When Machine Learning Isn’t Sure: Building Resilient ML-Based Computer Systems by Embracing Uncertainty

Varun Gohil
Nevena Stojkovic
Noman Bashir

MIT

Sundar Dev
Gaurang Upasani
David Lo

Google

Partha Ranganathan

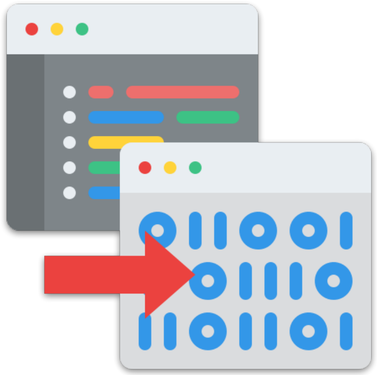
Christina Delimitrou

MIT

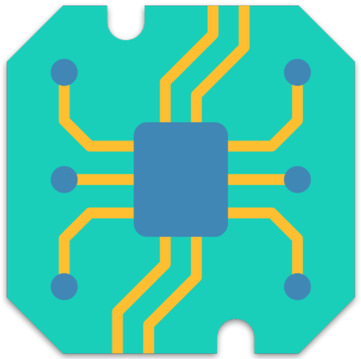
Machine Learning for Systems

Growing field with demonstrated effectiveness across every layer of systems stack.

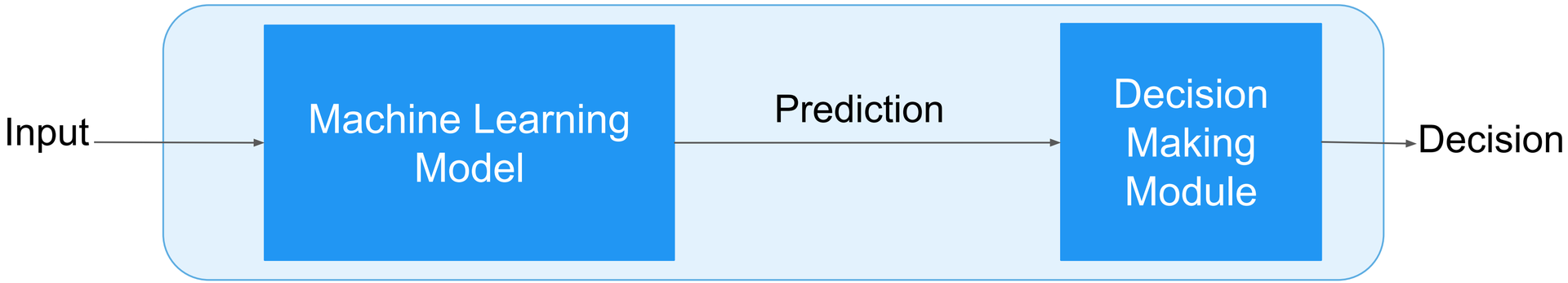
Datacenter Scheduling

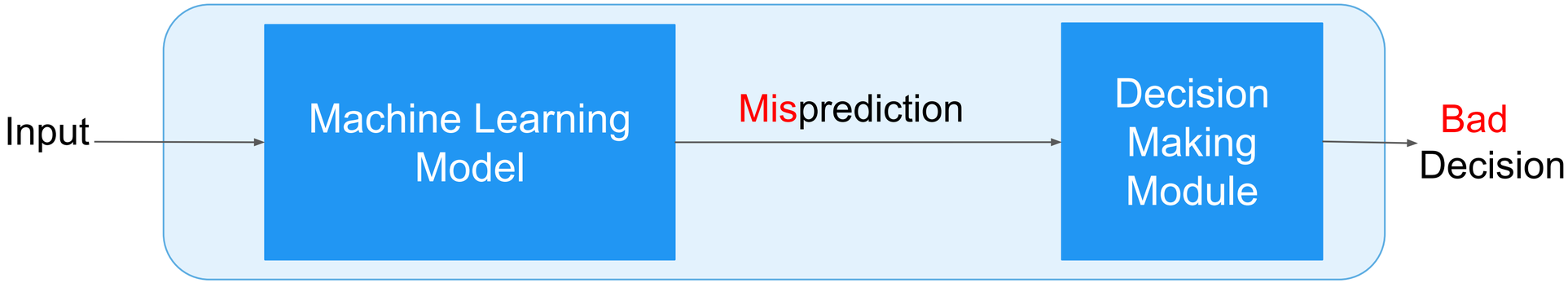


Compilers & Runtime

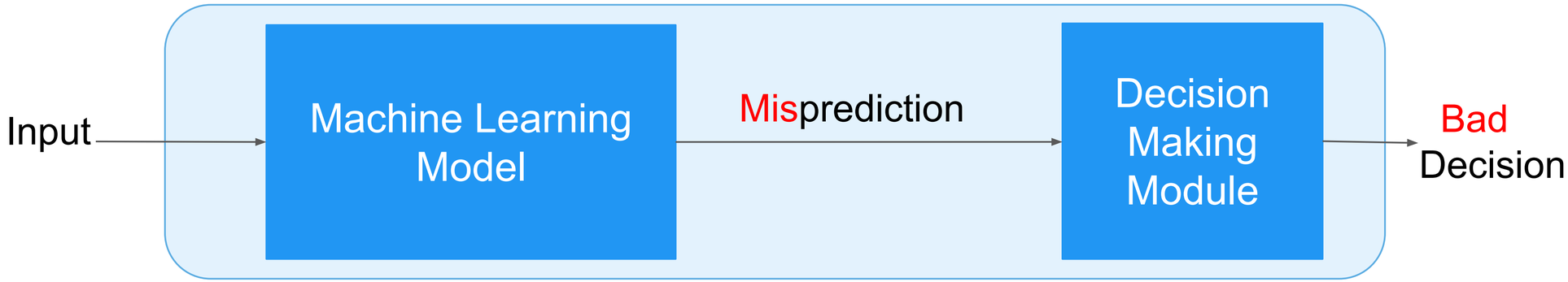


Chip Design


Input
Machine Learning Model
Prediction
Decision Making Module
Decision


Input
Machine Learning Model
Misprediction
Decision Making Module
Bad Decision

Mispredictions lead to poor decisions which hurt users and are difficult to debug.



Interpreting Deep Learning-Based Networking Systems, Meng et. al., SIGCOMM'20

Executive Summary

Problem:
ML model mispredictions lead to bad system decisions that hurt users.

Solution:
Workflow that **proactively uses uncertainty,** to guide model usage.

Contributions:

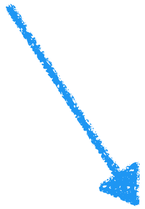
- Identify runtime and design tradeoffs offered by uncertainty estimators.
- Identify effective fallback mechanisms when the model is uncertain.
- Present guidelines to select optimal uncertainty estimator for a task.

When do models mispredict?

Models mispredict when they have poor generalizability.

When do models mispredict?

Models mispredict when they have poor generalizability.



Model's ability to predict accurately on data which is not independent and identically distributed as training data.

When do models mispredict?

Models mispredict when they have poor generalizability.

When test distribution is different from training distribution.

When do models mispredict?

Distribution changes can happen because of:

The Importance of Generalizability in Machine Learning for Systems, Gohil et. al., IEEE Computer Architecture Letters ‘24

When do models mispredict?

Distribution changes can happen because of:

- new hardware

The Importance of Generalizability in Machine Learning for Systems, Gohil et. al., IEEE Computer Architecture Letters ‘24

When do models mispredict?

Distribution changes can happen because of:

- new hardware
- changing software

The Importance of Generalizability in Machine Learning for Systems, Gohil et. al., IEEE Computer Architecture Letters ‘24

When do models mispredict?

Distribution changes can happen because of:

- new hardware
- changing software
- changing user load

The Importance of Generalizability in Machine Learning for Systems, Gohil et. al., IEEE Computer Architecture Letters ‘24

When do models mispredict?

Distribution changes can happen because of:

- new hardware
- changing software
- changing user load
- and many more reasons …

The Importance of Generalizability in Machine Learning for Systems, Gohil et. al., IEEE Computer Architecture Letters ‘24

When do models mispredict?

Impossible to have a model with perfect generalizability.

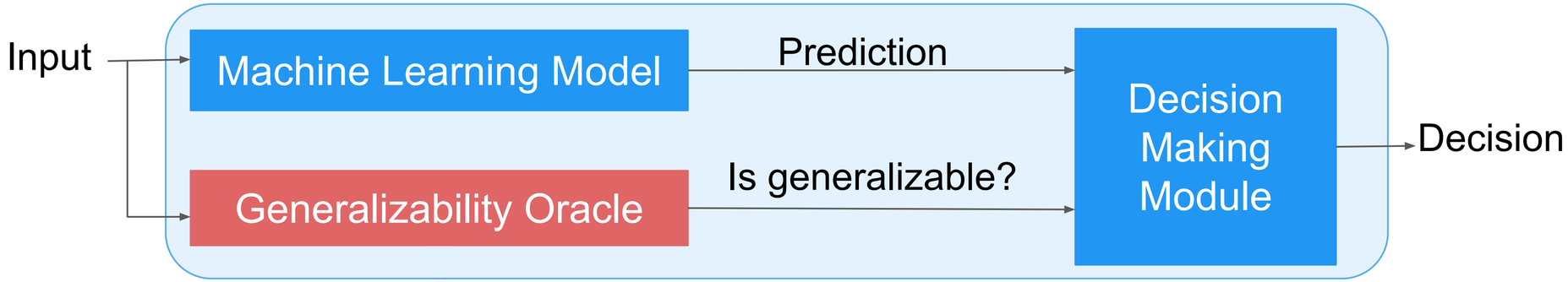
When do models mispredict?

Impossible to have a model with perfect generalizability.

Need to have a mechanism to deal with cases when model does not generalize.

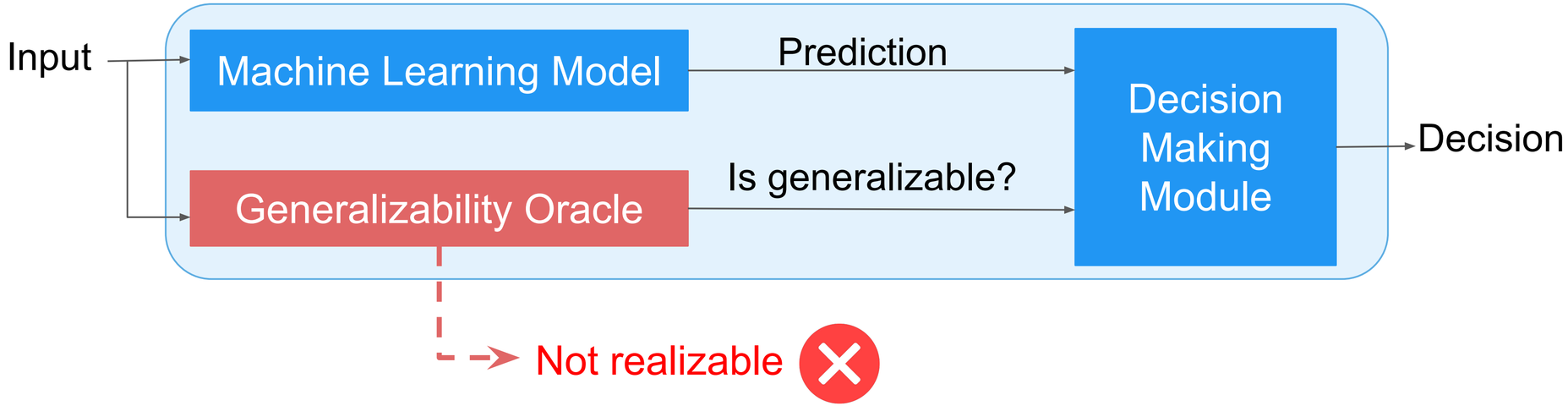
Workflow to handle misgeneralization

Proactively measure the model's **generalizability** on input data point and ignore model prediction if **generalizability is poor**.



