Uncertainty Quantification: Modern ML and LLMs

Tom Zollo, Spring 2025

Roadmap

- Introduction
  - Why quantify uncertainty?
  - Types and sources of uncertainty
- Uncertainty quantification in (image) classification models
  - Estimating uncertainty
  - Calibration
  - Selective classification, DFUQ, Bayesian approaches
- Uncertainty quantification in LLMs
  - Why is it harder?
  - Baselines
  - Advanced methods
  - Asking for more information
- Future directions and closing thoughts

Part 0:

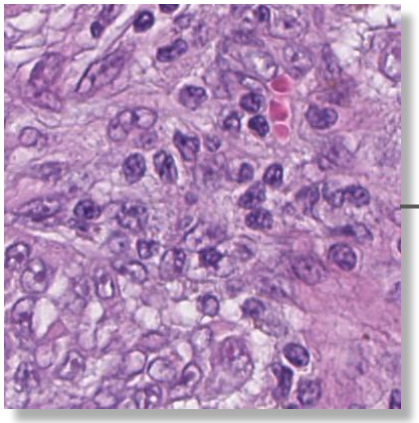
Introduction to Uncertainty Quantification in ML:

An Empiricist's Perscpective

Why Quantify Uncertainty?

In high-risk domains like **medicine,** education, and law, it is essential that we can quantify the uncertainty in the predictions of a machine learning model.

Patient Image



ML Model

70% probability of melanoma

Effective UQ enables downstream decision-makers and users to **trust predictions made by large, opaque neural networks**

Why Quantify Uncertainty?

Machine learning models are usually evaluated based on their **accuracy**.

➜ e.g., error rate at predicting “will this patient develop a diease?”

For a trustworthy system, must also know:

- How confident am I in this prediction?
- How does the quality of this prediction compare to other ones?
- Are there some predictions that should be ignored?
- Is there an unacceptably high likelihood of very bad outcomes?

Why Quantify Uncertainty?

Machine learning models are usually evaluated based on their **accuracy**.

➜ e.g., error rate at predicting “will this patient develop a diease?”

For a trustworthy system, must also know:

- How confident am I in this prediction?
- How does the quality of this prediction compare to other ones?
- Are there some predictions that should be ignored?
- Is there an unacceptably high likelihood of very bad outcomes?

Besides concerns about **copyright, fairness, robustness, alignment, etc.**

Types of Uncertainty

Aleatoric	Epistemic
 *Data Uncertainty*	*Model Uncertainty* 
 *Confidence in input data*	*Confidence in prediction* 
 *Randomness in data creation*	*Lack of knowledge* 

Sources of Uncertainty

Data	**Model**
● Missing ● Ambiguous ● Difficult ● etc.	Modelling decisions affect uncertainty estimates ● Architecture ● Ensembling ● etc.

Roadmap

- Introduction
  - Why quantify uncertainty?
  - Types and sources of uncertainty
- Uncertainty quantification in (image) classification models
  - Estimating uncertainty
  - Calibration
  - Selective classification, DFUQ, Bayesian approaches
- Uncertainty quantification in LLMs
  - Why is it harder?
  - Baselines
  - Advanced methods
  - Asking for more information
- Future directions and closing thoughts

Part 1:

Uncertainty Quantification in (Image) Classification

Image Classification

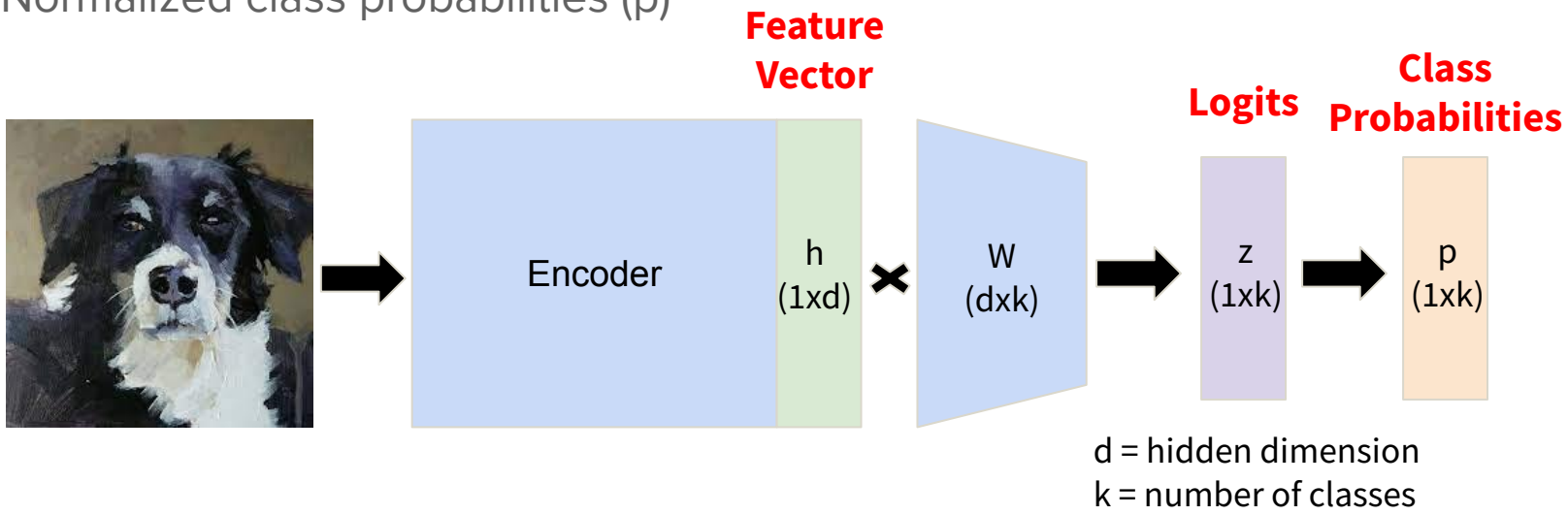
Image classification has driven much (most?) progress in deep learning

Source: ImageNet

Image Classification Models

For each image x in a k-class problem, model produces:

- Feature representation (h)
- (Unnormalized) logits (z)
- Normalized class probabilities (p)



Estimating Uncertainty with Image Models

For a given test instance (and without any additional data), how can we estimate (quantify) the uncertainty in the model's prediction?

➔ First, **what makes a good uncertainty estimate?**

- Higher scores for examples more likely to be wrong
- Lower scores for examples more likely to be correct

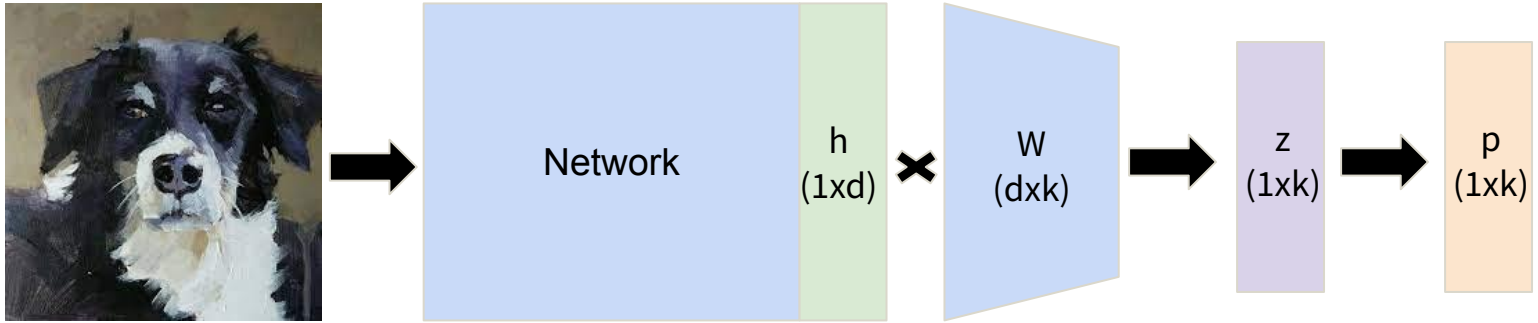
Given uncertainty scores and labels of whether or not the prediction is correct, **how can we measure the quality of (unscaled) uncertainty estimates?**

