Neural Control with Certificates for Safe Autonomy

Djordje Zikelic

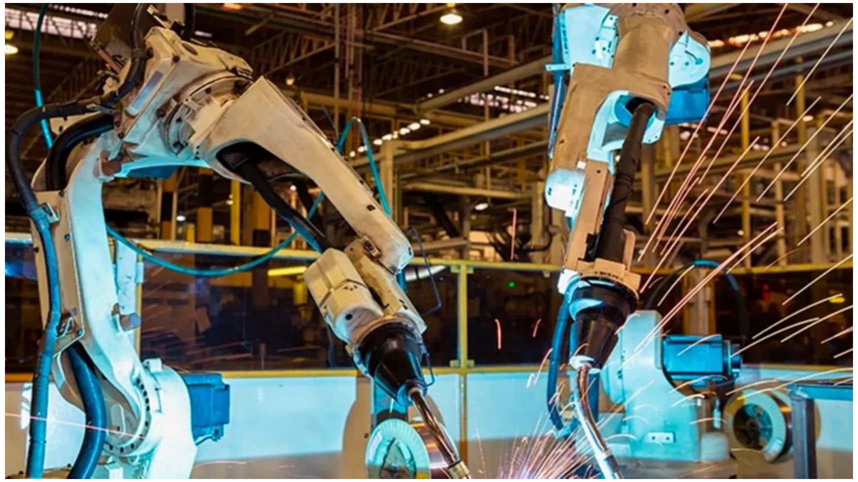
Singapore Management University

QuantFormal @ FSTTCS 2025

SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

Learning-enabled and neural control



- Reinforcement learning
- Other learning-based methods (e.g. supervised and unsupervised learning)
- Program synthesis (e.g. programmatic RL)

Safe autonomy requires **correctness guarantees**

How to learn correct neural controllers?

How to learn correct neural controllers?

Constrained reinforcement learning (RL)

+ Maximize expected reward in MDPs under safety constraints (constrained MDP formalism)
+ Focus on satisfying safety constraints in expectation
+ Recent work on almost-sure constraints (Sootla et al.) and VaR/CVaR constraints (Jiang et al.)

- No guarantees on safety constraint satisfaction

How to learn correct neural controllers?

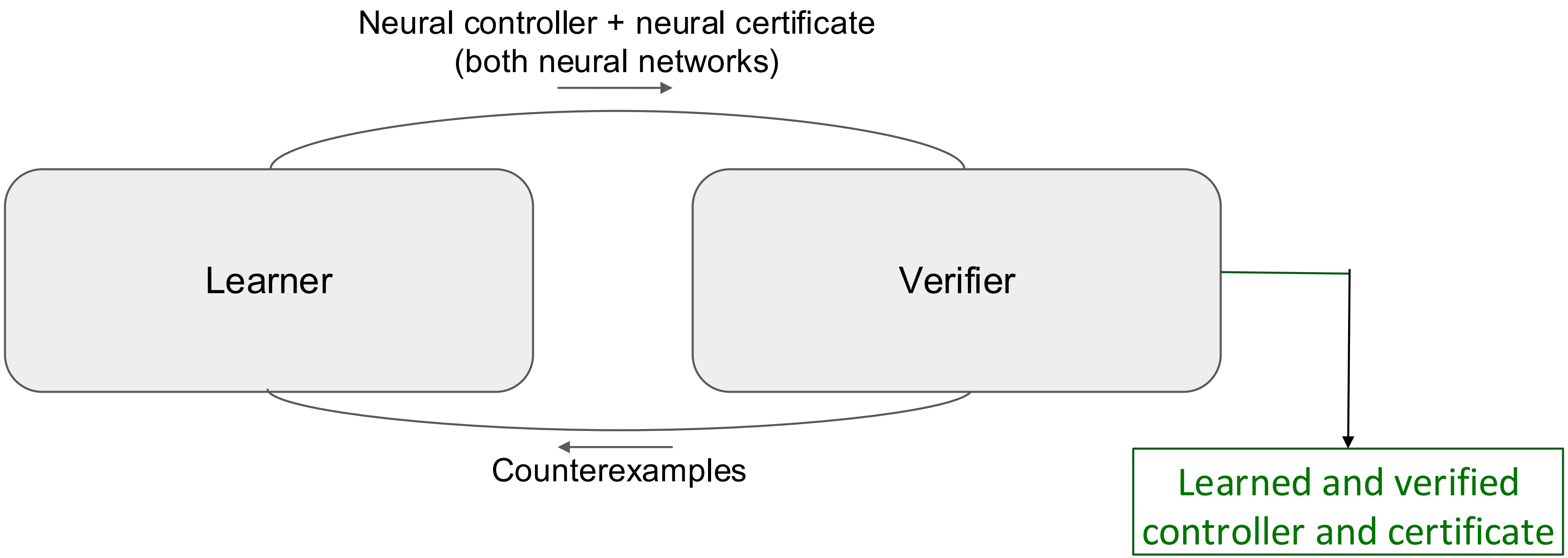
Constrained reinforcement learning (RL)

+ Maximize expected reward in MDPs under safety constraints (constrained MDP formalism)
+ Focus on satisfying safety constraints in expectation
+ Recent work on almost-sure constraints (Sootla et al.) and VaR/CVaR constraints (Jiang et al.)

- No guarantees on safety constraint satisfaction

Neural control with certificates

Neural control with certificates



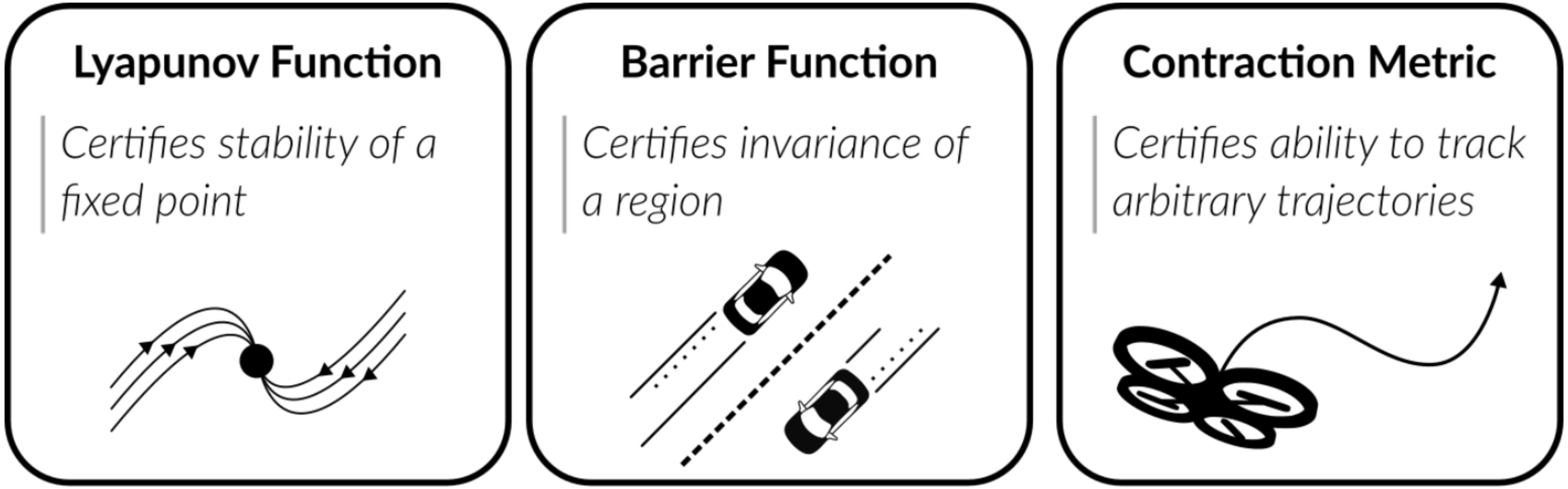
Idea: Learn controller + certificate for the specification

Machine learning + **Formal methods**

=

Formally verified learned controllers
(A certificate is a locally checkable witness of correctness)

Some examples of certificates



Learning of controllers with classical control theory certificates
+ verification by reduction to SMT-solving

*Image taken from: Dawson, Gao, Fan. *Safe Control with Learned Certificates: A Survey of Neural Lyapunov, Barrier, and Contraction Methods for Robotics and Control.* IEEE Transactions on Robotics

How to learn correct controllers?

Constrained reinforcement learning (RL)

+ Maximize expected reward in MDPs under safety constraints (constrained MDP formalism)
+ Focus on satisfying safety constraints in expectation
+ Recent work on almost-sure constraints (Sootla et al.) and VaR/CVaR constraints (Jiang et al.)

- No guarantees on safety constraint satisfaction

Neural control with certificates

+ Certificates act as formal proof of correctness
+ Formal certificates for reachability, safety, reach-avoidance
+ Formal guarantees by reducing verification to SMT-solving (Chang et al.; Abate et al.; Sankaranarayanan et al.; Fan et al.)

- Consider deterministic systems, no stochastic uncertainty

What is missing?

Theory

What should be the certificates for continuous stochastic systems?

Automation

How to learn and verify these new certificates as neural networks?

A Learner-verifier Framework for Neural Stochastic Control and Verification with Certificates

[AAAI'22, AAAI'23, NeurIPS'23, ATVA'23, TACAS'23, AAAI'25, CAV'25]

Joint work with Mathias Lechner, Tom Henzinger, Krishnendu Chatterjee
Matin Ansaripour, Abhinav Verma, Emily Yu

Requirements for neural controller synthesis

1. **Full automation**

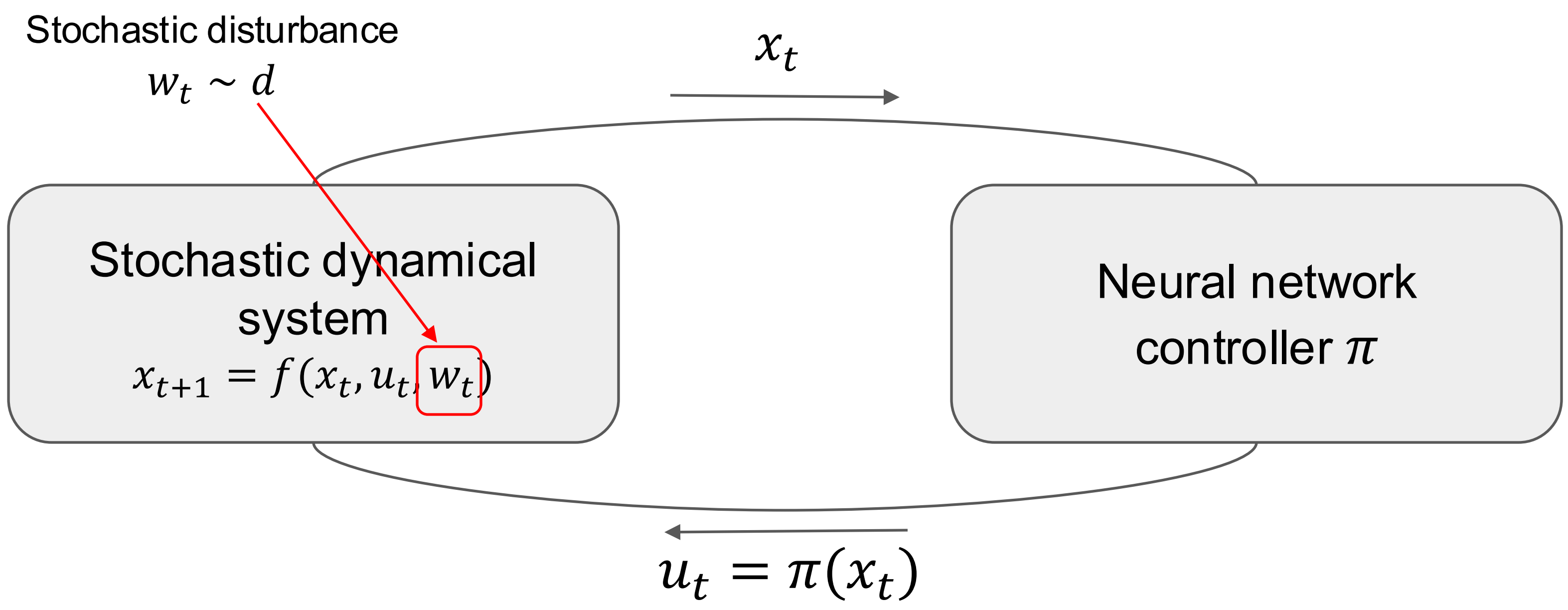
2. **General continuous systems**
   (classical automated control theory methods restricted to polynomial systems)