Workflow to handle misgeneralization

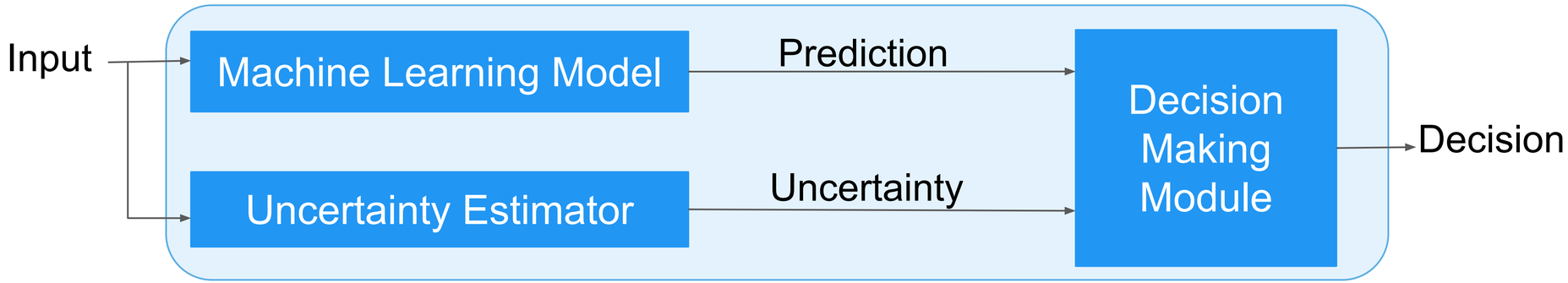
Proactively measure the model's **generalizability** on input data point and ignore model prediction if **generalizability is poor**.



Best measure of generalizability is accuracy which cannot be measured proactively.

Workflow to handle misgeneralization

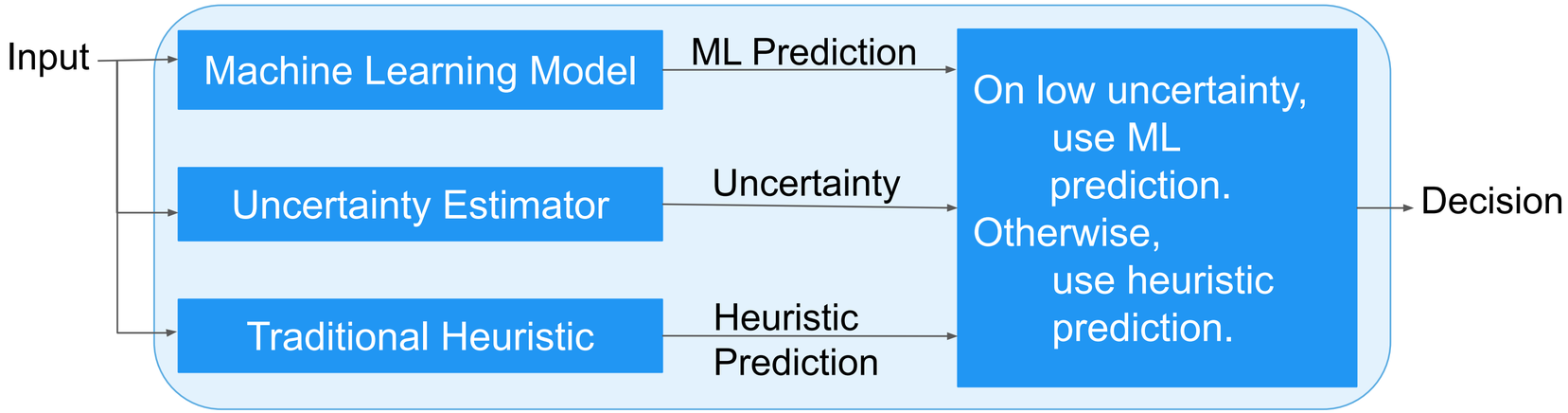
Proactively measure the model's **uncertainty** on input data point and ignore model prediction if **uncertainty is high**.



What to do when model is uncertain?

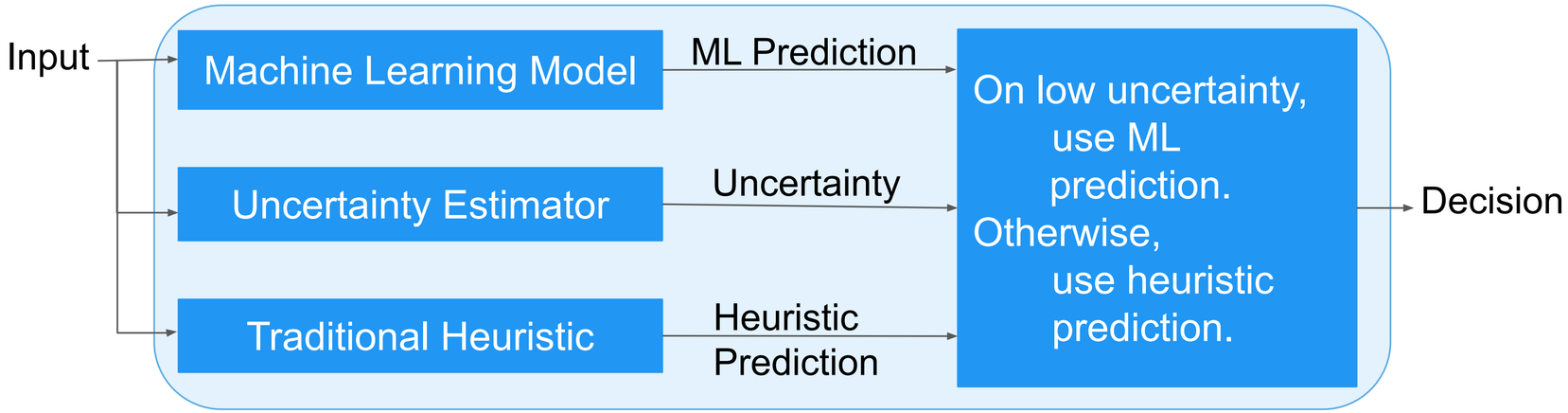
What to do when model is uncertain?

Fall back to a human or to non-ML based heuristics.



What to do when model is uncertain?

Fall back to a human or to non-ML based heuristics.

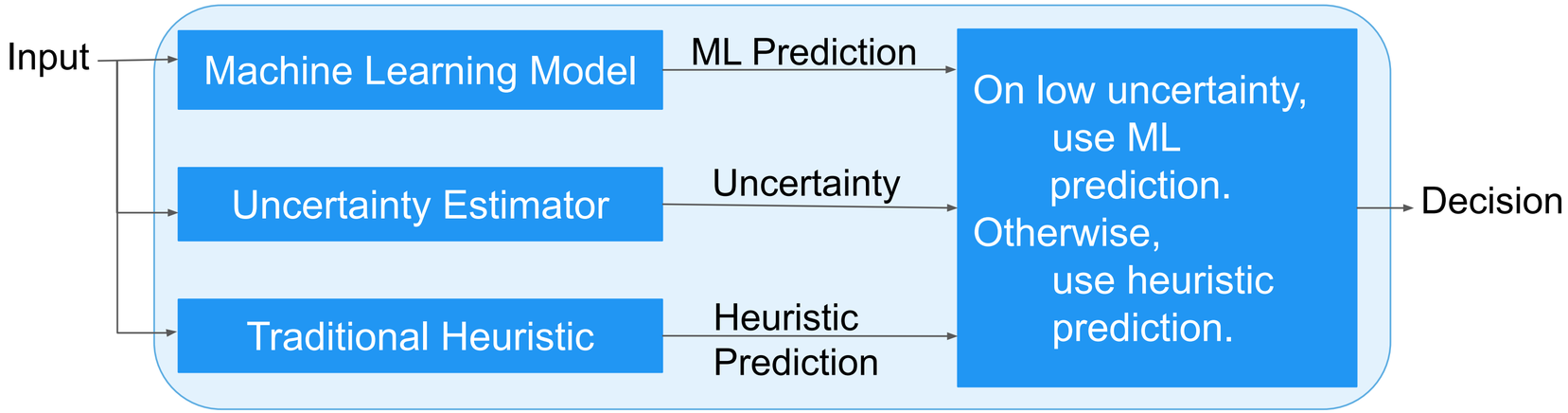


Falling back to heuristics helps because:

- Accuracy of heuristics does not degrade like the ML model on OOD data.

What to do when model is uncertain?

Fall back to a human or to non-ML based heuristics.

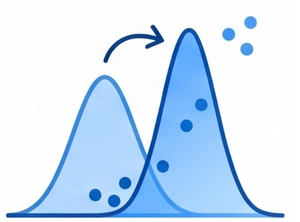


Falling back to heuristics helps because:

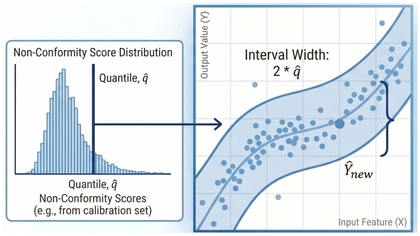
- Accuracy of heuristics does not degrade like the ML model on OOD data.
- Heuristics are interpretable.

Which uncertainty estimator to use?

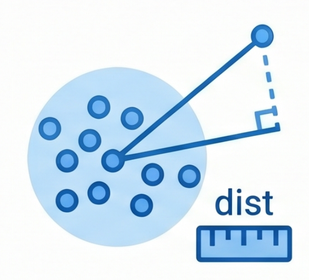
Which uncertainty estimator to use?



Bayesian
uncertainty estimator

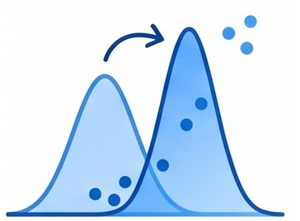


Conformal
Prediction

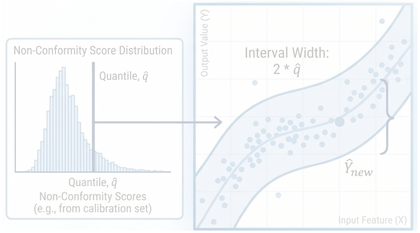


Distance-based
uncertainty estimator

Which uncertainty estimator to use?



Bayesian
uncertainty estimator



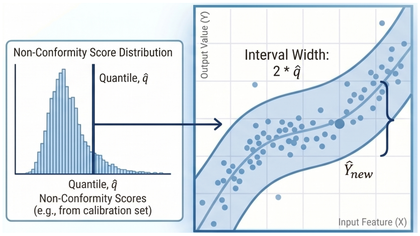
Conformal
Prediction

Distance-based
uncertainty estimator

- Outputs a distribution, instead of point estimate, whose standard deviation of can be interpreted as uncertainty.

Which uncertainty estimator to use?

Bayesian
uncertainty estimator



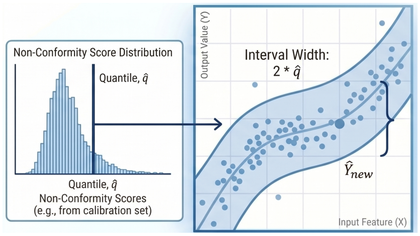
Conformal
Prediction

Distance-based
uncertainty estimator

- Outputs a prediction interval, instead of point estimate, whose width is the uncertainty estimate.

Which uncertainty estimator to use?

Bayesian
uncertainty estimator



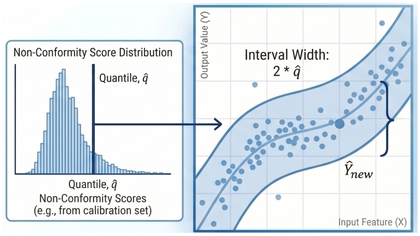
Conformal
Prediction

Distance-based
uncertainty estimator

- Outputs a prediction interval, instead of point estimate, whose width is the uncertainty estimate.
- Needs a calibration dataset.

Which uncertainty estimator to use?