- E.g., AUROC

Estimating Uncertainty with Image Models

Baseline: Prediction Entropy

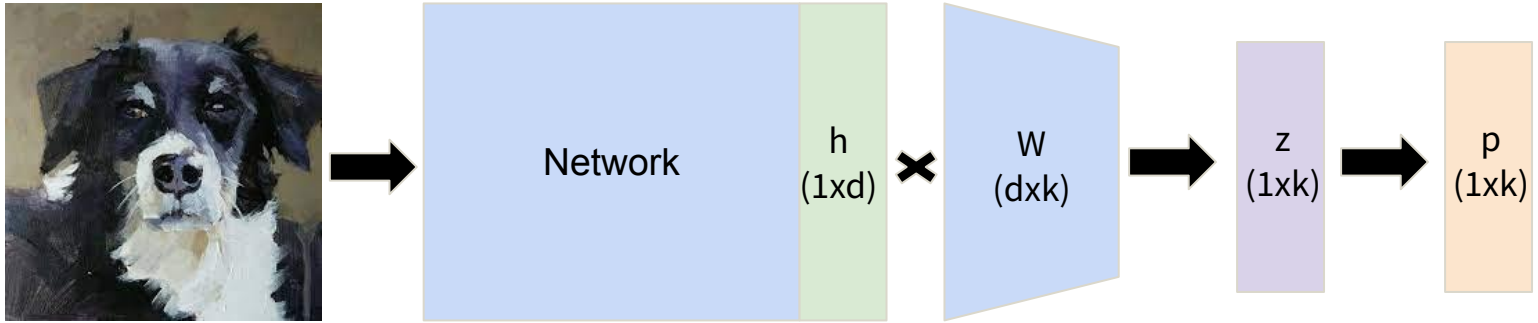
$$U = -\sum_{i=1}^{k} p_i \log(p_i)$$



Estimating Uncertainty with Image Models

Baseline: Top-class Confidence

$$U = 1 - \max_{1 \leq i \leq k} p_i$$

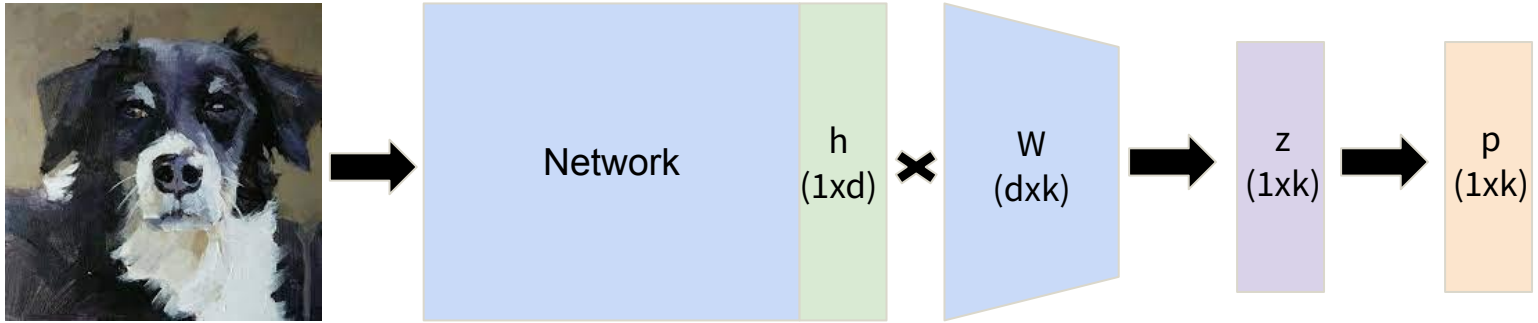


Estimating Uncertainty with Image Models

Baseline: Top-class Confidence

$$U = 1 - \max_{1 \leq i \leq k} p_i$$

“Confidence”



Calibration

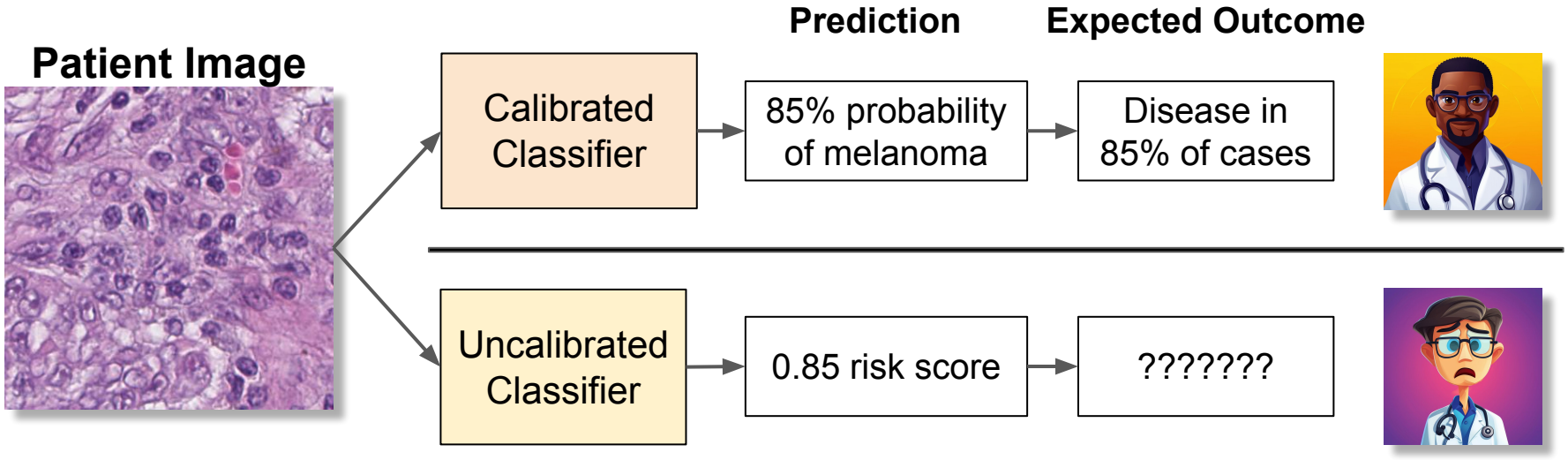
In machine learning, **calibration** typically refers to the quality where

prediction confidence matches the probability of correctness.

Then, e.g.,

for a well-calibrated ML model, 80% confidence implies 80% chance of correct prediction.

Calibration



Calibration

Definition: A model is perfectly calibrated if

$$P(\text{correct} \mid \text{confidence} = p) = p$$

Or...

$$P\big(y = \hat{y}(x) \mid \hat{p}(x) = c\big) = c, \quad \forall c \in [0, 1]$$

In other words, prediction confidence should match probability of correctness.

Measuring (Mis)calibration

Expected calibration error measures the expected difference between prediction confidence and correctness over the data distributuon

$$\mathrm{ECE} = \mathbb{E}\left[\left|P(y = \hat{y}(x) \mid \hat{p}(x)) - \hat{p}(x)\right|\right]$$

Accuracy **Conf.**

Measuring (Mis)calibration

Expected calibration error measures the expected difference between prediction confidence and correctness over the input distributuon.

