UiA University of Agder Norway

CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

Learning AND-OR hierarchies with Tsetlin machines

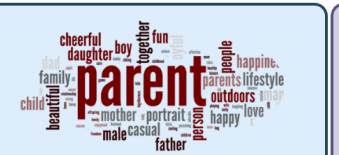
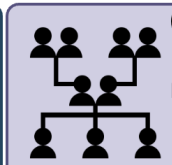
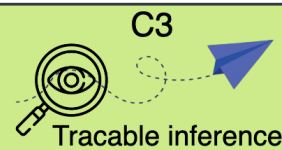
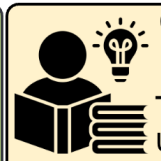
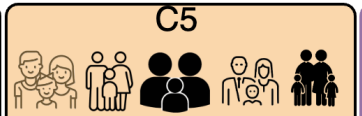
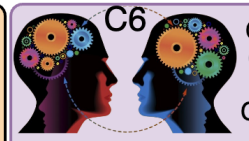
Ole-Christoffer Granmo

February 6, 2026

UiA University of Agder Norway

CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

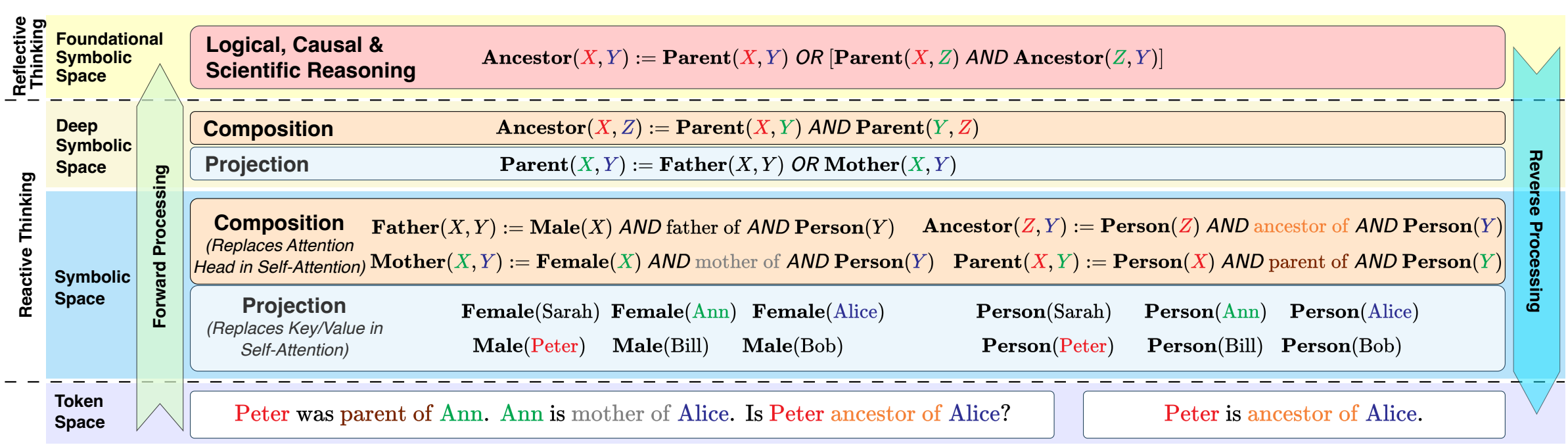
Bridging reactive and reflective thinking - An 8-year research agenda

 ***Token Space***	C1  Symbolic representation for reactive thinking ***Conceptualization***	C2  +  World knowledge ***Introspection***	C3  Tracable inference ***Transparency***	C4  +  New Knowledge Update on the fly ***Lifelong Learning***	C5  Go beyond the data ***Generalization***	C6  Combine C1 & C5 with C2 via C4 ***Reactive & Reflective Thinking***
Peter is parent of Ann. Ann is the mother of Alice.	Ancestor(Peter, Alice):= Parent(Peter, Ann); Mother(Ann, Alice)	Is Peter an ancestor of Paul? **Answer**: The answer cannot be derived from my current knowledge.	Why is Peter an ancestor of Alice? **Answer:** Because Peter is parent of Ann and Ann is parent of Alice, as documented.	Adding Ann is the mother of Paul, also adds Ancestor(Peter, Paul)	**If** X is parent of Y **or** (X is a parent of Z **and** Z is a ancestor of Y) **then** X is ancestor of Y.	**Reactive:** Is Peter an ancestor of Alice? **Reflective:** Must Paul and Alice always have the same ancestor?

