Finite model theory and logics for Machine learning

Grégoire Fournier

Logic seminar

March 3, 2025

Introduction

Lemma

Let $\mathcal{A}_m$ and $\mathcal{A}_n$ be two linear orders of size $n > m > 0$. $\mathcal{A}_m$ and $\mathcal{A}_n$ cannot be distinguished by an $\mathcal{C}_3^{(k)}$ sentence of size less than $\frac{\sqrt{m}}{k+1}$.

Topics: Model theory, complexity, logics, games, lower bounds.

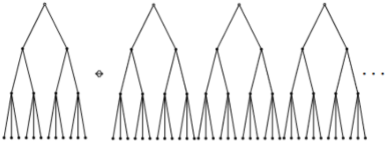

Figure 1: Left k-supplementing move on a pair of tree structures

Complexity theory background

Definition (Recursively enumerable set)

A subset L of Σ^* is recursively enumerable if there is a Turing machine that accepts it.

Definition (Recursive set)

A subset L of Σ^* is recursive if there is a Turing machine that decides it.

Complexity theory background

Definition (Polynomial time)

A subset L of Σ^* is in NTIME(f) if there is a Turing machine that accepts any $s \in L$ using at most $f(|s|)$ steps, and it is written $L \in$ DTIME(f).
The polynomial time computable problems PTIME is:

$$\mathrm{PTIME} = \bigcup_{c \in \mathbb{N}} \mathrm{DTIME}(n^c)$$

Model theory of finite structures

Theorem (Completeness)

ψ is a consequence of T iff ψ is formally provable from T.

For which we have the direct consequence:

Theorem

The set of logically valid sentences of first-order logic is recursively enumerable. (true in all structures, under all assignments)

But if we consider only finite models, this fails:

Theorem (Trakhtenbrot)

The set of sentences of first-order logic valid in all finite structures is not recursively enumerable.

Examples

Example

- There is a theory T such that T has no finite models, and every finite subset of T has a finite model.

Proof.

Consider $\{\lambda_n := \exists x_1 \dots x_n \bigwedge_{i \neq j} \neg(x_i = x_j) : n \in \mathbb{N}\}$.

$\square$

- An inexpressibility proof: Assume that $\sigma = \emptyset$, then *EVEN* is not FO definable.

Proof.

Suppose it is by Φ.

$T_1 = \{\Phi\} \cup \{\lambda_k | k > 0\}$

$T_2 = \{\neg\Phi\} \cup \{\lambda_k | k > 0\}$

By LS theorem, T_1 and T_2 have a countable model, $\mathfrak{A}_1$ and $\mathfrak{A}_2$ that are isomorphic and satisfy $\mathfrak{A}_1 \Vdash \Phi$ and $\mathfrak{A}_2 \Vdash \neg\Phi$. $\square$

Inexpressibility proofs

Lemma

For every finite structure $\mathfrak{A}$, there is a sentence $\Phi_{\mathfrak{A}}$ such that $\mathfrak{B} \Vdash \Phi_{\mathfrak{A}}$ iff $\mathfrak{B} \cong \mathfrak{A}$, where $\cong$ denotes an isomorphism.

In particular, two finite structures that agree on all FO sentences are isomorphic.

Inexpressibility proofs

Suppose we want to prove that a property $\mathcal{P}$ is not expressible in the logic $\mathcal{L}$.
We may partition $\mathcal{L}$ into countable many classes, $\mathcal{L}[0]$,$\mathcal{L}[1]$, ...
We find two family of structures $(\mathfrak{A}[k])_k$ and $(\mathfrak{B}[k])_k$, such that:

- $\mathfrak{A}_k \Vdash \Phi$ iff $\mathfrak{A}_k \Vdash \Phi$ for every $\Phi \in \mathcal{L}[k]$
- $\mathfrak{A}_k$ has property $\mathcal{P}$ and $\mathfrak{B}_k$ does not.

Then $\mathcal{P}$ is not expressible in the logic $\mathcal{L}$.