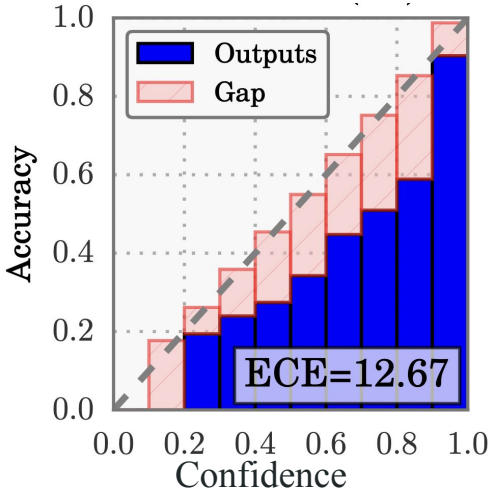
- Must estimate, typically via **binning**
  - Order by confidence estimate
  - Group by boundaries or mass
  - Calculate difference b/w average confidence and accuracy in each bin
  - Combine

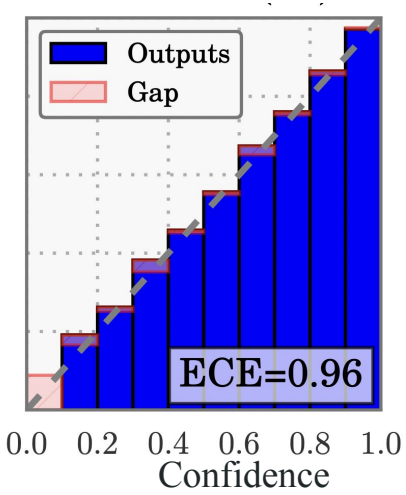
$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right|$$

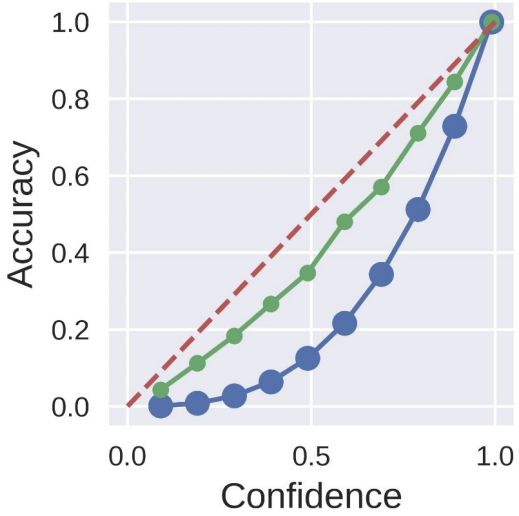
Visualizing (Mis)calibration

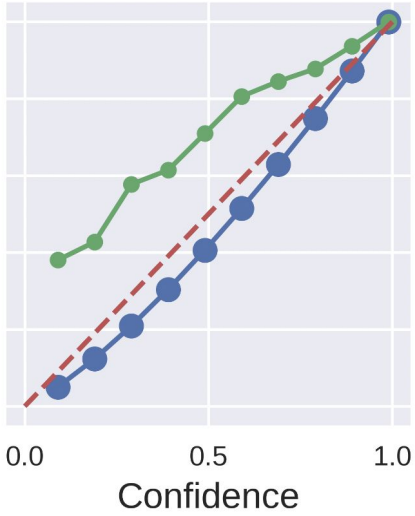
Reliability diagrams plot confidence vs. accuracy

➜ Perfect calibration lies on x=y line









src: https://arxiv.org/abs/1706.04599
and https://arxiv.org/abs/2410.05407

Measuring (Mis)calibration

ECE can be trivially minimized by predicting marginal class probabilities

$$\mathbf{ECE} = \mathbb{E}\left[\left|P(y = \hat{y}(x) \mid \hat{p}(x)) - \hat{p}(x)\right|\right]$$

e.g., always predict class at base rate.

Also helpful to consider some **proper scoring rules:**

- Brier score
- NLL

Recalibration

Sometimes, we have some extra target data lying around that we can use to **recalibrate** our model post-hoc

➜ Make the confidence scores more closely align with expected correctness

Usually this is a small amount of data, since if we had a lot we would do further fine-tuning

➜ Recalibration methods are typically simple and low-dimensional

Recalibration: Scaling

Platt scaling: learns sigmoidal scaling of the prediction confidence

$$p^{\text{Platt}}(x) = \frac{1}{1 + \exp(w\, p(x) + b)}$$

Tempature scaling: learns “temperature” for scaling logits

$$p^{\text{Temp}}(x) = \text{softmax}\left(\frac{z(x)}{T}\right)$$

Recalibration: Binning

Histogram binning:

- predictions assigned to bins (based on confidence ordering)
- all predictions in a bin share a score

$$p^{\text{Hist}}(x) = \theta_m \quad \text{if} \quad \hat{p}(x) \in (a_m, a_{m+1}]$$

The optimal bin predictions θ_m are determined by minimizing the bin-wise squared error:

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

src: https://dl.acm.org/doi/10.5555/645530.655658

Recalibration: Other Methods

Isotonic Regression: strict generalization of histogram binning in which the bin boundaries and bin predictions are jointly optimized.

Bayesian Binning into Quantiles: marginalizes out all possible binning schemes.

Many others... (e.g., based on KDEs)

Recalibration: Empirical Effectiveness

In general, recalibration with IID data improves miscalibration over pretrained model (exception is when ECE is already very low, <1-2%).

No “best” method (but I like temp. scaling and hist. binning).

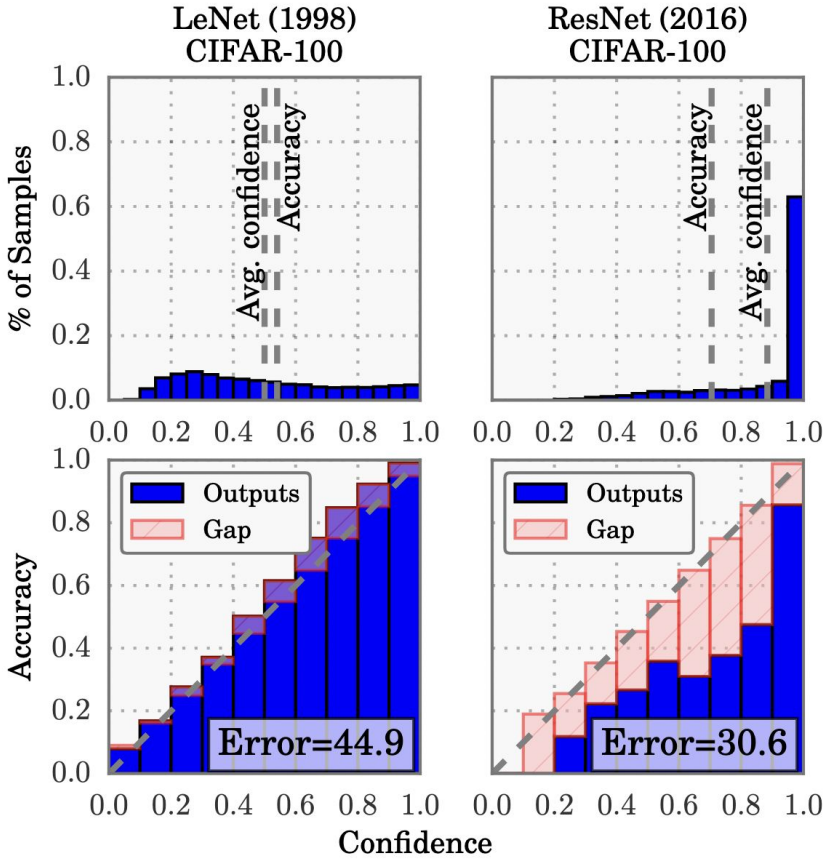
Dataset	Model	Uncalibrated	Hist. Binning	Isotonic	BBQ	Temp. Scaling
Birds	ResNet 50	9.19%	4.34%	5.22%	4.12%	**1.85%**
Cars	ResNet 50	4.3%	**1.74%**	4.29%	1.84%	2.35%
CIFAR-10	ResNet 110	4.6%	0.58%	0.81%	**0.54%**	0.83%
CIFAR-10	ResNet 110 (SD)	4.12%	0.67%	1.11%	0.9%	**0.6%**
CIFAR-10	Wide ResNet 32	4.52%	0.72%	1.08%	0.74%	**0.54%**

