Applying Data Collection Prioritization to Interpreting Neural Networks Using Polyhedral Abstract Domains

Michael Khalfin, Sandia National Labs

October 2025

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. SAND#2025-13346O.

Agenda

1. Uncertainty Quantification & Data Collection Prioritization
2. Neural Networks & Polyhedral Abstract Domains
3. Applying Data Collection Prioritization to Interpreting Neural Networks

1. Uncertainty Quantification & Data Collection Prioritization

(a) Overview of polyhedral inference models.

(b) Construction of taxonomies from sample data.

(c) Formulation of data collection prioritization using a risk parameter.

Intro & Motivation

Inference under Underspecification: Many OR problems (e.g., airlines, shipping, railroads) have incomplete data.

This leads to a **polyhedral inference model**:

$$\mathcal{P} = \{x \in \mathbb{R}^n : Ax \geq b\}$$

where every feasible x represents a possible state of the system.

Key Question: How do we *quantify* the uncertainty in such inference models?

Our approach: build a **taxonomy** that hierarchically categorizes the uncertainty.

The Marble Toy Problem

Consider a collection of marbles in three colors: blue (B), green (G), and yellow (Y).

We only have fragmentary information:

$$B + G \geq 8, \tag{1}$$

$$B + G + Y = 25, \tag{2}$$

$$B + Y \leq 21, \tag{3}$$

$$B \leq 10, \tag{4}$$

$$G \leq 10. \tag{5}$$

These constraints define an inference polytope.

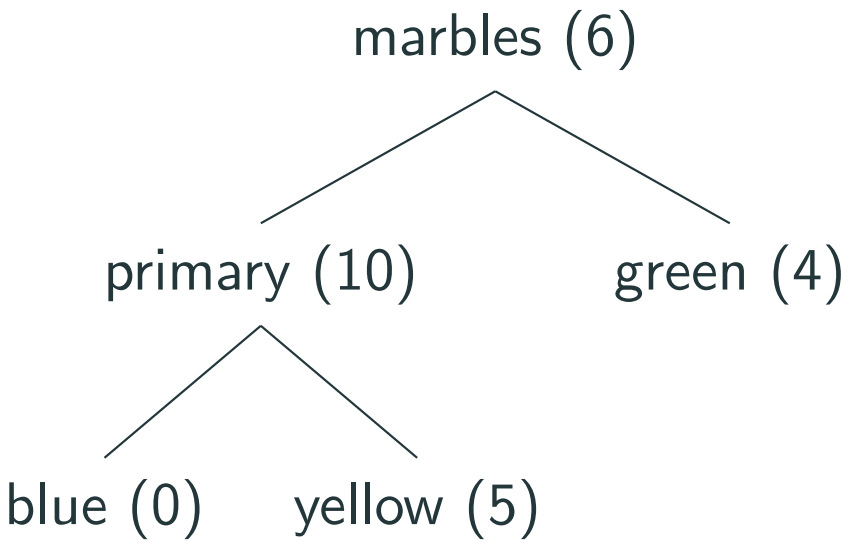
Each point in the polytope corresponds to a plausible marble collection.

Taxonomies as a Hierarchical Uncertainty Model

A **taxonomy** organizes uncertainty into levels.

By sampling from the inference polytope we can estimate the minimal counts at each node.

This hierarchical assignment captures both what we know and what remains ambiguous.



Constructing the Taxonomy

Procedure: Obtain a set S of sample points from the inference polytope.

For each taxonomic node g:

Compute the minimal count that can be *guaranteed* given the samples.

In formulas, we assign:

$$x_g = \min_{i \in S} \Big(\sum_{k \in K_g} N_{i,k} - \sum_{r \in D_g} x_r \Big)$$

where K_g are the colors under node g and D_g are its descendant groups.

The outcome is a taxonomy that summarizes the inference uncertainty.

Constructing the Taxonomy (Continued)

Equivalently we can solve the optimization problem:

$$\sum_{g \in \mathcal{G}} \ell(g) c_g \tag{1}$$

$$\text{s.t.} \quad c_g = \left\{ \begin{array}{ll} \min\limits_{x^g} & \sum\limits_{i \in M(g)} x_i^g \\ \text{s.t.} & A x^g \geq b \end{array} \right\} \qquad \forall g \in \mathcal{G} \tag{2}$$

$$\tag{3}$$

$\mathcal{G}$: Set of all groups in the taxonomy.

