Uncertainty and Calibration in AI Agents: A Review of Key Papers

Presenter: Andrew Zhou Miller

Date: September 11, 2025

Course: Common/Ground in AI Agent

Agenda

- Introduction to Calibration (Foundation)
- Common Features of the Papers
- Paper Sequence: General NN Calibration → Binning Methods → Transformer Calibration → Semantic Parsing Applications → LLM Robotics
- Q&A After Each Paper
- Conclusion

Foundation: What is Model Calibration?

- Calibration: Alignment between predicted probabilities and true likelihood of correctness.
- Well-calibrated model: If it predicts 80% confidence, it should be correct ~80% of the time.
- Metrics:
  - Expected Calibration Error (ECE): $\sum_{b=1}^{N} \frac{|B_b|}{n} |acc(B_b) - conf(B_b)|$,where $B_b$is bin b, acc is accuracy, conf is confidence.
  - Reliability Diagrams: Plot confidence vs. accuracy.
- Importance: Critical for AI agents in high-stakes actions (e.g., robotics) to avoid overconfidence.

Common Features of the 5 Papers

- All focus on uncertainty quantification and calibration to make models reliable for decision-making.
- Common Goal: Reduce miscalibration (e.g., overconfidence in modern deep models).
- Techniques: Post-processing (e.g., scaling, binning) for efficiency; applied to NNs, transformers, LLMs.
- Applications: From classification to sequence generation and robotic planning.
- Key Insight: Calibration improves trustworthiness in actions, especially under distribution shifts or ambiguity.
- Evolution: From general NNs (2017) to LLMs in robotics (2023).

Starting with General Calibration: "On Calibration of Modern Neural Networks" (Guo et al., 2017)

- Overview: Modern NNs (e.g., ResNets) are accurate but miscalibrated (overconfident).
- Factors: Deeper/wider nets worsen calibration; Batch Norm helps.
- Datasets: Image (CIFAR, ImageNet) and text classification.

On Calibration of Modern Neural Networks

$X \in \mathcal{X} \rightarrow$ input

$Y \in \mathcal{Y} = \{1, 2, \ldots, K\} \rightarrow$ label

$\pi(X, Y) = \pi(Y|X)\pi(X) \rightarrow$ ground-truth joint distribution

$h \rightarrow$ neural network

$(\hat{Y}, \hat{P}) = h(X) \rightarrow$ class prediction $\hat{Y}$ and associated confidence $\hat{P}$

We would like $\hat{P}$ to be calibrated (i.e., represent a true probability of correctness)

For example, given 100 predictions, each with confidence of 0.8, we expect that 80 should be correctly classified.

$$\underbrace{\mathbb{P}}_{\text{over the joint distribution}}\left(\hat{Y} = Y \mid \hat{P} = p\right) = p, \quad \forall p \in [0, 1] \rightarrow \text{perfect calibration}$$

Reliability Diagrams

$B_m \rightarrow$ set of indices of samples whose prediction confidence $\hat{p}$ falls into the interval $I_m = (\frac{m-1}{M}, \frac{m}{M}), m = 1, \ldots, M$

$$\text{acc}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}(\hat{y}_i = y_i) \rightarrow \text{consistent and unbiased estimator of } \mathbb{P}(\hat{Y} = Y | \hat{P} \in I_m)$$

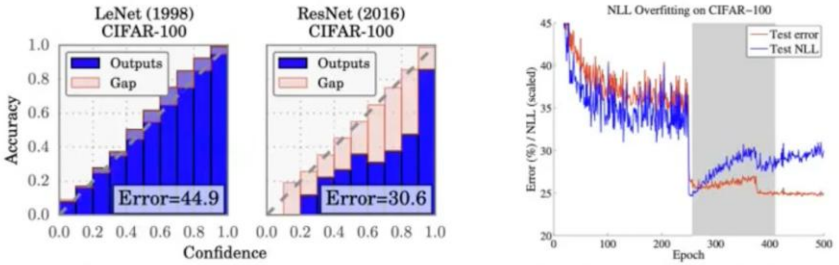
$$\text{conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_i \rightarrow \text{average confidence}$$

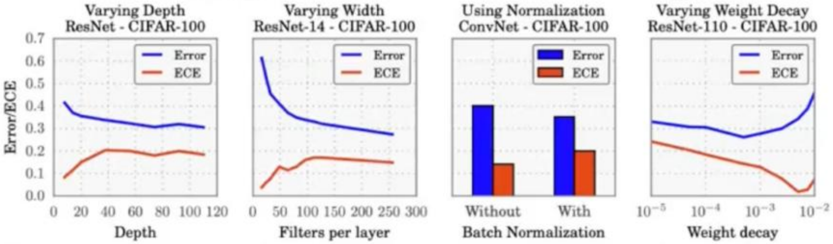
Expected Calibration Error (ECE)

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \underbrace{\left| \text{acc}(B_m) - \text{conf}(B_m) \right|}_{\text{calibration gap}}$$

$n \rightarrow$ number of samples

Modern neural networks are no longer well-calibrated!





Disconnect between NLL (Negative Log Likelihood) and 0/1 loss!

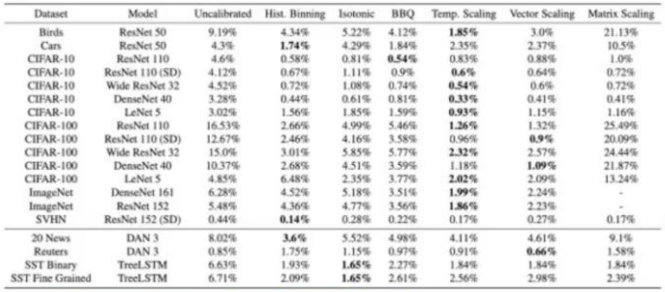
Temperature Scaling

$\mathbf{z}_i \rightarrow$ network logits

$T > 0 \rightarrow$ temperature scaler

$$\hat{q}_i = \max_k \text{softmax}(\frac{\mathbf{z}_i}{T})^{(k)} \rightarrow \text{calibrated confidence}$$

Optimize T using NLL on the validation data



Dataset	Model	Uncalibrated	Hist. Binning	Isotonic	BBQ	Temp. Scaling	Vector Scaling	Matrix Scaling
Birds	ResNet 50	9.19%	4.34%	5.22%	4.12%	**1.85%**	3.0%	21.13%
Cars	ResNet 50	4.3%	**1.74%**	4.29%	1.84%	2.35%	2.37%	10.5%
CIFAR-10	ResNet 110	4.6%	0.58%	0.81%	**0.54%**	0.83%	0.88%	1.0%
CIFAR-10	ResNet 110 (SD)	4.12%	0.67%	1.11%	0.9%	**0.6%**	0.64%	0.72%
CIFAR-10	Wide ResNet 32	4.52%	0.72%	1.08%	0.74%	**0.54%**	0.6%	0.72%
CIFAR-10	DenseNet 40	3.28%	0.44%	0.61%	0.81%	**0.33%**	0.41%	0.41%
CIFAR-10	LeNet 5	3.02%	1.56%	1.85%	1.59%	**0.93%**	1.15%	1.16%
CIFAR-100	ResNet 110	16.53%	2.66%	4.99%	5.46%	**1.26%**	1.32%	25.49%
CIFAR-100	ResNet 110 (SD)	12.67%	2.46%	4.16%	3.58%	0.96%	**0.9%**	20.09%
CIFAR-100	Wide ResNet 32	15.0%	3.01%	5.85%	5.77%	**2.32%**	2.57%	24.44%
CIFAR-100	DenseNet 40	10.37%	2.68%	4.51%	3.59%	1.18%	**1.09%**	21.87%
CIFAR-100	LeNet 5	4.85%	6.48%	2.35%	3.77%	**2.02%**	2.09%	13.24%
ImageNet	DenseNet 161	6.28%	4.52%	5.18%	3.51%	**1.99%**	2.24%	-
ImageNet	ResNet 152	5.48%	4.36%	4.77%	3.56%	**1.86%**	2.23%	-
SVHN	ResNet 152 (SD)	0.44%	**0.14%**	0.28%	0.22%	0.17%	0.27%	0.17%
20 News	DAN 3	8.02%	**3.6%**	5.52%	4.98%	4.11%	4.61%	9.1%
Reuters	DAN 3	0.85%	1.75%	1.15%	0.97%	0.91%	**0.66%**	1.58%
SST Binary	TreeLSTM	6.63%	1.93%	**1.65%**	2.27%	1.84%	1.84%	1.84%
SST Fine Grained	TreeLSTM	6.71%	2.09%	**1.65%**	2.61%	2.56%	2.98%	2.39%

Guo, Chuan, et al. "On calibration of modern neural networks." *International Conference on Machine Learning*. PMLR, 2017.

Methods of Post-Processing Calibration

- Histogram Binning: Bin predictions, map to empirical frequencies.

$$\min_{\theta_1,\ldots,\theta_M} \sum_{m=1}^{M} \sum_{i=1}^{n} \mathbf{1}(a_m \leq \hat{p}_i < a_{m+1}) \left(\theta_m - y_i\right)^2 , \quad (7)$$

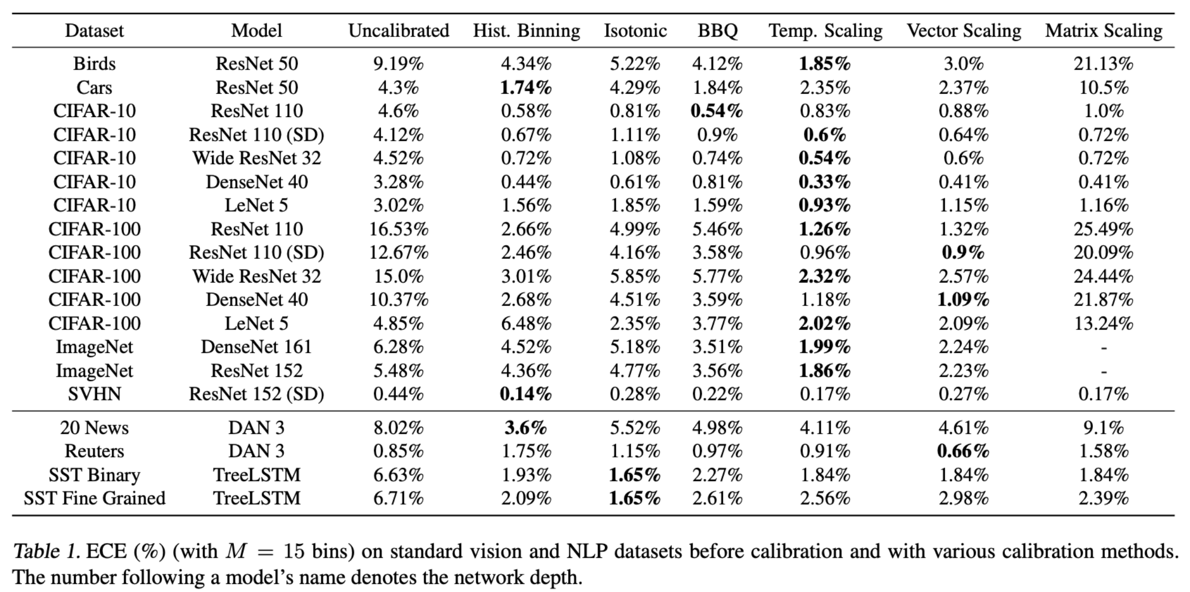
- Isotonic Regression: Non-parametric fit to sort predictions.

$$\begin{aligned} &\min_{\substack{M \\ \theta_1,\ldots,\theta_M \\ a_1,\ldots,a_{M+1}}} && \sum_{m=1}^{M} \sum_{i=1}^{n} \mathbf{1}(a_m \leq \hat{p}_i < a_{m+1}) \left(\theta_m - y_i\right)^2 \\ &\text{subject to} && 0 = a_1 \leq a_2 \leq \ldots \leq a_{M+1} = 1, \\ & && \theta_1 \leq \theta_2 \leq \ldots \leq \theta_M . \end{aligned}$$

- Platt Scaling: Logistic regression $\hat{q}_i = \sigma(az_i + b)$ on logits.
- Key Method: Temperature Scaling (single-parameter Platt variant).
  - Equation: $P(y = k|x) = \frac{\exp(z_k/T)}{\sum_j \exp(z_j/T)}$, optimize Ton validation set via NLL.