src: On Calibration of Modern Neural Networks

On Calibration of Neural Networks

Canonical paper in calibration for deep learning

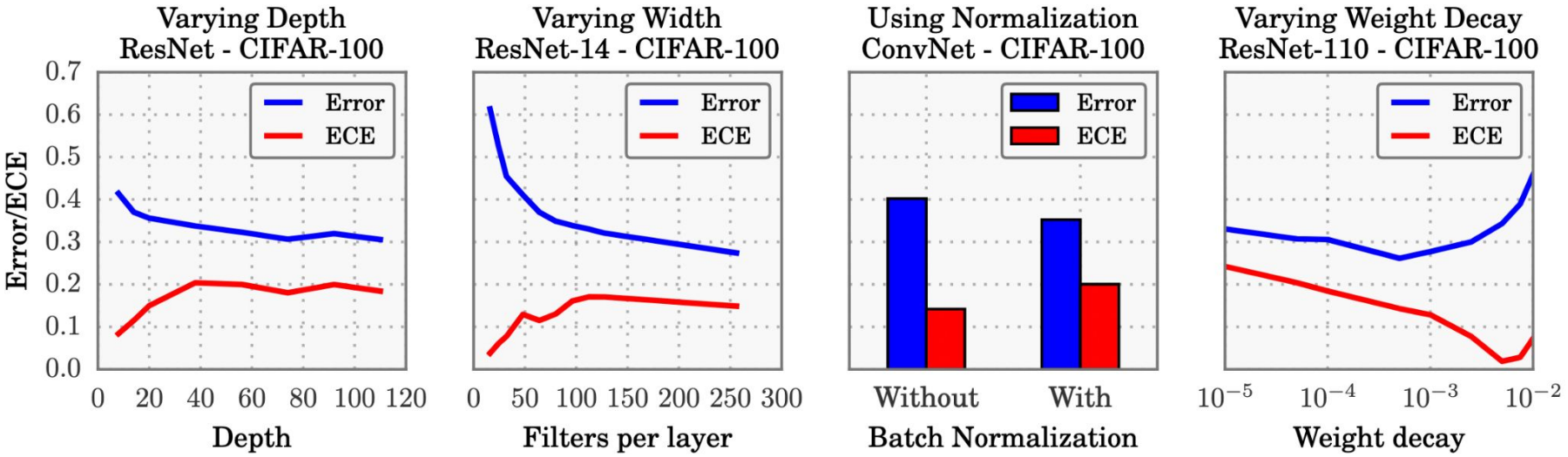
- Finds that advances in accuracy have come at the expense of increased miscalibration
- Also introduces temperature scaling



src: https://arxiv.org/abs/1706.04599

On Calibration of Neural Networks

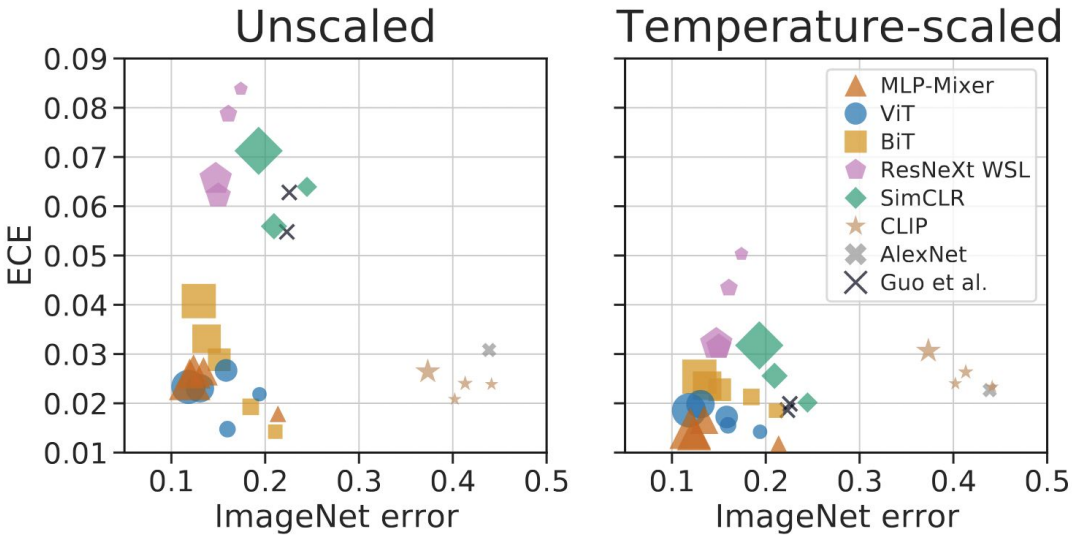
Find that **deep, wide networks** with **normalization** and **weight decay** (i.e., basically all modern networks) are more miscalibrated than networks without those qualities



Revisiting the Calibration of Neural Networks

Revisits findings of On Calibration:

- Find no tradeoff between accuracy and calibration
- Given temperature scaling, most accurate is (nearly) most calibrated
- Architecture matters



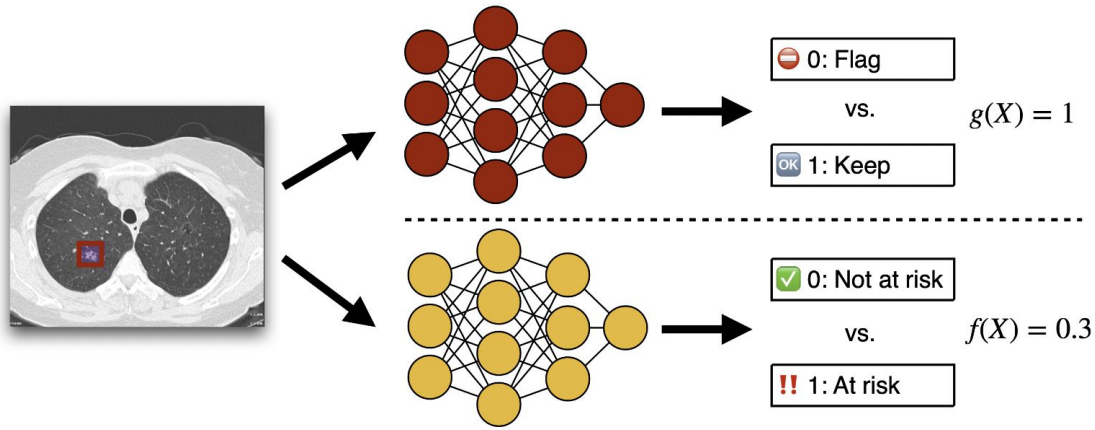
Now, generally accepted that supervised deep learning models will be pretty well-calibrated in the IID setting***

src: https://arxiv.org/abs/2106.07998

Selective Prediction

In selective prediction the goal is to build predictive systems that know when they should abstain from making a prediction