Bayesian
uncertainty estimator



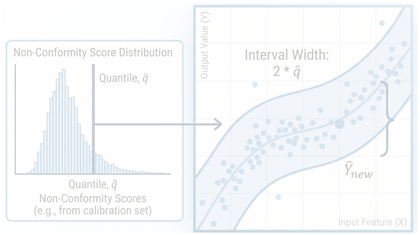
Conformal
Prediction

Distance-based
uncertainty estimator

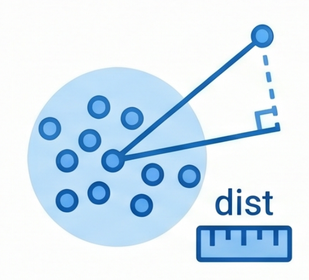
- Outputs a prediction interval, instead of point estimate, whose width is the uncertainty estimate.
- Needs a calibration dataset.
- Distribution-free and model-agnostic.

Which uncertainty estimator to use?

Bayesian
uncertainty estimator

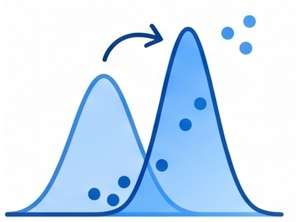


Conformal
Prediction

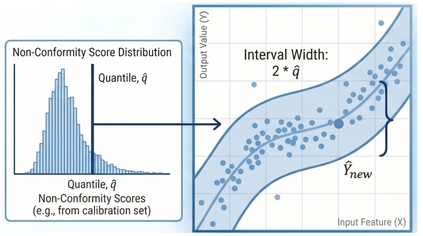


Distance-based
uncertainty estimator

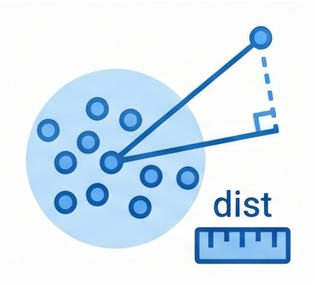
- Distance of input sample from training data distribution is interpreted as uncertainty.



Bayesian uncertainty estimator

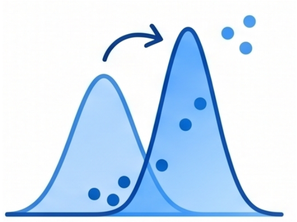


Conformal Prediction

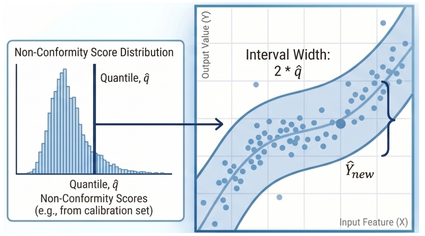


Distance-based uncertainty estimator

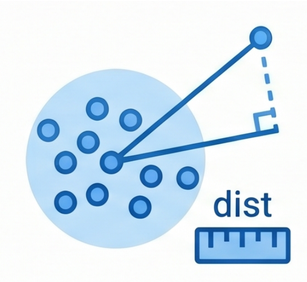
	Efficacy	High	High	Low-Medium
Runtime Tradeoffs	**Inference Latency**	High (ms-secs)	Medium (ms)	Low (µs)
	Memory Overhead	High (MB)	Medium-High	Low (bytes)
	Energy Usage	High	Low	Low
Design Tradeoffs	**Model-Agnostic**	No	Yes	Yes
	Requires Calibration Data	No	Yes	No
	Unit-Consistent	Yes	Yes	No



Bayesian uncertainty estimator

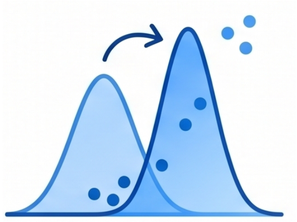


Conformal Prediction

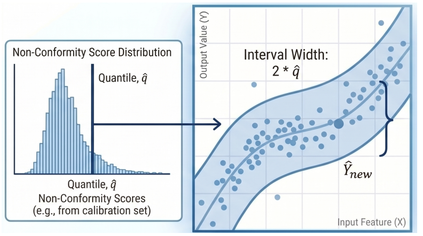


Distance-based uncertainty estimator

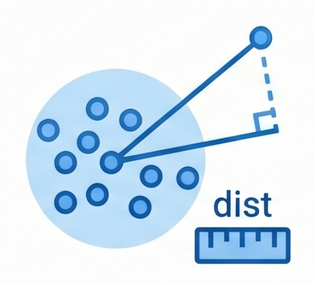
	Efficacy	High	High	Low-Medium
Runtime Tradeoffs	Inference Latency	High (ms-secs)	Medium (ms)	Low (µs)
	Memory Overhead	High (MB)	Medium-High	Low (bytes)
	Energy Usage	High	Low	Low
Design Tradeoffs	Model-Agnostic	No	Yes	Yes
	Requires Calibration Data	No	Yes	No
	Unit-Consistent	Yes	Yes	No



Bayesian uncertainty estimator

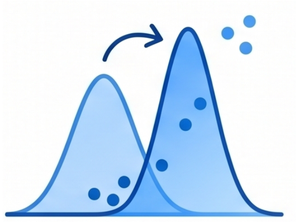


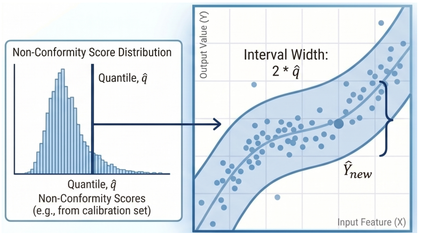
Conformal Prediction

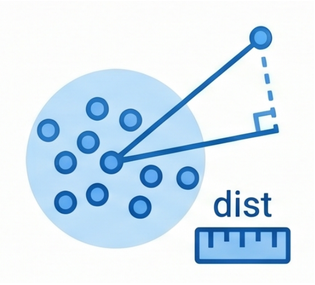


Distance-based uncertainty estimator

	Efficacy	High	High	Low-Medium
Runtime Tradeoffs	**Inference Latency**	High (ms-secs)	Medium (ms)	Low (µs)
	Memory Overhead	High (MB)	Medium-High	Low (bytes)
	Energy Usage	High	Low	Low
Design Tradeoffs	Model-Agnostic	No	Yes	Yes
	Requires Calibration Data	No	Yes	No
	Unit-Consistent	Yes	Yes	No







Bayesian uncertainty estimator

Conformal Prediction

Distance-based uncertainty estimator

	Efficacy	High	High	Low-Medium
Runtime Tradeoffs	Inference Latency	High (ms-secs)	Medium (ms)	Low (µs)
	Memory Overhead	High (MB)	Medium-High	Low (bytes)
	Energy Usage	High	Low	Low
Design Tradeoffs	**Model-Agnostic**	No	Yes	Yes
	Requires Calibration Data	No	Yes	No
	Unit-Consistent	Yes	Yes	No

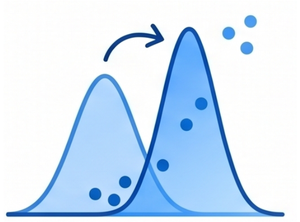
Bayesian uncertainty estimator

Conformal Prediction

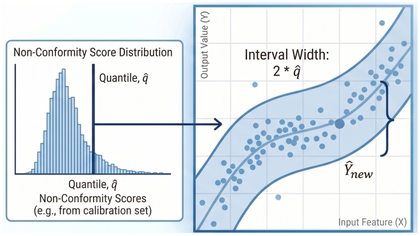
Distance-based uncertainty estimator

Uncertainty estimators offer a broad range of runtime and design tradeoffs.

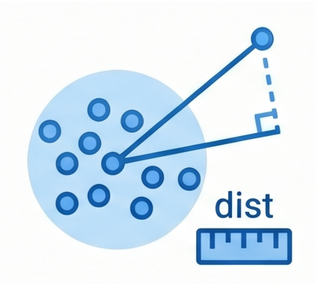
	Energy Usage	High	Low	Low
Design Tradeoffs	Model-Agnostic	No	Yes	Yes
	Requires Calibration Data	No	Yes	No
	Unit-Consistent	Yes	Yes	No



Bayesian
uncertainty estimator



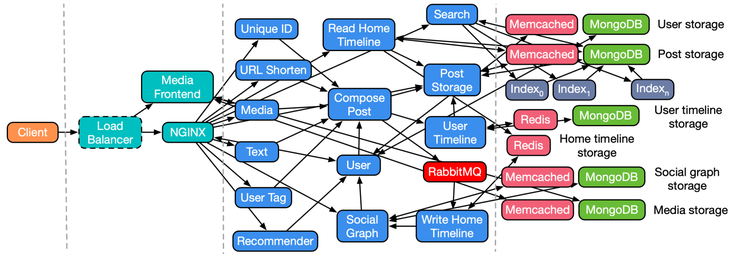
Conformal
Prediction



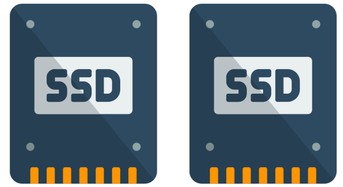
Distance-based
uncertainty estimator

×

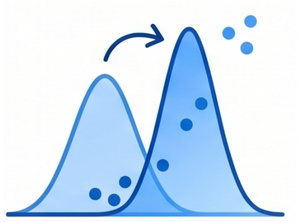
Server Resource
Capacity Provisioning



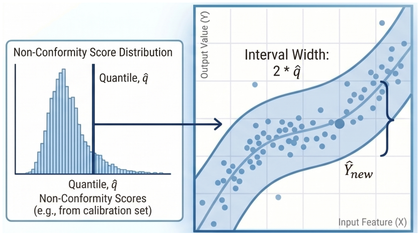
Microservice
Resource Management



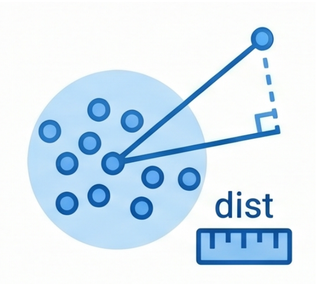
Storage
I/O Routing



Bayesian
uncertainty estimator



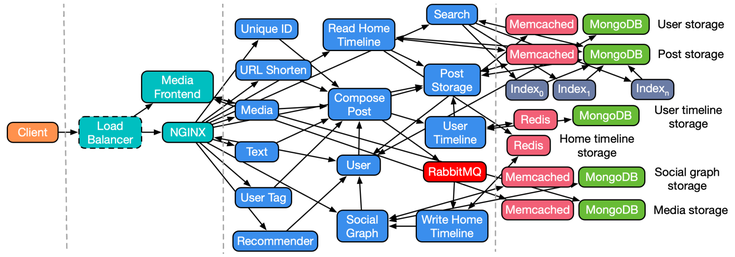
Conformal
Prediction



Distance-based
uncertainty estimator

×

Server Resource
Capacity Provisioning



Microservice
Resource Management

Storage
I/O Routing

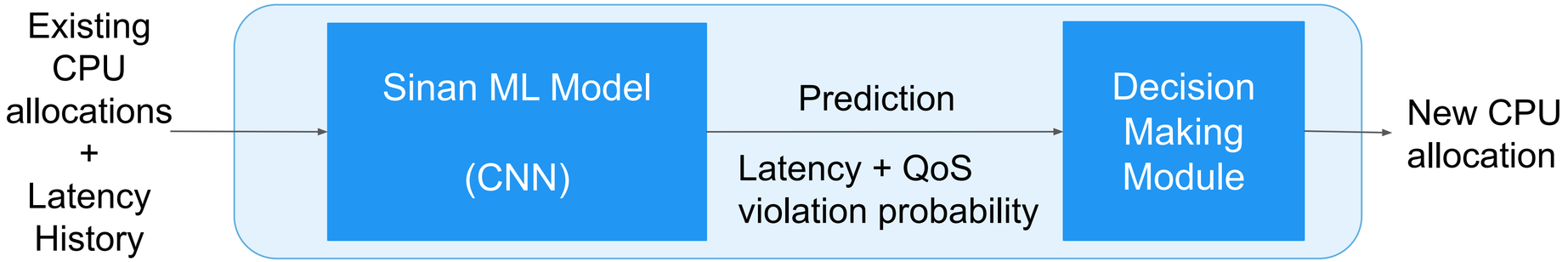
Sinan: Microservice Resource Management

Task: Minimize CPUs allocated to microservices while avoiding QoS violations.