UiA University of Agder Norway

CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

Bridging reactive and reflective thinking - An 8-year research agenda

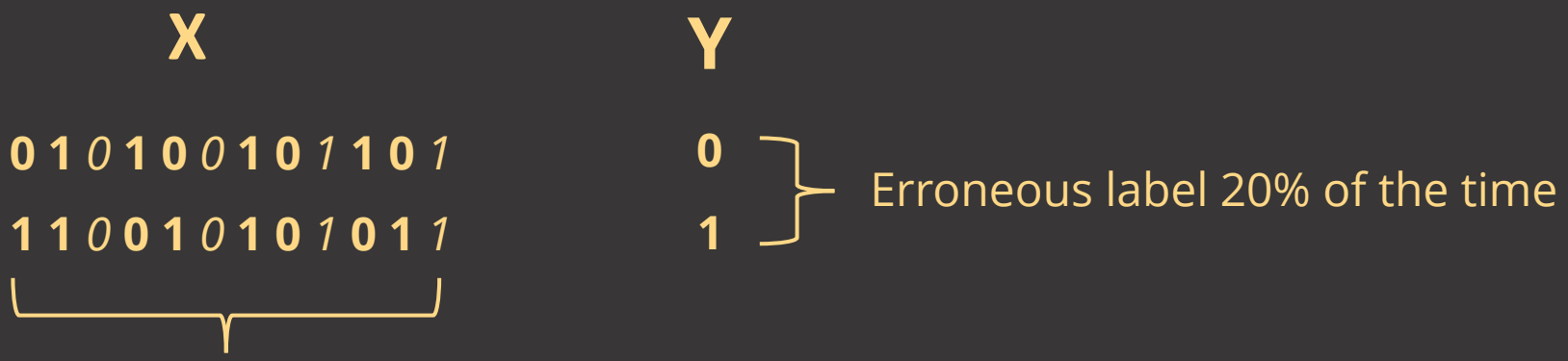


CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

The noise parity problem

Odd number of bits set → y = 1

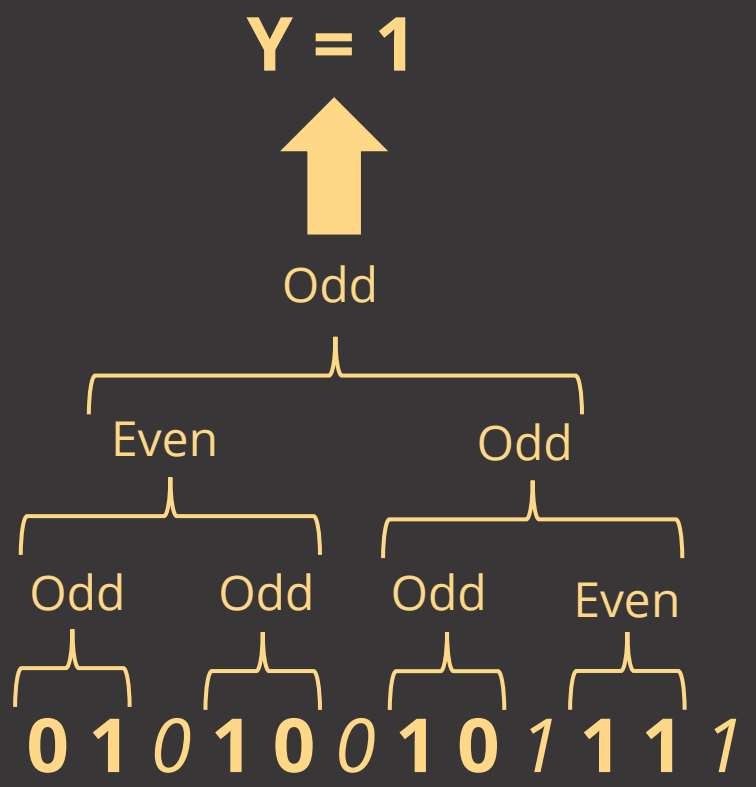
Even number of bits set → y = 0



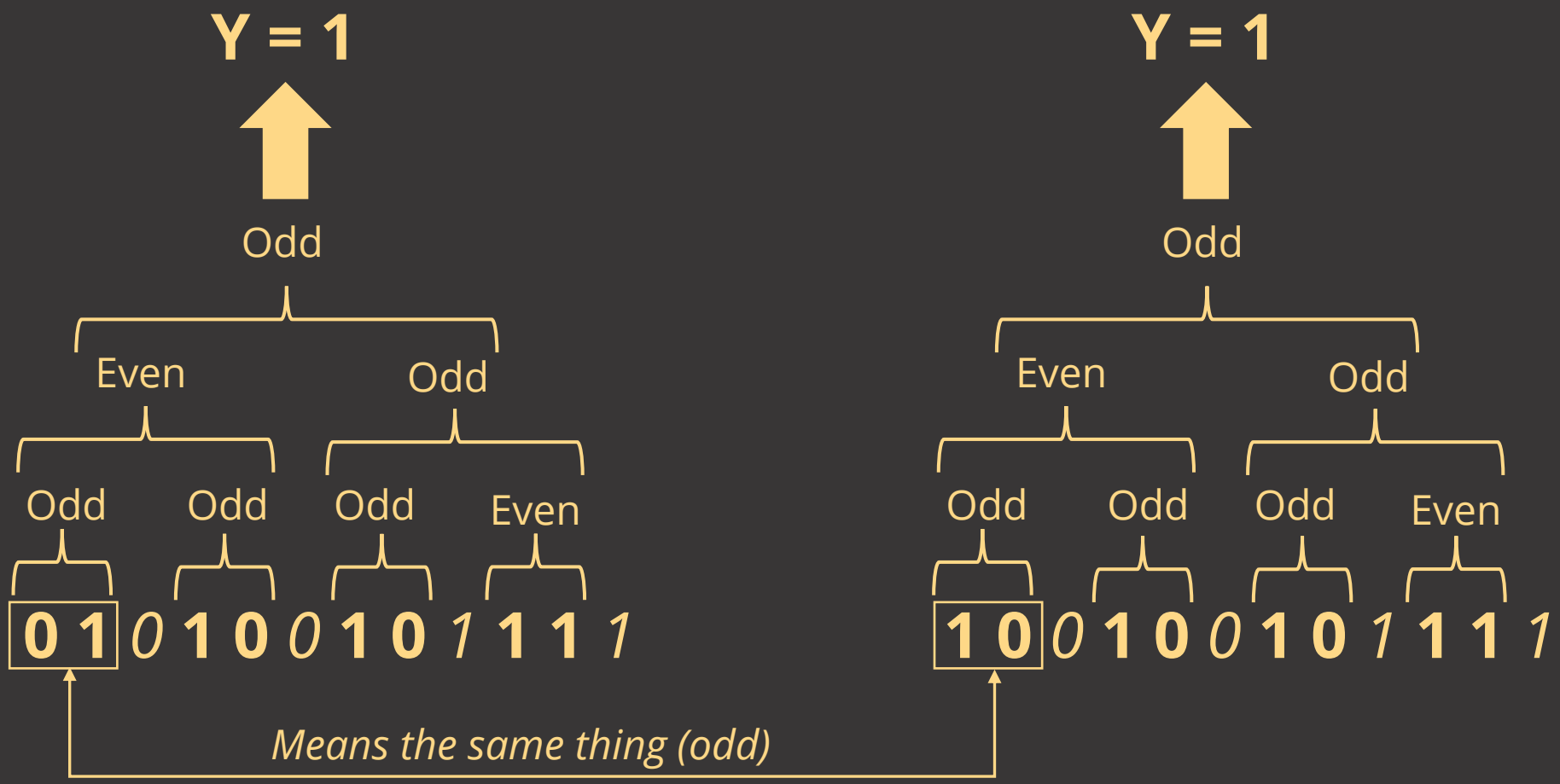
8 informative bits (in bold) → **256 unique patterns**