EF games

Definition (EF games)

The games is played on two relational structures $\mathfrak{A}$ and $\mathfrak{B}$, and it has two player, a spoiler and a duplicator. It goes as follows:

- For n rounds:
  - ⋆ The spoiler makes a move by picking an element of $\mathfrak{A}$ or $\mathfrak{B}$.
  - ⋆ The duplicator responds by picking an element in the other structure.
- The n-rounds game ends in the position $\vec{a} = (a_1, \ldots, a_n)$, $\vec{b} = (b_1, \ldots, b_n)$. Duplicator wins if:

  $((\vec{a}, \vec{c}^{\mathfrak{A}}), (\vec{b}, \vec{c}^{\mathfrak{B}}))$ is a partial isomorphism btw. $\mathfrak{A}$ and $\mathfrak{B}$.

EF games

Theorem

EF Denoting $FO[k]$ for all FO formula sof quantifier rank up to k. Let $\mathfrak{A}$, $\mathfrak{B}$ be two structures on a relational vocabulary, then tfae:

- $\mathfrak{A}$ *and* $\mathfrak{B}$ *agree on* $FO[k]$.
- $\mathfrak{A} \equiv_k \mathfrak{B}$ *(duplicator has a winning strategy in the k-round EF game).*

What is a winning strategy for an EF game?

An application of EF games

Theorem

Let $k > 0$, L_1, L_2 be linear orders of length at least 2^k, then $L_1 \equiv_k L_2$.

The proof is by induction.

An application of EF games

Proof.

We prove that $L_1 \equiv_k L_2$ in the expanded vocabulary with the constants *min* and *max*. Assume $L_1 = [n]$ and $L_2 = [m]$m with $n, m \geq 2^k + 1$. After round i, the moves are denoted $\vec{a} = (a_{-1}, a_0, a_1, \ldots, a_i)$ such that $(a_{-1}, a_0) = (min^{L_1}, max_1^{L})$, similarly for $\vec{b}$.

We claim that duplicator can maintain after move i, for $-1 \leq j, l \leq i$:

- If $d(a_j, a_l) < 2^{k-i}$, then $d(b_j, b_l) = d(a_j, a_l)$.
- If $d(a_j, a_l) \geq 2^{k-i}$, then $d(b_j, b_l) \geq 2^{k-i}$.
- $a_j \leq a_l$ iff $b_j \leq b_l$.

$\square$

Applications of EF games

EF games have been applied to study different kind of logics under different types of partitions, such as:

- Pebble games to describe $\mathcal{L}_k$, the set of FO formulas such that the quantified variables are a subset of $x_1, \ldots, x_k$, or $\mathcal{C}_k$.
- EF game for formula size, in logics such as $\mathrm{FO}^2(\mathrm{TC})$, propositional FO logic, predicate FO logic.
- EF game for formula size, on FO, C with bounded counting rank, and applications to linear orders.

Guarded Counting Logic

We interpret formulas over labelled graphs; variables range over the vertices.

$\mathcal{C}_k$ denotes the fragment of $\mathcal{C}$ consisting of all formulas with at most k variables, and by $\mathcal{C}^{(q)}$ denotes the fragment consisting of all formulas of quantifier rank at most q. Combine the two, $\mathcal{C}_k^{(q)} = \mathcal{C}_k \cap \mathcal{C}^{(q)}$.

Example

For every ℓ, a $C_3^{(\ell)}$ formula stating that the diameter of a graph is at most 2ℓ is:

$$\delta_{2n}(x, y) = \exists z \, (\delta_n(x, z) \wedge \delta_n(z, y))$$

Guarded Counting Logic

The guarded fragment $\mathcal{GC}$ restricts quantifiers to range over the neighbours of the current nodes.
$\mathcal{GC}_2$ is also known as graded modal logic .