$\ell(g)$: Level of group g (larger $\ell(g)$ for more specific levels).

c_g: The minimal sum of assignments for group g.

$M(g)$: The set of indices corresponding to the measurement variables associated with group g.

x^g: The vector of decision variables associated with group g.

Applications

Airlines:

Dataset obtained from Bureau of Transportation Statistics.
Focused on a single day (January 15, 2011) and one airline (Southwest Airlines), resulting in 69 airports and 2,381 flights.
The model infers passenger itineraries and layovers based on fragmented and underspecified flight data.

Container Ships:

Dataset constructed from openly available vessel visit logs and commercially available TEUS data.
Involves 235,000 shipment records, later filtered to a dataset with 36,970 ports and 18,985 shipments.
The model infers the flow of commodities from limited info on shipment origins, transshipment points, and port visit logs.

Data Collection Prioritization: Motivation

Our taxonomy quantifies uncertainty in the inference polytope.

Key Idea: Additional data (new constraints) can shrink the polytope.

Question: Which extra data reduces uncertainty the most?

This guides optimal sensor placement and data acquisition.

Defining the Uncertainty Function $Q(y)$

We introduce a function $Q(y)$ that quantifies the residual uncertainty under extra data specified by y:

$$Q(y) = \min_{z,x} \sum_{g \in \mathcal{G}} \ell(g) \sum_{i \in M(g)} x_i^g \tag{4}$$

$$\text{s.t.} \quad Ax^g \geq b, \quad \forall\, g \in \mathcal{G}, \tag{5}$$

$$y_i \left(x_i^g - z_i \right) = 0, \quad \forall\, i \in I,\ g \in \mathcal{G}, \tag{6}$$

$$y_i \in \{0, 1\}, \quad \forall\, i \in I. \tag{7}$$

y_i: Binary indicator for whether extra data (or a sensor) is assigned to query i.

z_i: Used to fix x_i^g to be the same for all $g \in \mathcal{G}$.

Best- and Worst-Case Uncertainty

With a fixed data budget B (i.e. $\sum_{i \in I} y_i = B$), we define:

$$V_{\text{best}} = \min_{y:\, \sum_i y_i = B} Q(y), \qquad V_{\text{worst}} = \max_{y:\, \sum_i y_i = B} Q(y).$$

V_{best}: The lowest achievable uncertainty (optimistic case).

V_{worst}: The highest (conservative) uncertainty under the budget (worst-case).

Risk-Controlled Formulation

Our final data collection prioritization formulation balances optimism and conservatism via the parameter $\alpha \in [0, 1]$:

$$\min_{x} \quad Q(y) \quad \text{s.t.} \quad Q(y) \geq (1 - \alpha)\, V_{\text{best}} + \alpha\, V_{\text{worst}}, \quad \sum_{i \in I} y_i = B.$$

For a given budget B, we search for y such that the uncertainty $Q(y)$ exceeds a weighted average of the best and worst cases.

The α-lever enables decision-makers to adjust between risk-taking and risk-aversion in data collection.

Practical Example

For this example let $\alpha = 1$ (worst-case).

We focus on inferring the number of people west (N_w) and east (N_e) of the Mississippi at noon.

The total number is fixed:

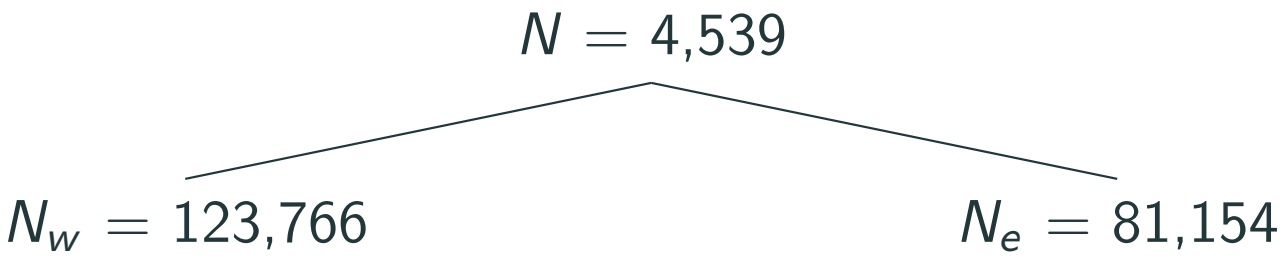
$$N_w + N_e = 209{,}459.$$

Each data query takes the form: how many people are at airport p at noon?

