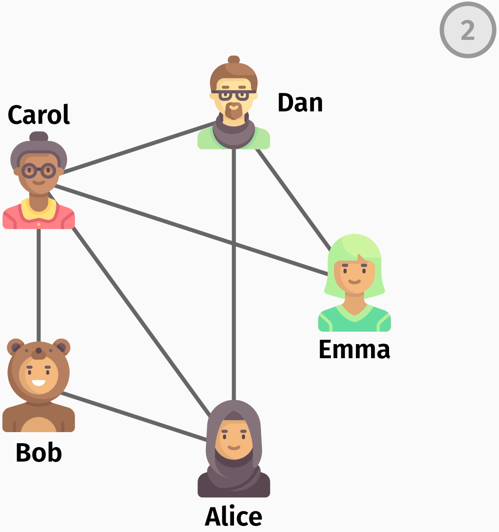
Possible hypothesis

Alice’s friends

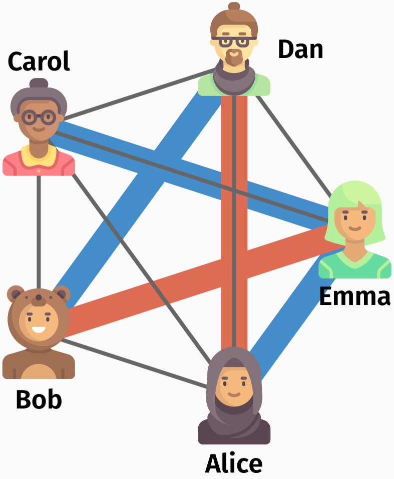
$\varphi(x; y) = F(x, y)$

parameter: $w =$ Alice

Learning from Examples on Relational Structures



Learning from Examples on Relational Structures



2

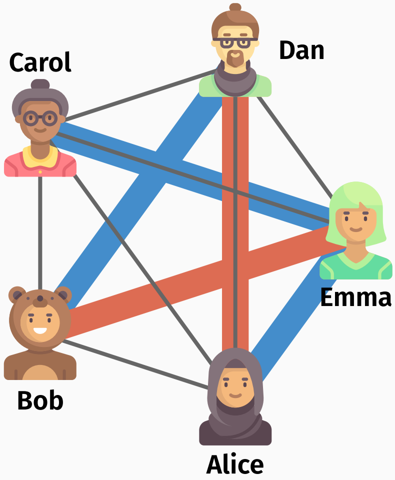
Positive examples

(Alice, Emma)
(Bob, Dan)
(Carol, Emma)

Negative examples

(Alice, Dan)
(Bob, Emma)

Learning from Examples on Relational Structures



2

Positive examples

(Alice, Emma)
(Bob, Dan)
(Carol, Emma)

Negative examples

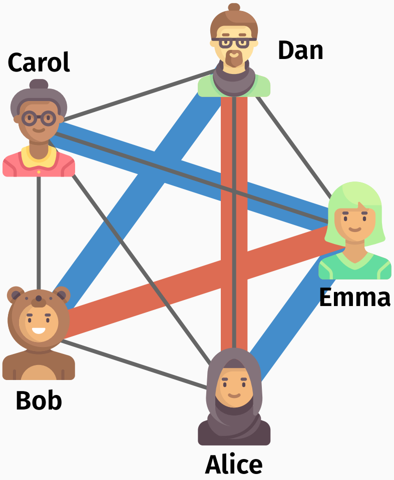
(Alice, Dan)
(Bob, Emma)

Possible hypothesis

having a common friend who is not Carol

Icons made by Freepik from Flaticon

Learning from Examples on Relational Structures



2

Positive examples

(Alice, Emma)
(Bob, Dan)
(Carol, Emma)

Negative examples

(Alice, Dan)
(Bob, Emma)

Possible hypothesis

having a common friend who is not Carol

$$\varphi(x_1, x_2; y) = \exists z \left(F(x_1, z) \wedge F(x_2, z) \wedge z \neq y\right)$$

parameter: $w =$ Carol

Icons made by Freepik from Flaticon

Main result:

In general, learning first-order logic is hard

Main result:

In general, learning first-order logic is hard, but there are a lot of tractable cases.