3. **Hard formal guarantees**
   (sampling, numerical methods, testing provide soft correctness guarantees)

4. **Long or even infinite-time horizon**
   (sampling, numerical methods, testing only applicable to finite horizon problems)

5. **Consideration of stochastic environment uncertainty**
   (formal guarantees require system model, but the model may be imprecise or contain noise)

Model: Stochastic dynamical system (a.k.a. infinite-state discrete-time MDP)

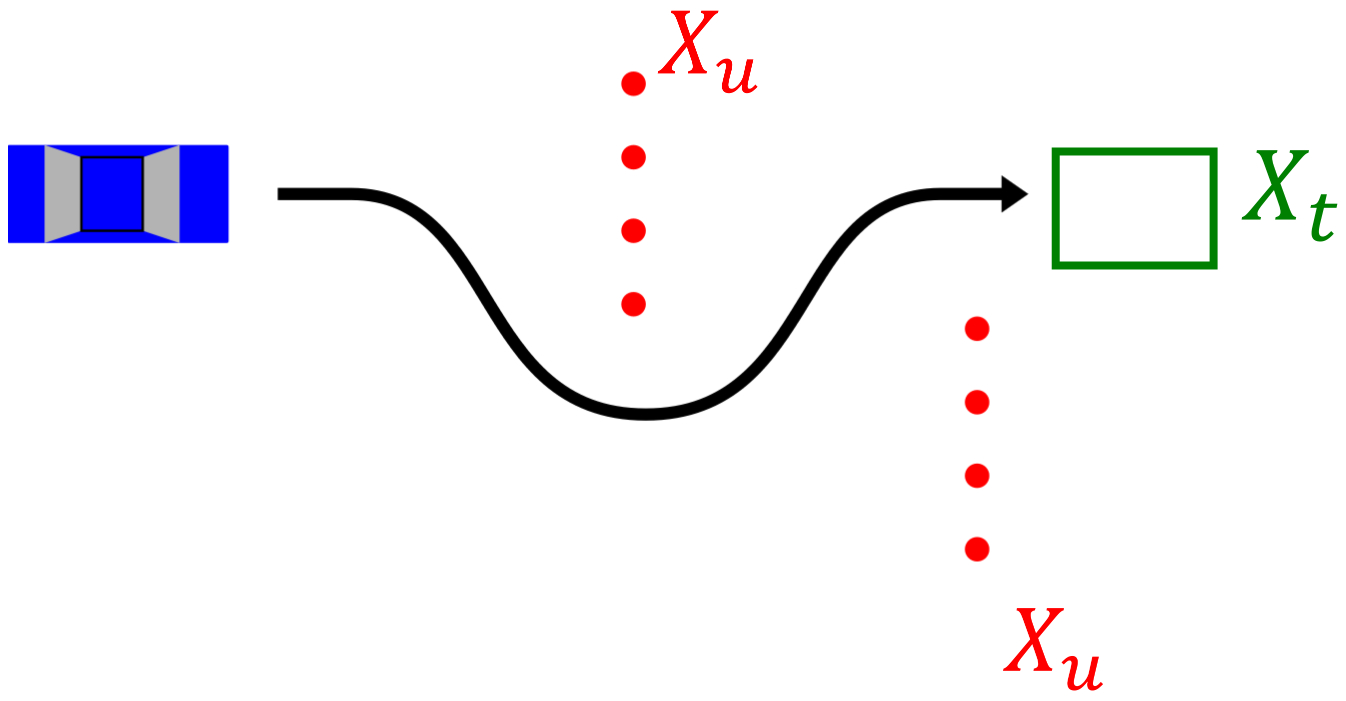


Given: Initial region X_0 ,specification ϕ defining a set of “good” traces, probability threshold $p \in [0,1]$

Control problem: Neural controller + certificate that guarantee $\mathbb{P}_{x_0}^{\pi}[x_0, x_1, x_2, \ldots \models \phi] \geq p$ for all $x_0 \in X_0$

Verification problem: Neural certificate that guarantees $\mathbb{P}_{x_0}^{\pi}[x_0, x_1, x_2, \ldots \models \phi] \geq p$ for all $x_0 \in X_0$

Most of this talk: Reach-avoid specifications



Reachability = reach the target set of states
Safety = do not reach the unsafe set of states
Reach-avoidance = reach the target set while avoiding the unsafe set of states

Research questions that need to be answered

Theory

What should be the certificates for continuous stochastic systems?

Supermartingale certificates

Automation

How to learn and verify these new certificates as neural networks?

Abstract interpretation + Lipschitz analysis

What are {super,sub}martingales?

Martingale – stochastic process constant in expectation $\mathbb{E}[X_{n+1}|X_n] = X_n$

Supermartingale – stochastic process decreasing in expectation $\mathbb{E}[X_{n+1}|X_n] \leq X_n$

Submartingale – stochastic process increasing in expectation $\mathbb{E}[X_{n+1}|X_n] \geq X_n$

What are {super,sub}martingales?

Martingale – stochastic process constant in expectation $\mathbb{E}[X_{n+1}|X_n] = X_n$

Supermartingale – stochastic process decreasing in expectation $\mathbb{E}[X_{n+1}|X_n] \leq X_n$

Submartingale – stochastic process increasing in expectation $\mathbb{E}[X_{n+1}|X_n] \geq X_n$

Martingale certificates in stochastic control

Ranking supermartingales (RSMs) for probability 1 reachability

[Kushner, Transactions on Automatic Control 1966, Chakarov, Sankaranarayanan, CAV 2013]

A measurable function $V: X \to \mathbb{R}$ for a target set X_t such that:

1. Nonnegativity. $V(x) \geq 0$ for $x \in X$
2. Strict expected decrease. $\exists \epsilon > 0$ s.t. $\mathbb{E}_{w \sim d}[V(f(x, \pi(x), w))] \leq V(x) - \epsilon$ for $x \in X \backslash X_t$

Stochastic barrier functions for probability $p \in [0,1]$ safety

[Prajna, Jadbabaie, Pappas. CDC 2004]

Automated synthesis of polynomial supermartingale certificates

Probability $p \in [0,1]$ safety in stochastic control

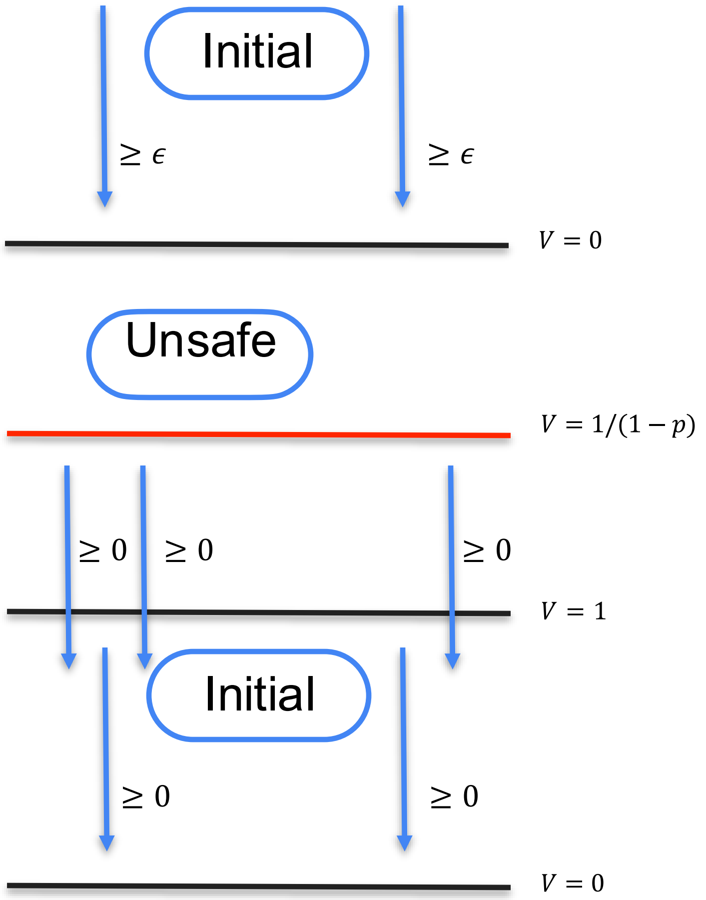
[Prajna, Jadbabaie, Pappas. CDC 2004]

Probability 1 reachability for probabilistic program verification

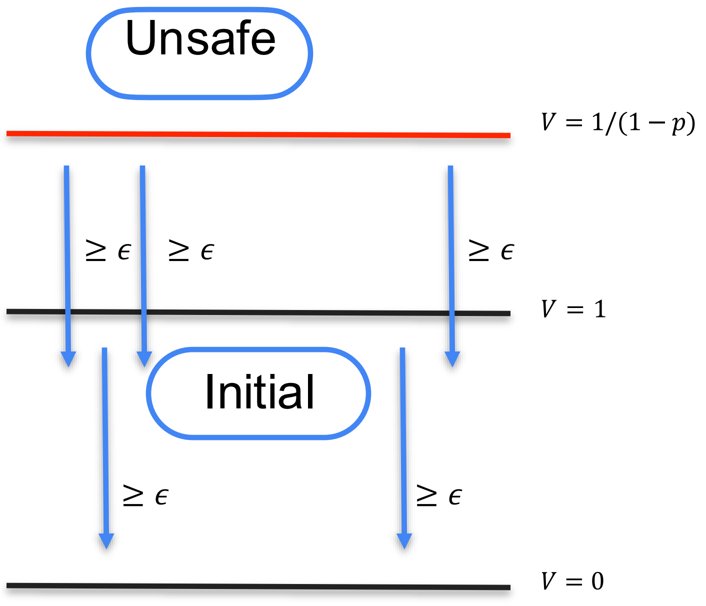
[Chakarov, Sankaranarayanan, CAV 2013]

Probability $p \in [0,1]$ reachability for probabilistic program verification

[Chatterjee, Novotny, Zikelic, POPL 2017; Chatterjee, Goharshady, Meggendorfer, Zikelic, CAV 2022]



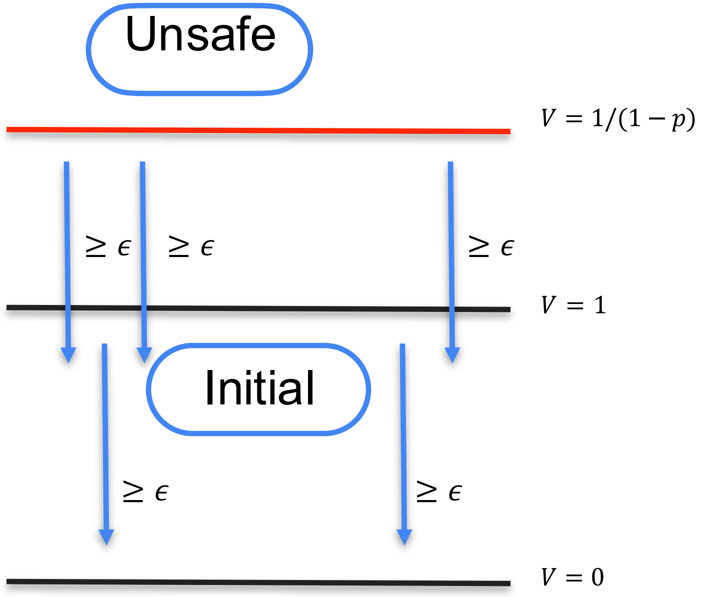
Reach-avoid supermartingale



RASM is a measurable function $V: X \to \mathbb{R}$ such that:

1. Nonnegativity. $V(x) \geq 0$ for each $x \in X$.
2. Initial condition. $V(x) \leq 1$ for each initial state $x \in X_0$.
3. Safety condition. $V(x) \geq 1/(1-p)$ for each unsafe state $x \in X_u$.
4. Strict expected decrease. There exists $\epsilon > 0$ such that $V(x) \geq \mathbb{E}_{\omega \sim d}[V(f(x, \pi(x), w)] + \epsilon$ for $x \in X \backslash X_t$ at which $V(x) \leq 1/(1-p)$.