- Improve performance on remaining examples
- Ideal for human in-the-loop
  - Enhance trust of decision-makers
  - Efficiently allocate human/AI resources



src: https://arxiv.org/abs/2208.12084

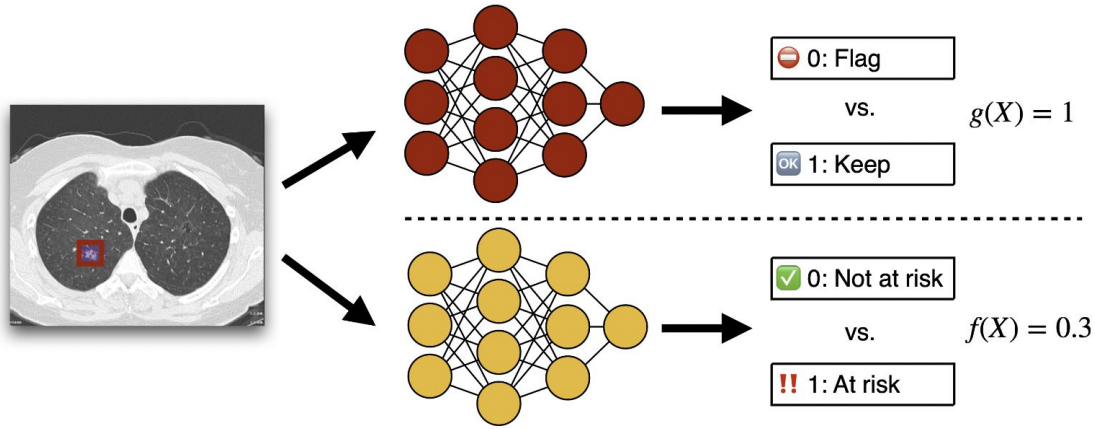
Selective Prediction

In selective prediction the goal is to build predictive systems that know when they should abstain from making a prediction

Baseline: Uncertainty scores

- Predictive entropy
- Top confidence

Also: LR on embeddings



SelectiveNet

Jointly optimize predictor ***f*** and selector ***g***, so that system is optimal for covered domain

$$(f, g)(x) \triangleq \begin{cases} f(x), & \text{if } g(x) = 1; \\ \text{don't know}, & \text{if } g(x) = 0. \end{cases}$$

$$\theta^* = \arg\min_{\theta \in \Theta} (R(f_\theta, g_\theta))$$ **Selective risk**

$$s.t.\ \phi(g_\theta) \geq c.$$ **Coverage**

src: https://arxiv.org/abs/1901.09192

SelectiveNet

The true selective risk and true coverage have corresponding empirical counterparts that can be calculated for any given labeled set S_m. The *empirical selective risk* is

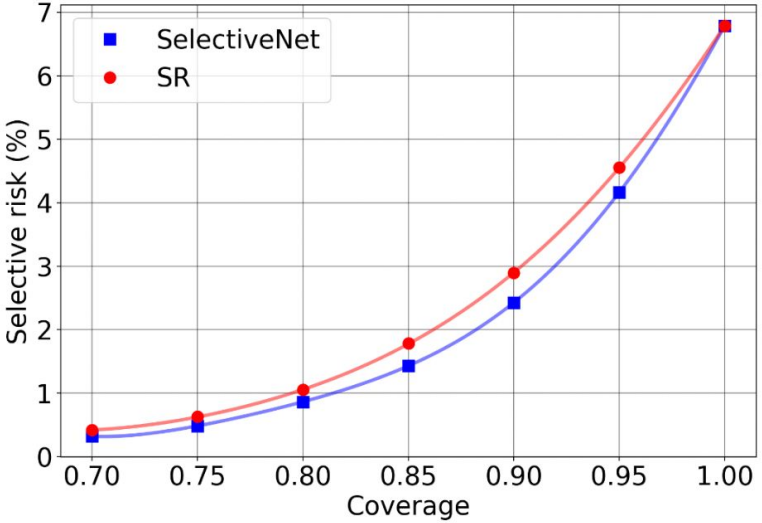
$$\hat{r}(f, g|S_m) \triangleq \frac{\frac{1}{m}\sum_{i=1}^{m} \ell(f(x_i), y_i) g(x_i)}{\hat{\phi}(g|S_m)},$$

and the *empirical coverage* is

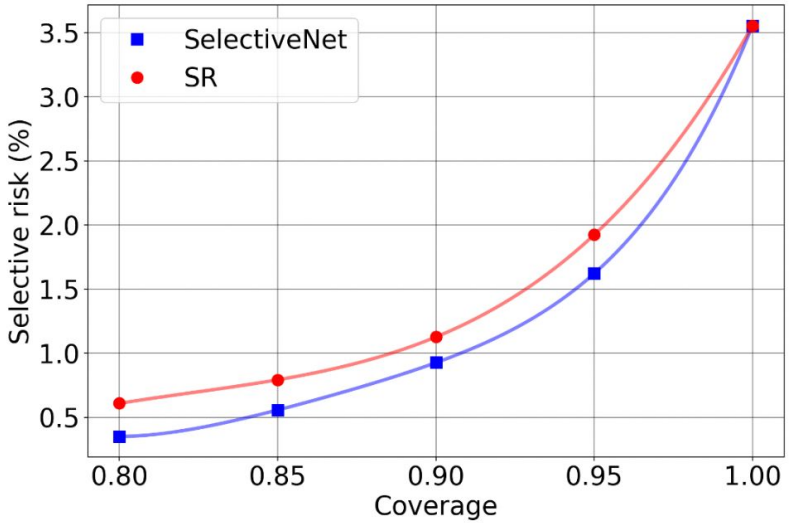
$$\hat{\phi}(g|S_m) \triangleq \frac{1}{m}\sum_{i=1}^{m} g(x_i).$$

Learn with soft constraint, find threshold with validation set...

SelectiveNet



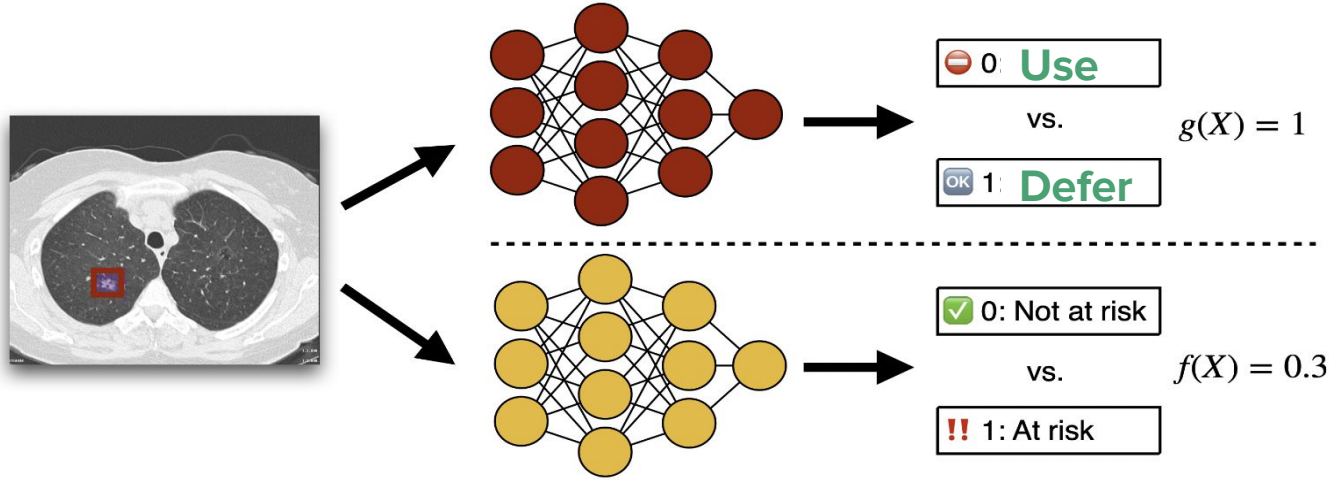
(a) Cifar-10



(b) Cats vs. dogs

Learning to Defer

Train selection model to consider not only what the ML model knows, but also what the expert in the loop knows