Analysis of Calibration Methods in Guo et al.

- ECE as Primary Metric: Lower is better.
- Results: Temperature scaling outperforms others.
- Why Effective: Softens probabilities without changing accuracy.
- Limitations: Assumes validation set represents test distribution.



Dataset	Model	Uncalibrated	Hist. Binning	Isotonic	BBQ	Temp. Scaling	Vector Scaling	Matrix Scaling
Birds	ResNet 50	9.19%	4.34%	5.22%	4.12%	**1.85%**	3.0%	21.13%
Cars	ResNet 50	4.3%	**1.74%**	4.29%	1.84%	2.35%	2.37%	10.5%
CIFAR-10	ResNet 110	4.6%	0.58%	0.81%	**0.54%**	0.83%	0.88%	1.0%
CIFAR-10	ResNet 110 (SD)	4.12%	0.67%	1.11%	0.9%	**0.6%**	0.64%	0.72%
CIFAR-10	Wide ResNet 32	4.52%	0.72%	1.08%	0.74%	**0.54%**	0.6%	0.72%
CIFAR-10	DenseNet 40	3.28%	0.44%	0.61%	0.81%	**0.33%**	0.41%	0.41%
CIFAR-10	LeNet 5	3.02%	1.56%	1.85%	1.59%	**0.93%**	1.15%	1.16%
CIFAR-100	ResNet 110	16.53%	2.66%	4.99%	5.46%	**1.26%**	1.32%	25.49%
CIFAR-100	ResNet 110 (SD)	12.67%	2.46%	4.16%	3.58%	0.96%	**0.9%**	20.09%
CIFAR-100	Wide ResNet 32	15.0%	3.01%	5.85%	5.77%	**2.32%**	2.57%	24.44%
CIFAR-100	DenseNet 40	10.37%	2.68%	4.51%	3.59%	1.18%	**1.09%**	21.87%
CIFAR-100	LeNet 5	4.85%	6.48%	2.35%	3.77%	**2.02%**	2.09%	13.24%
ImageNet	DenseNet 161	6.28%	4.52%	5.18%	3.51%	**1.99%**	2.24%	-
ImageNet	ResNet 152	5.48%	4.36%	4.77%	3.56%	**1.86%**	2.23%	-
SVHN	ResNet 152 (SD)	0.44%	**0.14%**	0.28%	0.22%	0.17%	0.27%	0.17%
20 News	DAN 3	8.02%	**3.6%**	5.52%	4.98%	4.11%	4.61%	9.1%
Reuters	DAN 3	0.85%	1.75%	1.15%	0.97%	0.91%	**0.66%**	1.58%
SST Binary	TreeLSTM	6.63%	1.93%	**1.65%**	2.27%	1.84%	1.84%	1.84%
SST Fine Grained	TreeLSTM	6.71%	2.09%	**1.65%**	2.61%	2.56%	2.98%	2.39%

Table 1. ECE (%) (with $M = 15$ bins) on standard vision and NLP datasets before calibration and with various calibration methods. The number following a model's name denotes the network depth.

Q&A for Guo et al.

- Questions?

Binning Methods: "Obtaining Well Calibrated Probabilities Using Bayesian Binning" (Naeini et al., 2015)

- Overview: Non-parametric method for binary classifiers.
- Issue Addressed: Limitations in prior binning (e.g., fixed bins miss data density).
- BBQ: Bayesian Binning into Quantiles.

Methods in Naeini et al. - BBQ Details

- Process: Bin predictions into quantiles; use Bayesian update for calibrated probs.
- Equation (Bayesian Update Example):

$$P(\theta \mid D) \propto P(D \mid \theta) \cdot P(\theta)$$

where θ is bin probability, D is data.

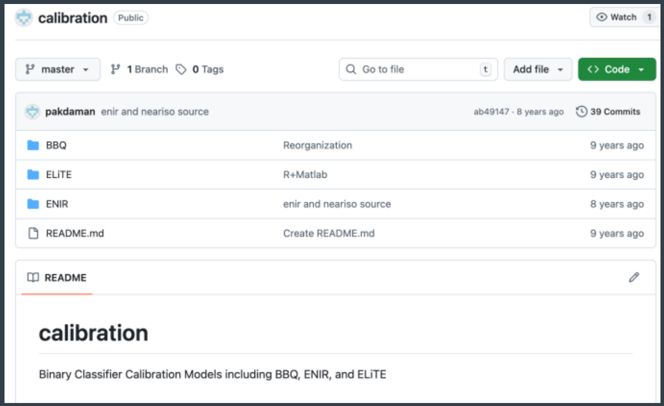
- Advantages: Computationally tractable, empirically accurate on real/simulated data.

Abstract

Key Points:

- Need for well-calibrated probabilities in AI decision-making (e.g., medicine, business).
- Introduces Bayesian Binning into Quantiles (BBQ): Non-parametric post-processing for binary classifiers.
- Addresses limitations of histogram binning, Platt scaling, isotonic regression.
- Combines multiple equal-frequency binnings via Bayesian averaging (BDeu score).
- Empirically superior on simulated/real data

$$P(z=1|y)=\sum_{i=1}^{T}\frac{Score(M_i)}{\sum_{j=1}^{T} Score(M_j)}P(z=1|y,M_i),$$



pakdaman enir and neariso source ab49147 · 8 years ago 39 Commits

BBQ	Reorganization	9 years ago
ELiTE	R+Matlab	9 years ago
ENIR	enir and neariso source	8 years ago
README.md	Create README.md	9 years ago

calibration

Binary Classifier Calibration Models including BBQ, ENIR, and ELiTE

Introduction and Motivation

Key Points: Rational agents maximize utility; need accurate probabilities (Russell & Norvig 1995).

- Miscalibration: Predictions p should match observed frequency (e.g., p=0.5 → 50% positives).
- Reliability Curve: Plots predicted vs. observed; ideal is diagonal (y=x).
- Existing ML focuses on discrimination (AUC), not calibration.
- Post-processing preferred over ab initio redesign (avoids objective changes).

(Figure 1: Solid line deviates slightly low; dotted is ideal).

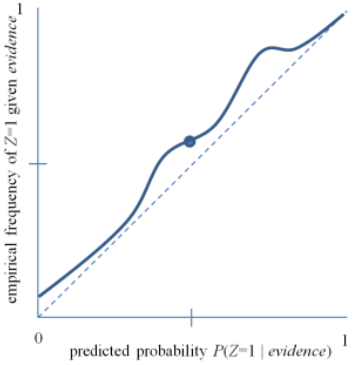


Figure 1: The solid line shows a calibration (reliability) curve for predicting $Z = 1$. The dotted line is the ideal calibration curve.

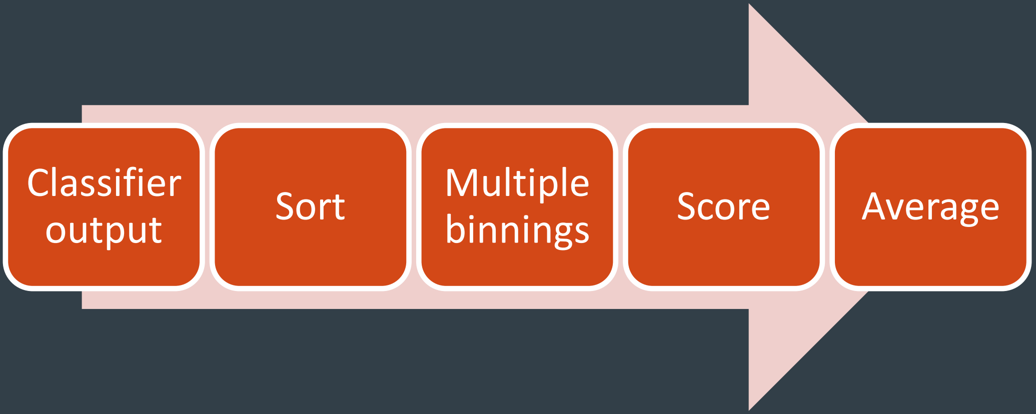
Existing Calibration Methods

- **Key Points**: **Parametric**: Platt scaling (1999)—sigmoid map: $\hat{p} = \frac{1}{1+\exp(Ay+B)}$ limited by shape.
- **Non-Parametric**:
  - Histogram/Quantile Binning (Zadrozny & Elkan 2001): Sort, equal-size bins, replace y with bin positive fraction. Fixed bins = limitation.
  - Isotonic Regression (Zadrozny & Elkan 2002): Monotonic mapping via PAV; good but assumes isotonicity (violated in noise).
- Limitations: Parametric too rigid; non-parametric fixed or assumptive.
- **Visual**: Mapping example (paper's Figure 1 description).

The BBQ Method Overview

Key Points:

- Extends histogram binning: Multiple equal-frequency models varying by B (bin count).
- Model $M = \{B, P_a, \Theta\}$:$P_a$partitions sorted predictions S; $\Theta = \{\theta_b\}$binomial params.
- Range: $B \in \left[\ , \frac{\sqrt[3]{N}}{C}, C\sqrt[3]{N} \right]$ (C=10; inspired by minimax rates).
- Refine: Optional top models via score drop threshold ρ=0.001.



Scoring Binning Models

Key Points:

- Score(M) = P(M) · P(D|M) (uniform P(M)).
- Marginal Likelihood:
- $P(D \mid M) = \prod_{b=1}^{B} \frac{\Gamma(N'/B)\Gamma(m_b+\alpha_b)\Gamma(n_b+\beta_b)}{\Gamma(N_b+N'/B)\Gamma(\alpha_b)\Gamma(\beta_b)}$
  - Priors: $\alpha_b = \frac{N'}{B} p_b$,$\beta_b = \frac{N'}{B}(1 - p_b)$)N'=2, p_b=midpoint).
  - m_b=positives, n_b=negatives in bin b.

Model Averaging for Calibration

Key Points:

- Calibrated P(z=1|y):
  - $P(z=1 \mid y) = \sum_{i=1}^{T} \frac{\text{Score}(M_i)}{\sum_j \text{Score}(M_j)} P(z\text{'} = 1 \mid yM_i)$
  - P(z=1|y, M_i) = Smoothed bin fraction: $\frac{m_b+\alpha_b}{N_b+N'/B}$.