Learning Problem

Input

- $k, \ell, q \in \mathbb{N}$
- graph G
- labelled examples $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$

Output

- FO-query $\varphi(x_1, \dots, x_k; y_1, \dots, y_\ell)$ with $\mathrm{qr}(\varphi) \leqslant q$
- parameters $w_1, \dots, w_\ell \in V(G)$

Learning Problem

Input

- $k, \ell, q \in \mathbb{N}$
- graph G
- labelled examples $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$

Output

- FO-query $\varphi(x_1, \dots, x_k; y_1, \dots, y_\ell)$ with $\mathrm{qr}(\varphi) \leqslant q$
- parameters $w_1, \dots, w_\ell \in V(G)$

PAC learning / hypotheses that generalise well?

Fundamental Theorem of Statistical Learning Theory

We can perform **PAC learning**
if and only if we can solve the
Empirical Risk Minimisation Problem.

Fundamental Theorem of Statistical Learning Theory

We can perform **PAC learning**
if and only if we can solve the
Empirical Risk Minimisation Problem.

PAC learning: find hypotheses that generalise well

Empirical Risk Minimisation: (approximately) minimise the error we make on the training examples

Learning Problem

Input

- $k, \ell, q \in \mathbb{N}$
- graph G
- labelled examples $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$

Output

- FO-query $\varphi(x_1, \dots, x_k; y_1, \dots, y_\ell)$ with $\mathrm{qr}(\varphi) \leqslant q$
- parameters $w_1, \dots, w_\ell \in V(G)$

PAC learning / hypotheses that generalise well?

Empirical Risk Minimisation!

Empirical Risk Minimisation Problem

Input

- $k, \ell, q \in \mathbb{N}, \quad \varepsilon > 0$
- graph G
- labelled examples $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$

Output

- FO-query $\varphi(x_1, \dots, x_k; y_1, \dots, y_\ell)$ with $\mathrm{qr}(\varphi) \leqslant q$
- parameters $w_1, \dots, w_\ell \in V(G)$

such that $\mathrm{err}_T(\varphi, \bar{w}) \leqslant \varepsilon^* + \varepsilon$

PAC learning / hypotheses that generalise well?

Empirical Risk Minimisation!

Theorem (Grohe and Ritzert, 2017)

Concepts definable in FO on structures of small degree can be learned in sublinear time (in the size of the structure).

Learning Concepts Definable in Logics

Theorem (Grohe and Ritzert, 2017)

Concepts definable in FO on structures of small degree can be learned in sublinear time (in the size of the structure).

- unary MSO-queries on strings (Grohe, Löding, and Ritzert, 2017)
- unary MSO-queries on trees (Grienenberger and Ritzert, 2017)

Learning Concepts Definable in Logics

Theorem (Grohe and Ritzert, 2017)

Concepts definable in FO on structures of small degree can be learned in sublinear time (in the size of the structure).

- unary MSO-queries on strings (Grohe, Löding, and Ritzert, 2017)
- unary MSO-queries on trees (Grienenberger and Ritzert, 2017)
- FO with counting (v. B., 2019)
- FO with weight aggregation (v. B. and Schweikardt, 2021)

Beyond Structures of Small Degree

Theorem (Grohe and Ritzert, 2017)

Concepts definable in FO on structures of small degree can be learned in sublinear time (in the size of the structure).

Beyond structures of small degree / sublinear time:

Beyond Structures of Small Degree

Theorem (Grohe and Ritzert, 2017)

Concepts definable in FO on structures of small degree can be learned in sublinear time (in the size of the structure).

Beyond structures of small degree / sublinear time:

- k, ℓ, q are considered fixed

Beyond Structures of Small Degree

Theorem (Grohe and Ritzert, 2017)

Concepts definable in FO on structures of small degree can be learned in sublinear time (in the size of the structure).