Sinan: ML-Based & QoS-Aware Resource Management for Cloud Microservices, Zhang et. al., ASPLOS'21

Sinan: Microservice Resource Management

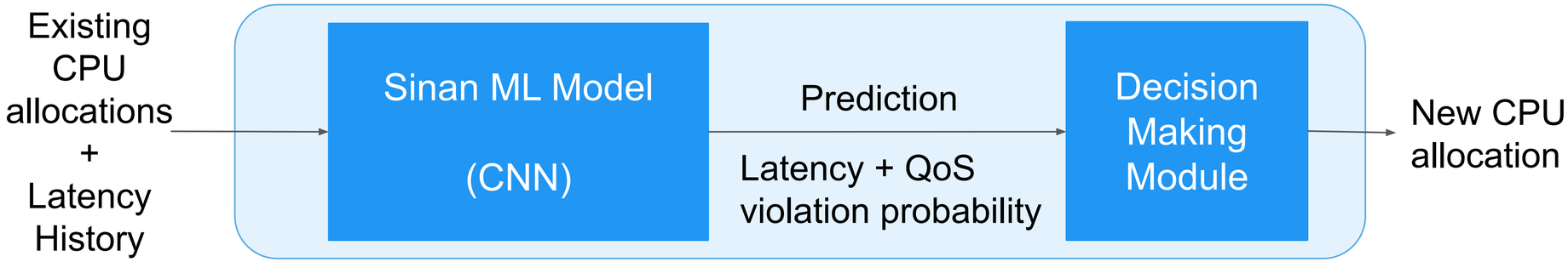
Task: Minimize CPUs allocated to microservices while avoiding QoS violations.



Sinan: ML-Based & QoS-Aware Resource Management for Cloud Microservices, Zhang et. al., ASPLOS'21

Sinan: Microservice Resource Management

Task: Minimize CPUs allocated to microservices while avoiding QoS violations.

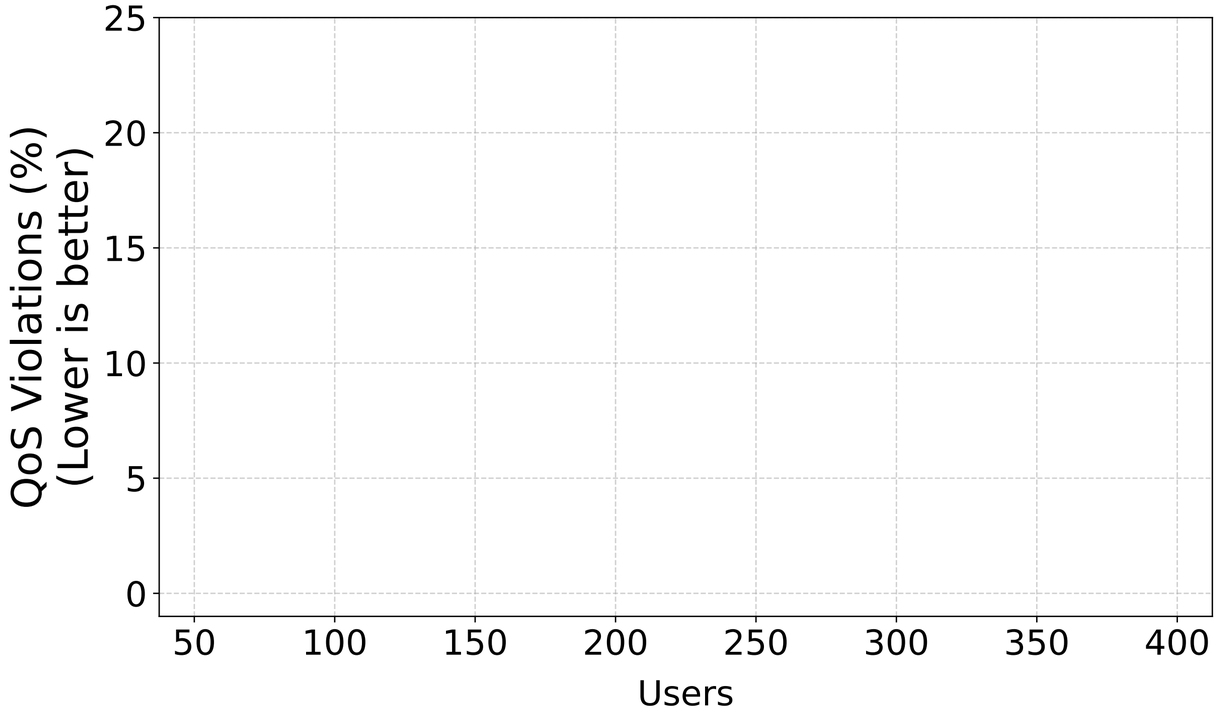


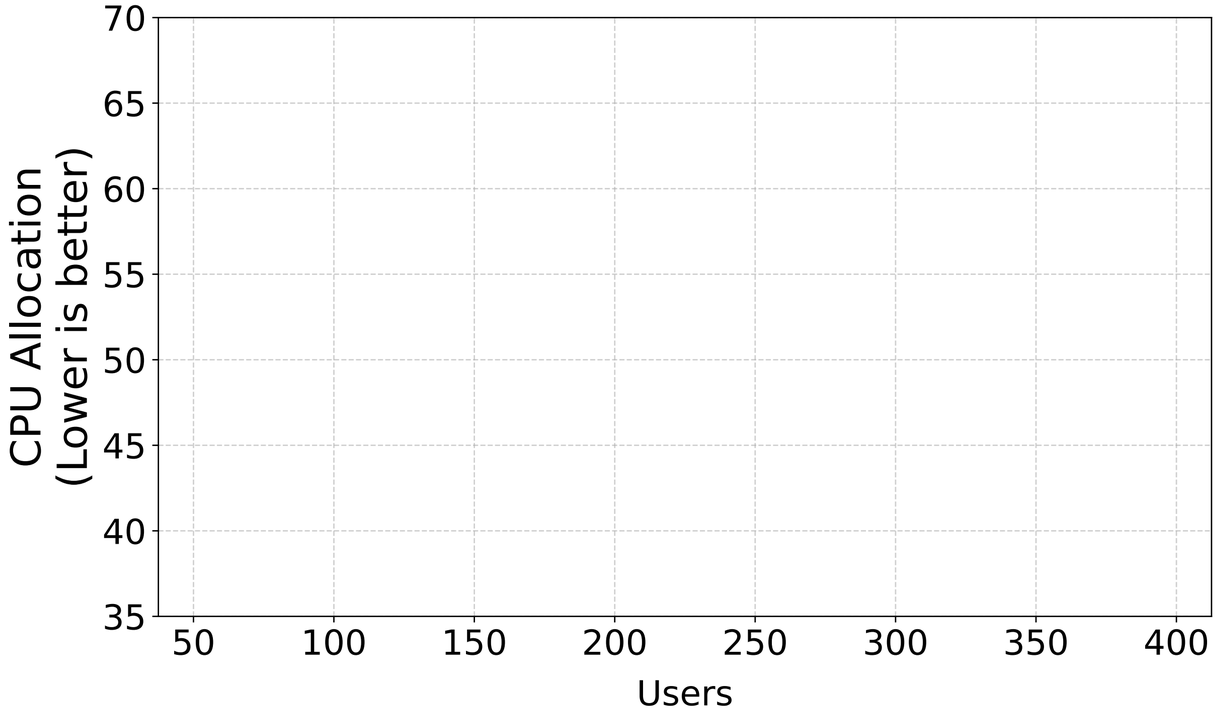
Setup:

- Using social network application from DeathStarBench deployed on a 7-node cluster.

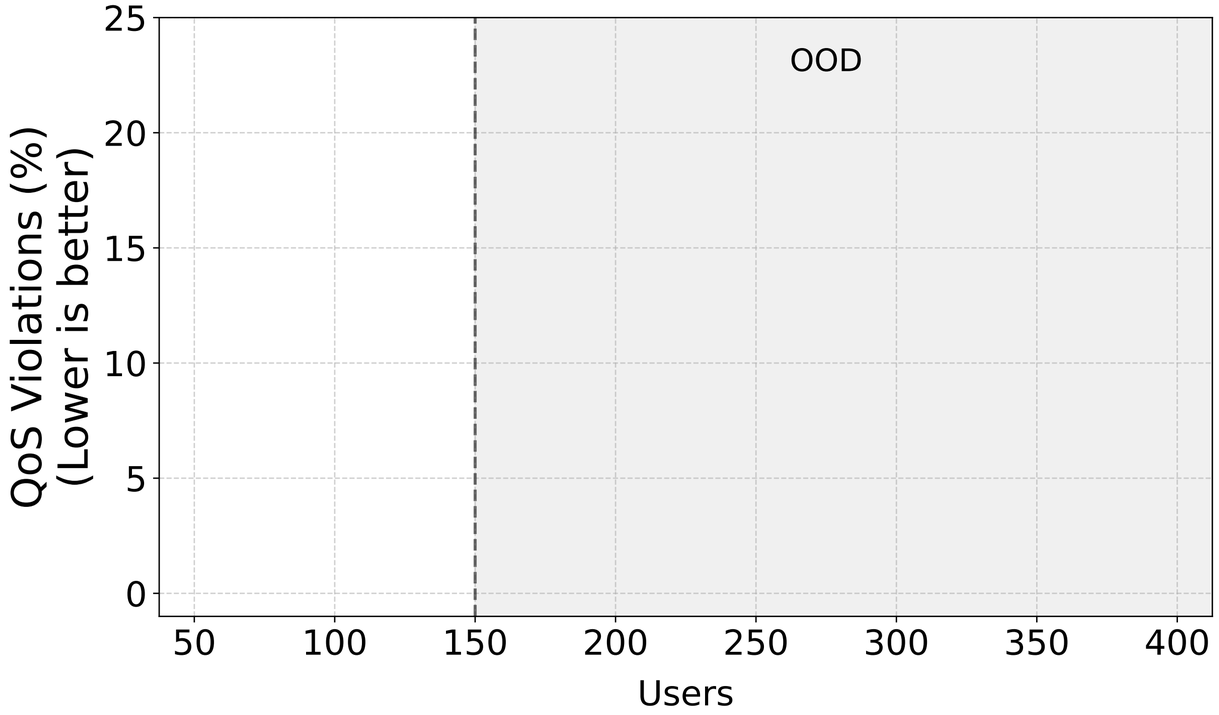
Sinan: ML-Based & QoS-Aware Resource Management for Cloud Microservices, Zhang et. al., ASPLOS'21