X	Y
0 1 *0* **1 0** *0* **1 0** *1* **? 0** *1*	**?**

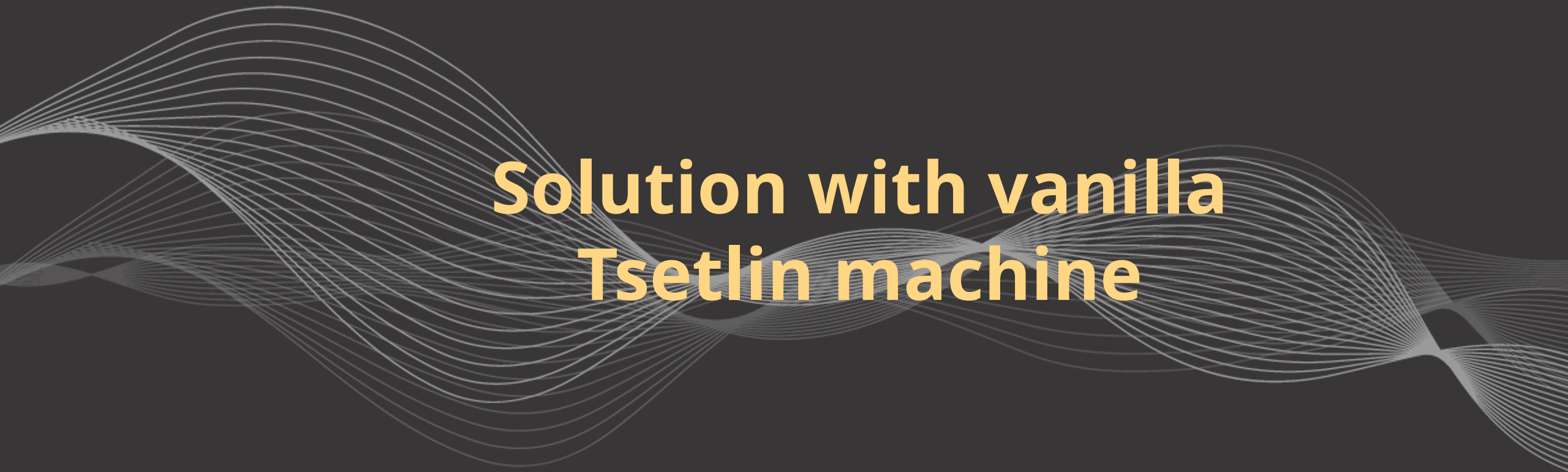
Non-linear: All information is lost if just a single informative bit is missing!



Captures stepwise abstraction (cf. Transformers)



Rewards generalization capability



Solution with vanilla Tsetlin machine

Clauses for Y = 1

$$C_1^+ = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge x_7 \wedge \neg x_8$$

$$C_2^+ = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6 \wedge \neg x_7 \wedge x_8$$

$$C_3^+ = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge \neg x_6 \wedge x_7 \wedge x_8$$

$$C_4^+ = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge \neg x_5 \wedge x_6 \wedge \neg x_7 \wedge x_8$$

$$\vdots$$

$$C_{128}^+ = x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_4 \wedge \neg x_5 \wedge \neg x_6 \wedge \neg x_7 \wedge \neg x_8$$