Beyond structures of small degree / sublinear time:

- k, ℓ, q are considered fixed
- Algorithms running in time $|V(G)|^{\ell}$ would still be polynomial

Parameterized Complexity of Learning

Empirical Risk Minimisation Problem

Input

- $k, \ell, q \in \mathbb{N}, \quad \varepsilon > 0$
- graph G
- labelled examples $T = \big((\bar{v}_1, \lambda_1), \ldots, (\bar{v}_m, \lambda_m)\big)$

Output

- FO-query $\varphi(x_1, \ldots, x_k; y_1, \ldots, y_\ell)$ with $\mathrm{qr}(\varphi) \leqslant q$
- parameters $w_1, \ldots, w_\ell \in V(G)$

such that $\mathrm{err}_T(\varphi, \bar{w}) \leqslant \varepsilon^* + \varepsilon$

Empirical Risk Minimisation Problem

Input

- $k, \ell, q \in \mathbb{N}, \quad \varepsilon > 0$
- graph G
- labelled examples $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$

Parameter

$$k + \ell + q + 1/\varepsilon$$

Output

- FO-query $\varphi(x_1, \dots, x_k; y_1, \dots, y_\ell)$ with $\mathrm{qr}(\varphi) \leqslant q$
- parameters $w_1, \dots, w_\ell \in V(G)$

such that $\mathrm{err}_T(\varphi, \bar{w}) \leqslant \varepsilon^* + \varepsilon$

Fixed-Parameter Tractable (FPT)

running in time $f(k + \ell + q + 1/\varepsilon) \cdot |V(G)|^c$

Fixed-Parameter Tractable (FPT)

running in time $f(k + \ell + q + 1/\varepsilon) \cdot |V(G)|^c$

XP

running in time $|V(G)|^{f(k+\ell+q+1/\varepsilon)}$

In general, learning first-order logic is hard

Hardness of Learning

Theorem

Learning concepts definable in FO is hard for the parameterized complexity class $\mathsf{AW}[*]$ *under parameterized Turing reductions.*

Under the assumption $\mathsf{FPT} \neq \mathsf{W}[1]$:

$\implies$ **Learning FO is not fixed-parameter tractable.**

Hardness of Learning

Theorem

Learning concepts definable in FO is hard for the parameterized complexity class $\mathsf{AW}[*]$ *under parameterized Turing reductions.*

Under the assumption $\mathsf{FPT} \neq \mathsf{W}[1]$:

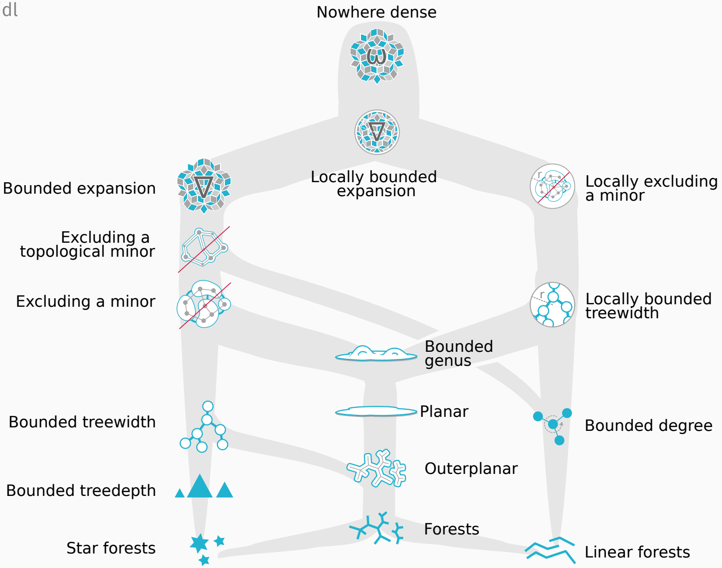
$\implies$ **Learning FO is not fixed-parameter tractable.**