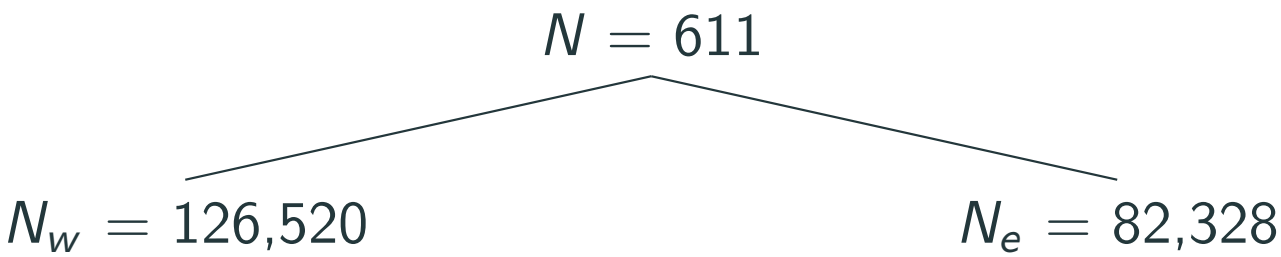
Budget: The number of queries we can deploy.

Taxonomy: Learning by Asking Questions

Budget $= 0...16$

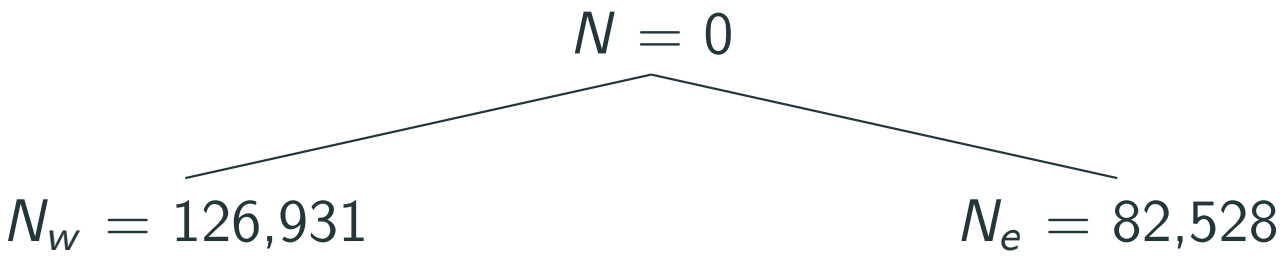


Budget $= 31$



Taxonomy: Learning by Asking Questions (Continued)