Target model for Y = 1

$$C_1^+ \vee C_2^+ \vee C_3^+ \vee C_4^+ \vee \cdots \vee C_{128}^+$$

Key learning aspect:
Replace **OR** with **Addition**
→ **Clause vote sum**

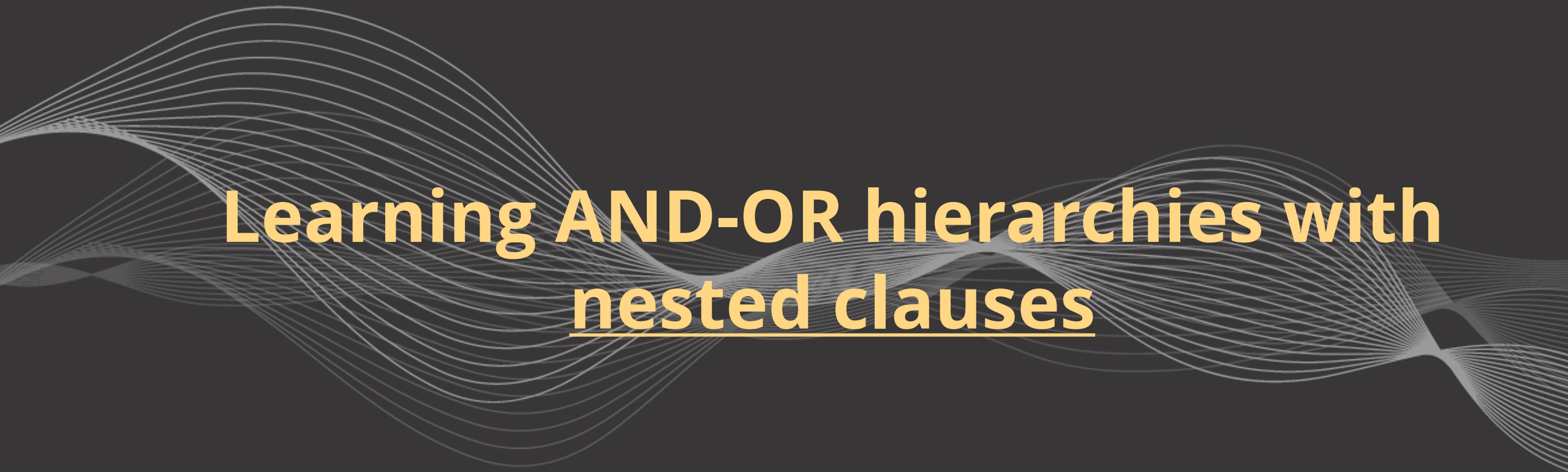
$$C_1^+ \vee C_2^+ \vee C_3^+ \vee C_4^+ \vee \cdots \vee C_{128}^+$$

$$C_1^+ + C_2^+ + C_3^+ + C_4^+ + \cdots + C_{128}^+$$

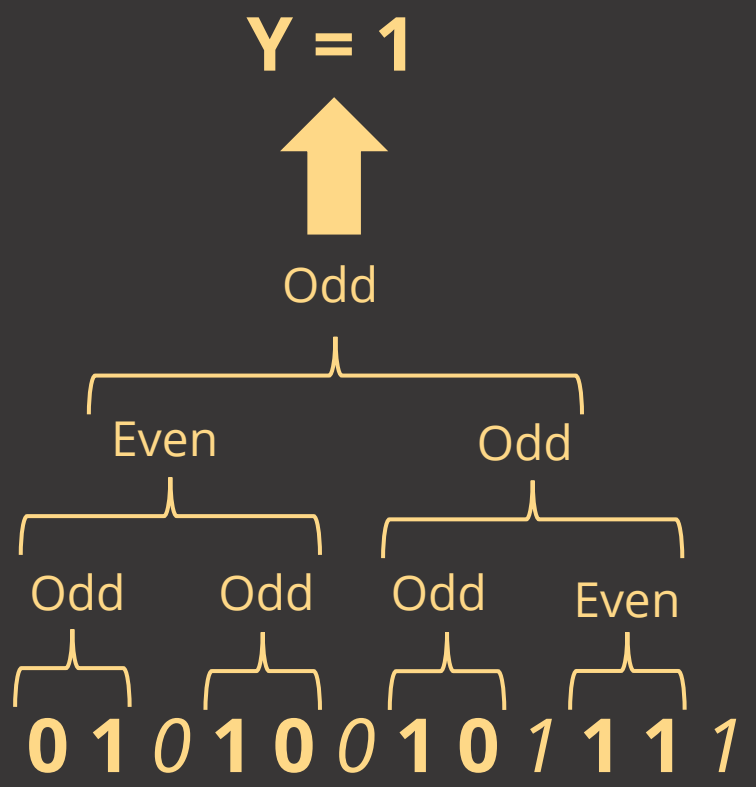
Clause vote sum favouring Y = 1

UiA University of Agder Norway

CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH



Learning AND-OR hierarchies with nested clauses

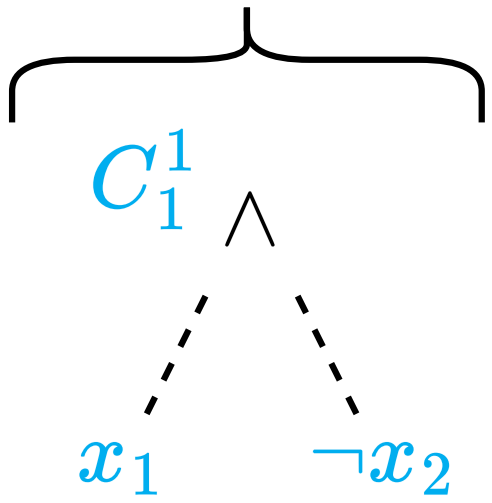


Captures stepwise abstraction (cf. Transformers)