Calibration Measures

Key Points:

- ECE (Expected Calibration Error):
  - ECE $= \sum_{i=1}^{K} P\,(i) \cdot | \; o_i - e_i \; | \;\; (K = 10)$
  - o_i=true positive fraction, e_i=mean predicted in bin i.
- MCE (Maximum Calibration Error):
  - MCE $= \max_{i} \quad | \; o_i - e_i \; |$
- Also: RMSE, AUC, ACC (discrimination).

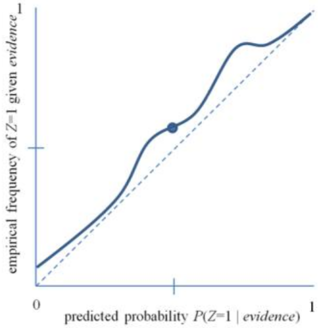
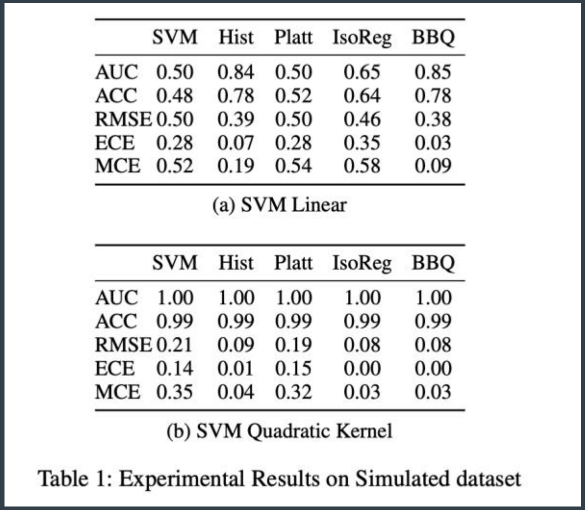


Figure 1: The solid line shows a calibration (reliability) curve for predicting $Z = 1$. The dotted line is the ideal calibration curve.

Empirical Results - Simulated Data

Key Points:

- Dataset: Nonlinear separable (Figure 2: Moon-like scatter, 1000 train/test).
- Bases: Linear SVM (poor), Quadratic SVM (ideal).
- Table 1a (Linear): BBQ ECE=0.03 (vs. raw 0.28, IsoReg 0.35); preserves AUC=0.85.
- Table 1b (Quadratic): Matches IsoReg, improves raw calibration.



	SVM	Hist	Platt	IsoReg	BBQ
AUC	0.50	0.84	0.50	0.65	0.85
ACC	0.48	0.78	0.52	0.64	0.78
RMSE	0.50	0.39	0.50	0.46	0.38
ECE	0.28	0.07	0.28	0.35	0.03
MCE	0.52	0.19	0.54	0.58	0.09

(a) SVM Linear

	SVM	Hist	Platt	IsoReg	BBQ
AUC	1.00	1.00	1.00	1.00	1.00
ACC	0.99	0.99	0.99	0.99	0.99
RMSE	0.21	0.09	0.19	0.08	0.08
ECE	0.14	0.01	0.15	0.00	0.00
MCE	0.35	0.04	0.32	0.03	0.03

(b) SVM Quadratic Kernel

Table 1: Experimental Results on Simulated dataset

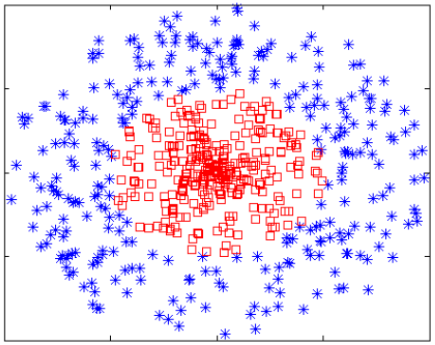


Figure 2: Scatter plot of non-linear separable simulated data

Q&A for Naeini et al.

- Questions?

Overview of "Calibration of Pre-trained Transformers" (Desai & Durrett, 2020)

- Key Focus: Evaluate calibration of BERT and RoBERTa on NLP tasks; compare in-domain (ID) vs. out-of-domain (OD) performance.
- Tasks: Natural Language Inference (SNLI/MNLI), Paraphrase Detection (QQP/TwitterPPDB), Commonsense Reasoning (SWAG/HellaSWAG).
- Findings: Pre-trained models calibrated ID; 3.5× lower OD ECE than baselines (e.g., DA, ESIM).
- Methods: Temperature scaling for ID, label smoothing for OD.
- Importance: Highlights pre-training's role in robust calibration for trustworthy NLP.

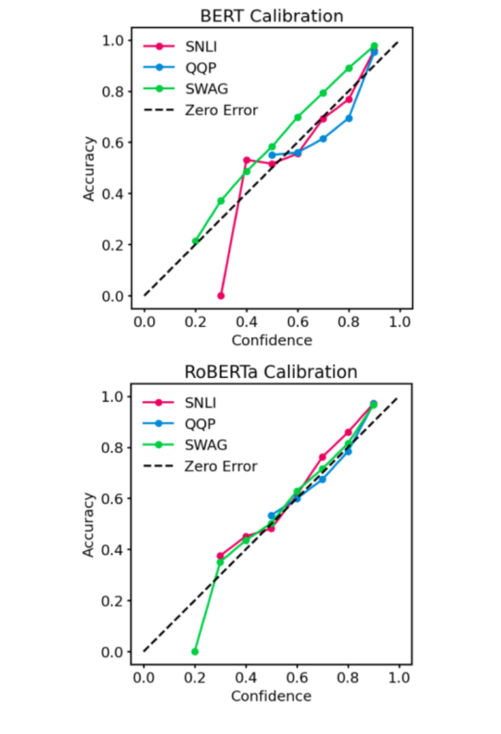


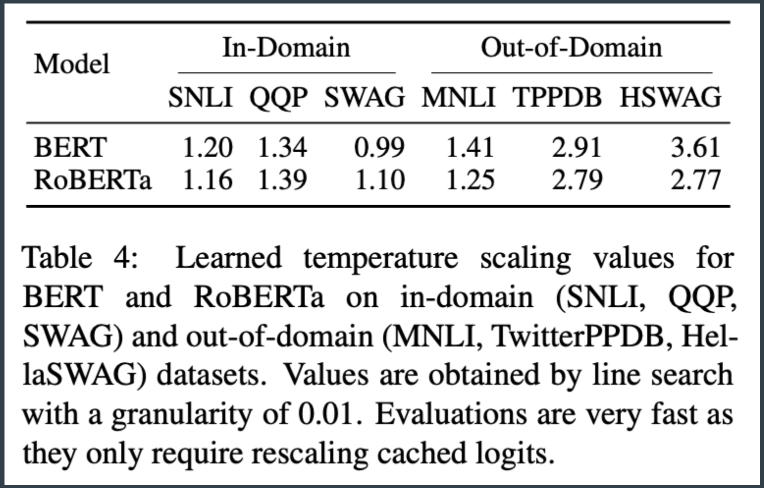
Figure 1: In-domain calibration of BERT and RoBERTa when used out-of-the-box. Models are both trained and evaluated on SNLI, QQP, and SWAG, respectively. ZERO ERROR depicts perfect calibration (e.g., expected calibration error = 0). Note that low-confidence buckets have zero accuracy due to a small sample count; however, as a result, these buckets do not influence the expected error as much.

Calibration Definition and Metrics

- Definition: Posterior probabilities align with empirical likelihoods; perfect if $P(Y = y \mid Q = q) = q$,where $Q$is confidence, $Y$is label.
- Primary Metric: Expected Calibration Error (ECE)
  - Formula: ECE $= \sum_{k=1}^{M} \frac{b_k}{n} |acc(k) - conf(k)|$,where $M = 10$ bins, b_k is bin size, n total examples, acc(k) bin accuracy, conf(k) average confidence.
- Other: Reliability diagrams plot acc(k)vs. conf(k) ;ideal is diagonal line.
- Context: Modern models overconfident; pre-training helps mitigate.

Method 1 - Temperature Scaling

- Post-hoc Technique: Adjust logits before softmax with scalar $T > 0$.
- Equation: $p(y|x) = \frac{\exp(z_y/T)}{\sum_{y'} \exp(z_{y'}/T)}$, where z are logits.
- Optimization: Learn T on ID dev set via NLL minimization.
- Effects: $T < 1$: Sharpens (more confident); $T > 1$: Softens (less confident, reduces overconfidence).
- Advantages: Single parameter, no retraining; effective for ID calibration.

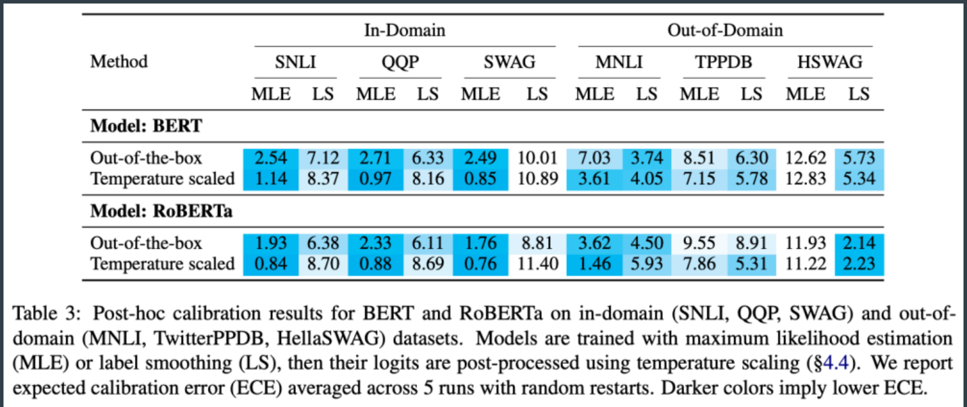


Model	In-Domain			Out-of-Domain		
	SNLI	QQP	SWAG	MNLI	TPPDB	HSWAG
BERT	1.20	1.34	0.99	1.41	2.91	3.61
RoBERTa	1.16	1.39	1.10	1.25	2.79	2.77

Table 4: Learned temperature scaling values for BERT and RoBERTa on in-domain (SNLI, QQP, SWAG) and out-of-domain (MNLI, TwitterPPDB, HellaSWAG) datasets. Values are obtained by line search with a granularity of 0.01. Evaluations are very fast as they only require rescaling cached logits.

Method 2 - Label Smoothing

- During Fine-Tuning: Replace hard one-hot labels with smoothed distribution.
- Equation: Target $q(y) = (1 - \alpha) \cdot \mathbf{1}_{y=y_{true}} + \frac{\alpha}{K-1} \cdot \mathbf{1}_{y \neq y_{true}}$,where $\alpha = 0.1$, Kclasses.
- Objective: Minimize KL divergence instead of MLE, encouraging uncertainty.
- Effects: Increases entropy in predictions, better for OD where shifts cause overconfidence.
- Advantages: Parameter-efficient; combines with temperature scaling.