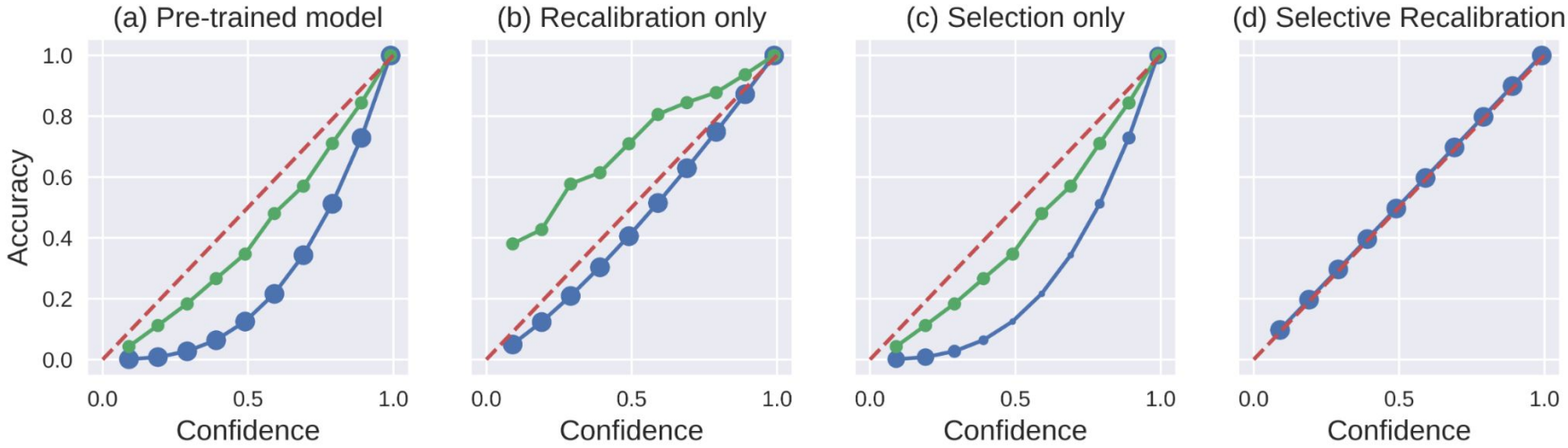


$$-\sum_i [(1 - s_i)\ell(Y_i, \hat{Y}_{M,i}) + s_i \ell(Y_i, \hat{Y}_{D,i})]$$

src: https://arxiv.org/abs/1711.06664

Selective Recalibration

Since many recalibrators are simple functions, can benefit from ignoring some complexity in input space, especially if jointly optimized with selector



src: https://arxiv.org/abs/2410.05407

Out-of-Distribution Detection

- Similar in spirit to selective prediction
  - Same baselines
- Aims to reject all examples that are OOD, vs. a pre-set percentage

Distribution-Free Uncertainty Quantification

Family of UQ methods gaining increasing popularity in the DL community.

Uncertainty quantification performed on the population/distribution level, e.g.:

- 90% probability the correct class will be in prediction set
- 95% proability toxicity will be below some chosen threshold

Distribution-free means:

- does not depend on the underlying model
- does not depend on the underlying distribution
- just need i.i.d. samples from target distribution (valid in finite samples)

Conformal prediction

- Simple framework for creating distribution-free prediction sets that come with rigorous statistical guarantees on coverage.
  - Prediction set: for a k-class problem, return m predictions, 1<=m<=k
  - e.g. produce sets that contain ground truth class 90% of the time.
- When the model is less certain, reflect that with larger prediction sets.
- Simple, effective algorithms using **softmax output of classifier.**

src: https://arxiv.org/abs/2107.07511

Forming Conformal Prediction Sets

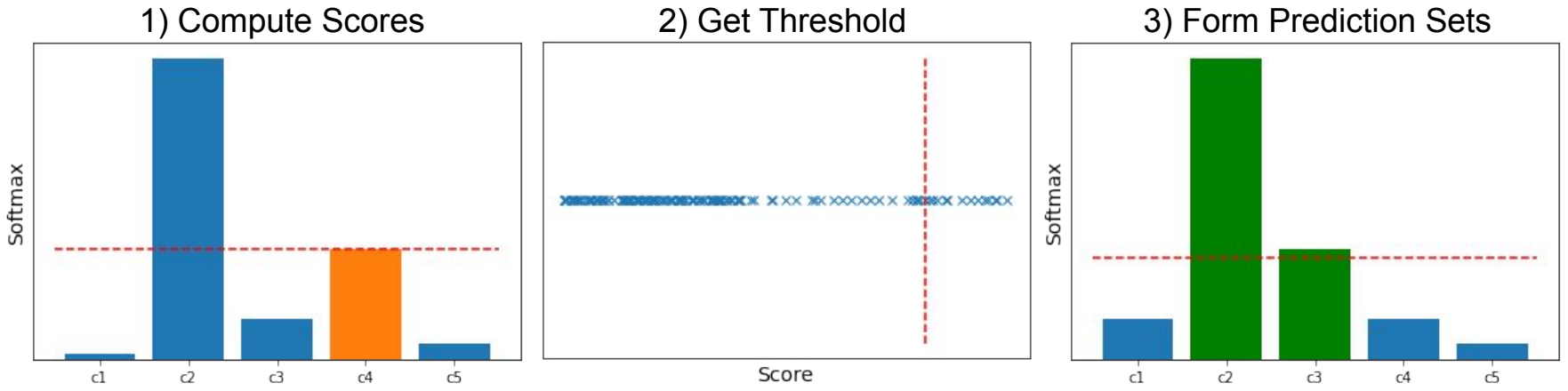
For each item in validation set, calculate non-conformity score

Compute $\hat{q}$ as the $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$ quantile of non-conformity scores

Form prediction sets using quantile score:

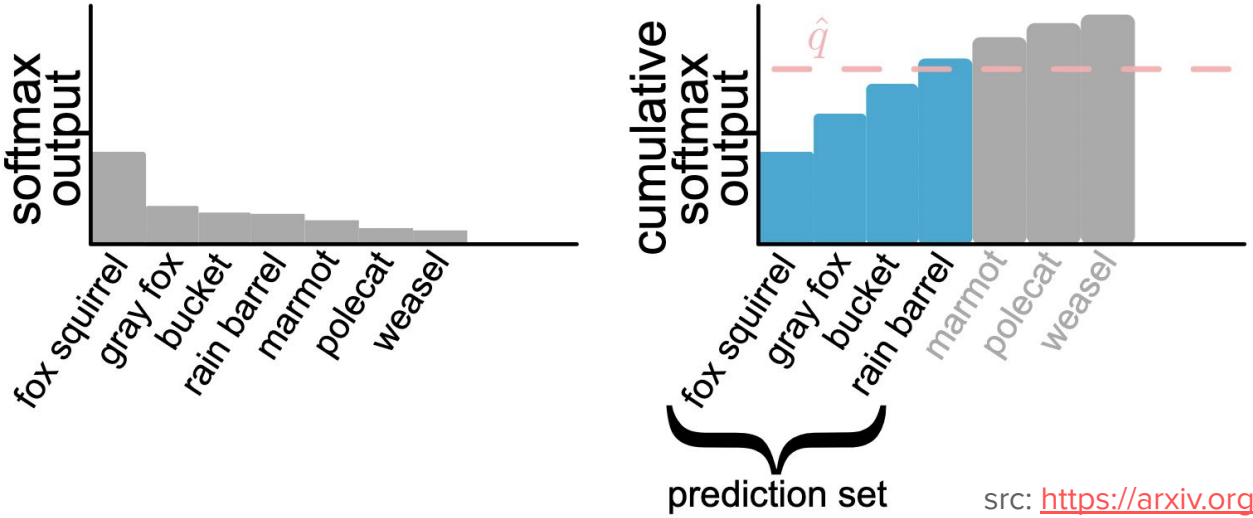
$$C(X_{test}) = \{y : s(X_{test}, y) \leq \hat{q}\}$$