Hierarchical clause structure

Leaves: clause component

Leaf

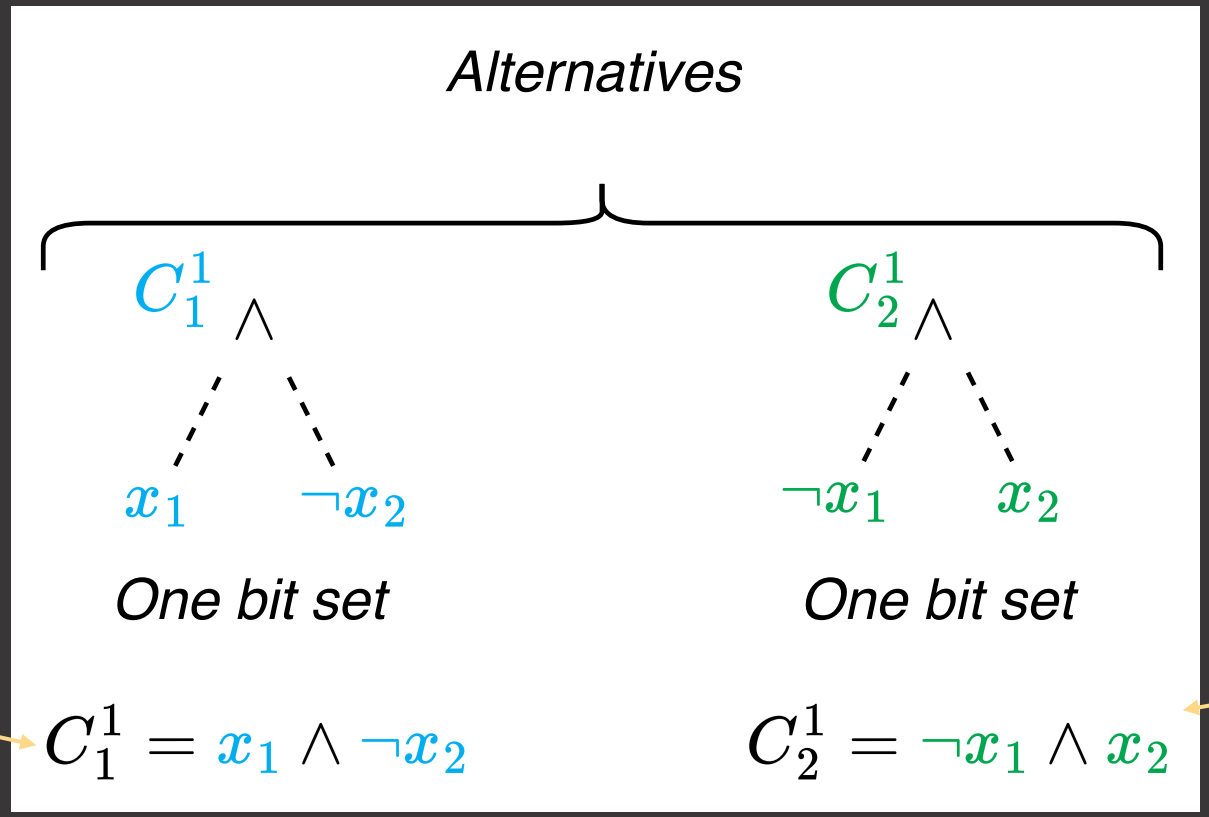


Feature group
(two features - hyper-parameter)