	In-Domain						Out-of-Domain					
Method	SNLI		QQP		SWAG		MNLI		TPPDB		HSWAG	
	MLE	LS	MLE	LS	MLE	LS	MLE	LS	MLE	LS	MLE	LS
Model: BERT												
Out-of-the-box	2.54	7.12	2.71	6.33	2.49	10.01	7.03	3.74	8.51	6.30	12.62	5.73
Temperature scaled	1.14	8.37	0.97	8.16	0.85	10.89	3.61	4.05	7.15	5.78	12.83	5.34
Model: RoBERTa												
Out-of-the-box	1.93	6.38	2.33	6.11	1.76	8.81	3.62	4.50	9.55	8.91	11.93	2.14
Temperature scaled	0.84	8.70	0.88	8.69	0.76	11.40	1.46	5.93	7.86	5.31	11.22	2.23

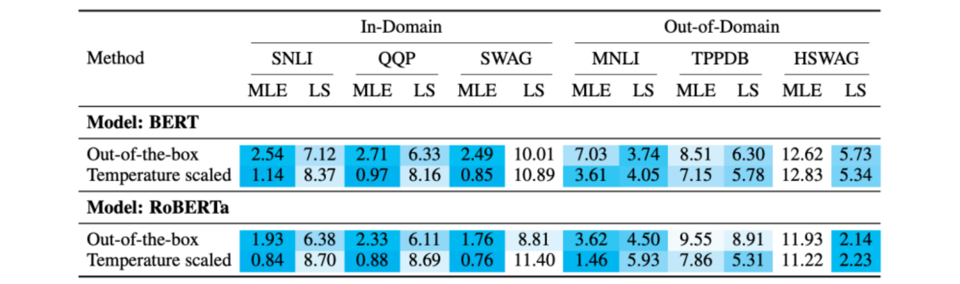
Table 3: Post-hoc calibration results for BERT and RoBERTa on in-domain (SNLI, QQP, SWAG) and out-of-domain (MNLI, TwitterPPDB, HellaSWAG) datasets. Models are trained with maximum likelihood estimation (MLE) or label smoothing (LS), then their logits are post-processed using temperature scaling (§4.4). We report expected calibration error (ECE) averaged across 5 runs with random restarts. Darker colors imply lower ECE.

Experiments and Results

- Models: DA (382K params), ESIM (4M), BERT (110M), RoBERTa (110M).
- Datasets: ID/OD pairs per task (e.g., SNLI/MNLI); sizes in Appendix Table 5.
- Out-of-the-Box Results (Table 2): Pre-trained models higher acc, lower ECE OD (e.g., RoBERTa 78.79% acc, 3.62 ECE on MNLI vs. DA's 70.69% acc, 9.38 ECE).
- Post-hoc Results (Table 3): Temp scaling lowers ID ECE (RoBERTa 0.84 on SNLI); LS + temp lowers OD ECE (e.g., 1.49 on HellaSWAG).
- Key: Pre-training reduces OD miscalibration by increasing uncertainty.

Analysis and Conclusions

- Analysis: BERT/RoBERTa calibrated ID due to pre-training; OD benefits from uncertainty boost via LS (higher T values OD indicate needed softening).
- Limitations: ECE binning sensitive; assumes dev set representative.
- Conclusions: Pre-trained transformers reliable for trust; use temp for ID, LS for OD; future: explore larger models, more tasks.
- Broader Impact: Enables safer AI agents in NLP applications.



Method	In-Domain						Out-of-Domain					
	SNLI		QQP		SWAG		MNLI		TPPDB		HSWAG	
	MLE	LS	MLE	LS	MLE	LS	MLE	LS	MLE	LS	MLE	LS
Model: BERT												
Out-of-the-box	2.54	7.12	2.71	6.33	2.49	10.01	7.03	3.74	8.51	6.30	12.62	5.73
Temperature scaled	1.14	8.37	0.97	8.16	0.85	10.89	3.61	4.05	7.15	5.78	12.83	5.34
Model: RoBERTa												
Out-of-the-box	1.93	6.38	2.33	6.11	1.76	8.81	3.62	4.50	9.55	8.91	11.93	2.14
Temperature scaled	0.84	8.70	0.88	8.69	0.76	11.40	1.46	5.93	7.86	5.31	11.22	2.23

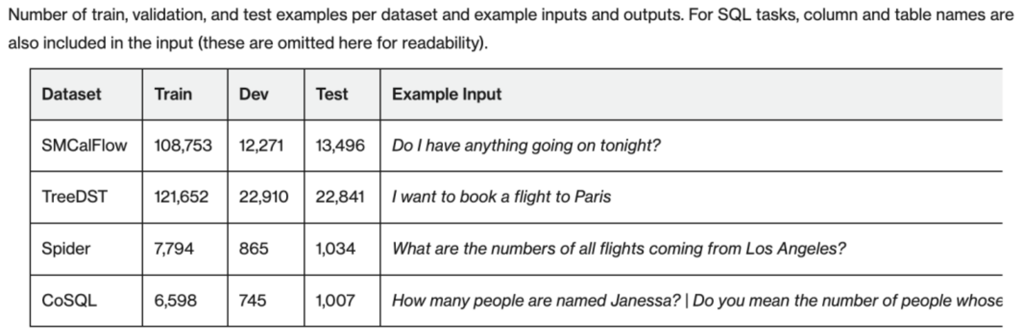
Table 3: Post-hoc calibration results for BERT and RoBERTa on in-domain (SNLI, QQP, SWAG) and out-of-domain (MNLI, TwitterPPDB, HellaSWAG) datasets. Models are trained with maximum likelihood estimation (MLE) or label smoothing (LS), then their logits are post-processed using temperature scaling (§4.4). We report expected calibration error (ECE) averaged across 5 runs with random restarts. Darker colors imply lower ECE.

Q&A for Desai & Durrett

- Questions?

Overview of "Calibrated Interpretation: Confidence Estimation in Semantic Parsing"

- Key Focus: First large-scale study of calibration in executable semantic parsing; measures alignment of model confidence with correctness in sequence generation.
- Tasks: Task-Oriented Dialogue (TOD) parsing (SMCalFlow, TreeDST) and Text-to-SQL (Spider, CoSQL).
- Findings: TOD models well-calibrated (low ECE); SQL models poorly calibrated (high overconfidence); calibration varies by model scale, paradigm (fine-tuning vs. ICL).
- Methods: Subword max-prob confidence; min aggregation for tokens/sequences; adaptive ECE.
- Contributions: Analysis of factors (data size, input/output difficulty); release of challenge splits (EASY/HARD subsets) and calibration metrics library.
- Importance: Enhances safety in AI agents for real-world actions via reliable confidence estimates.

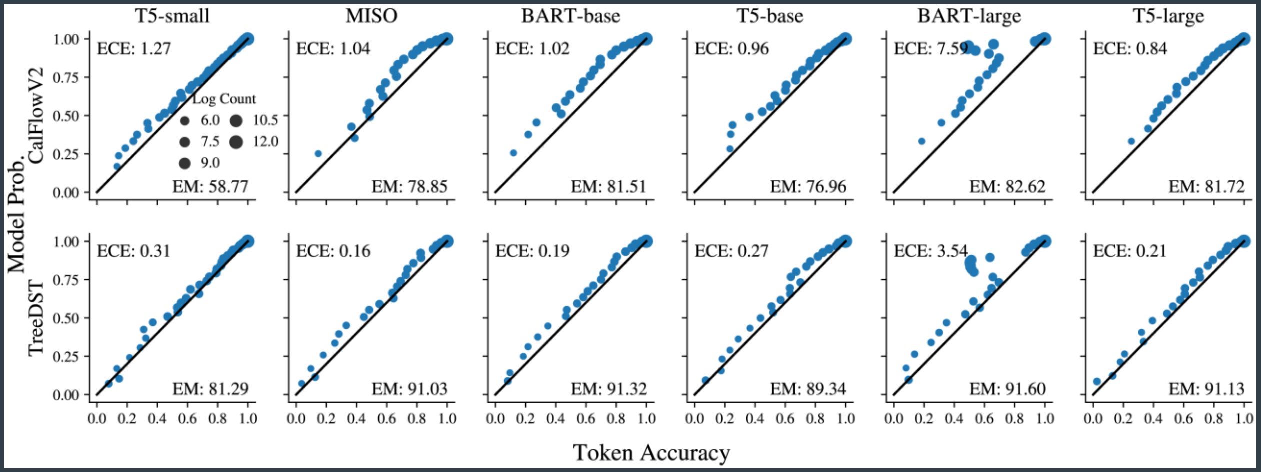


Number of train, validation, and test examples per dataset and example inputs and outputs. For SQL tasks, column and table names are also included in the input (these are omitted here for readability).

Dataset	Train	Dev	Test	Example Input
SMCalFlow	108,753	12,271	13,496	*Do I have anything going on tonight?*
TreeDST	121,652	22,910	22,841	*I want to book a flight to Paris*
Spider	7,794	865	1,034	*What are the numbers of all flights coming from Los Angeles?*
CoSQL	6,598	745	1,007	*How many people are named Janessa? | Do you mean the number of people whose*

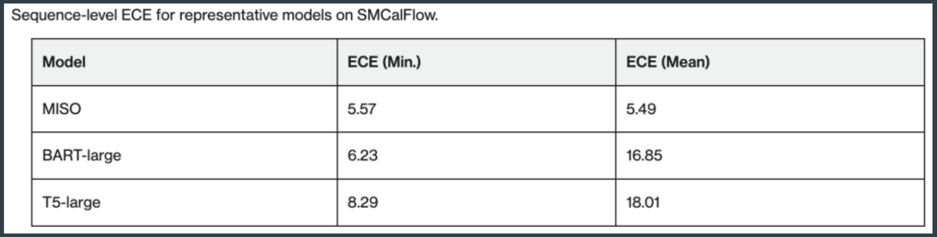
Calibration in Sequence Generation

- Challenge: Unlike classification, semantic parsing involves multi-timestep dependencies; errors propagate.
- Definition: Confidence c_j should match accuracy likelihood; overconfidence risky for executable outputs.
- Token-Level: Confidence per program token (aggregated from subwords).
- Sequence-Level: Aggregate over entire predicted program $\hat{P}$.
- Metrics Focus: Expected Calibration Error (ECE) with adaptive binning for non-uniform distributions.
- Why Semantic Parsing? Constrained outputs enable exact/execution accuracy measurement.



Method 1 - Confidence Estimation

- Subword-Level: Max probability at each timestep: $\hat{C} = \max \hat{Z}$,where $\hat{Z}$is distribution over vocabulary V; $\hat{Y} = \arg\max \hat{Z}$.
- Token-Level Aggregation: Min over subwords (e.g., "SELECT" split into 3: min of their max-probs), as weakest link.
- Sequence-Level Aggregation: Min over tokens in $\hat{P}$)preferred over mean, which inflates confidence in long sequences).
- Alternatives Tested: Mean pooling increases ECE (e.g., BART-large: 16.85 mean vs. 6.23 min on SMCalFlow).
- No Teacher Forcing for Sequences: Uses predicted outputs for realistic confidence.

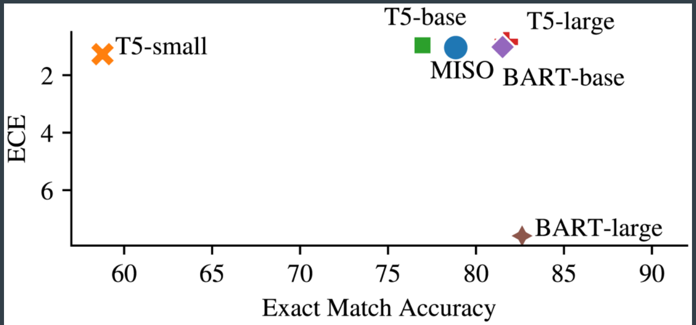


Sequence-level ECE for representative models on SMCalFlow.