Budget $= 32$

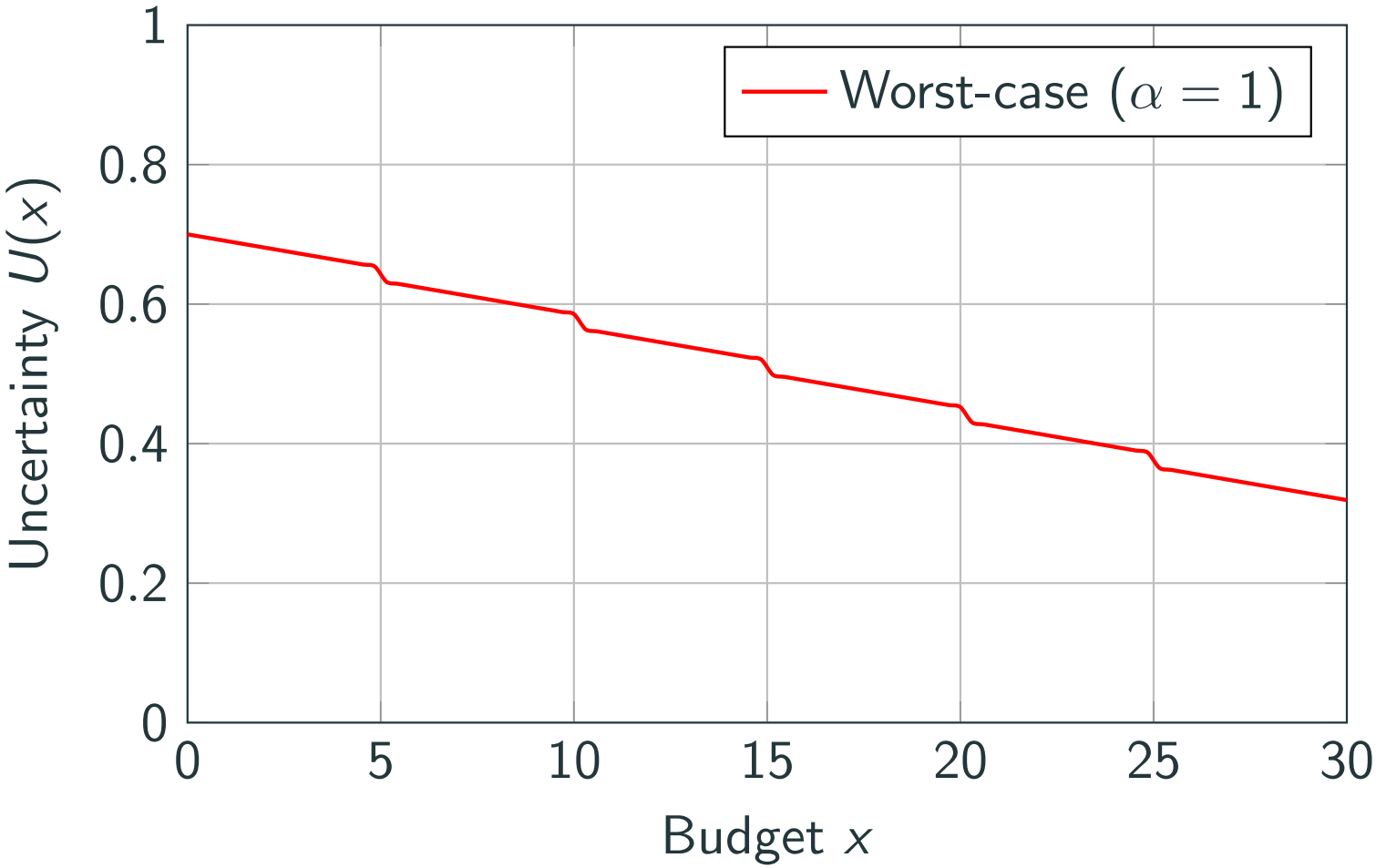


Although the total $N_w + N_e = 209{,}459$ remains constant, increasing the budget reduces the uncertainty N.

Even in the worst-case we still gain useful information on the passenger distribution.