$$C_1^1 = x_1 \wedge \neg x_2$$

Clause Component

Alternatives

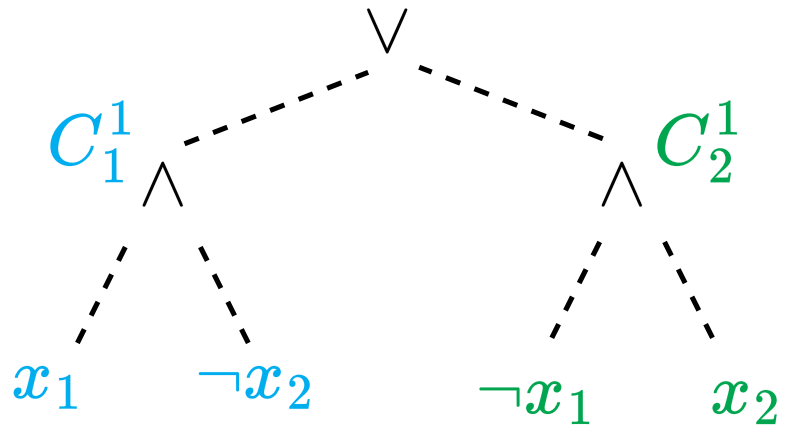


Clause component

Clause component

OR-structure for organizing alternatives

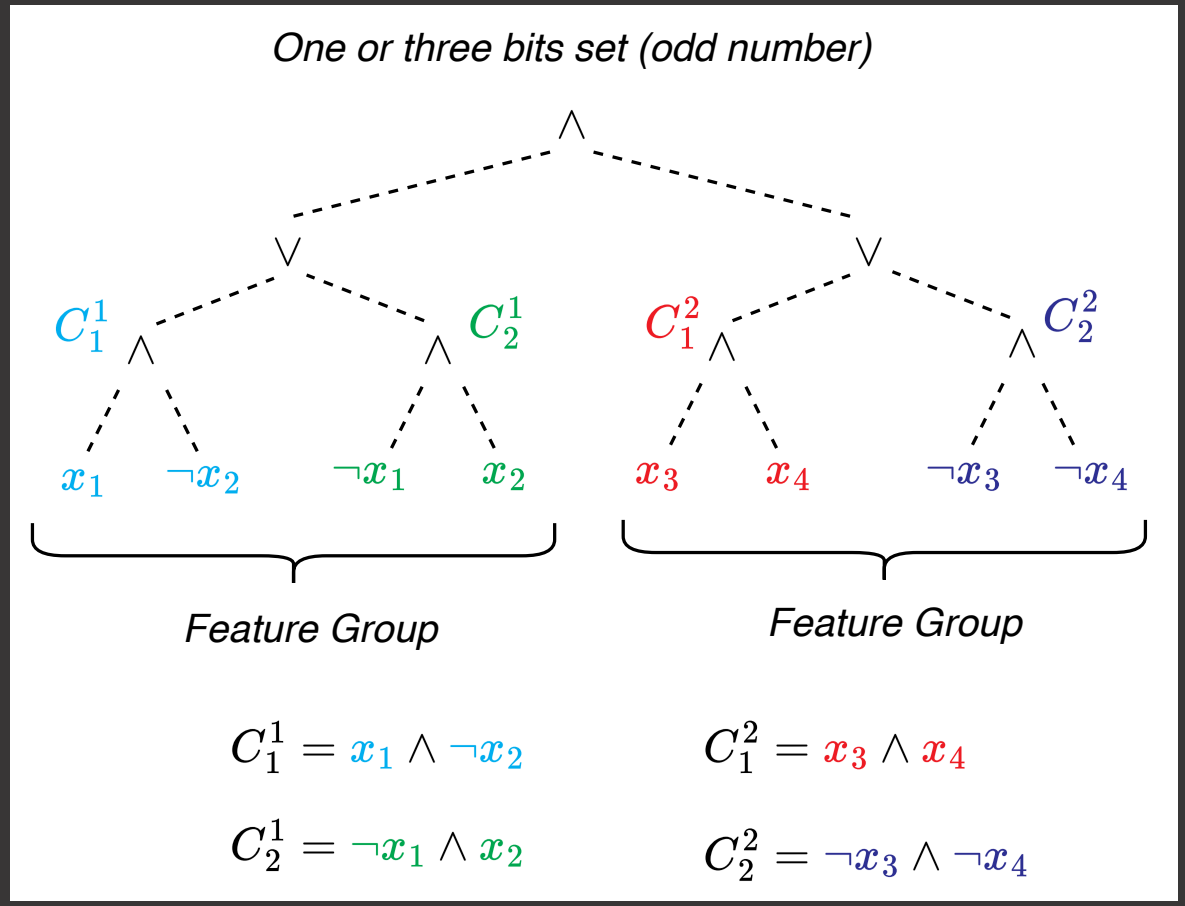
One bit set



$$C_1^1 = x_1 \wedge \neg x_2 \qquad C_2^1 = \neg x_1 \wedge x_2$$

$$\text{Clause} = (C_1^1 \vee C_2^1)$$

AND-structure for combining feature groups



$$\text{Clause} = (C_1^1 \vee C_2^1) \wedge (C_1^2 \vee C_2^2)$$

Inference

Inference is exactly as for vanilla Tsetlin machine, except for clause evaluation...

Clause evaluation

A clause provides multiple votes:

- Replace **OR** with **Addition**
- Replace **AND** with **Multiplication**!