Model	ECE (Min.)	ECE (Mean)
MISO	5.57	5.49
BART-large	6.23	16.85
T5-large	8.29	18.01

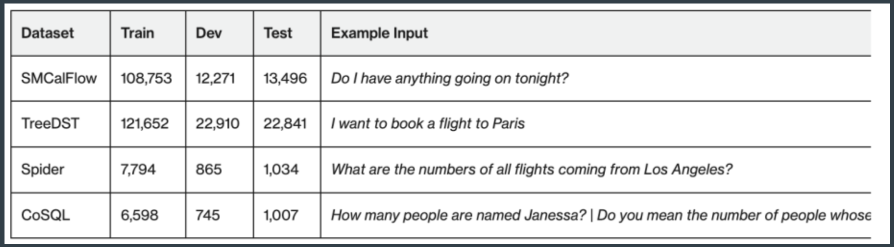
Method 2 - Calibration Metrics

- Primary: Expected Calibration Error (ECE)
  - Equation: ECE$(B) = 100 \times \sum_{i=1}^{N} \frac{|B_i|}{n} \left| \frac{\sum_{j \in B_i} a_j}{|B_i|} - \frac{\sum_{j \in B_i} c_j}{|B_i|} \right|$where $B_i$is bin i, $n$total examples, $a_j = \delta(\hat{y}_j, y_j)$)binary accuracy), $c_j$confidence.
- Adaptive Binning: Bin size $n_b = Z_{\alpha/2}^2 \cdot \frac{p(1-p)}{\epsilon^2}$,with $Z_{\alpha/2}$Z-score, $\epsilon$for stability.
- Accuracy Measures: Exact Match (EM); for SQL, also execution accuracy (executes against DB).
- Qualitative: Reliability plots (accuracy vs. binned confidence); ideal diagonal.
- Library: Released for ECE computation and plotting, integrated with Transformers.



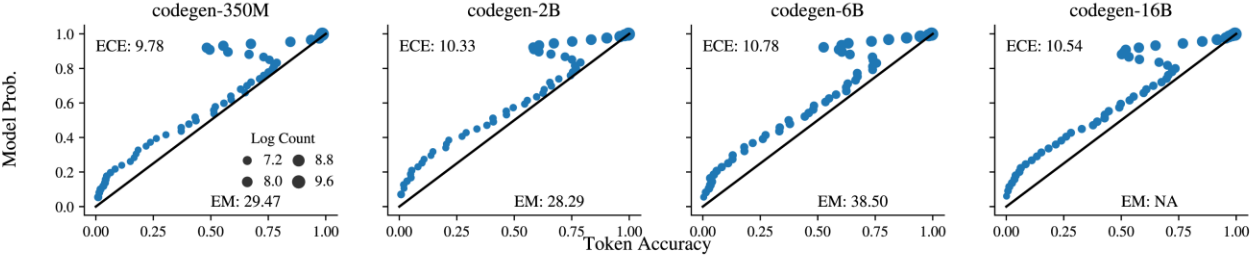
Experiments - Datasets and Models

- Datasets: TOD (SMCalFlow: 108k train; TreeDST: 122k); SQL (Spider: 8k; CoSQL: 7k). All English; multi-turn for TreeDST/CoSQL.



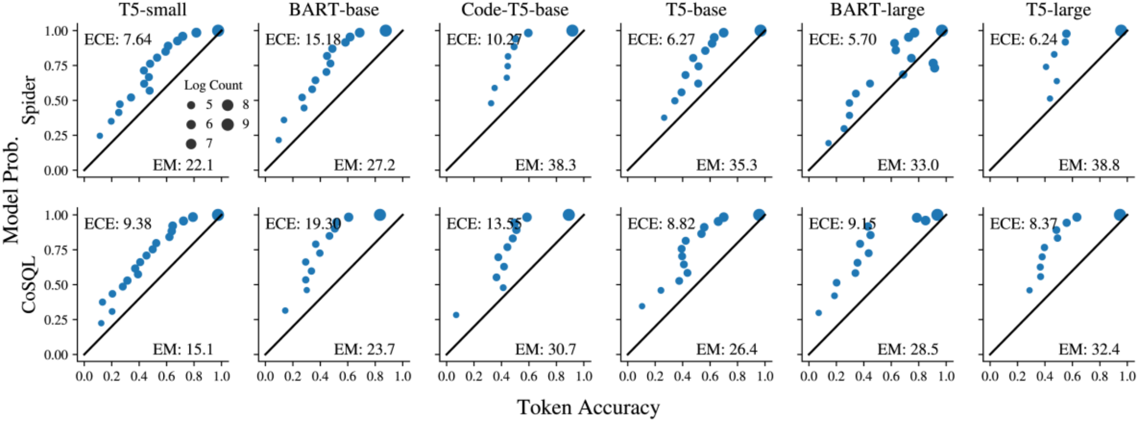
Dataset	Train	Dev	Test	Example Input
SMCalFlow	108,753	12,271	13,496	*Do I have anything going on tonight?*
TreeDST	121,652	22,910	22,841	*I want to book a flight to Paris*
Spider	7,794	865	1,034	*What are the numbers of all flights coming from Los Angeles?*
CoSQL	6,598	745	1,007	*How many people are named Janessa? | Do you mean the number of people whose*

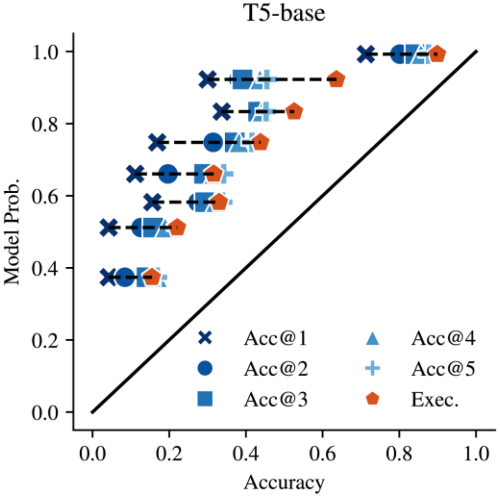
- Models:
  - Fine-Tuning: MISO (127M), T5 (60M-770M), BART (139M-406M), Code-T5 (220M for SQL).
  - ICL: Codegen (350M-16B), with 5-shot prompts, BM25 retrieval, constrained decoding.
- Setups: Unconstrained decoding for fine-tuning; inputs include dialogue history/schema; no post-hoc calibration.



Results - Calibration Findings

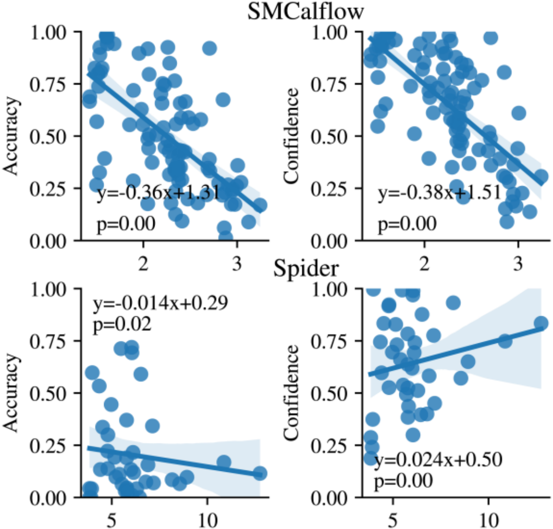
- TOD: Low ECE (e.g., SMCalFlow: 0.84 BART-large to 7.59 T5-base; TreeDST: 0.16 T5-small to 3.54 T5-base); improves with T5 scale.
- SQL: High ECE (Spider: 5.70 T5-base to 15.18 BART-base; CoSQL: 8.37 T5-large to 19.30 BART-base); overconfident.
- ICL: Higher ECE than fine-tuning (SMCalFlow: ~10); U-shaped with scale.
- Sequence vs. Token: Higher ECE at sequence (e.g., T5-large SMCalFlow: 8.29 vs. token 1.04).
- SQL Acc: Execution better calibrated than EM (Figure 7).

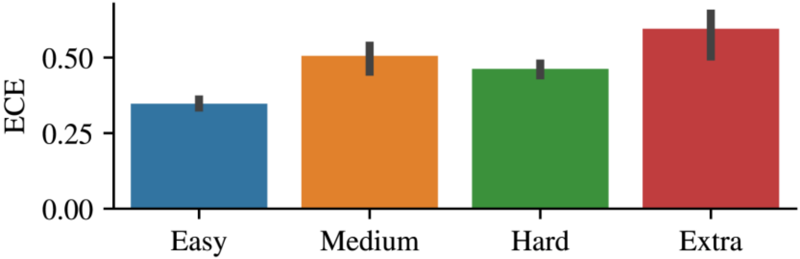


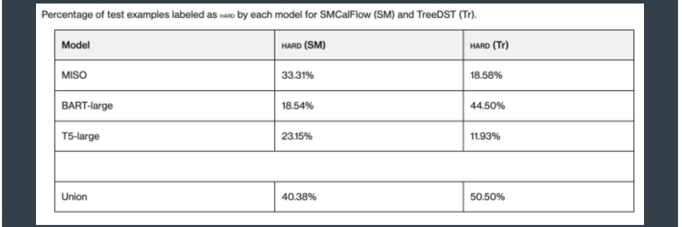


Results - Factors Affecting Calibration

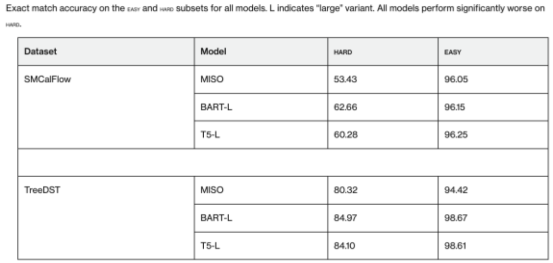
- Data Size: Not primary (reducing SMCalFlow to Spider size raises ECE 1.27→2.40, still < Spider 7.64).
- Input Difficulty: TOD confidence drops with perplexity (OOD); SQL overconfident on OOD.
- Output Difficulty: ECE rises with SQL hardness (easy: ~5; extra-hard: ~10 for T5-large).
- Challenge Splits: EASY (high confidence, low acc); HARD (low confidence, high acc); unions 40-50% of test.






Percentage of test examples labeled as HARD by each model for SMCalFlow (SM) and TreeDST (Tr).