Reach-avoid supermartingale

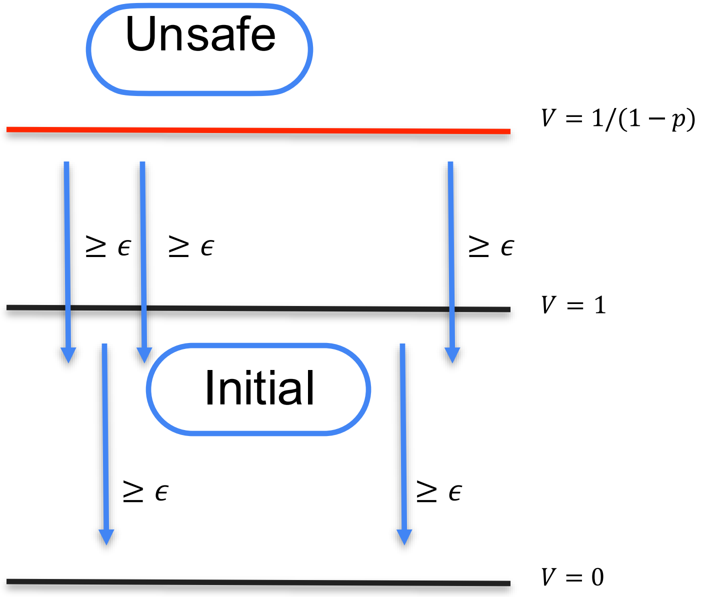


RASM is a measurable function $V: X \to \mathbb{R}$ such that:

1. Nonnegativity. $V(x) \geq 0$ for each $x \in X$.
2. Initial condition. $V(x) \leq 1$ for each initial state $x \in X_0$.
3. Safety condition. $V(x) \geq 1/(1-p)$ for each unsafe state $x \in X_u$.
4. Strict expected decrease. There exists $\epsilon > 0$ such that $V(x) \geq \mathbb{E}_{\omega \sim d}[V(f(x, \pi(x), w)] + \epsilon$ for $x \in X \backslash X_t$ at which $V(x) \leq 1/(1-p)$.

Theorem (Soundness). Suppose that the system admits a RASM. Then $\mathbb{P}_{x_0}^{\pi}[ReachAvoid(X_t, X_u)] \geq p$ for all $x \in X_0$.

Reach-avoid supermartingale



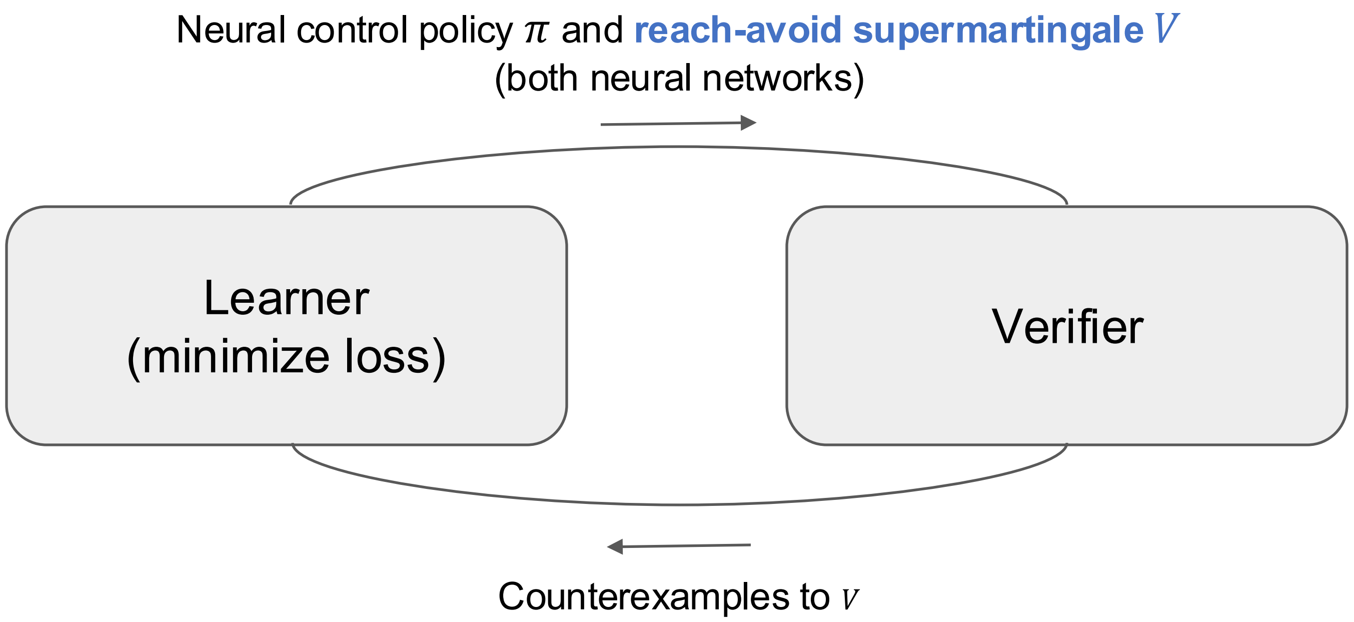
RASM is a measurable function $V: X \to \mathbb{R}$ such that:

1. Nonnegativity. $V(x) \geq 0$
2. Initial c… $x \in X_0$.
3. Safety c… …nsafe state $x \in X_u$.
4. Strict exp… …xists $\epsilon > 0$ such that $V(x) \geq \mathbb{E}_{\omega\sim d}[V(f(x, \pi(x), w)] + \epsilon$ for $x \in X$ \ … which $V(x) \leq 1/(1-p)$.

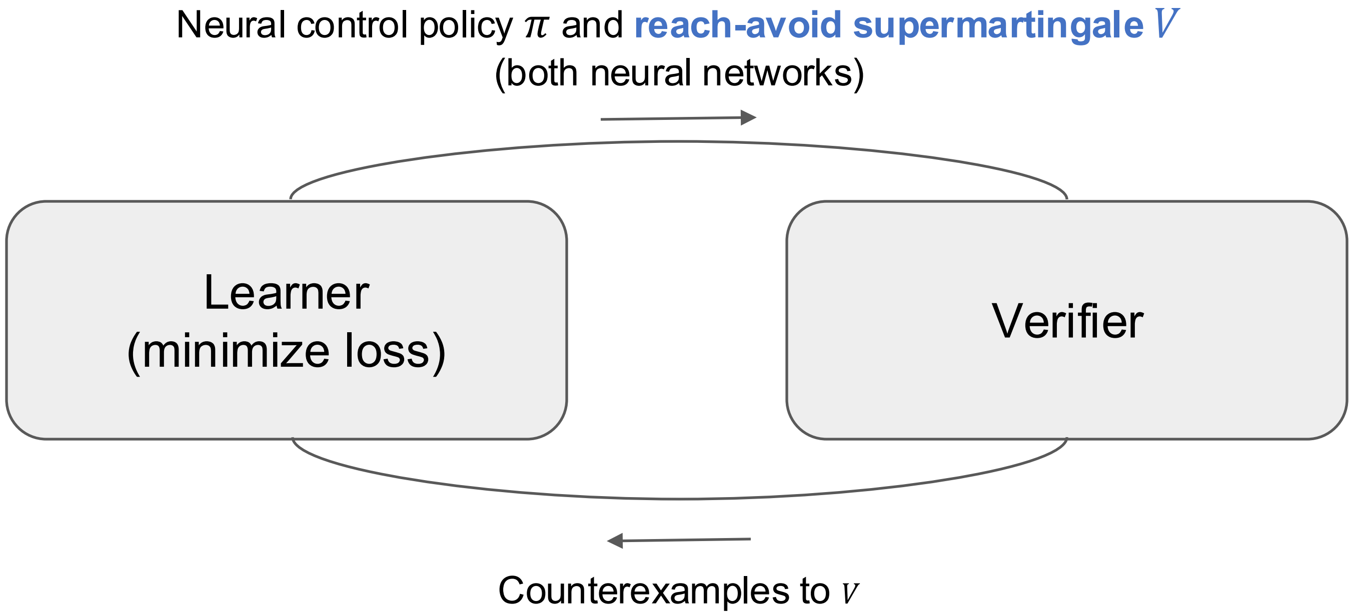
RASMs unify and generalize ranking supermartingales and stochastic barrier functions

Theorem (Soundness). Suppose that the system admits a RASM. Then $\mathbb{P}^{\pi}_{x_0}[ReachAvoid(X_t, X_u)] \geq p$ for all $x \in X_0$.

Neural control with supermartingale certificates



Neural control with supermartingale certificates

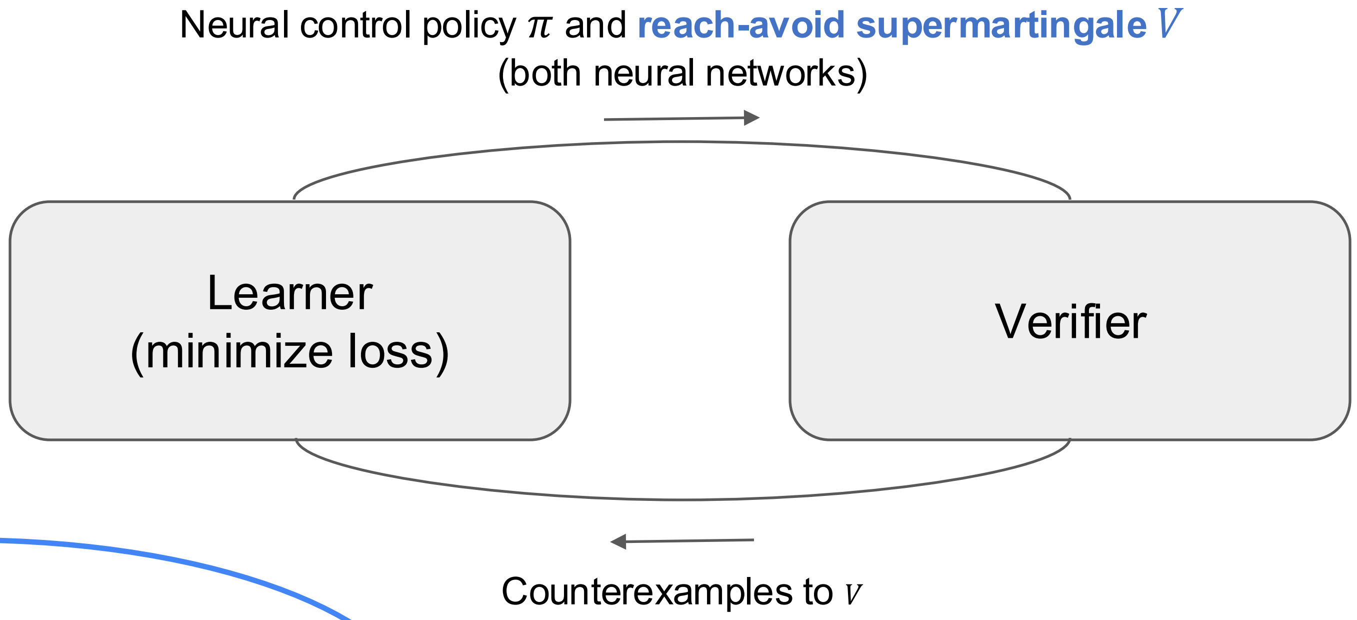


Assumptions (needed for automation):