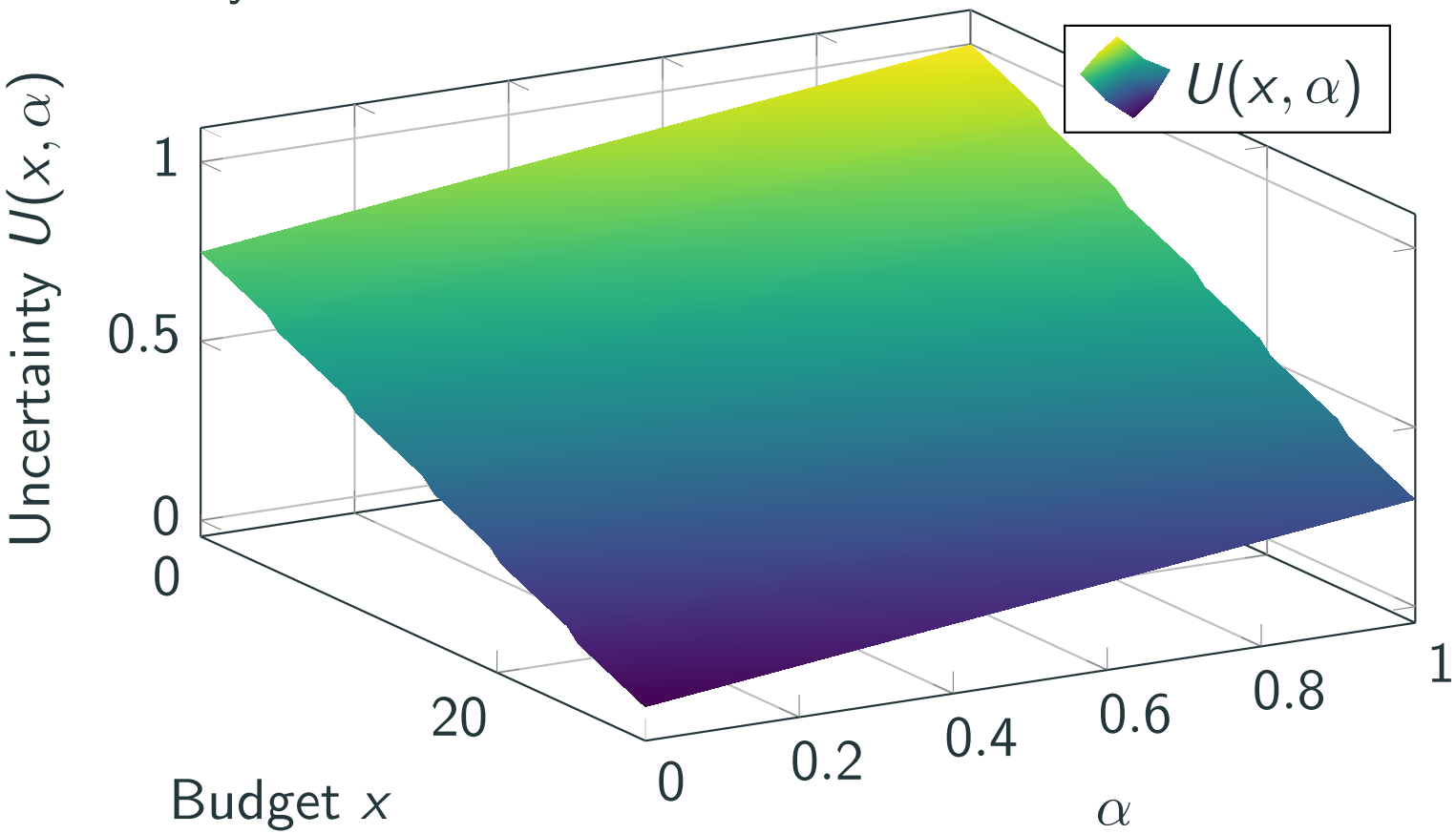
Budget vs. Worst-Case Uncertainty

Here is a sketch of how the worst-case uncertainty might vary with the budget. (Of course, depending on structure of the data)



Risk Spectrum as a 3D Manifold

If we work harder then we can characterize the entire spectrum of uncertainty!



2. Neural Networks for LLMs & Polyhedral Abstract Domains

(a) Introduction to feedforward architectures.

(b) Overview of the universal approximation theorem.

(c) Preview of polyhedral abstract domains for capturing network behavior.

Neural Network Architecture

A neural network implements a function $f : \mathbb{R}^n \to \mathbb{R}^m$ via composition.

For a feedforward network with L layers:

$$a^{(0)} = x, \quad a^{(l)} = \sigma\big(W^{(l)}\, a^{(l-1)} + b^{(l)}\big), \quad l = 1, \ldots, L,$$

where:

$W^{(l)}$ is the weight matrix,
$b^{(l)}$ is the bias vector,
$\sigma(\cdot)$ is the activation function (e.g., sigmoid, ReLU).

The output is given by $f(x) = a^{(L)}$.

Universal Approximation Theorem

Statement: A feedforward neural network with a single hidden layer containing a finite number of neurons can approximate any continuous function on a compact subset of $\mathbb{R}^n$ arbitrarily well.

Requirements:

- Activation function σ must be nonconstant, bounded, and continuous.
- The approximation is in the uniform norm.

Implication: Even simple architectures (with sufficiently many neurons) have immense expressive power.

Reality: We often use "deep" networks since they allow us to better **learn** how to approximate functions.

Polyhedral Abstract Domains: Overview

Abstract Interpretation: A method from program analysis that over-approximates the set of all possible values.

Polyhedral Domain: Represents sets as polyhedra:

$$\{x \in \mathbb{R}^n \mid Ax \leq b\},$$

where each row of A and b defines a linear constraint.

Purpose: Captures linear relationships and constraints among variables.

Used to safely over-approximate the behavior of systems—in our case, neural activations.

Applying Polyhedral Abstract Domains to Neural Networks

Idea: Track the set of possible activations at each layer as a polyhedron.