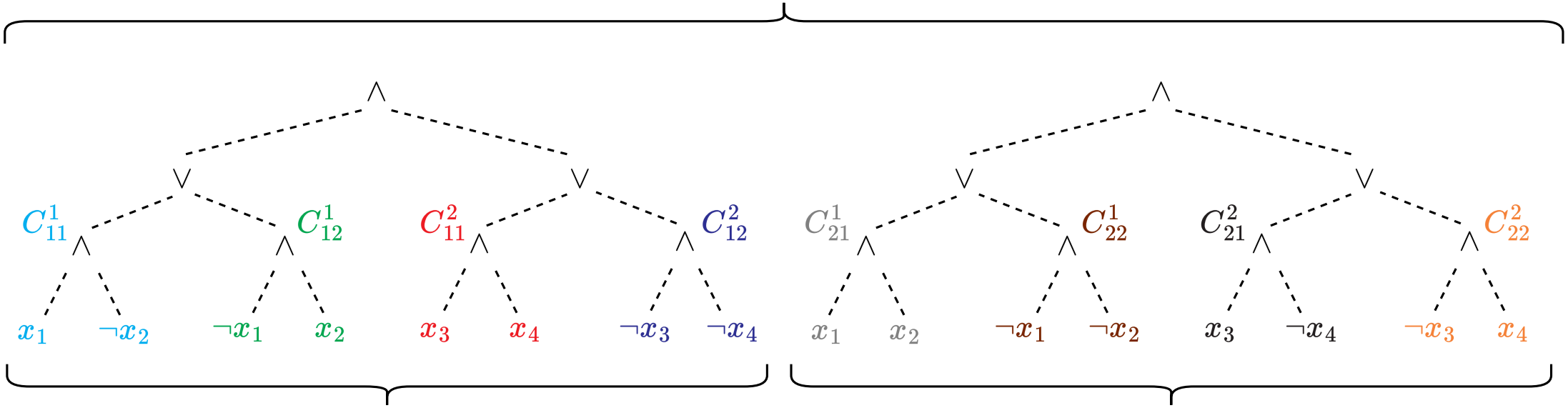
→ **Clause votes**

$$\text{Clause} = (C_1^1 \vee C_2^1) \wedge (C_1^2 \vee C_2^2)$$

$$\text{Clause Votes} = (C_1^1 + C_2^1) \times (C_1^2 + C_2^2)$$

Scaling up

Alternatives spanning larger groups of features



Two feature groups combined

Two feature groups combined

$$(C_{11}^1 \vee C_{12}^1) \wedge (C_{12}^2 \vee C_{12}^2)$$

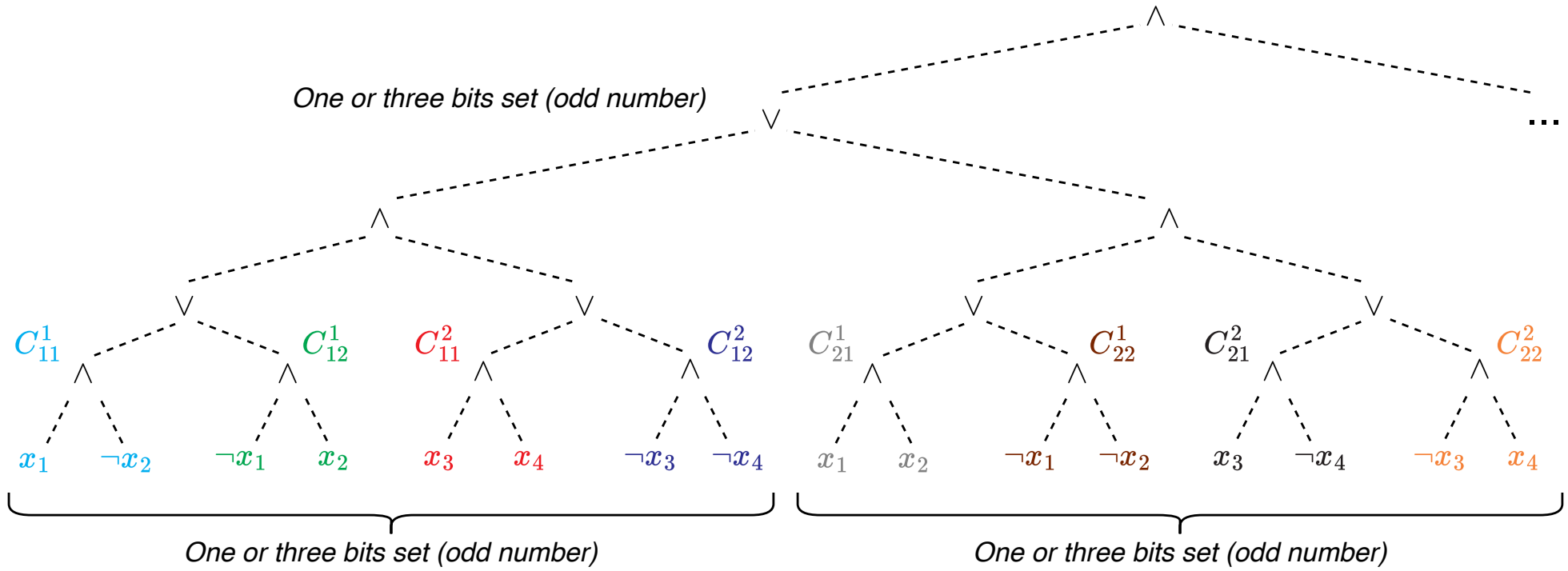
$$(C_{21}^1 \vee C_{22}^1) \wedge (C_{21}^2 \vee C_{22}^2)$$

$$C_{11}^1 = x_1 \wedge \neg x_2 \qquad C_{11}^2 = x_3 \wedge x_4$$

$$C_{12}^1 = \neg x_1 \wedge x_2 \qquad C_{12}^2 = \neg x_3 \wedge \neg x_4$$

$$C_{21}^1 = x_1 \wedge x_2 \qquad C_{21}^2 = x_3 \wedge \neg x_4$$

$$C_{22}^1 = \neg x_1 \wedge \neg x_2 \qquad C_{22}^2 = \neg x_3 \wedge x_4$$



$$C_{11}^1 = x_1 \wedge \neg x_2 \qquad C_{11}^2 = x_3 \wedge x_4 \qquad C_{21}^1 = x_1 \wedge x_2 \qquad C_{21}^2 = x_3 \wedge \neg x_4$$

$$C_{12}^1 = \neg x_1 \wedge x_2 \qquad C_{12}^2 = \neg x_3 \wedge \neg x_4 \qquad C_{22}^1 = \neg x_1 \wedge \neg x_2 \qquad C_{22}^2 = \neg x_3 \wedge x_4$$

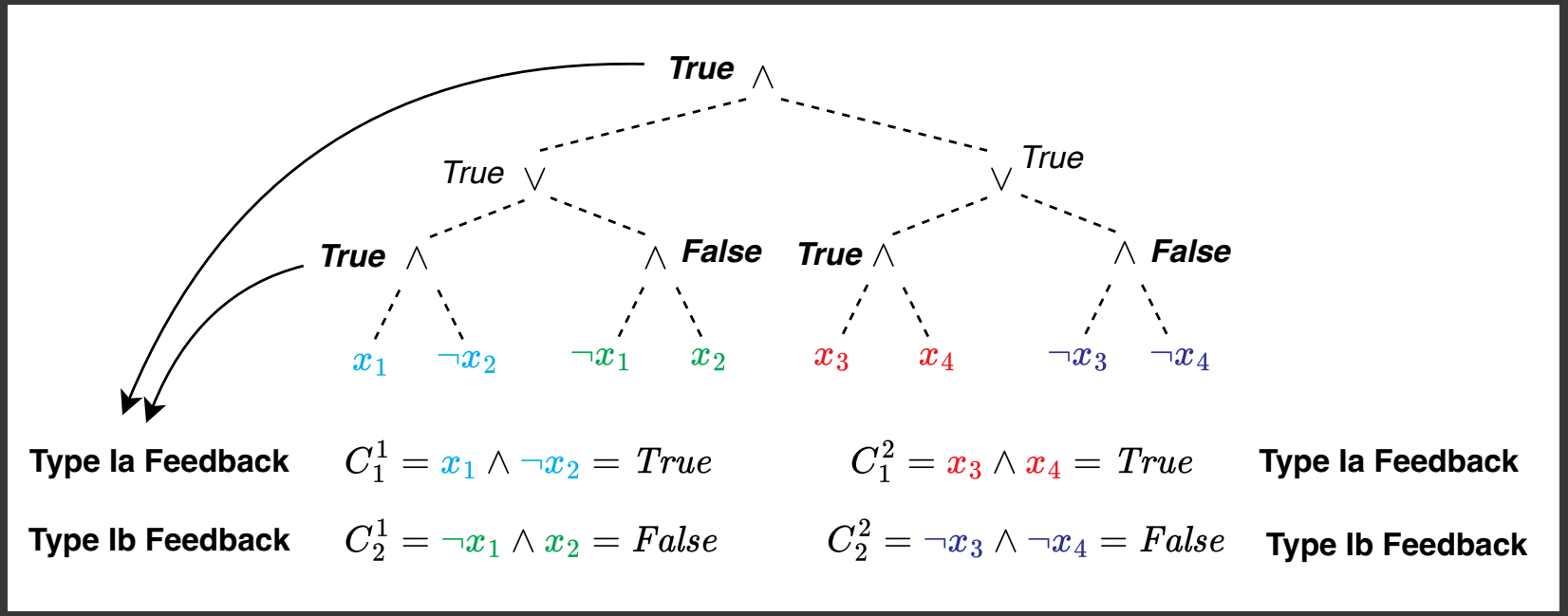
$$\text{Clause} = \left[\left(\left(C_{11}^1 \vee C_{12}^1\right) \wedge \left(C_{11}^2 \vee C_{12}^2\right)\right) \vee \left(\left(C_{21}^1 \vee C_{22}^1\right) \wedge \left(C_{21}^2 \vee C_{22}^2\right)\right)\right] \wedge \cdots$$