(1) State space X of the system is compact

(2) Dynamics function f is (Lipschitz) continuous with Lipschitz constant L_f

Neural control with supermartingale certificates



Loss function incurs loss if V_θ violates the RASM definition

Learner module

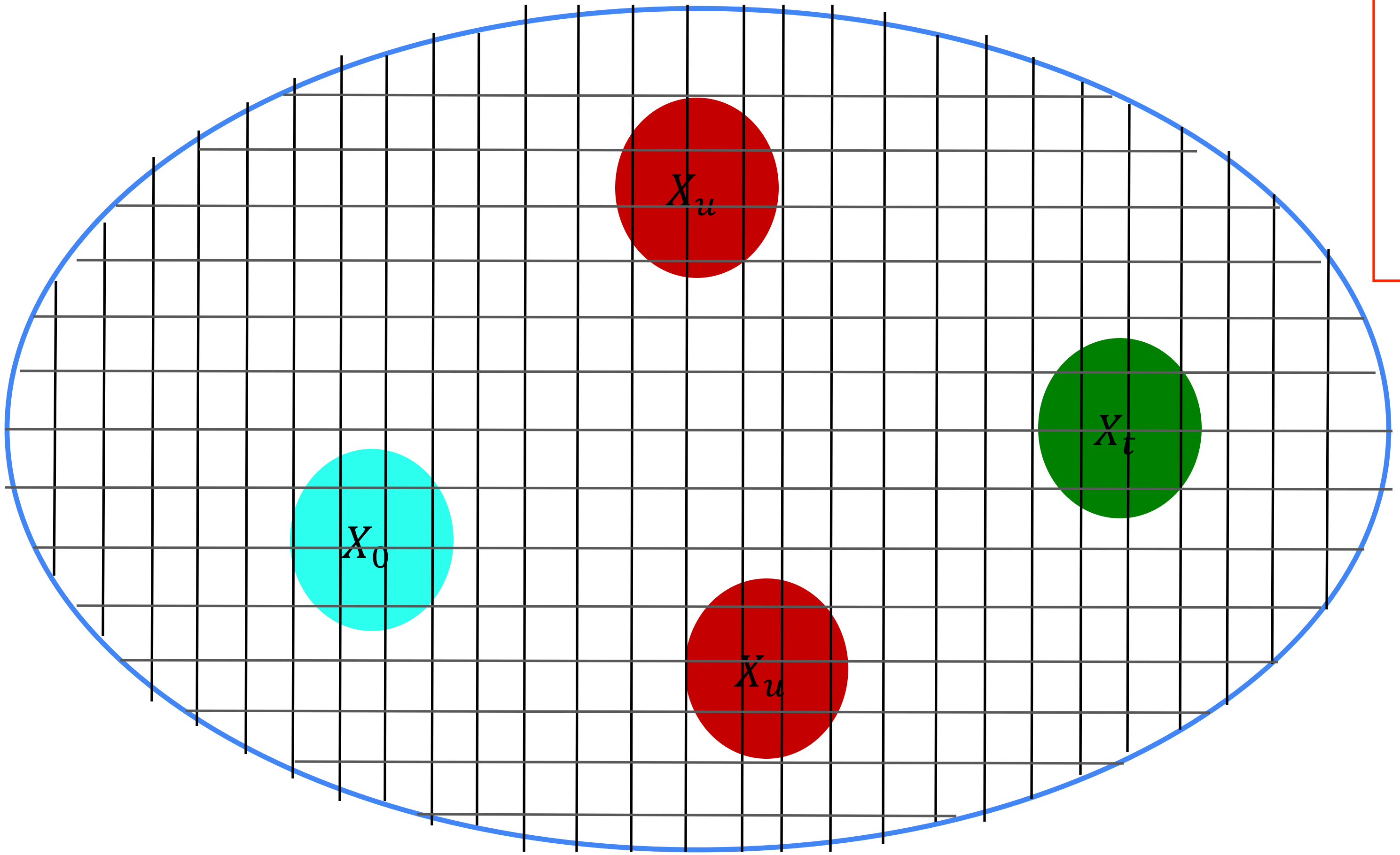
Unsupervised learning task, loss function encodes defining conditions of RASMs

(Non-negativity imposed by default, by applying ReLU/softplus on neural RASM output)

$$\mathcal{L}(\theta, \nu) = \mathcal{L}_{Init}(\nu) + \mathcal{L}_{Unsafe}(\nu) + \mathcal{L}_{Decrease}(\theta, \nu)$$

Intuition: The loss function empirically encodes all RASM defining conditions. Hence, it guides the learner to learn a neural controller that admits a neural RASM and thus guaranteeing reach-avoidance with the desired probability.

Training set: Discretization



$\tilde{X}$ = hyperrectangular discretization of X

$C_{init} = X_0 \cap \tilde{X}$

$C_{unsafe} = X_U \cap \tilde{X}$

$C_{decrease} = \tilde{X} \setminus (X_T \cup X_U)$

Loss function

$$\mathcal{L}(\theta,\nu) = \mathcal{L}_{Init}(\nu) + \mathcal{L}_{Unsafe}(\nu) + \mathcal{L}_{Decrease}(\theta,\nu)$$

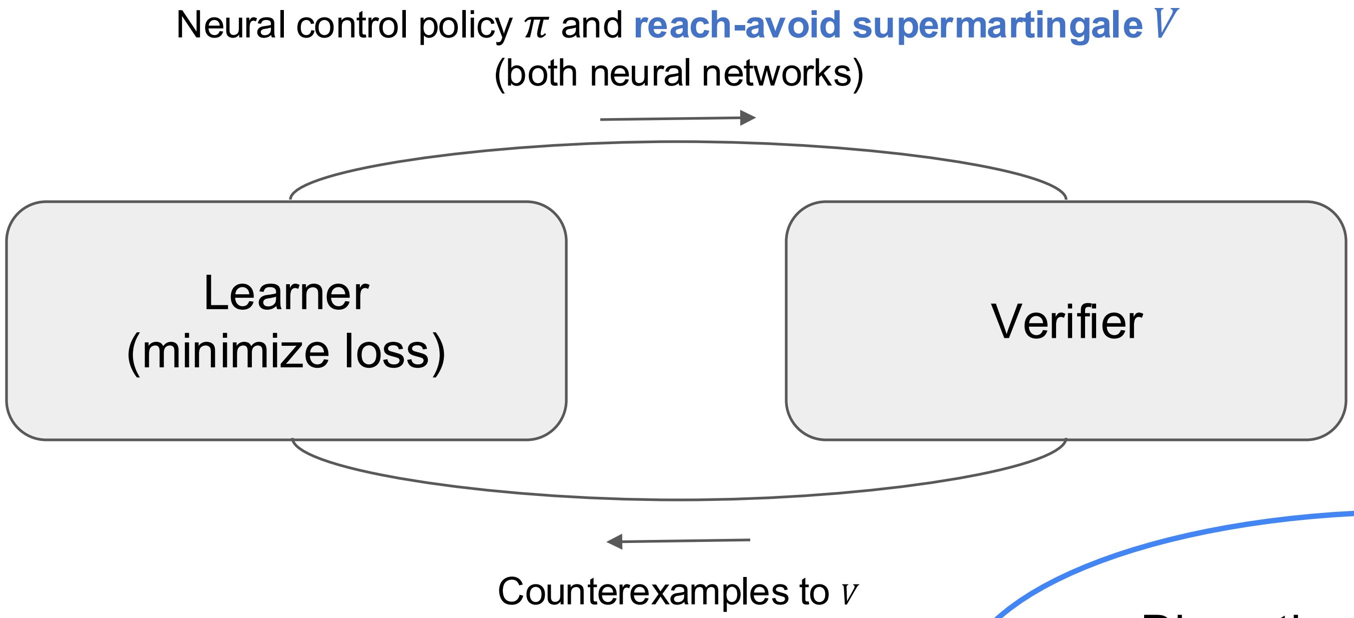
Empirically enforce RASM defining conditions

$$\mathcal{L}_{Init}(\nu) = \max_{\mathbf{x}\in C_{init}} \{V_\nu(\mathbf{x}) - 1, 0\}$$

$$\mathcal{L}_{Unsafe}(\nu) = \max_{\mathbf{x}\in C_{unsafe}} \{\frac{1}{1-p} - V_\nu(\mathbf{x}), 0\}$$

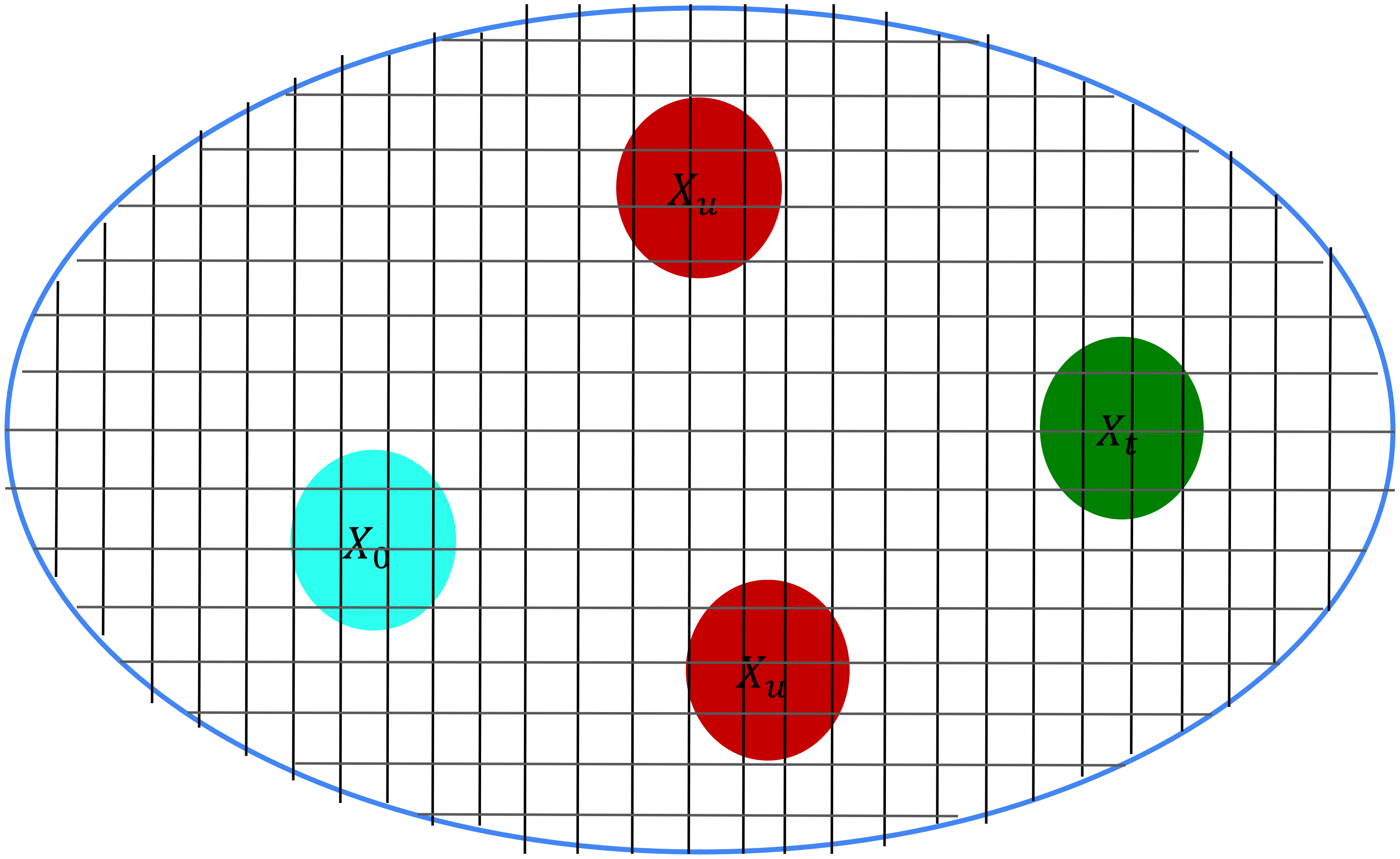
$$\mathcal{L}_{Decrease}(\theta,\nu) = \frac{1}{|C_{decrease}|} \cdot \sum_{\mathbf{x}\in C_{decrease}} (max\{ \sum_{\omega_1,\ldots,\omega_N\sim\mathcal{N}} \frac{V_\nu(f(\mathbf{x},\pi_\theta(\mathbf{x}),\omega_i))}{N} - V_\nu(\mathbf{x}) + \tau\cdot K, 0\})$$