Proof idea

Reduction from model checking

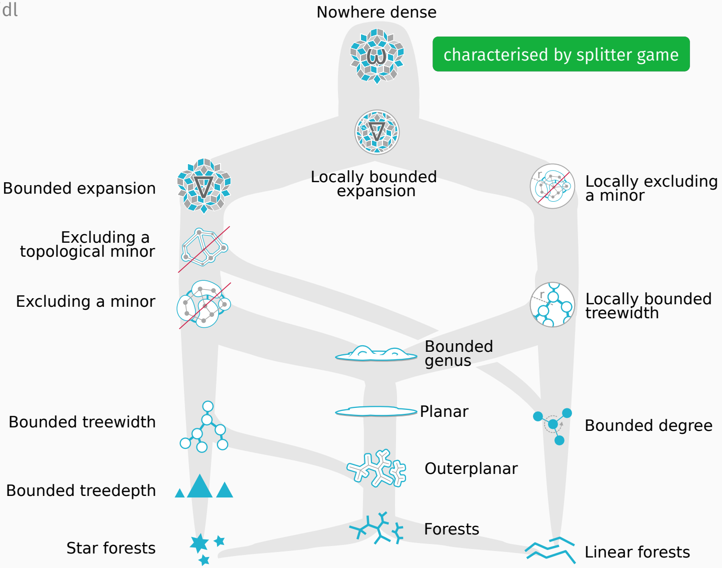
Classes with a Tractable Model-Checking Problem

Illustration by Felix Reidl



Classes with a Tractable Model-Checking Problem

Illustration by Felix Reidl



Learning first-order logic
on nowhere dense classes is tractable

Tractability of Learning

Theorem

For every nowhere dense class $\mathcal{C}$ of graphs, learning concepts definable in FO is fixed-parameter tractable on $\mathcal{C}$.

Tractability of Learning

Theorem

For every nowhere dense class $\mathcal{C}$ of graphs, learning concepts definable in FO is fixed-parameter tractable on $\mathcal{C}$.

Proof idea

- find hypothesis by trying all combinations of formulas and parameter tuples

Tractability of Learning

Theorem

For every nowhere dense class $\mathcal{C}$ of graphs, learning concepts definable in FO is fixed-parameter tractable on $\mathcal{C}$.

Proof idea

- find hypothesis by trying all combinations of formulas and parameter tuples
- number of formulas only depends on k, ℓ, q

Tractability of Learning

Theorem

For every nowhere dense class $\mathcal{C}$ of graphs, learning concepts definable in FO is fixed-parameter tractable on $\mathcal{C}$.

Proof idea

- find hypothesis by trying all combinations of formulas and parameter tuples
- number of formulas only depends on k, ℓ, q
- find small number of candidate parameter tuples to check

Tractability of Learning

Theorem

For every nowhere dense class $\mathcal{C}$ of graphs, learning concepts definable in FO is fixed-parameter tractable on $\mathcal{C}$.

Proof idea

- find hypothesis by trying all combinations of formulas and parameter tuples
- number of formulas only depends on k, ℓ, q
- find small number of candidate parameter tuples to check
  - heavily depends on the splitter game
  - we control one player
  - vertices chosen by the other player are good parameters

Conclusion

Conclusion

Theorem (Hardness)

Learning concepts definable in FO is hard for the parameterized complexity class $\mathsf{AW}[*]$ *under parameterized Turing reductions.*

Conclusion

Theorem (Hardness)

Learning concepts definable in FO is hard for the parameterized complexity class $\mathrm{AW}[*]$ *under parameterized Turing reductions.*

Theorem (Tractability)

For every nowhere dense class $\mathcal{C}$ *of graphs, learning concepts definable in FO is fixed-parameter tractable on* $\mathcal{C}$.

Conclusion

Theorem (Hardness)

Learning concepts definable in FO is hard for the parameterized complexity class $\mathsf{AW}[*]$ *under parameterized Turing reductions.*

Theorem (Tractability)

For every nowhere dense class $\mathcal{C}$ *of graphs, learning concepts definable in FO is fixed-parameter tractable on* $\mathcal{C}$.

Future Work

- Can we avoid increasing ℓ and q?

Conclusion

Theorem (Hardness)

Learning concepts definable in FO is hard for the parameterized complexity class $\mathrm{AW}[*]$ *under parameterized Turing reductions.*

Theorem (Tractability)

For every nowhere dense class $\mathcal{C}$ *of graphs, learning concepts definable in FO is fixed-parameter tractable on* $\mathcal{C}$.