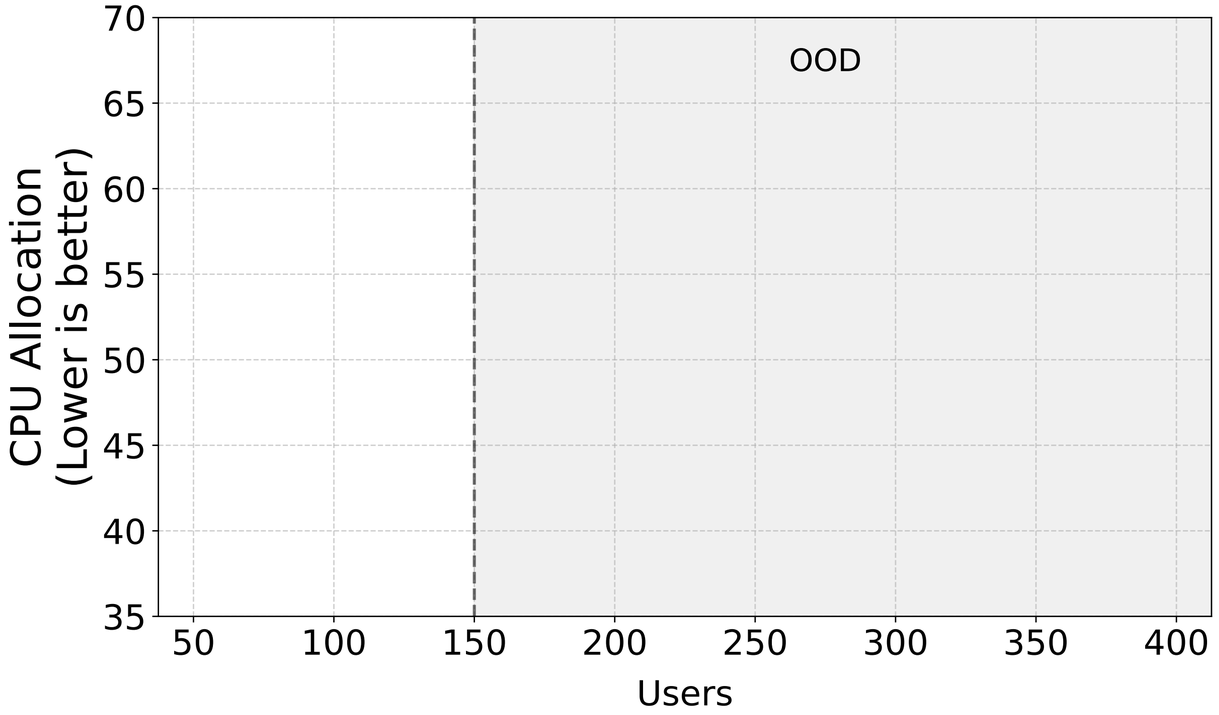
Sinan: Microservice Resource Management



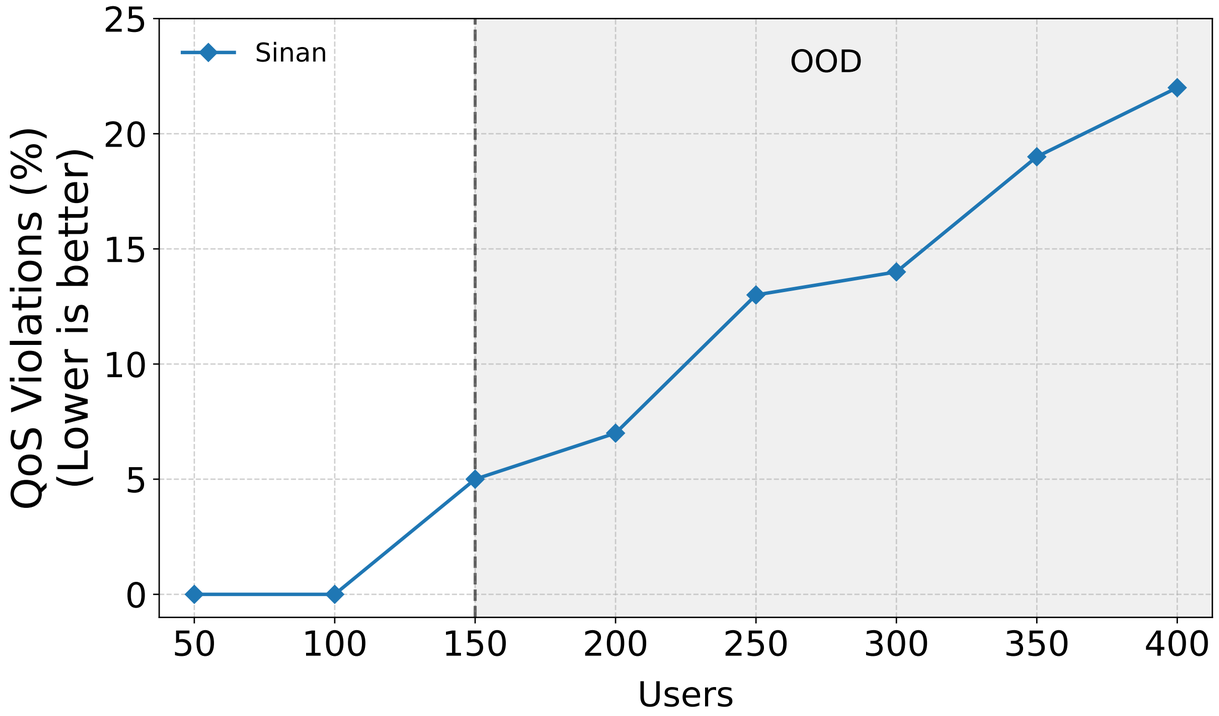


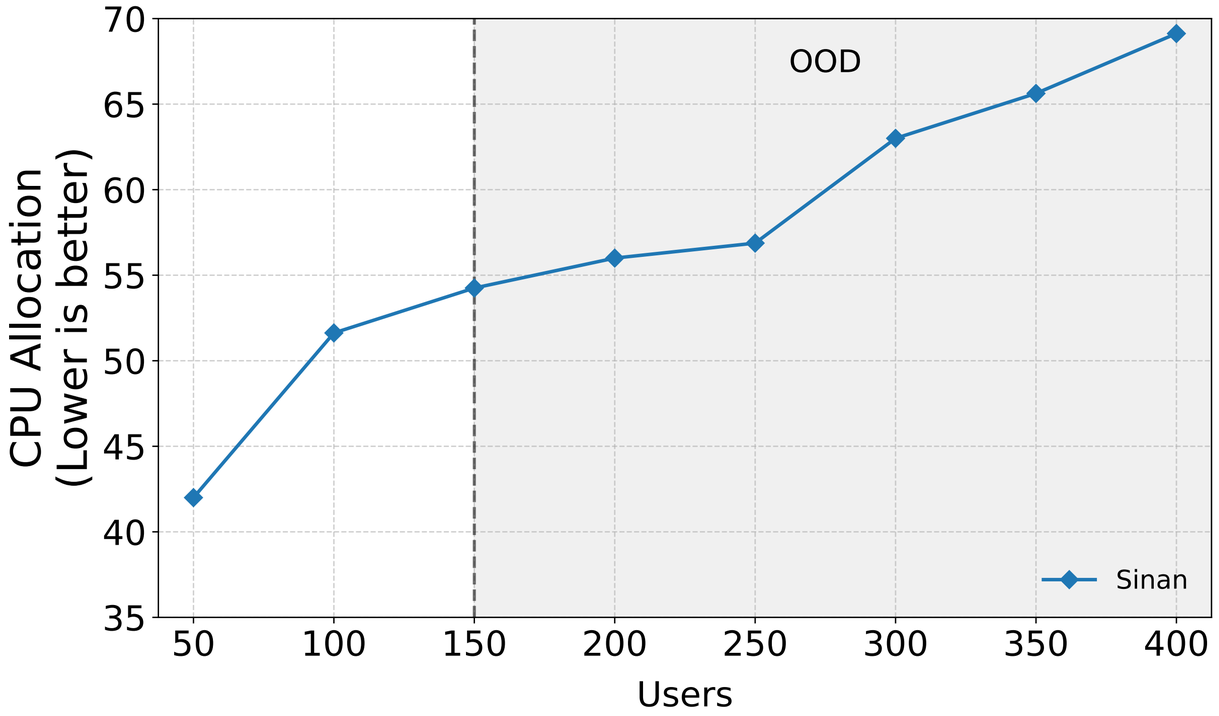
Sinan: Microservice Resource Management





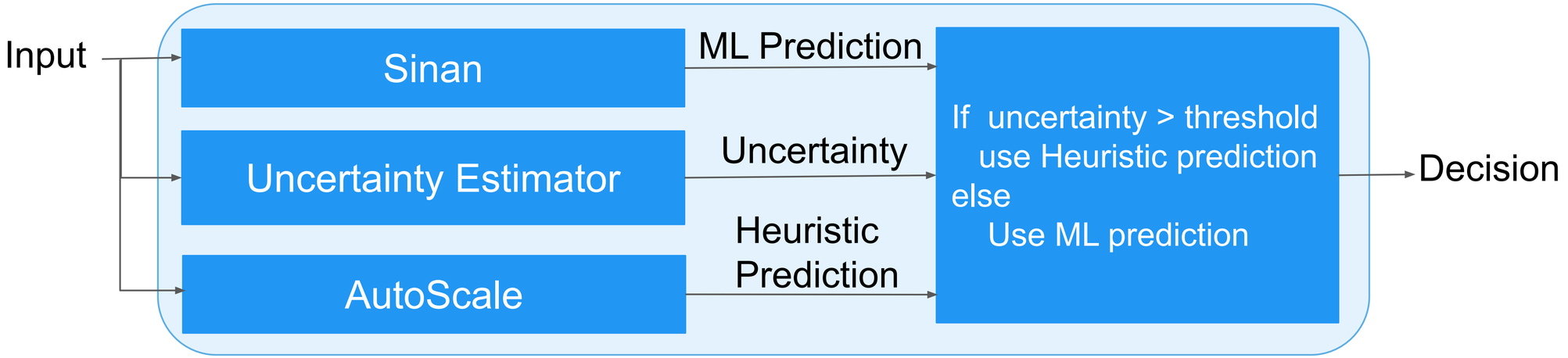
Sinan: Microservice Resource Management



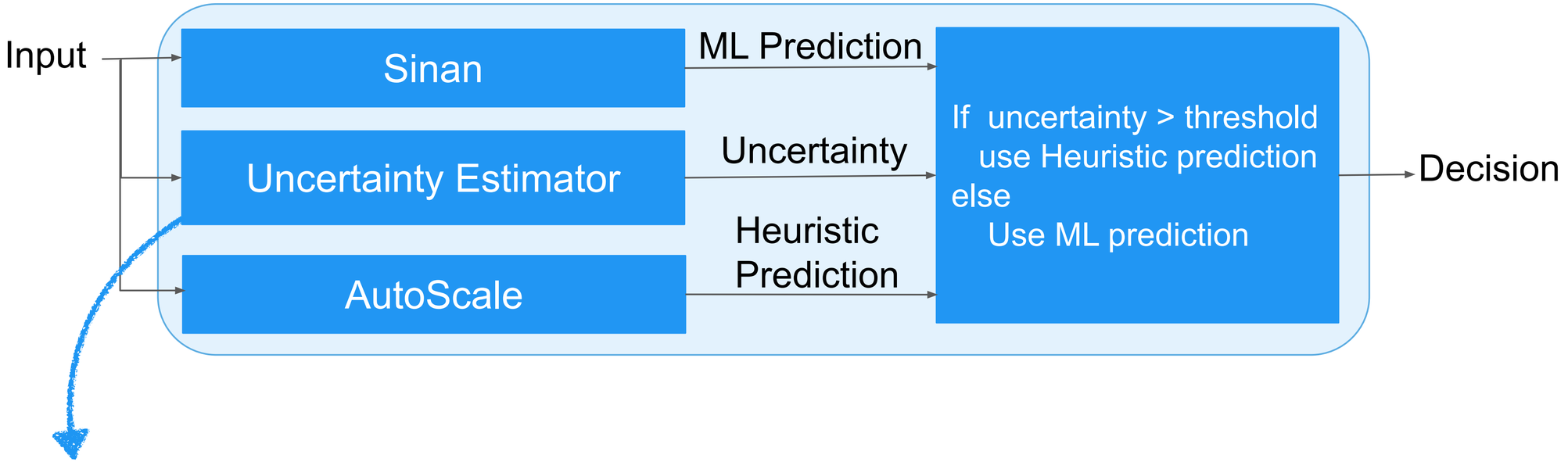


ML-driven Sinan suffers in OOD regimes.