Neural control with supermartingale certificates

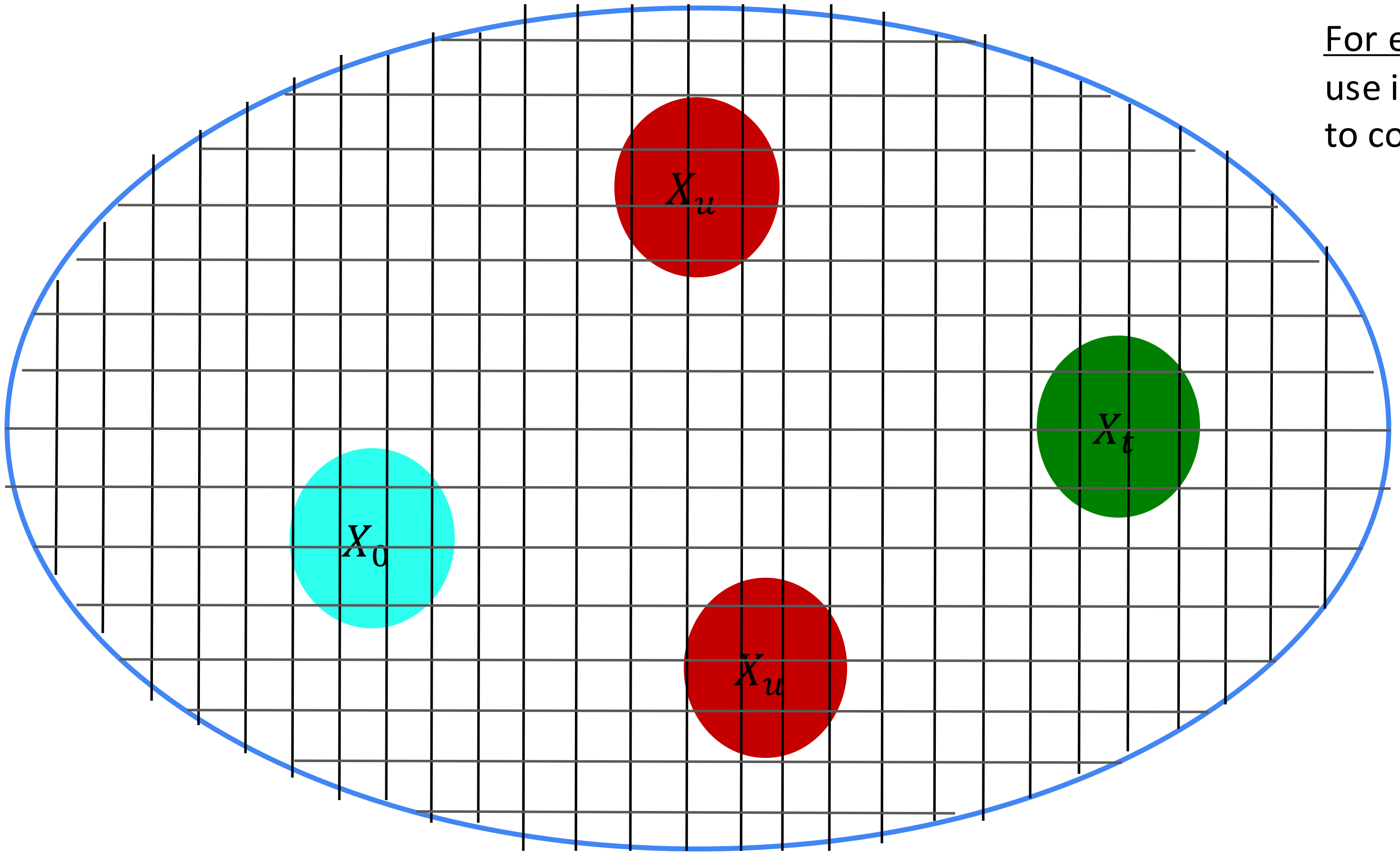


Discretize the state space and use abstract interpretation

Verifier module



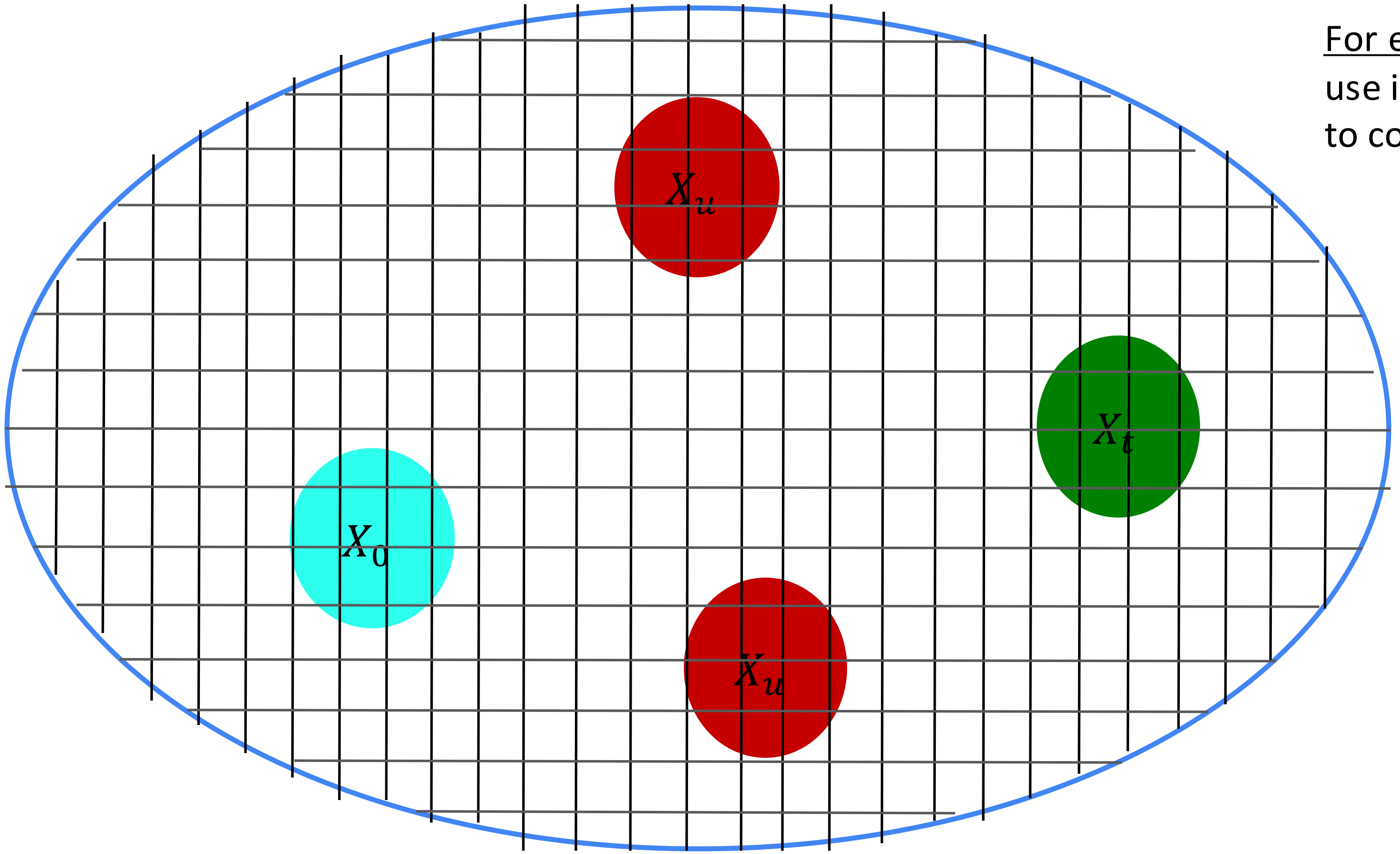
Verifier module



<u>For each discretization cell:</u>
use interval arithmetic abstract interpretation (IAAI) [1] to compute bounds on the RASM over each cell

[1] Gowal et al. *On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models.* arXiv 2018

Verifier module

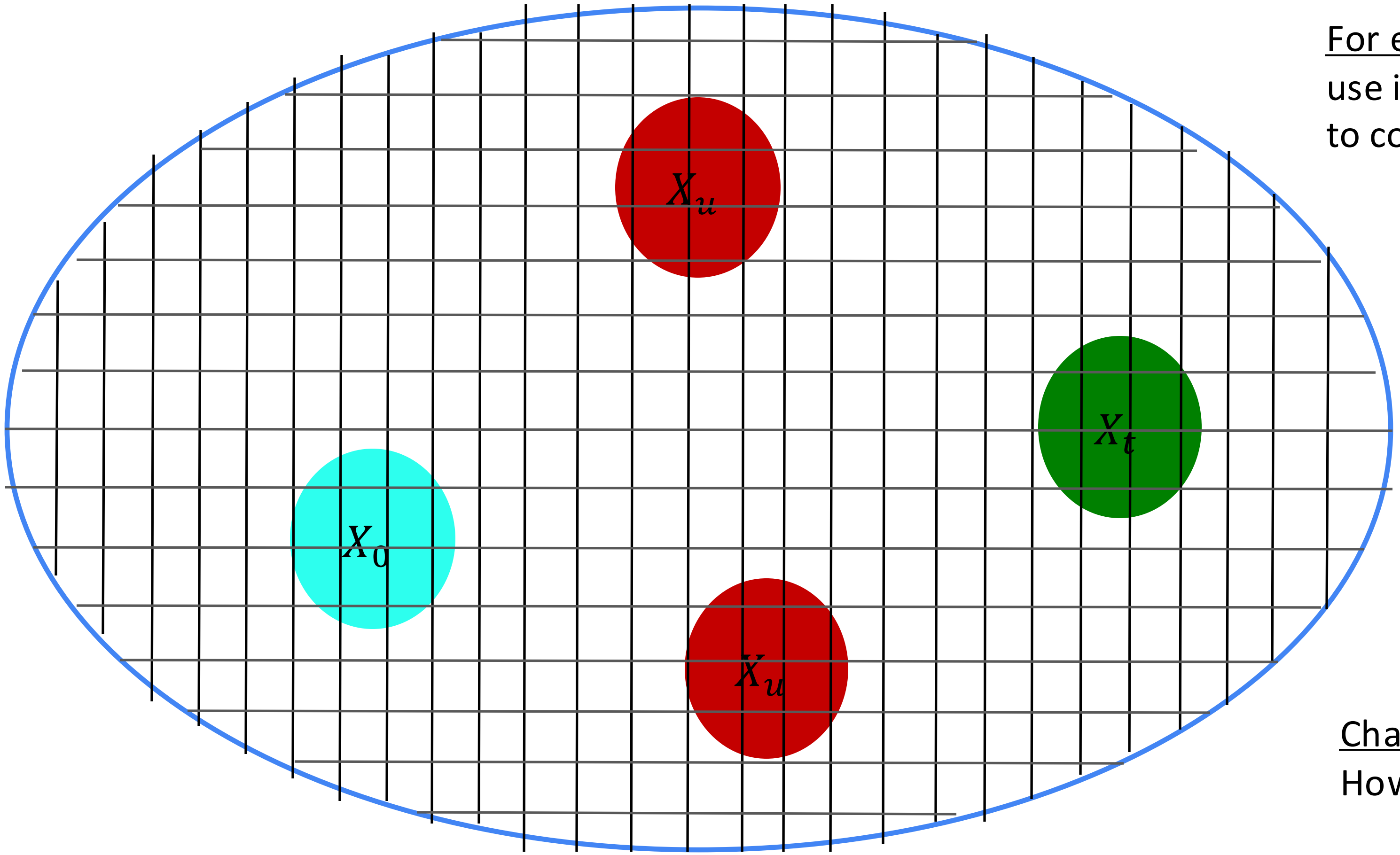


For each discretization cell:
use interval arithmetic abstract interpretation (IAAI) [1] to compute bounds on the RASM over each cell

Check Initial and Safety conditions of RASMs over all grid cells that intersect X_0 or X_u

[1] Gowal et al. *On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models.* arXiv 2018

Verifier module



For each discretization cell:
use interval arithmetic abstract interpretation (IAAI) [1] to compute bounds on the RASM over each cell

Check Initial and Safety conditions of RASMs over all grid cells that intersect X_0 or X_u

Challenge:
How to verify the expected decrease condition?