Future Work

- Can we avoid increasing ℓ and q?
- Are learning and model checking equivalent?

Conclusion

Theorem (Hardness)

Learning concepts definable in FO is hard for the parameterized complexity class AW$[*]$ *under parameterized Turing reductions.*

Theorem (Tractability)

For every nowhere dense class $\mathcal{C}$ of graphs, learning concepts definable in FO is fixed-parameter tractable on $\mathcal{C}$.

Future Work

- Can we avoid increasing ℓ and q?
- Are learning and model checking equivalent?
- Extend results to richer logics.

Hardness of Learning — Reduction from Model Checking

Solve model checking using learning problem

- evaluate formula recursively
- negation and Boolean connectives are easy

Hardness of Learning — Reduction from Model Checking

Solve model checking using learning problem

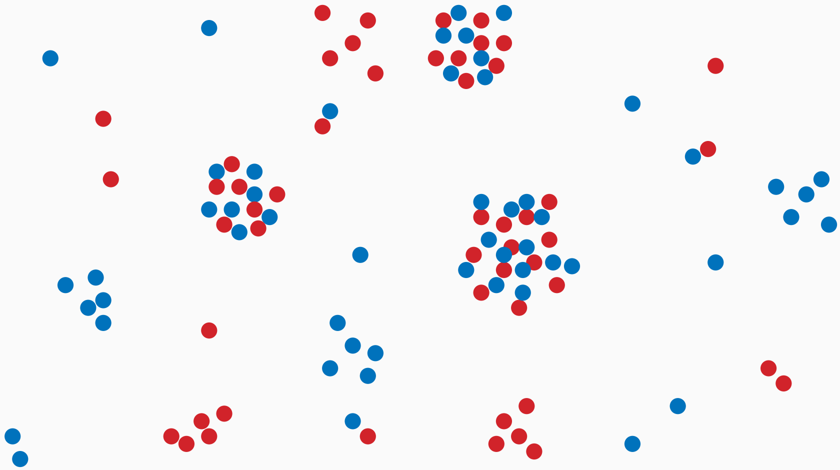
- evaluate formula recursively
- negation and Boolean connectives are easy
- use learning for existential quantification
- find small set of representatives that suffice to be checked

Hardness of Learning — Reduction from Model Checking

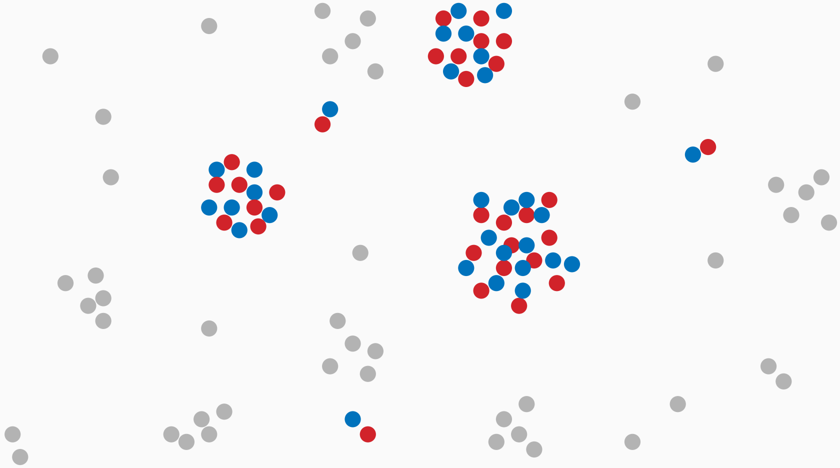
Solve model checking using learning problem

- evaluate formula recursively
- negation and Boolean connectives are easy
- use learning for existential quantification
- find small set of representatives that suffice to be checked
    - run learning algorithm for every pair of vertices, one as a positive and one as a negative example
    - use answers to find and remove vertices that are already represented
    - by Ramsey's Theorem, this works until the set is small

Tractability of Learning — Finding the Right Parameter Tuples



Tractability of Learning — Finding the Right Parameter Tuples



Tractability of Learning — Finding the Right Parameter Tuples