Sinan: Microservice Resource Management

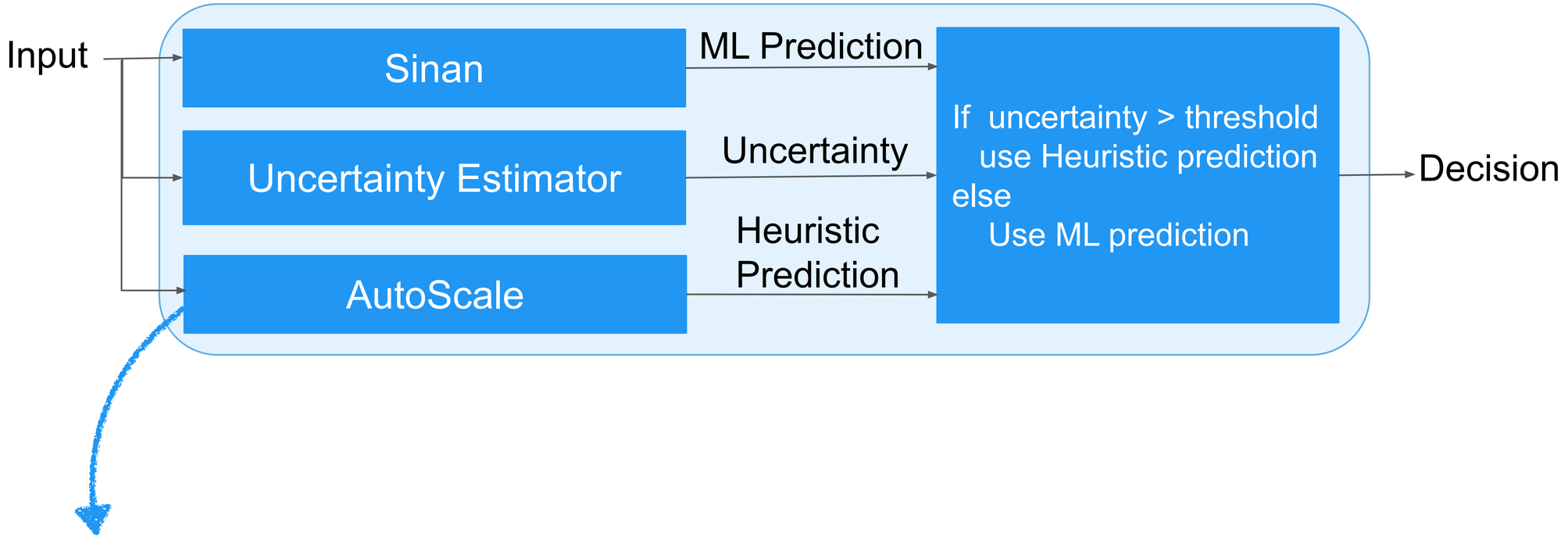


Sinan: Microservice Resource Management



Run uncertainty estimator in parallel to ML model to mask its latency.

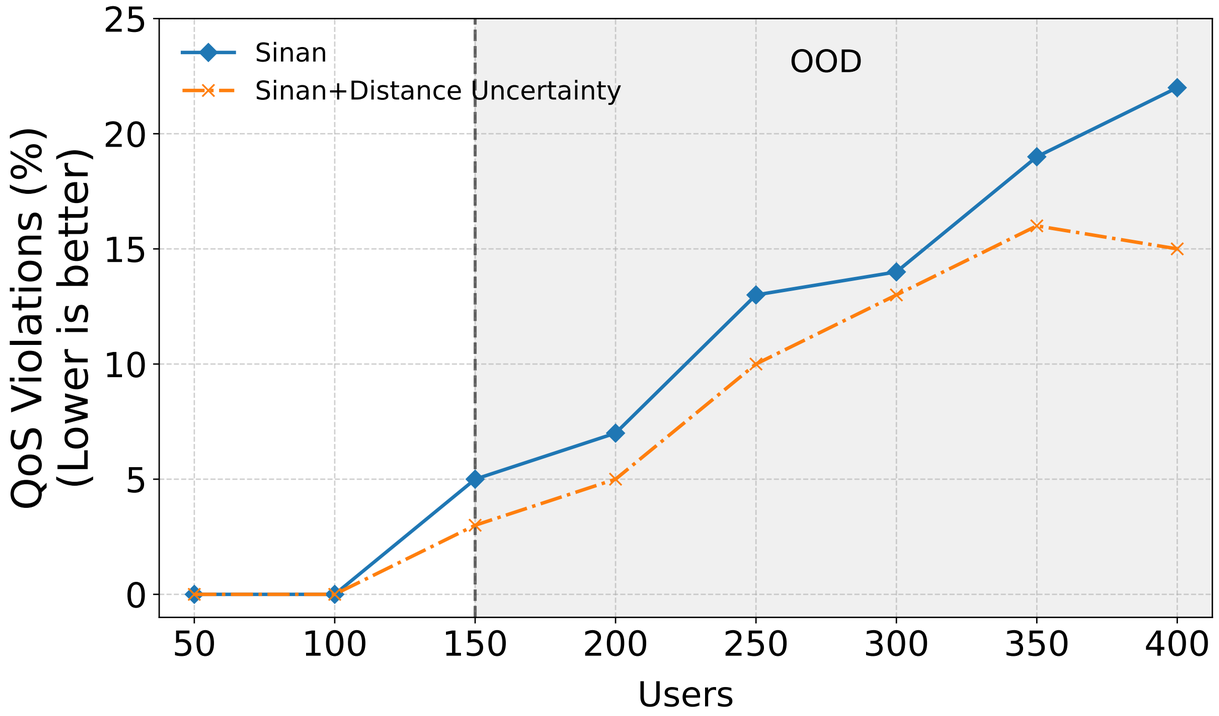
Sinan: Microservice Resource Management

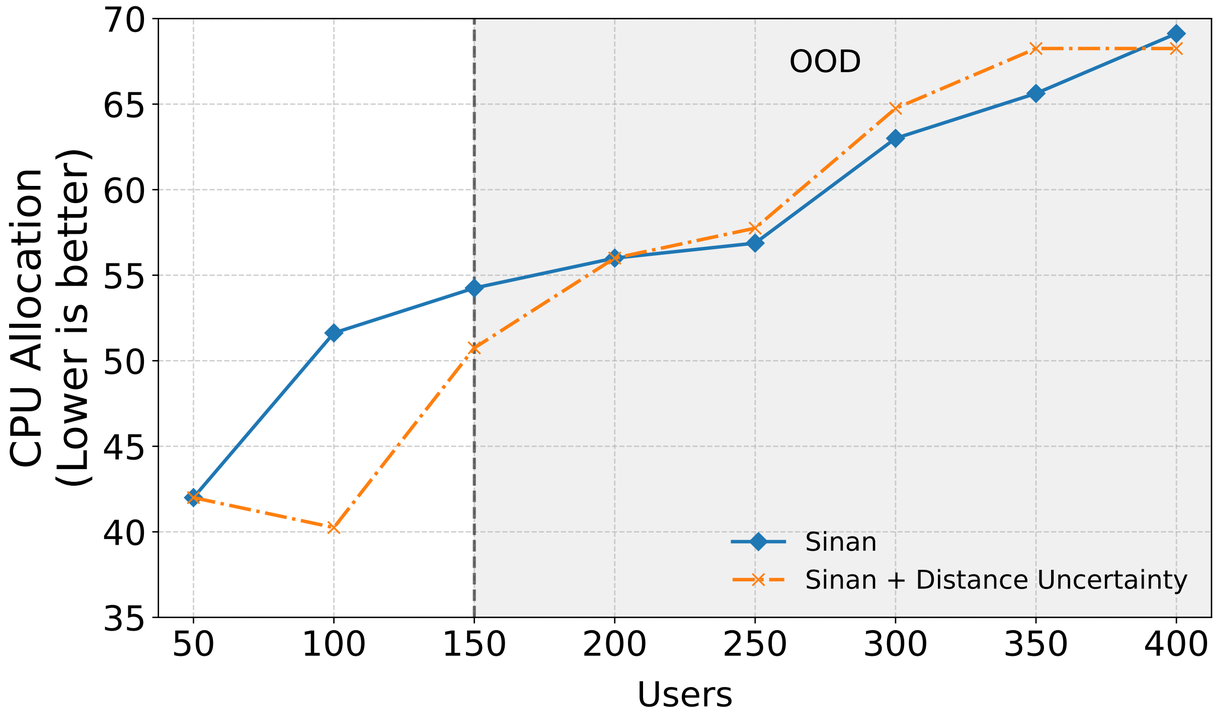


Fall back to AutoScale, a heuristic that allocates CPUs based on CPU utilization.

Task's latency constraint and heuristic's lightweight nature allows running it in parallel.