[1] Gowal et al. *On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models.* arXiv 2018

Verifier module

Solution: Check a stricter condition at the centers of the discretization cells

$$\mathbb{E}_{\omega\sim d}[V_\nu(f(\mathbf{x},\pi_\theta(\mathbf{x}),\omega))] < V_\nu(\mathbf{x}) - \tau \cdot K$$

Lipschitz error term

Expected value computation

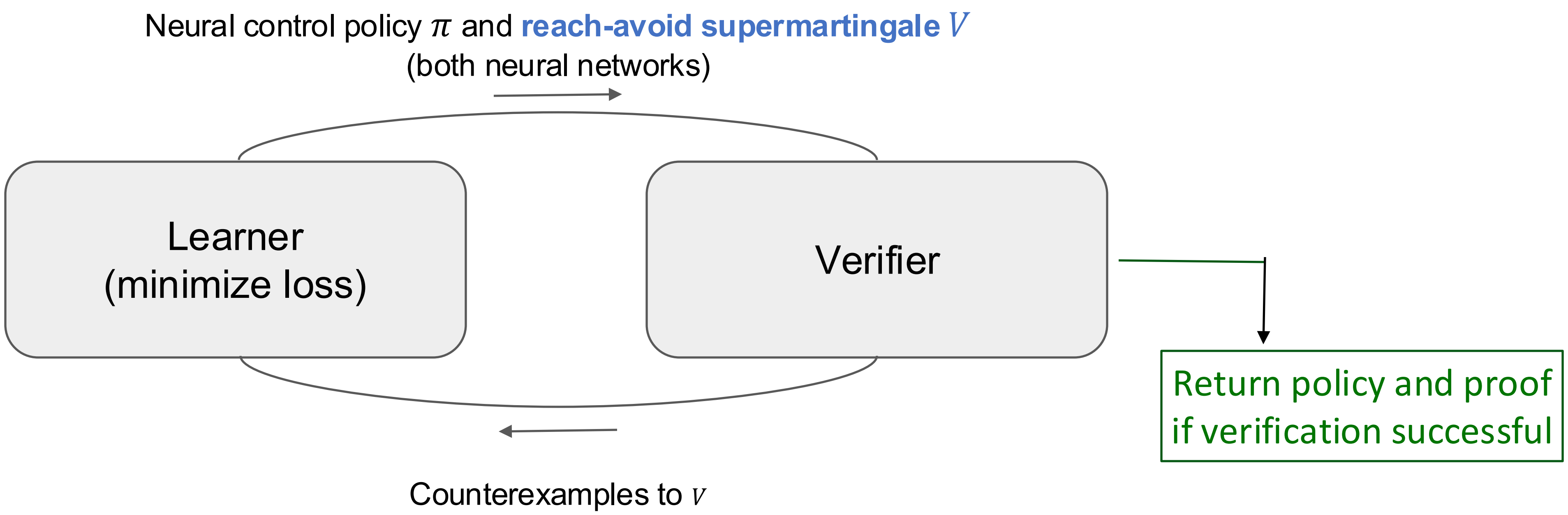
Compute: $\mathbb{E}_{\omega\sim d}\left[V_\nu\left(f(\mathbf{x},\pi_\theta(\mathbf{x}),\omega)\right)\right]$ for a <u>fixed</u> $\mathbf{x}\in X$

Problem: V is a neural network, so no closed form solution in general

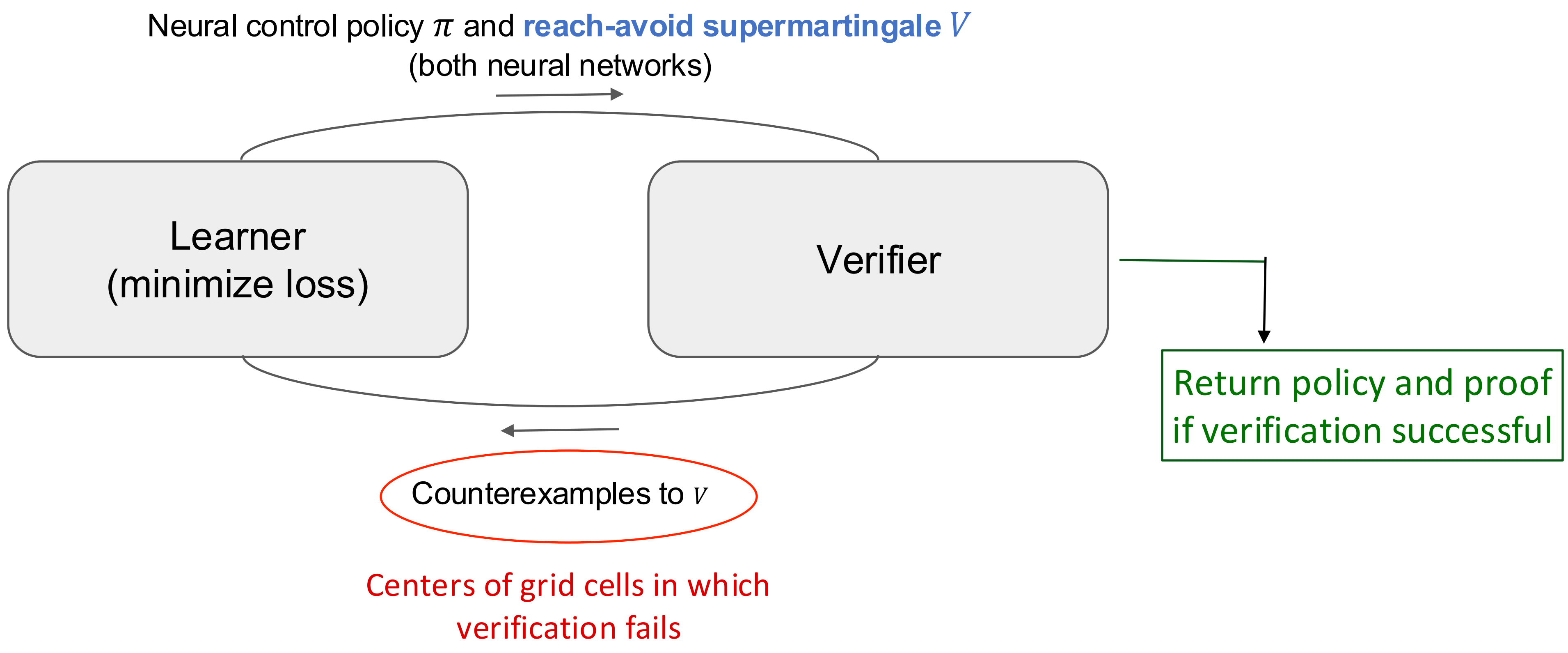
Solution: Discretize the support of d, expand as a sum, then bound the summands via IAAI

$$\mathbb{E}_{\omega\sim d}[V_\nu(f(\mathbf{x},\pi_\theta(\mathbf{x}),\omega))] \leq \sum_{C\in cells} maxvol \cdot \sup_{\mathbf{x}\in C} V_\nu(\mathbf{x})$$

Neural control with supermartingale certificates



Neural control with supermartingale certificates



Verifier guides the learner

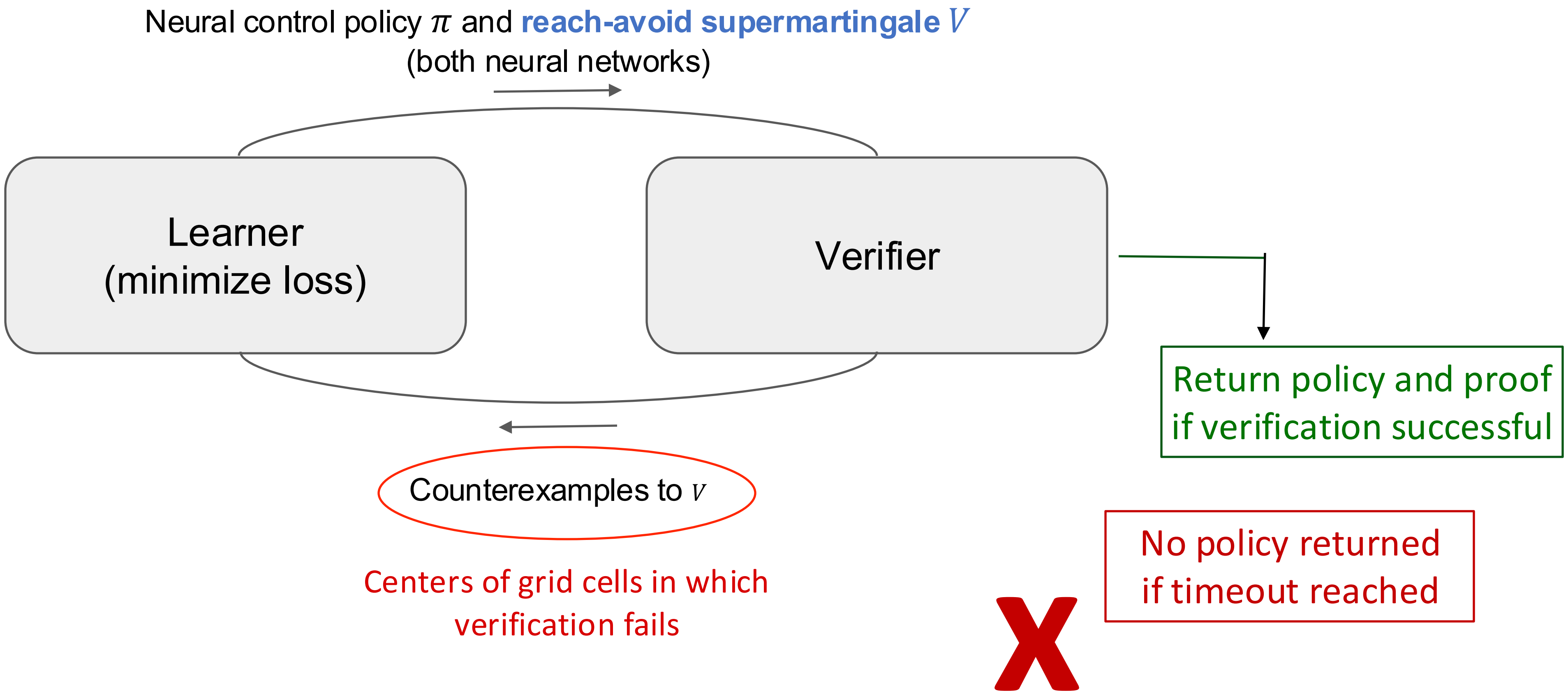
1. Counterexample guided inductive synthesis (CEGIS)