For an affine layer, if the input is approximated by

$$\mathcal{P}_{\text{in}} = \{x \mid A_{\text{in}} x \leq b_{\text{in}}\},$$

then the output $z = Wx + b$ is over-approximated by

$$\{z \mid A_{\text{in}} W^{-1}(z - b) \leq b_{\text{in}}\}.$$

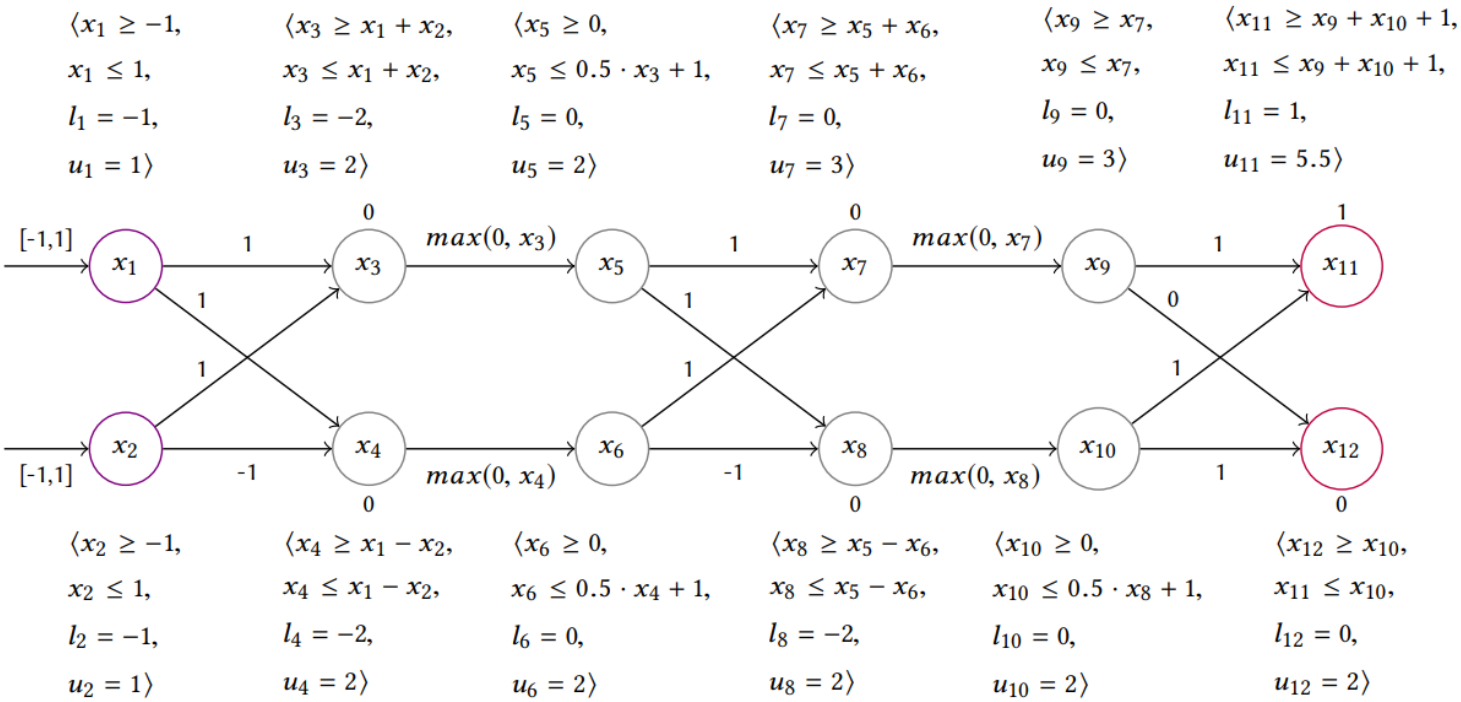
ReLU Layers: The ReLU function $\text{ReLU}(z) = \max(0, z)$ can be handled by splitting cases or using relaxations, yielding a new polyhedral over-approximation.

This framework also handles tanh, sigmoid, GELU, Softmax, etc.

3. Applying Data Collection Prioritization to Interpreting Neural Networks

(a) Integrating uncertainty quantification with neural network models.

Singh et al 2019



The figure illustrates how the abstract domain propagates linear constraints through a feedforward network.

From Polytope to Data Collection Prioritization

The overall system of inequalities derived from abstract transformers forms a polytope in the neural network's activation space.

Our data collection prioritization framework leverages this polytope, allowing us to choose additional constraints, and observe how the polytope shrinks.

This shrinking reflects increased certainty in the network's behavior, offering avenues for better understanding the high-dimensional latent spaces.

Thank you!

Questions?