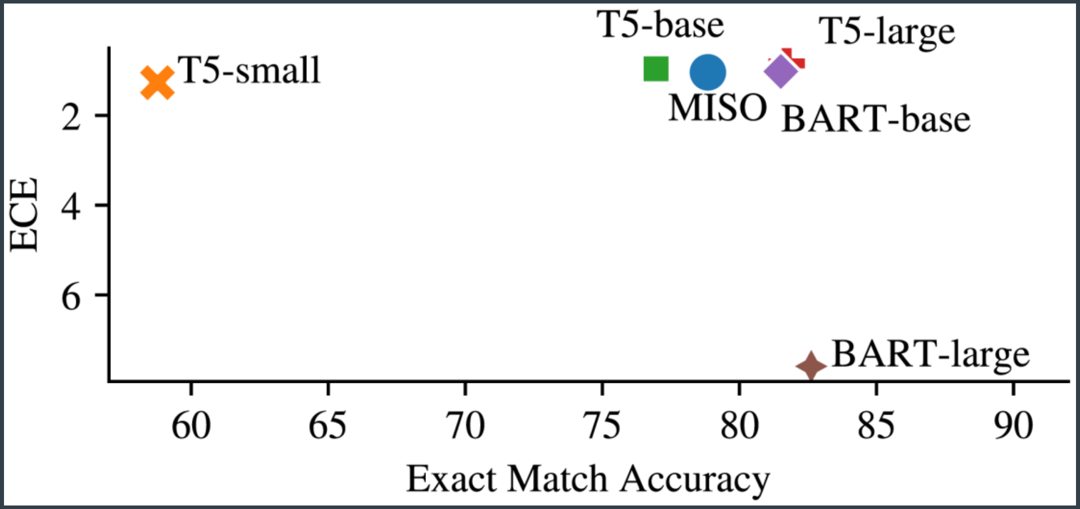
Model	HARD (SM)	HARD (Tr)
MISO	33.31%	18.58%
BART-large	18.54%	44.50%
T5-large	23.15%	11.93%
Union	40.38%	50.50%


Exact match accuracy on the EASY and HARD subsets for all models. L indicates "large" variant. All models perform significantly worse on HARD.

Dataset	Model	HARD	EASY
SMCalFlow	MISO	53.43	96.05
	BART-L	62.66	96.15
	T5-L	60.28	96.25
TreeDST	MISO	80.32	94.42
	BART-L	84.97	98.67
	T5-L	84.10	98.61

Analysis and Conclusions

- Analysis: Non-monotonic curves (BART) hard to fix; ICL resembles BART; execution acc promising for calibration.
- Limitations: English-only; no post-hoc methods; EM strict.
- Conclusions: Calibration feasible in parsing; varies by domain/model; integrate metrics for safer agents; releases aid research.
- Future: Post-hoc calibration; multilingual; other generation tasks.

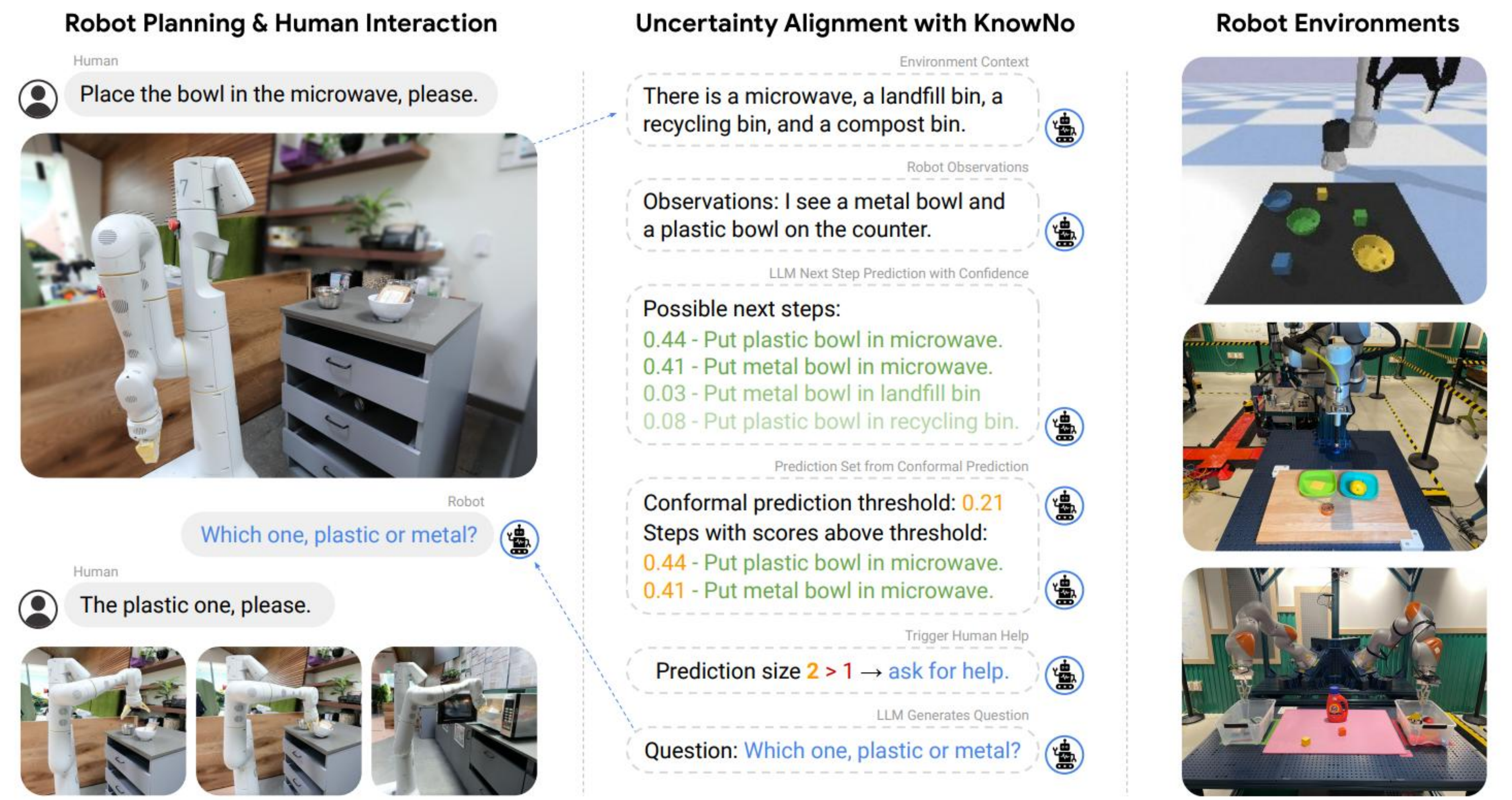


Q&A

- Questions?

Overview of "Robots That Ask For Help: Uncertainty Alignment for Large Language Model Planners"

- Key Focus: Align LLM uncertainty in robotic planning to enable robots to ask for help when uncertain, reducing hallucinations and improving safety/efficiency.
- Problem: LLMs excel in planning but confidently hallucinate on ambiguous tasks (e.g., spatial, numeric, preferences, Winograd schemas).
- Contributions: KnowNo framework using conformal prediction (CP) for statistical guarantees on task success (≥1-ε) with minimal human help; works out-of-box with LLMs, no fine-tuning.
- Tasks: Simulated (e.g., table-top manipulation) and real-robot experiments with multi-step planning under ambiguity.
- Importance: Enhances autonomy in AI agents for real-world robotics by quantifying and acting on uncertainty.



Robot Planning & Human Interaction

Human

Place the bowl in the microwave, please.

Robot

Which one, plastic or metal?

Human

The plastic one, please.

Uncertainty Alignment with KnowNo

Environment Context

There is a microwave, a landfill bin, a recycling bin, and a compost bin.

Robot Observations

Observations: I see a metal bowl and a plastic bowl on the counter.

LLM Next Step Prediction with Confidence

Possible next steps:
0.44 - Put plastic bowl in microwave.
0.41 - Put metal bowl in microwave.
0.03 - Put metal bowl in landfill bin
0.08 - Put plastic bowl in recycling bin.

Prediction Set from Conformal Prediction

Conformal prediction threshold: 0.21
Steps with scores above threshold:
0.44 - Put plastic bowl in microwave.
0.41 - Put metal bowl in microwave.

Trigger Human Help

Prediction size 2 > 1 → ask for help.

LLM Generates Question

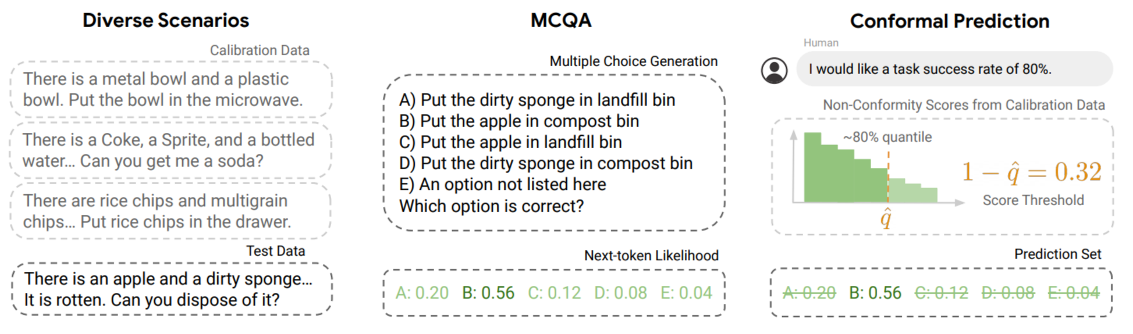
Question: Which one, plastic or metal?

Robot Environments

Figure 1: KnowNo uses Conformal Prediction (CP) to align the uncertainty of LLM planners. Given a language instruction, an LLM generates possible next steps and its confidences (scores) in these options. CP then provides a prediction set that includes the options with scores above a certain quantile. If there is more than one option in the set, the robot asks for help. Experiments across multiple embodiments and a variety of ambiguous situations show that KnowNo significantly improves efficiency and autonomy compared to baselines.

Problem and Motivation

- LLM Planners in Robotics: Use LLMs for step-by-step reasoning and action generation, but prone to errors in ambiguous scenarios (e.g., "place the bowl in the microwave" with multiple bowls).
- Uncertainty Misalignment: LLMs output uncalibrated probabilities, leading to unsafe actions without recognition of unknowns.
- Goal: Calibrate uncertainty so robots ask for help only when needed, balancing autonomy and success.
- Ambiguity Types: Spatial (object disambiguation), Numeric (e.g., quantities), Preferences (user-specific), Winograd (commonsense).
- Why CP? Provides distribution-free guarantees without retraining.



There is a metal bowl and a plastic bowl. Put the bowl in the microwave.

There is a Coke, a Sprite, and a bottled water... Can you get me a soda?

There are rice chips and multigrain chips... Put rice chips in the drawer.

There is an apple and a dirty sponge... It is rotten. Can you dispose of it?

A) Put the dirty sponge in landfill bin
B) Put the apple in compost bin
C) Put the apple in landfill bin
D) Put the dirty sponge in compost bin
E) An option not listed here
Which option is correct?

A: 0.20 B: 0.56 C: 0.12 D: 0.08 E: 0.04

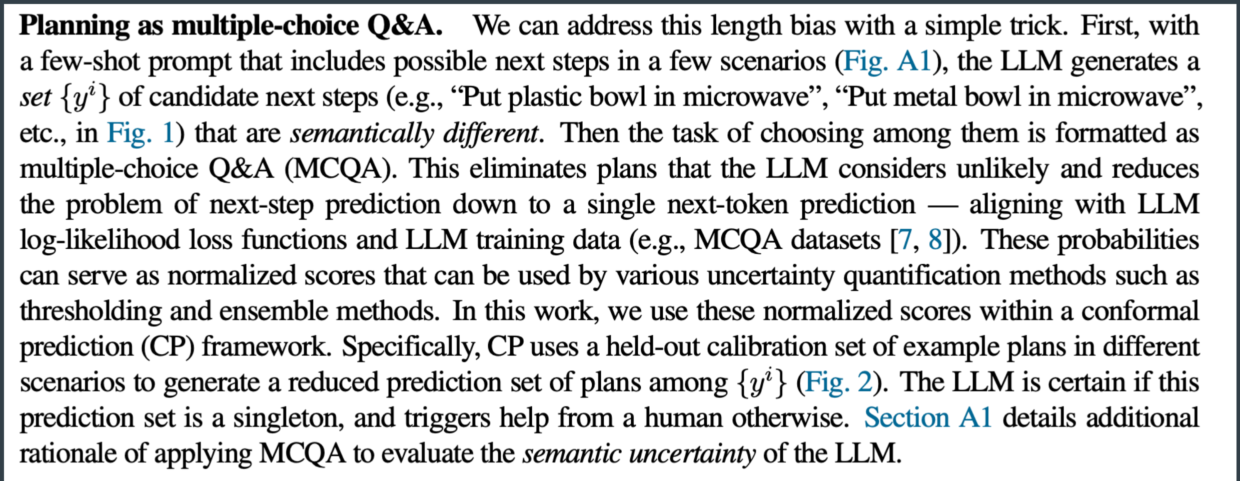
I would like a task success rate of 80%.