— counterexamples cell centers are added to training sets used by the learner

2. Adaptive grid refinement

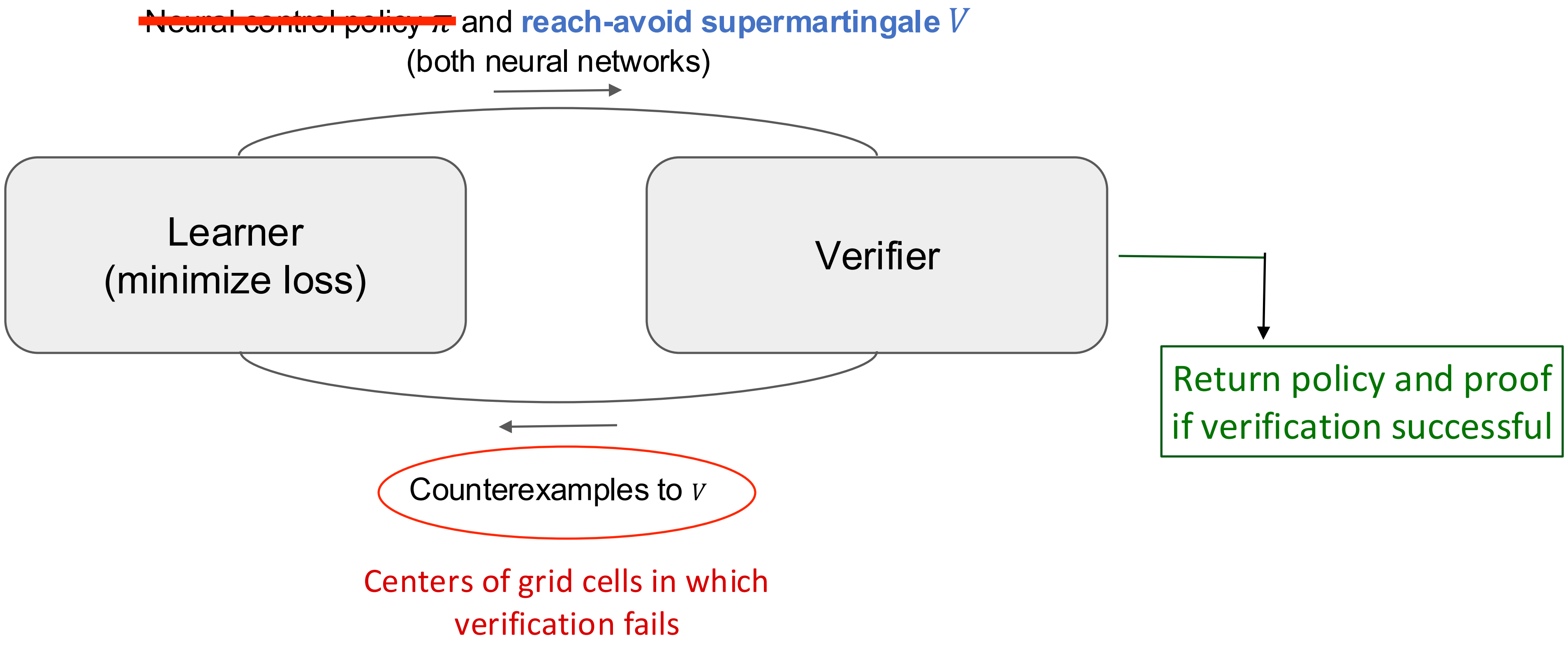
— grid cells that contain spurious counterexamples are refined

Neural control with supermartingale certificates

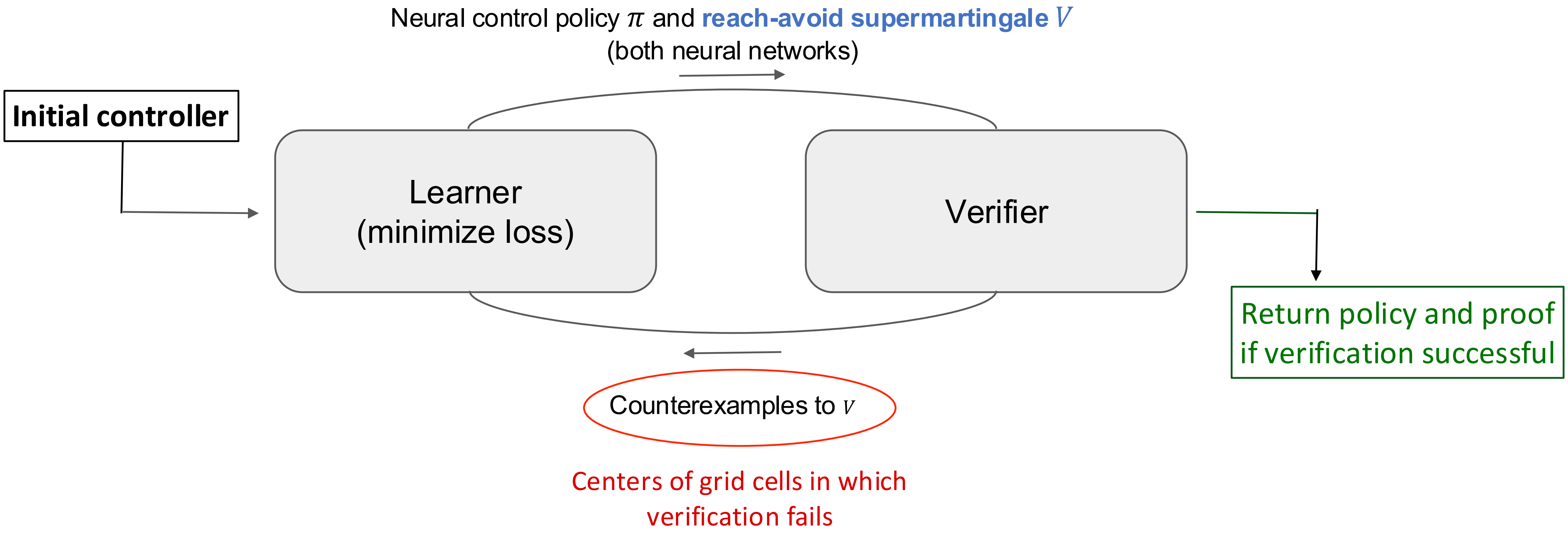


Theorem (Soundness). If the algorithm outputs a control policy π, then V is a valid RASM and reach-avoidance is satisfiedwith probability at least p.

Formal verification of neural policies



Repair of neural policies



Experimental evaluation

- Initialize neural network policy using 100 iterations of PPO or SAC
- Policy network: [256,256] hidden dimension with ReLU activations
- RASM network: [256,256] hidden dimension with ReLU activations

Benchmark	Original tool (AAAI 2023)		With improvements	
	Probability	# iterations	Probability	# iterations
2D Linear	96.65	6	99.51	16
Inverted Pendulum	95.90	15	98.95	8
Collision avoidance	95.00	13	96.22	5
3D Linear	Fail	-	94.13	10
Humanoid.	Fail	-	69.44	32

See also: Badings et al. "Policy Verification in Stochastic Dynamical Systems Using Logarithmic Neural Certificates". CAV 2025

Other learner-verifier CEGIS frameworks

Y.-C. Chang, N. Roohi, S. Gao
A. Abate, M. Giacobbe, et al.
S. Sankaranarayanan et al.

Continuous-time, deterministic

Certificates are Lyapunov functions and control barrier functions

Closed-form/symbolic reasoning
Verification reduced to SMT-solving

Our framework

Discrete-time, **stochastic**

Certificates are **supermartingales**

Discretization, Lipschitz continuity
Verification via **abstract interpretation**

Extension to more general specifications

1. **Compositional reasoning** about reach-avoidance [NeurIPS'23]

2. **Stability** (a.k.a. co-Büchi or reach-and-stay) with prob. 1 [ATVA'23]

3. Supermartingale certificates for **general omega-regular specifications** [CAV'25]

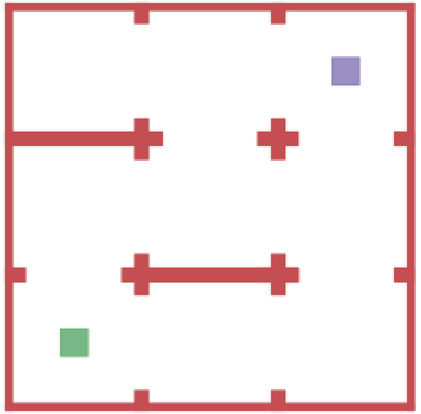
Compositional policy learning [NeurIPS'23]

- RL algorithms **struggle** with long-horizon tasks and complex logical specifications
- Compositional policy learning:
  1. Decompose complex logical specifications into simpler subtasks
  2. Solve subtasks
  3. Compose subtask policies into a global policy
- Prior work: Either no formal guarantees or restricted to deterministic systems

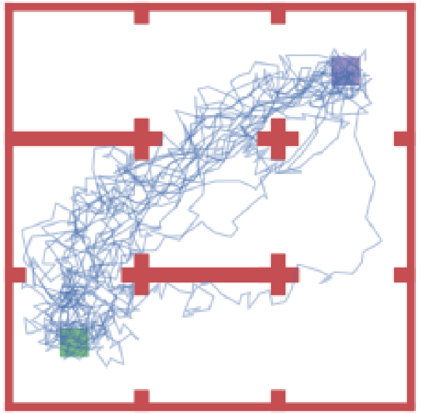
Our contribution

Compositional policy learning framework
for **stochastic** control systems
with **formal guarantees** on correctness

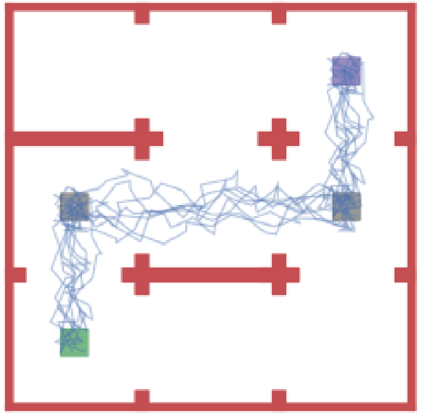
Example: Stochastic 9-rooms environment



Goal: Move from green to purple without hitting a wall



Challenge: End-to-end policy hard to formally verify

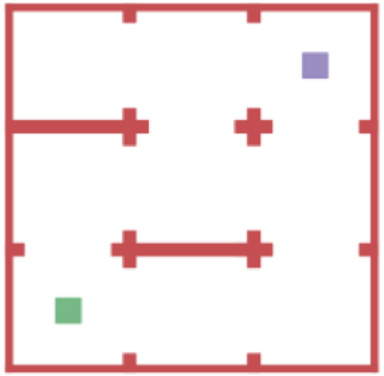


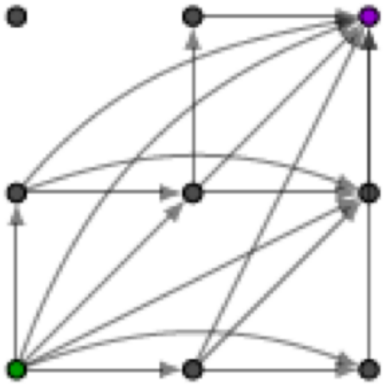
Solution: Decompose the task into simpler subtasks

Compositional policy learning problem

Given: Stochastic dynamical system, SpectRL specification ϕ, probability $p \in [0,1]$

(SpectRL [1,2] = all boolean and sequential compositions of reach-avoid tasks)





Goal: Learn compositional policy that satisfies specification ϕ with probability $\geq p$

(compositional policy = policies for a subset of edges that together solve the task)