$$\text{Clause} = \left[\left(\left(C_{11}^{1} \vee C_{12}^{1}\right) \wedge \left(C_{11}^{2} \vee C_{12}^{2}\right)\right) \vee \left(\left(C_{21}^{1} \vee C_{22}^{1}\right) \wedge \left(C_{21}^{2} \vee C_{22}^{2}\right)\right)\right] \wedge \cdots$$

$$\text{Clause Votes} = \left[\left(\left(C_{11}^{1} + C_{12}^{1}\right) \times \left(C_{11}^{2} + C_{12}^{2}\right)\right) + \left(\left(C_{21}^{1} + C_{22}^{1}\right) \times \left(C_{21}^{2} + C_{22}^{2}\right)\right)\right] \times \cdots$$

Learning

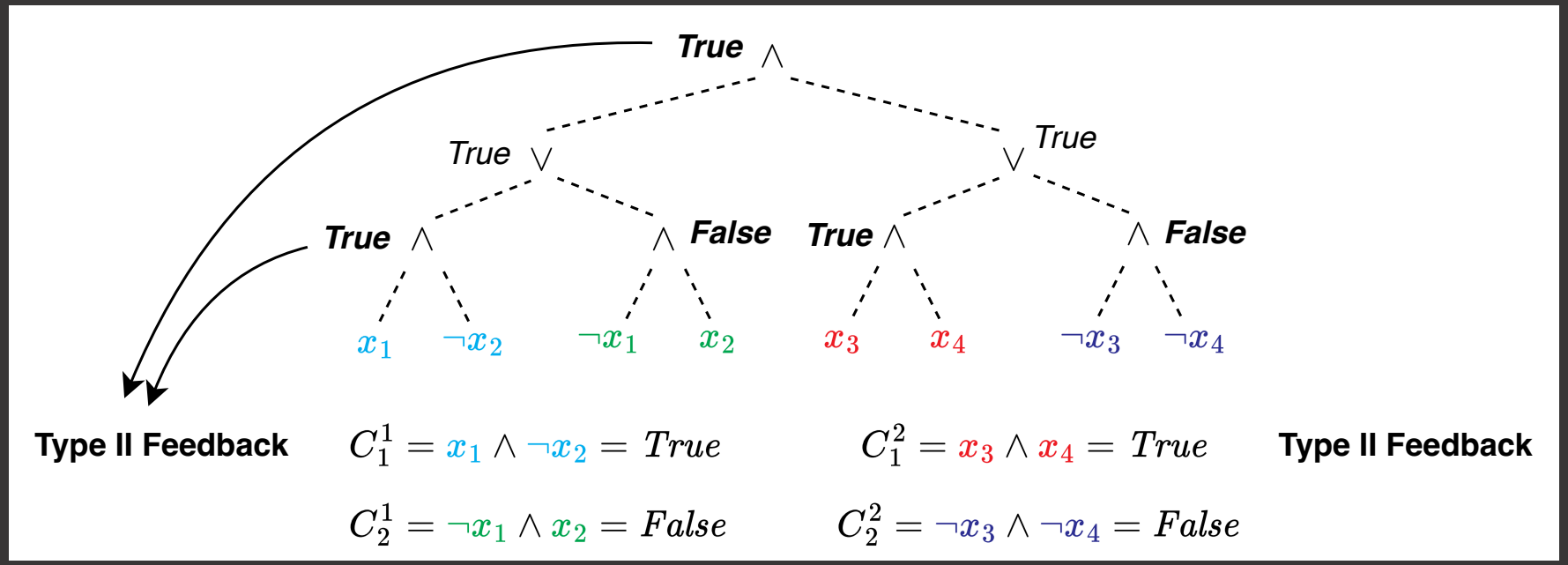
A **clause component** is updated exactly as a standard **clause**, apart from...



Input: $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 1$ **Target:** $y = 1$

Type I Feedback (Target Class)

- Record truth value of all AND-sub-expressions from leaf (**clause component**) to root (**clause**)
- If all AND-sub-expressions on path are True → Give **Type Ia feedback** to **clause component**
- Otherwise, give it **Type Ib feedback**



Input: $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 1$ **Target:** $y = 0$ (noise)

Type II Feedback (Other Classes)

- Record truth value of all AND-sub-expressions from leaf (**clause component**) to root (**clause**)
- If all AND-sub-expressions on path are True → Give **Type II feedback** to **clause component**

Further work

- Full implementation in PyTsetlinMachineCUDA, TMU, PyTsetlinMachine, and GraphTsetlinMachine
- Reuse/sharing of clause components
- Flexible architecture composition (1D, 2D, 3D, graphs, ...)
- Proof of concept applications (image understanding, natural language understanding, signal processing, etc.)
- Theoretical analysis