~~A: 0.20~~ B: 0.56 ~~C: 0.12~~ ~~D: 0.08~~ ~~E: 0.04~~

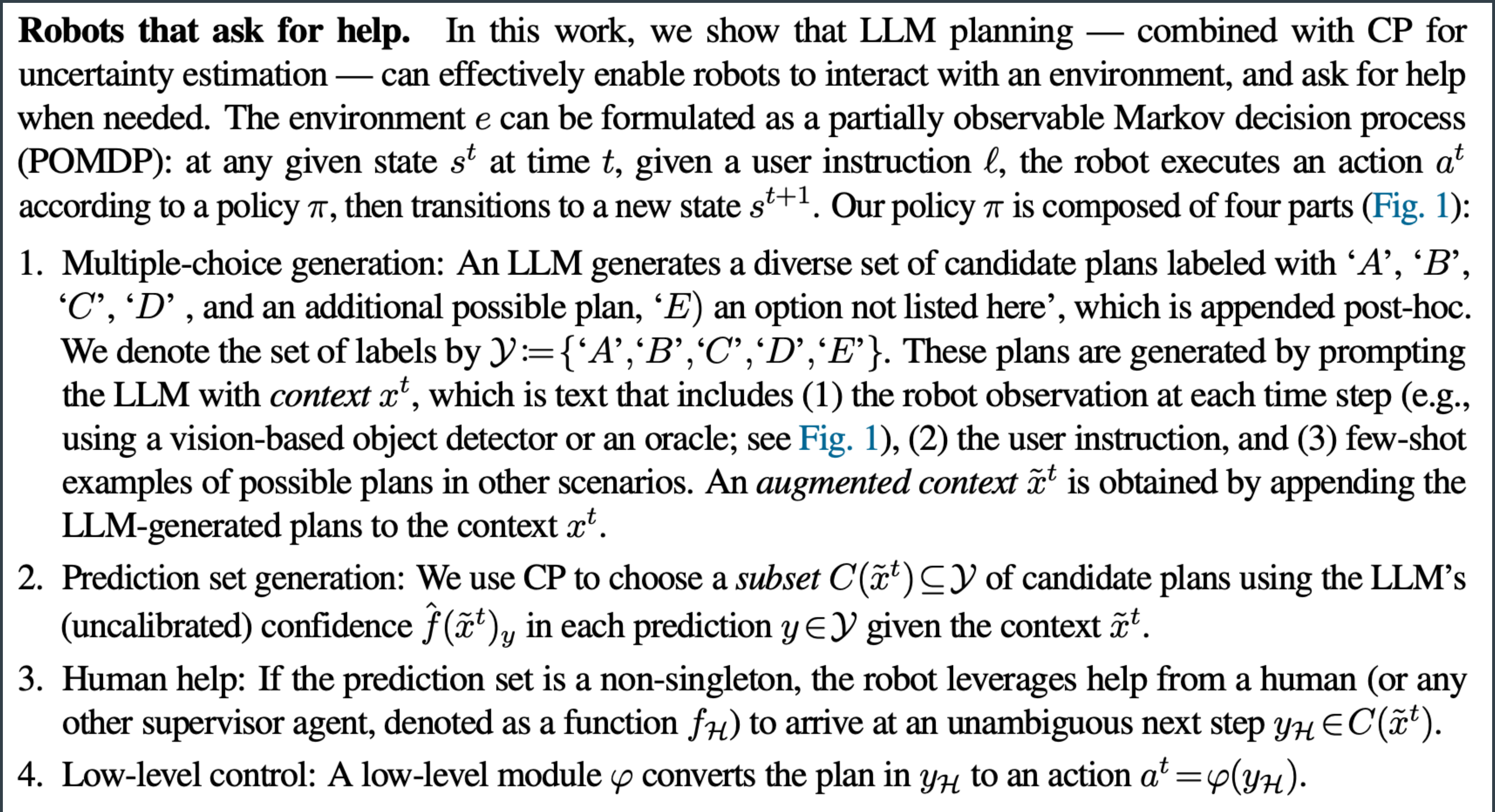
Figure 2: KnowNo formulates LLM planning as MCQA by first prompting an LLM to generate plausible options, and then asking it to predict the correct one. Based on the next-token likelihoods from a calibration dataset, CP finds the quantile value $\hat{q}$ such that all options with a score $\geq 1 - \hat{q}$ are included in the prediction set in a test scenario. The resulting sets are guaranteed to cover the true option with the user-specified probability.

Method 1 - Uncertainty Estimation via MCQA

- Prompting: Format planning as Multiple-Choice Question Answering (MCQA) to generate fixed candidates and score them.
- Generation: LLM proposes semantically diverse options (e.g., 5-10 candidates) via few-shot prompting.
- Scoring: Use normalized next-token likelihoods as probabilities $\hat{f}\left(\tilde{x}\right)_y$ for each candidate y.
- Equation (Joint Probability Issue): Avoid length-biased $p(y) = \prod_{i=1}^{k} p\,(\sigma_i \mid \sigma_1 \ldots \sigma_{i-1})$;MCQA normalizes across fixed set.
- Multi-Step: Recursively apply for chained plans, asking help per step if uncertain.



Planning as multiple-choice Q&A. We can address this length bias with a simple trick. First, with a few-shot prompt that includes possible next steps in a few scenarios (Fig. A1), the LLM generates a *set* $\{y^i\}$ of candidate next steps (e.g., "Put plastic bowl in microwave", "Put metal bowl in microwave", etc., in Fig. 1) that are *semantically different*. Then the task of choosing among them is formatted as multiple-choice Q&A (MCQA). This eliminates plans that the LLM considers unlikely and reduces the problem of next-step prediction down to a single next-token prediction — aligning with LLM log-likelihood loss functions and LLM training data (e.g., MCQA datasets [7, 8]). These probabilities can serve as normalized scores that can be used by various uncertainty quantification methods such as thresholding and ensemble methods. In this work, we use these normalized scores within a conformal prediction (CP) framework. Specifically, CP uses a held-out calibration set of example plans in different scenarios to generate a reduced prediction set of plans among $\{y^i\}$ (Fig. 2). The LLM is certain if this prediction set is a singleton, and triggers help from a human otherwise. Section A1 details additional rationale of applying MCQA to evaluate the *semantic uncertainty* of the LLM.

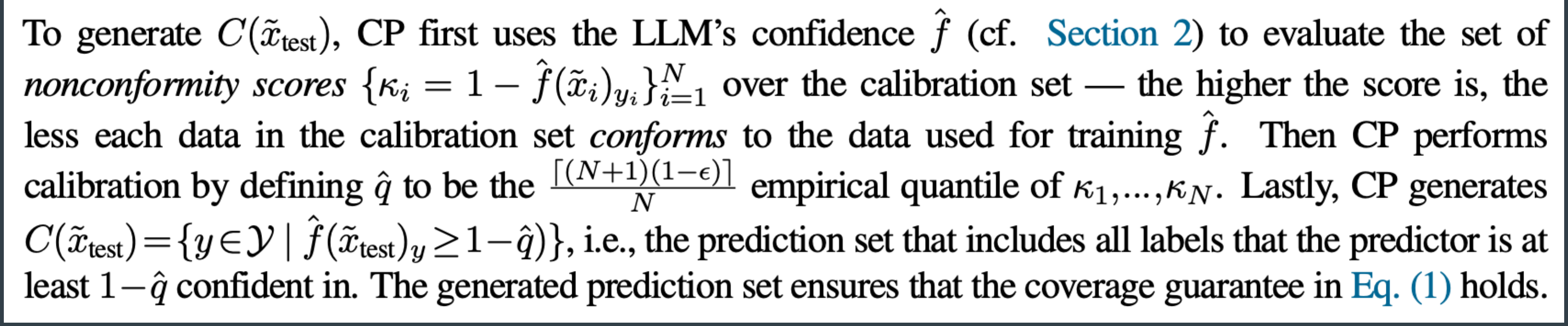


Robots that ask for help. In this work, we show that LLM planning — combined with CP for uncertainty estimation — can effectively enable robots to interact with an environment, and ask for help when needed. The environment e can be formulated as a partially observable Markov decision process (POMDP): at any given state s^t at time t, given a user instruction ℓ, the robot executes an action a^t according to a policy π, then transitions to a new state s^{t+1}. Our policy π is composed of four parts (Fig. 1):

1. Multiple-choice generation: An LLM generates a diverse set of candidate plans labeled with 'A', 'B', 'C', 'D' , and an additional possible plan, 'E) an option not listed here', which is appended post-hoc. We denote the set of labels by $\mathcal{Y} := \{$'A','B','C','D','E'$\}$. These plans are generated by prompting the LLM with *context* x^t, which is text that includes (1) the robot observation at each time step (e.g., using a vision-based object detector or an oracle; see Fig. 1), (2) the user instruction, and (3) few-shot examples of possible plans in other scenarios. An *augmented context* $\tilde{x}^t$ is obtained by appending the LLM-generated plans to the context x^t.
2. Prediction set generation: We use CP to choose a *subset* $C(\tilde{x}^t) \subseteq \mathcal{Y}$ of candidate plans using the LLM's (uncalibrated) confidence $\hat{f}(\tilde{x}^t)_y$ in each prediction $y \in \mathcal{Y}$ given the context $\tilde{x}^t$.
3. Human help: If the prediction set is a non-singleton, the robot leverages help from a human (or any other supervisor agent, denoted as a function $f_{\mathcal{H}}$) to arrive at an unambiguous next step $y_{\mathcal{H}} \in C(\tilde{x}^t)$.
4. Low-level control: A low-level module φ converts the plan in $y_{\mathcal{H}}$ to an action $a^t = \varphi(y_{\mathcal{H}})$.