Example

The following $\mathcal{GC}_2$-formula $\Phi(x)$ says that vertex x has at most 1 neighbour that has more than 10 neighbours with label P_1:

$$\Phi(x) := \neg \exists^{\geq 2} y \left(E(x, y) \wedge \exists^{\geq 11} x \left(E(y, x) \wedge P_1(x) \right) \right)$$

Invariants, colouring and the Weisfeiler Lehman algorithm

Definition (Graph invariant)

For $k \geq 1$, a k-ary graph invariant is a function ξ that associates with each graph G a function ξ(G) defined on $V(G)^k$ in such a way that for all graphs G, G', all isomorphisms f from G to G', and all tuples $v \in V(G)^k$ it holds that:

$$\xi(G)(v) = \xi(f(G))(f(v))$$

For ξ be a k-ary invariant. If $\xi(G, v) \neq \xi(G', v')$, then we say that ξ distinguishes (G, v) and (G', v').

Invariants, colouring and the Weisfeiler Lehman algorithm

Definition (Colouring algorithm (or 1-WL))

For every graph G, we define a sequence of vertex colourings $cr^{(t)}(G)$ as follows:

- For every $v \in V(G)$, let $cr^{(0)}(G, v) := col(G, v)$.

  $$cr^{(t+1)}(G, v) := (cr^{(t)}(G, v), \{\{cr^{(t)}(G, w) | w \in N(v)\}\})$$

- The stable colouring is denoted $cr^{(\infty)}(G)$.
- $cr^{(t)}$ is a vertex invariant.

In the same way we can define the k-dimensional WL algorithm ($wl_k^{(\infty)}$).

Invariants, colouring and the Weisfeiler Lehman algorithm

Theorem

For all graphs G, G' tfae:

- $wl_k^{(\infty)}$ *does not distinguish G and G'.*
- *G and G' satisfy the same $\mathcal{C}_{k+1}$-sentences.*

Theorem

Let $t \geq 0$. Then for all graphs G, G' and vertices $v \in V(G), v' \in V(G')$ tfae:

- $cr^{(t)}(G, v) = cr^{(t)}(G', v')$;
- *for all formulas $\phi(x) \in \mathcal{GC}_2^{(t)}$, $G \Vdash \phi(v)$ iff $G' \Vdash \phi(v')$.*

Graph neural networks

We want to learn graph invariant function. Graph neural networks (GNNs) are neural network architectures that guarantee invariance by their design.

The combination function *comb* is learned, and represented by a neural network. A layer of a GNN acts on a graph, where each vertex v is assigned a state, $\zeta(v)$. It produces an output $\eta(v)$ on each vertex v, given by the equation:

$$\eta(v) := comb(\zeta(v), \sum_{w \in N_G(v)} \zeta(w))$$

Graph neural networks

We say that a GNN expresses a unary query if it approximates the corresponding vertex invariant.

For ξ a vertex invariant computed by a GNN, we say that the GNN expresses Q if there is an $\varepsilon < 1/2$ such that for all graphs G and vertices $v \in V(G)$:

- $\xi(G, v) \geq 1 - \varepsilon$ if $v \in Q(G)$,
- $\xi(G, v) \leq \varepsilon$ if $v \notin Q(G)$.

Graph neural networks

Theorem

Let Q be a unary query expressible in graded modal logic $\mathcal{GC}_2$. Then there is a GNN that expresses Q.

Theorem

Let Q be a unary query expressible by a GNN and also expressible in first-order logic. Then Q is expressible in $\mathcal{GC}_2$.

References

[1] Micah Adler and Neil Immerman. An n! lower bound on formula size. 2003.

[2] Pablo Barceló, Egor V. Kostylev, Mikael Monet, Jorge Pérez, Juan Reutter, and Juan Pablo Silva. The logical expressiveness of graph neural networks. 2020.

[3] Martin Grohe. The logic of graph neural networks. 2021.

[4] Lauri Hella. Logical hierarchies in ptime. 1996.

[5] Neil Immerman and Eric Lander. Describing graphs: A first-order approach to graph canonization. 1990.

[6] Leonid Libkin. *Elements of Finite Model Theory*. 2004.

[7] D. Marker. Math 502 model theory lecture notes, 2015.

[8] Martin Otto. Graded modal logic and counting bisimulation. 2019.