[1,2] Jothimurugan, Bansal, Alur, Bastani. NeurIPS 2019, NeurIPS 2022

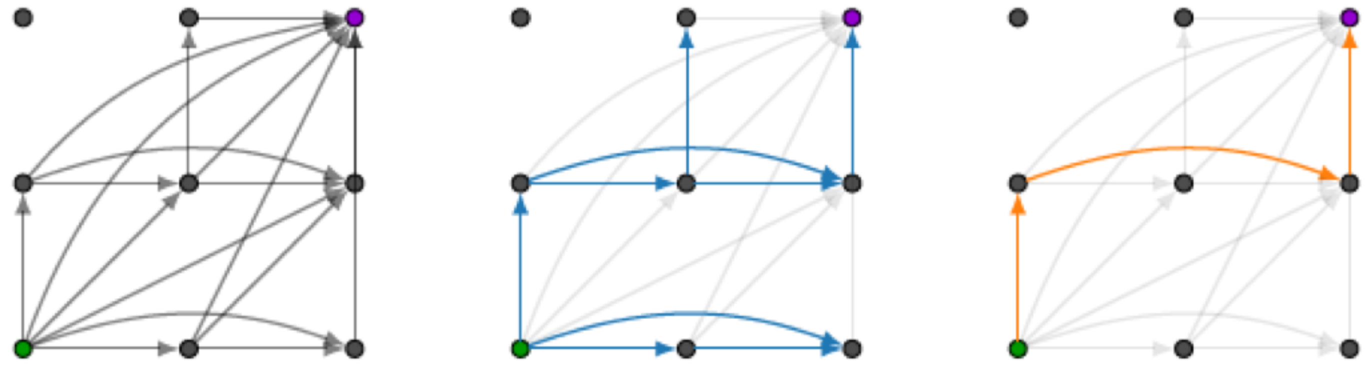
Outline of our approach

1. High-level planning: Decompose the specification into a graph of reach-avoid subtasks

2. Low-level policy learning: Learn policies + reach-avoid supermartingales for subtasks

On-demand low-level policy learning

3. Composition: Traverse the graph to compose low-level policies into a global policy, (while **composing formal guarantees** provided by formal certificates)



Theory

(Supermartingale certificates)

→

Neural control design

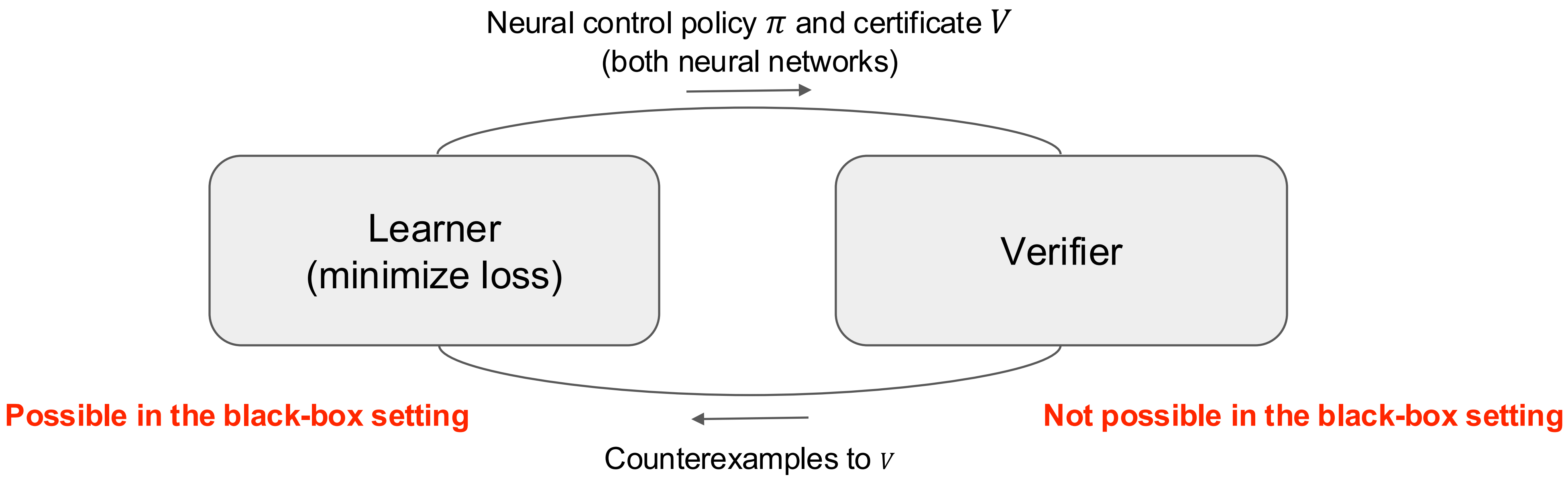
(Correct-by-construction design via learner-verifier loop)

→

Safe deployment

(Monitoring and shielding with certificates)

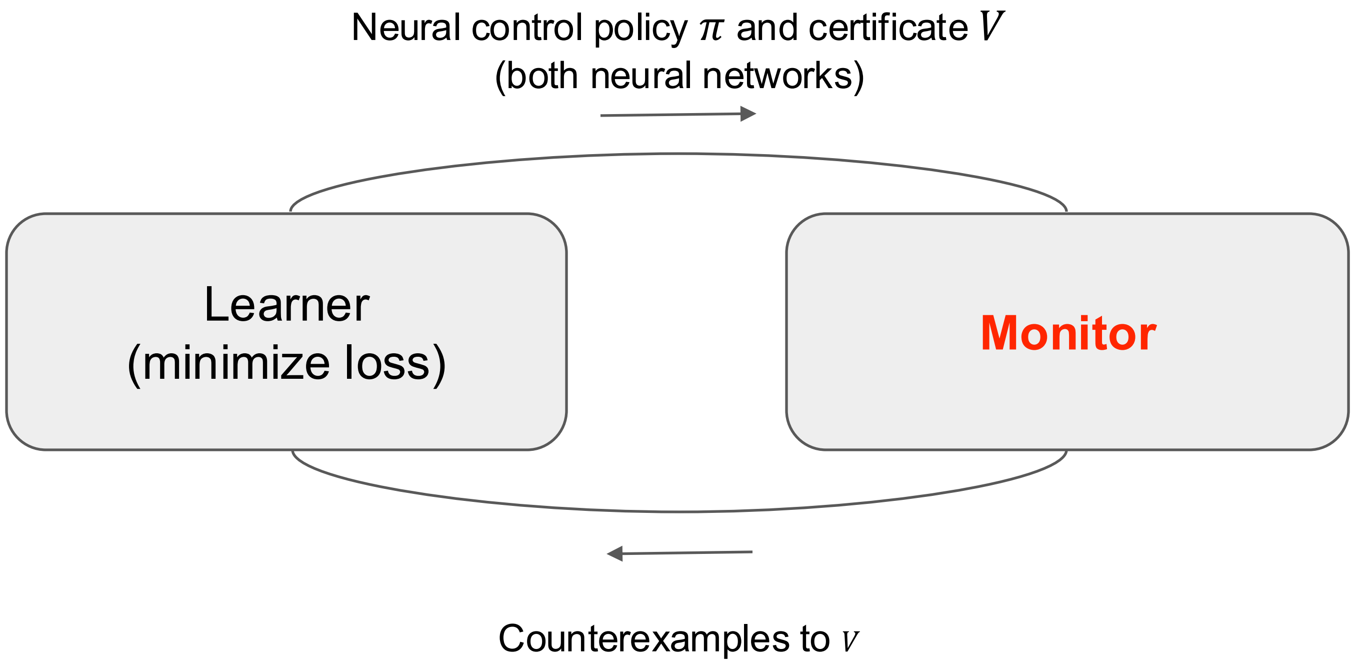
Runtime monitoring in black-box systems [AAAI'25]



How to certify correctness of learned controllers and certificates?

How to repair them if they are incorrect?

Runtime monitoring in black-box systems [AAAI’25]



Monitor certificate condition violations

➞ Use **certificate violations** as new training data

➞ **Repair** by re-learning

Runtime monitoring in black-box systems [AAAI'25]

	$\#D_{\text{NEW}}$	**SR (%)**	**BR (%)**	**NDR (%)**
Initialized	-	93.99	87.03	45.38
Baseline	878	96.61	-	-
CertPM	548	**99.13**	**100.00**	90.66
PredPM [0,0,-1]	146	99.06	**100.00**	90.11
PredPM [0,1,-5]	355	98.67	**100.00**	90.12
PredPM [2,2,0]	**1000**	99.09	**100.00**	**91.67**

Drone Environment [1]

(Navigate a drone among 1024 other drones)

8D encoding

Initialized: Neural controller and barrier certificate learned via SABLAS [1]

Baseline: Monitor and repair only with hard safety violations

CertPM + PredPM: Monitor and repair with hard safety violations + certificate violations

[1] Qin et al. "SABLAS: Learning Safe Control for Black-box Dynamical Systems". IEEE Robotics and Automation Letters 2022

Future directions

- Neural control under richer specifications (omega-regular specifications) [CAV’25]
- Scalability challenge [Work in progress]
- Compositional reasoning with respect to state space and specification [NeurIPS’23]
- Runtime monitoring and shielding of neural controllers [AAAI’25, TOSEM’26]

SMU SINGAPORE MANAGEMENT UNIVERSITY

Open positions

School of
Computing and Information Systems

CSRankings (2020-2025)
#57 in general CS,
#3 in software engineering, #25 in AI

❗Multiple **PhD positions** in computer science (fully funded)

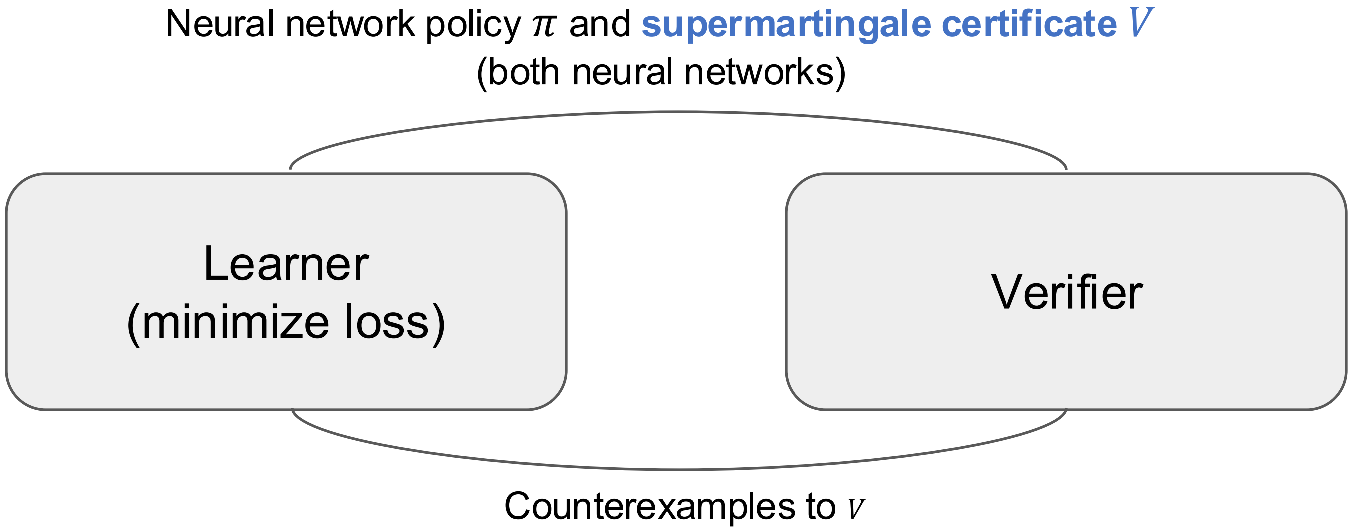
❗Multiple **Visiting Research Student** positions, 6 months

Possible topic include (but not limited to):

- Formal verification and synthesis in Markov models
- Probabilistic program verification
- Certification of neural control systems
- Runtime monitoring and safeguarding
- Safe reinforcement learning
- Runtime monitoring and safeguarding of LLM agents

Contact: dzikelic@smu.edu.sg
More details: https://djordjezikelic.github.io/openings/
PhD application deadline: Jan 31, 2026

Conclusion



1. **Full automation**
2. **General continuous systems**
3. **Hard formal guarantees**
4. **Long or even infinite-time horizon**
5. **Consideration of stochastic environment uncertainty**

Safe autonomy
(towards guaranteed safe AI)