Method 2 - Alignment via Conformal Prediction

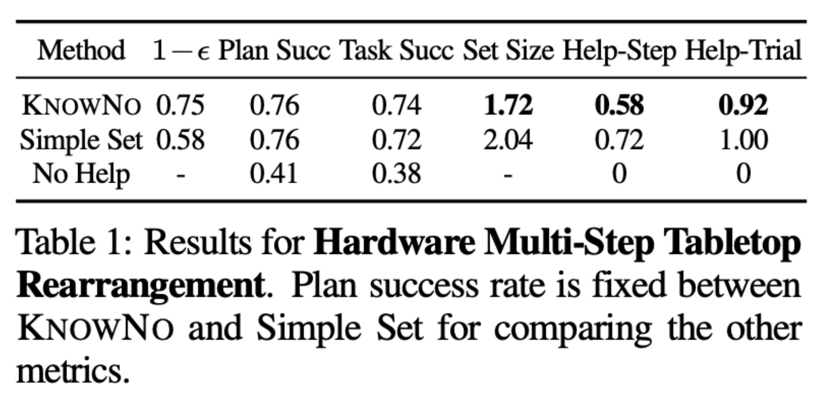
- Core: CP builds prediction sets C with guarantee $P\left(y_{test} \in C(\tilde{x}_{test})\right) \geq 1-\epsilon$.
- Nonconformity Scores: $\kappa_i = 1-\hat{f}\left(\tilde{x}_i\right)_{y_i}$on calibration set Z of N=400 examples.
- Quantile: $\hat{q} = \lceil (N+1)(1-\epsilon) \rceil / N$;Set: $C(\tilde{x}_{test}) = \left\{y \in Y \mid \hat{f}\left(\tilde{x}_{test}\right)_y \geq 1-\hat{q}\right\}$
- Decision: If |C|=1, execute; else, ask help (human selects or clarifies).
- Dataset-Conditional: Tighter bound $P(y_{test} \in C \mid Z) \geq Beta^{-1}_{N+1-v,v}(\delta)$,v = \lfloor (N+1)\hat{\epsilon} \rfloor.
- Multi-Step Guarantee: End-to-end success via product of per-step probs.



To generate $C(\tilde{x}_{\text{test}})$, CP first uses the LLM's confidence $\hat{f}$ (cf. Section 2) to evaluate the set of *nonconformity scores* $\{\kappa_i = 1-\hat{f}(\tilde{x}_i)_{y_i}\}_{i=1}^{N}$ over the calibration set — the higher the score is, the less each data in the calibration set *conforms* to the data used for training $\hat{f}$. Then CP performs calibration by defining $\hat{q}$ to be the $\frac{\lceil (N+1)(1-\epsilon) \rceil}{N}$ empirical quantile of $\kappa_1,\ldots,\kappa_N$. Lastly, CP generates $C(\tilde{x}_{\text{test}}) = \{y \in \mathcal{Y} \mid \hat{f}(\tilde{x}_{\text{test}})_y \geq 1-\hat{q})\}$, i.e., the prediction set that includes all labels that the predictor is at least $1-\hat{q}$ confident in. The generated prediction set ensures that the coverage guarantee in Eq. (1) holds.

Experiments - Setups and Baselines

- Tasks: 4 simulated (spatial, numeric, preference, Winograd); 2 real-robot (table-top with UR5e arm: spatial, preference).
- Calibration Data: 400 examples per task, i.i.d. from distribution.
- Models: PaLM 540B for sim; Flan-PaLM 540B for real.
- Metrics: Success Rate (≥1-ε, e.g., 90%), Help Rate (fraction of interventions), Efficiency (success/help).
- Baselines: Vanilla LLM (no help), Threshold (on raw probs), MC Dropout, Ensemble (5 LLMs), Socratic (prompt tuning for uncertainty).



Method	$1-\epsilon$	Plan Succ	Task Succ	Set Size	Help-Step	Help-Trial
KnowNo	0.75	0.76	0.74	**1.72**	**0.58**	**0.92**
Simple Set	0.58	0.76	0.72	2.04	0.72	1.00
No Help	-	0.41	0.38	-	0	0

Table 1: Results for **Hardware Multi-Step Tabletop Rearrangement**. Plan success rate is fixed between KnowNo and Simple Set for comparing the other metrics.

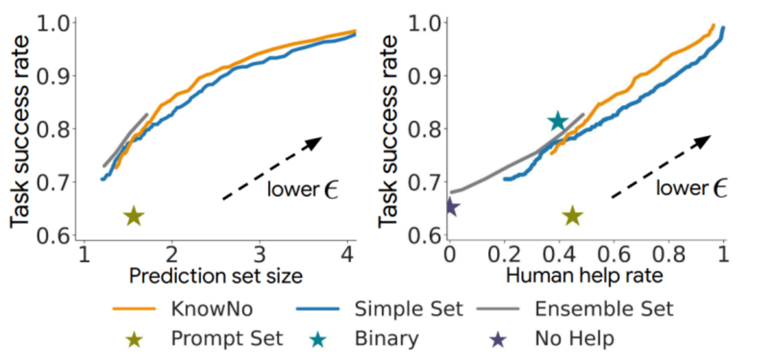
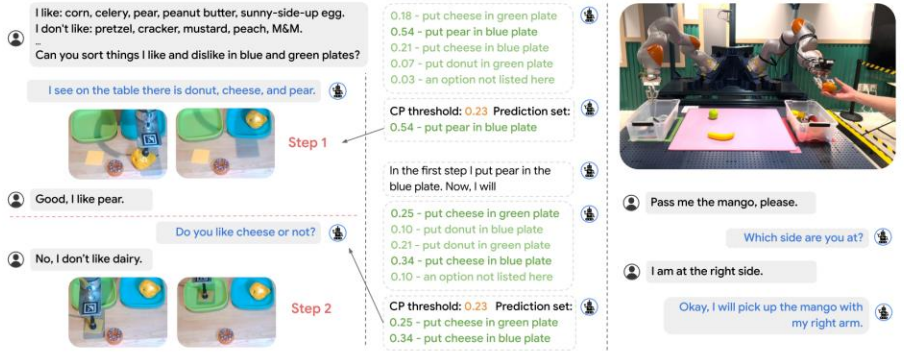


Figure 4: Comparison of task success rate vs average prediction set size (Left) and vs. human help rate (Right) in **Simulation** averaged over the three settings. 200 trials are run for each method. ϵ is varied from 0.25 to 0.01 for KnowNo, and from 0.6 to 0.01 for Simple Set and Ensemble Set. Binary and No Help are not shown on the left since prediction sets are not provided.

Results - Performance

- Simulated: KnowNo achieves 90% success with 10-24% help; outperforms baselines (e.g., Ensemble 28% help, Threshold 35%).
- Real-Robot: 90% success with 15% help on spatial, 20% on preference; better than Socratic (25% help).
- Multi-Step: End-to-end guarantee holds; help reduces with steps (compounds uncertainty).
- Ablations: CP tighter than baselines; MCQA better than sequence scoring.
- Key: 1.5-2x fewer interventions than prompt-tuned methods.



I like: corn, celery, pear, peanut butter, sunny-side-up egg.
I don't like: pretzel, cracker, mustard, peach, M&M.
...
Can you sort things I like and dislike in blue and green plates?

I see on the table there is donut, cheese, and pear.

0.18 - put cheese in green plate
0.54 - put pear in blue plate
0.21 - put cheese in blue plate
0.07 - put donut in green plate
0.03 - an option not listed here

CP threshold: 0.23 Prediction set:
0.54 - put pear in blue plate

Good, I like pear.

In the first step I put pear in the blue plate. Now, I will

0.25 - put cheese in green plate
0.10 - put donut in blue plate
0.21 - put donut in green plate
0.34 - put cheese in blue plate
0.10 - an option not listed here

Do you like cheese or not?

No, I don't like dairy.

CP threshold: 0.23 Prediction set:
0.25 - put cheese in green plate
0.34 - put cheese in blue plate

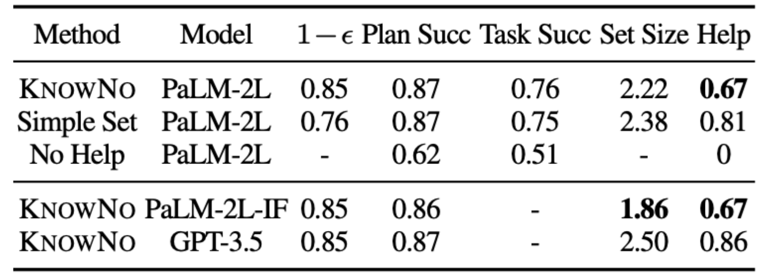
Pass me the mango, please.

Which side are you at?

I am at the right side.

Okay, I will pick up the mango with my right arm.

Figure 5: (Left) Multi-step CP is applied in **Hardware Tabletop Rearrangement**. (Right) CP models ambiguity in possible human locations and triggers clarification from the human in **Bimanual**.



Method	Model	$1-\epsilon$	Plan Succ	Task Succ	Set Size	Help
KnowNo	PaLM-2L	0.85	0.87	0.76	2.22	**0.67**
Simple Set	PaLM-2L	0.76	0.87	0.75	2.38	0.81
No Help	PaLM-2L	-	0.62	0.51	-	0
KnowNo	PaLM-2L-IF	0.85	0.86	-	**1.86**	**0.67**
KnowNo	GPT-3.5	0.85	0.87	-	2.50	0.86

Table 2: Results for **Hardware Mobile Manipulation**. Plan success rate is fixed between KnowNo and Simple Set to compare the other metrics.

Analysis and Conclusions

- Analysis: CP provides formal guarantees absent in baselines; effective on diverse ambiguities; scales with LLM size; help correlates with true uncertainty.
- Limitations: Requires calibration data (though small N=400); assumes human help available; may over-ask on edge cases.
- Conclusions: KnowNo is lightweight, out-of-box solution for calibrated LLM planners; promotes safe robotics; future: integrate with vision, more ambiguities.
- Broader Impact: Aligns uncertainty for trustworthy AI agents in action-oriented domains.

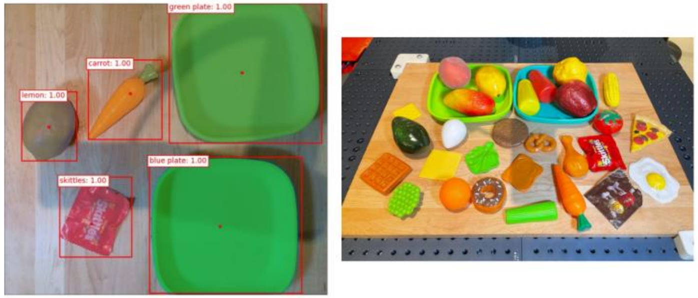
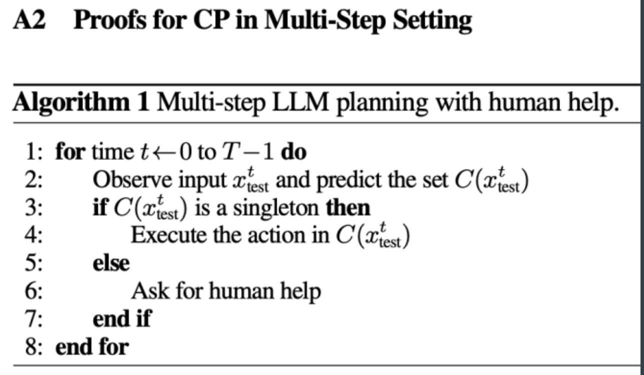


Figure A7: (Left) MDTER [60] object detection with Segment Anything [62] and most distant internal point (red dots) for Hardware Tabletop Rearrangement. (Right) The total 28 toy items used for the experiments.



A2 Proofs for CP in Multi-Step Setting

Algorithm 1 Multi-step LLM planning with human help.

```
for time $t \leftarrow 0$ to $T-1$ do
    Observe input $x_{\text{test}}^t$ and predict the set $C(x_{\text{test}}^t)$
    if $C(x_{\text{test}}^t)$ is a singleton then
        Execute the action in $C(x_{\text{test}}^t)$
    else
        Ask for human help
    end if
end for
```

Q&A

- Question?