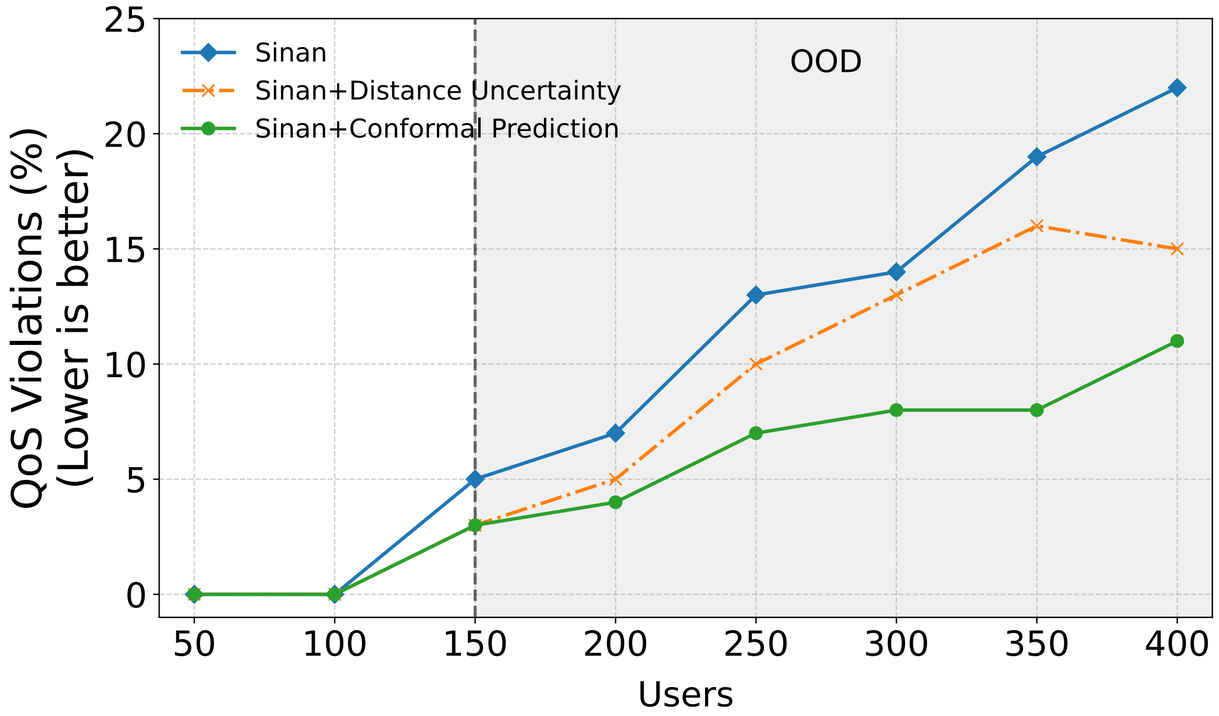
Sinan: Microservice Resource Management

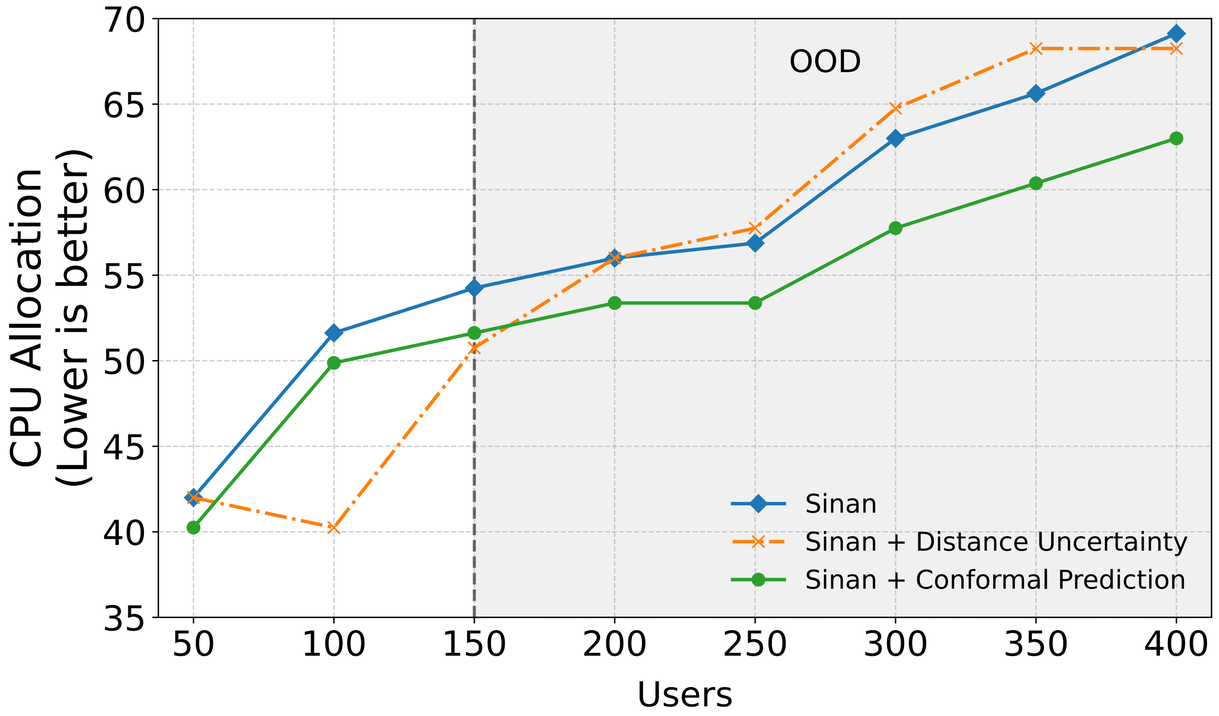




With distance-based uncertainty, our workflow has fewer QoS violations than Sinan with similar CPU allocations.

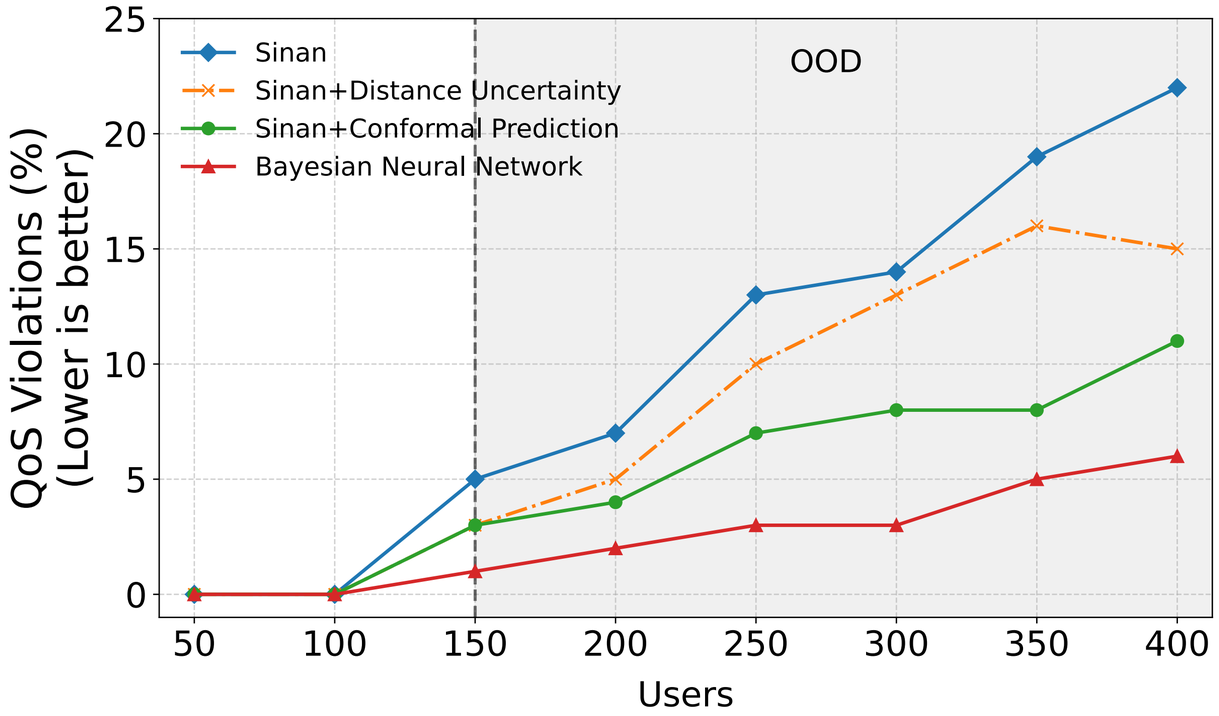
Sinan: Microservice Resource Management

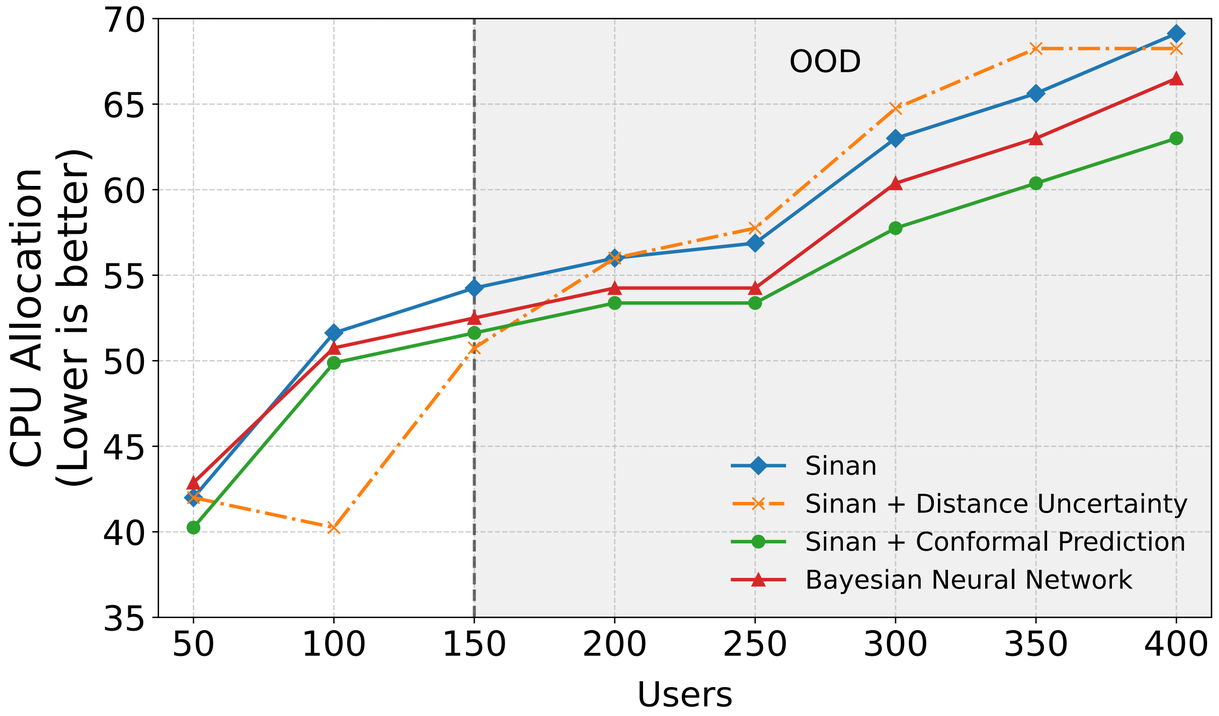




With conformal prediction, our workflow has fewer QoS violations than Sinan with lower CPU allocations.

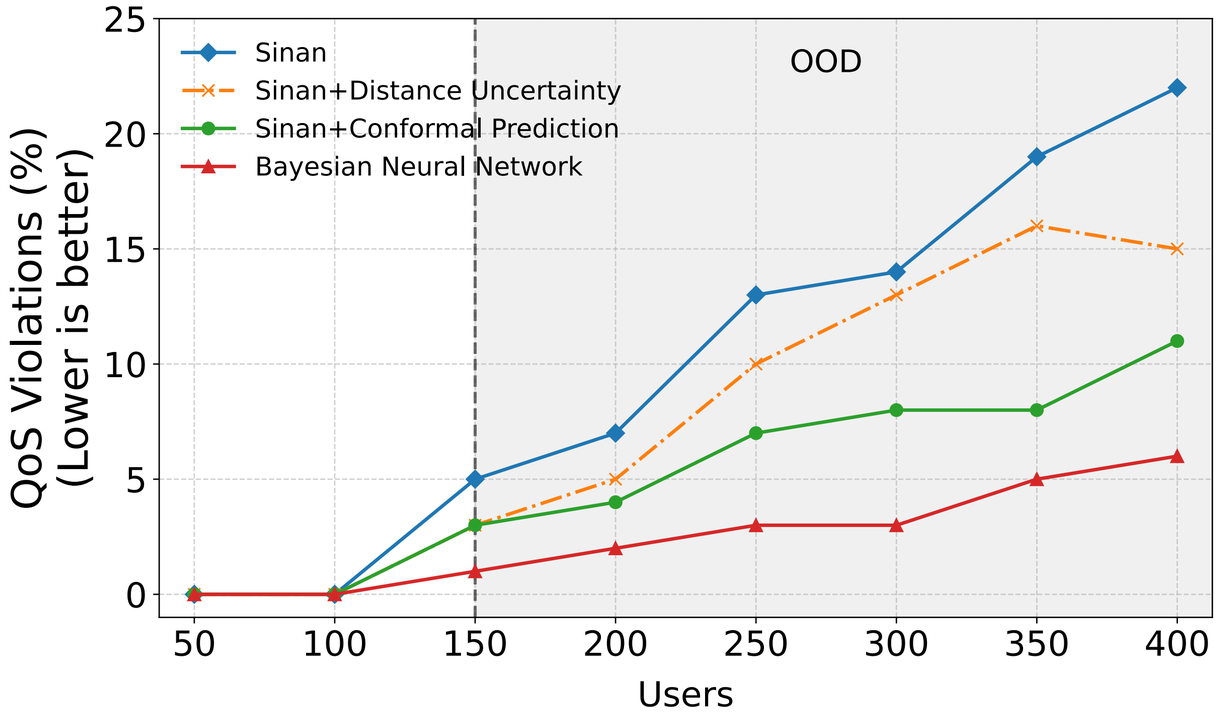
Sinan: Microservice Resource Management

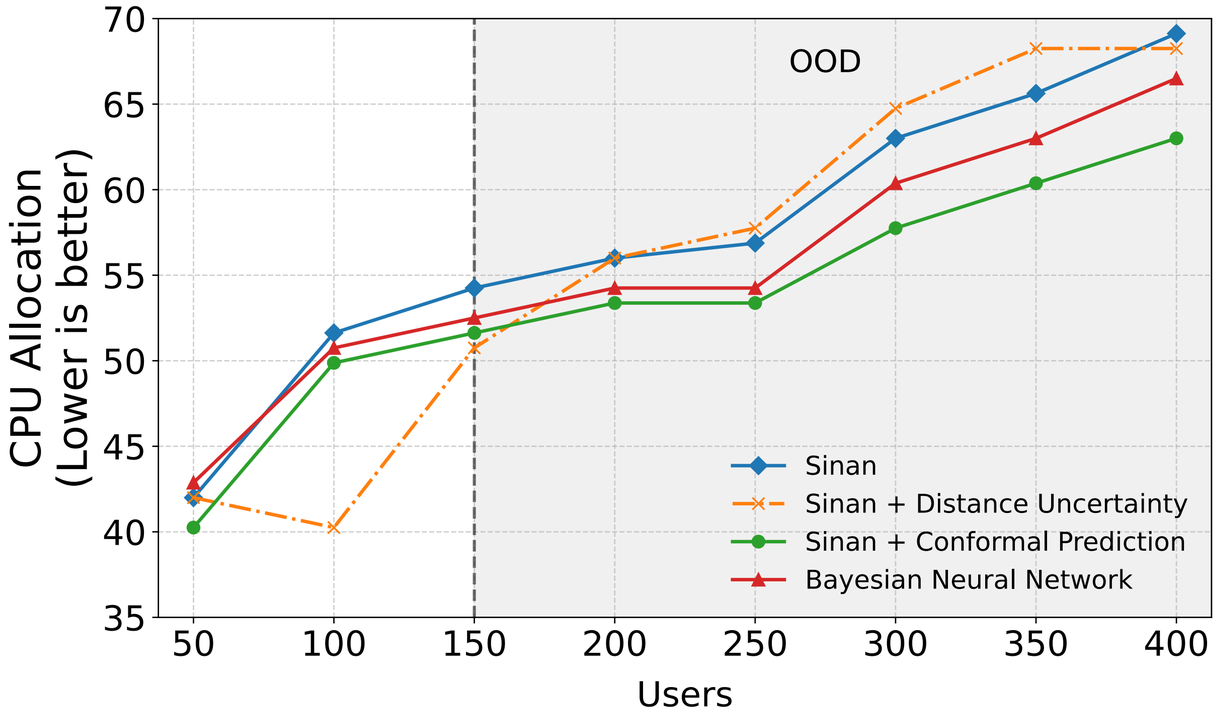




Replacing CNN with a BNN and using bayesian uncertainty results in lower QoS violations with lower CPU allocations compared to Sinan.