Non-conformity score:
1 - (top confidence)



Adaptive Prediction Sets

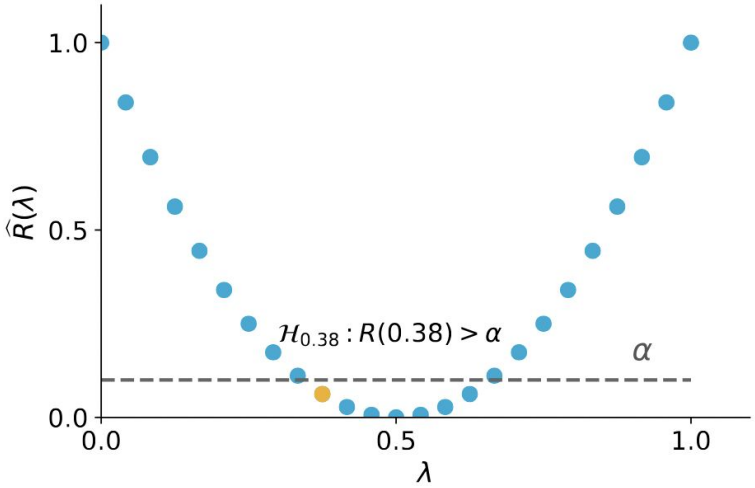
- Typical CP uses (one minus) softmax response as non-conformity score
- In APS, classes are included from most to least likely until their cumulative softmax output exceeds the quantile.
  - Performs favorably on **conditional coverage**

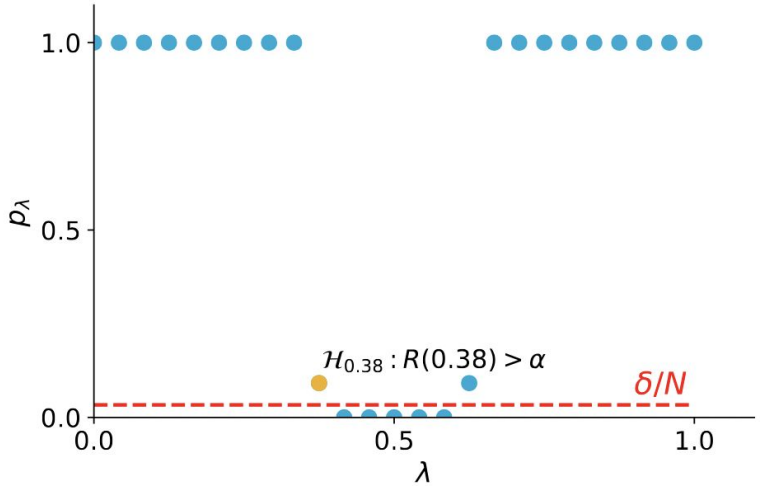


src: https://arxiv.org/abs/2006.02544

Learn Then Test

- Test finite set of hypotheses for control (high-prob. risk bound below some threshold) of a **general loss function**
- Multiple hypothesis testing
- Returns a set of “safe” predictors (possibly empty)

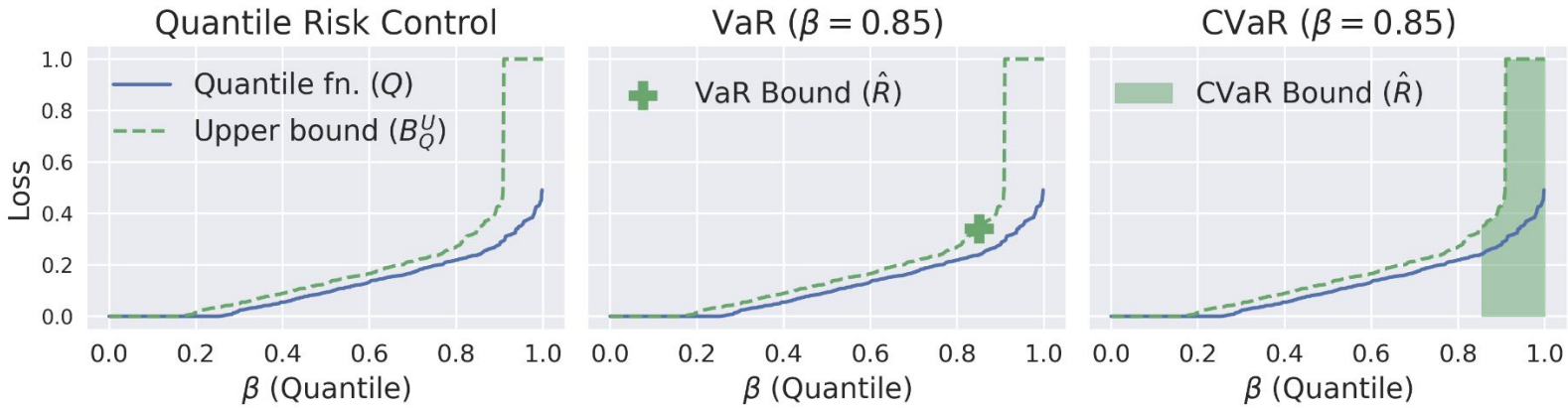




src: https://arxiv.org/abs/2110.01052

More General Risk Control: Quantile-Based Measures

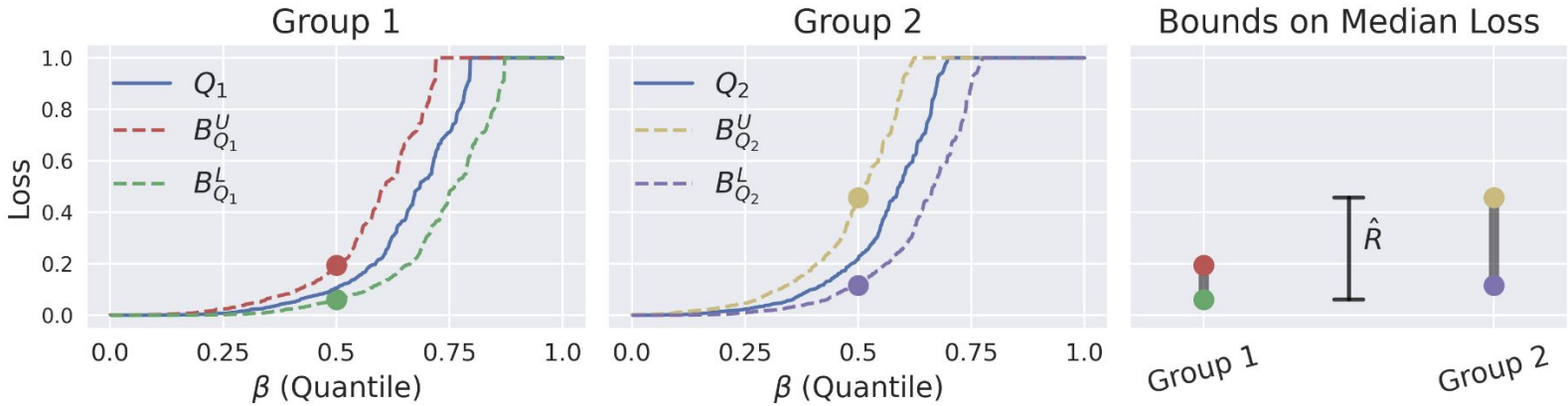
More recent work has extended these high-probability bounds to cover important tail quantities like VaR and CVaR...



src: https://arxiv.org/abs/2212.13629

More General Risk Control: Dispersion Measures

...as well as statistical dispersion measures, such as the Gini coefficient or difference in median loss between groups



src: https://arxiv.org/abs/2309.13786

Bayesian Approaches

Popular Bayesian approaches to UQ in neural networks:

- Bayesian neural networks
- MC-Dropout
- Deep Ensembles

Bayesian Neural Networks

- Place probability distributions over weights, allowing direct uncertainty estimation.
- Inference usually requires approximate methods (e.g., variational inference, Monte Carlo Dropout, Markov Chain Monte Carlo) due to computational intractability of exact Bayesian inference.