Sinan: Microservice Resource Management

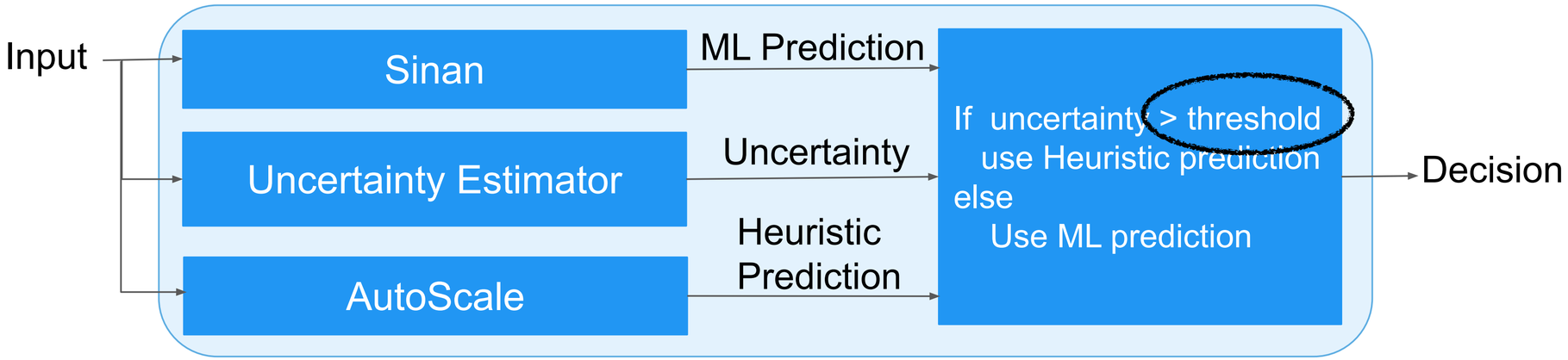




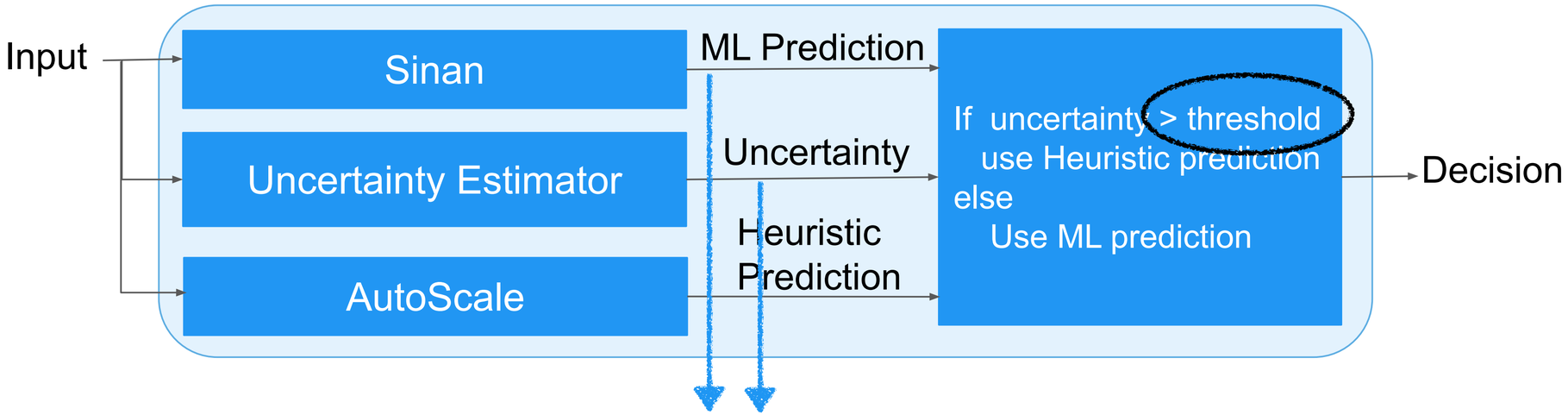
Design Constraint: Cannot change model architecture.

Conformal prediction is model-agnostic and hence best uncertainty estimator for this task.

How to set uncertainty threshold?



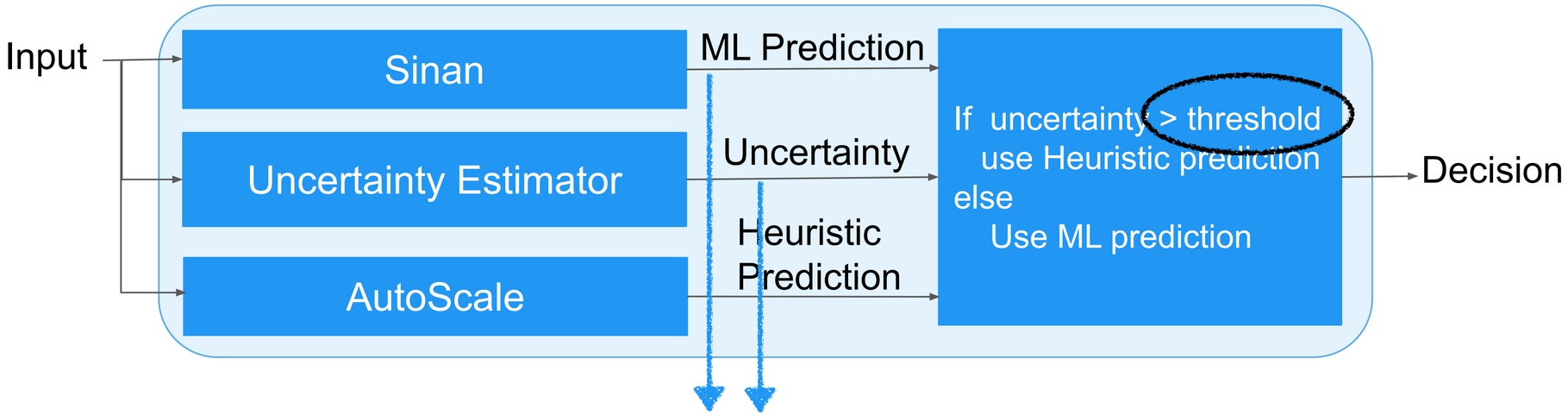
How to set uncertainty threshold?



Same unit: ms (Uncertainty of 1 = Uncertainty of 1ms)

Unit-consistent uncertainty: Unit of uncertainty is same as unit of ML prediction.

How to set uncertainty threshold?

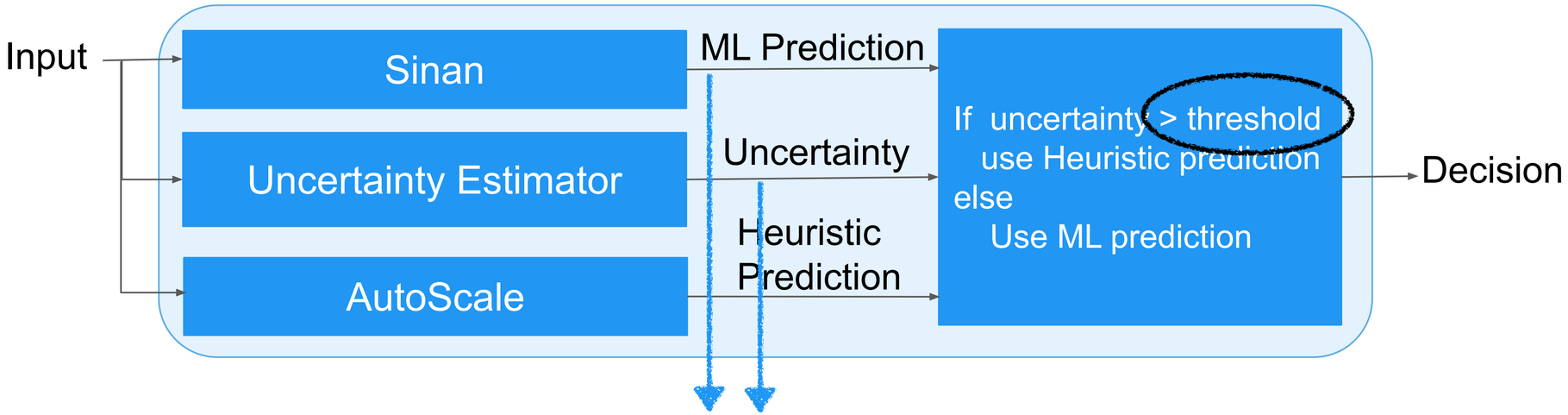


Same unit: ms (Uncertainty of 1 = Uncertainty of 1ms)

Unit-consistent uncertainty: Unit of uncertainty is same as unit of ML prediction.

Domain experts have a sense of acceptable uncertainty that decides threshold.

How to set uncertainty threshold?

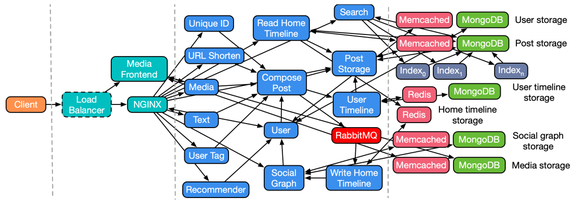


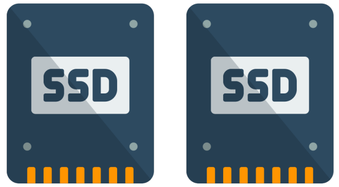
Same unit: ms (Uncertainty of 1 = Uncertainty of 1ms)

Unit-consistent uncertainty: Unit of uncertainty is same as unit of ML prediction.

Domain experts have a sense of acceptable uncertainty that decides threshold.

We use 15% relative uncertainty (~10ms latency interval).





Task	Server Resource Capacity Provisioning	Microservice Resource Management	Storage I/O Routing
Latency Budget	~hours	~milliseconds	~microseconds
Design Constraint	None	Fixed model architecture	Fixed model architecture
Best Uncertainty Estimator	Bayesian	Conformal Prediction	Distance-based

Design Constraint	None	Fixed model architecture	Fixed model architecture
Best Uncertainty Estimator	Bayesian	Conformal Prediction	Distance-based

The optimal uncertainty estimator depends on the task's runtime and design constraints.

Key Takeaways

- Poor generalizability of models makes ‘ML for Systems’ techniques unreliable.
- Proactively use uncertainty, as a proxy of generalizability, to guide model usage.
- Align uncertainty estimator characteristics with task’s runtime and design constraints.