Strength: principled uncertainty estimates.

Weaknesses: higher computational complexity and sensitivity to choice of priors.

Bayesian Neural Networks

Some popular works:

- Evidential Deep Learning to Quantify Classification Uncertainty
  - https://arxiv.org/abs/1806.01768
- Predictive Uncertainty Estimation via Prior Networks
  - https://arxiv.org/abs/1802.10501
- Epistemic Neural Networks
  - https://arxiv.org/abs/2107.08924

Monte-Carlo Dropout

Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning https://arxiv.org/abs/1506.02142

- Cast dropout training in deep neural networks (NNs) as approximate Bayesian inference in deep Gaussian processes.
- Algorithm:
  - Perform dropout at training **and test time**
  - During test, ensemble over repeated samplings with dropout
  - In practice, this is equivalent to performing stochastic forward passes through the network and averaging the results.

Deep Ensembles

Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles
https://arxiv.org/abs/1612.01474

- Deep Ensembles can be viewed as drawing multiple “samples” from a model space—mirroring the idea of placing distributions over parameters in Bayesian approaches.
- Each network in the ensemble is trained independently, avoiding computationally demanding techniques often seen in Bayesian neural networks.
- Empirically, Deep Ensembles often outperform alternative uncertainty quantification methods (e.g., MC Dropout), delivering better calibration and predictive accuracy in both classification and regression tasks.

Deep Ensembles

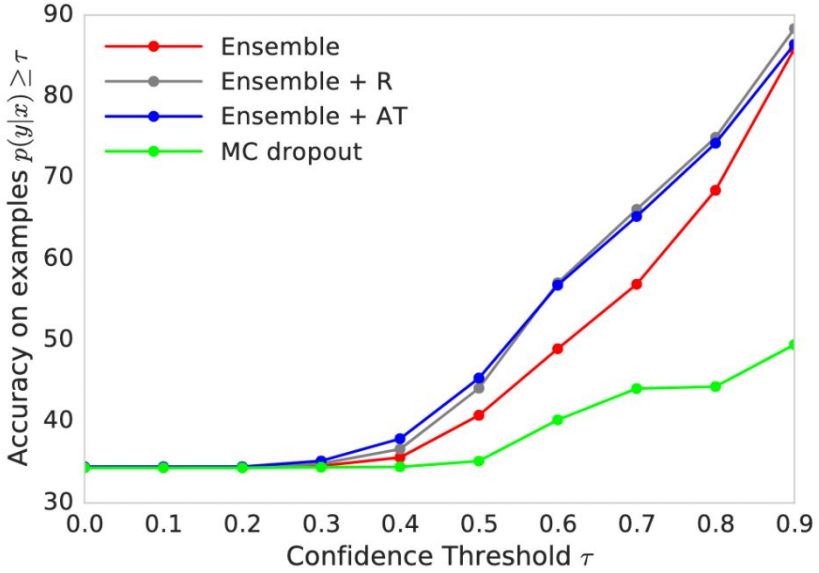


Figure 6: Accuracy vs Confidence curves: Networks trained on MNIST and tested on both MNIST test containing known classes and the NotMNIST dataset containing unseen classes. MC-dropout can produce overconfident wrong predictions, whereas deep ensembles are significantly more robust.

Part 1: Summary

Uncertainty quantification in image classification models

- Why quantify uncertainty?
- Estimating uncertainty
- Calibration and recalibration
- Selective prediction
- Distribution-free uncertainty quantification
- Bayesian approaches

Roadmap

- Introduction
  - Why quantify uncertainty?
  - Types and sources of uncertainty
- Uncertainty quantification in (image) classification models
  - Estimating uncertainty
  - Calibration
  - Selective classification, DFUQ, Bayesian approaches
- Uncertainty quantification in LLMs
  - Why is it harder?
  - Baselines
  - Advanced methods
  - Asking for more information
- Future directions and closing thoughts

Part 2:

Uncertainty Quantification in Large Language Models

Uncertainty in Natural Language

Given some input context to a language model, there are usually many possible valid responses.

- Input may be reasonably interpreted to have multiple different meanings.
  - Vague (“She watched the man with binoculars”)
  - Complex (as some reading comprehension questions are, even for humans)
  - Contain spelling or other errors.
- Some queries are inherently open-ended and allow for many reasonable responses.
  - request to complete a fictional story, tell a joke, or give a position on some political or social issue.
- For fixed input interpretation, equivalent answers may be expressible in many ways.
  - For example, given the input context “What is the capital of Rwanda?”...
    - “Kigali is the capital of Rwanda” and “The capital of Rwanda is Kigali”
  - ...offer semantically equivalent answers with different surface forms

src: https://arxiv.org/abs/2307.15703

Uncertainty in Natural Language

Training	Test
● Dataset is small or not sufficiently general. ● Data may include errors, such as “The capital of Rwanda is Berlin.” ● Large amount of ambiguous language. ● Also: conflicting or outdated information.	● Queries could be ambiguous. ● Tasks or instructions could be open-ended or under-prescribed. ● Relevant information could be excluded from the context. ● Users may produce input errors.

Why is UQ Harder in LLMs?

- Image clf. models give us direct access to their (task) predictive distribution.
- LLMs make a sequence of classification decisions over tokens to produce strings of arbitrary length:

$$P(X) = \prod_{i=1}^{n} P(x_i | x_1, x_2, \ldots, x_{i-1})$$

- Comprehensively measuring the uncertainty in language model generations may require **knowledge of the distribution over all possible sequences.**
- Often a disconnect between token prediction task and actual task being performed.

Naive Approach: Token Probabilities

- LLMs make a sequence of classification decisions over tokens:

$$P(X) = \prod_{i=1}^{n} P(x_i | x_1, x_2, \ldots, x_{i-1})$$

- Probability assigned to each token reflects its **plausibility** as the next word given the context under the training distribution.
- Joint probability can give uncertainty score for a given response.
  - Possibly length normalized

However, this may not be the “uncertainty” that we’re interested in...

Plausibility vs. correctness

Given the context

Question: What is the shape of the earth?

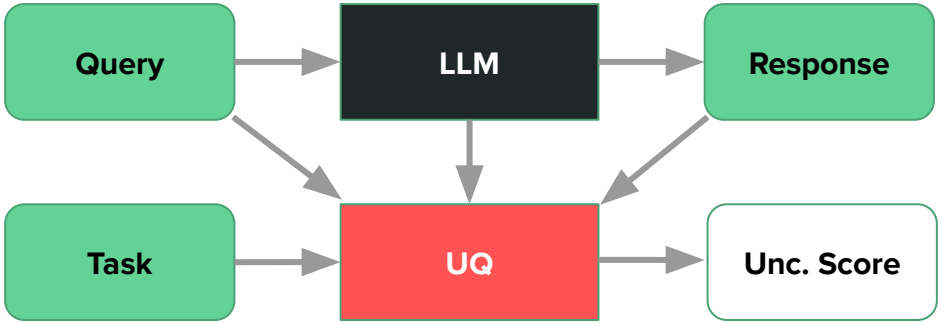
the completion

Answer: The shape of the earth is flat.

- may be plausible under training distribution scraped from the internet.
- should be assigned (near) zero probability as correct answer in QA task.

We need some notion of **task uncertainty.**

Task uncertainty



Given the **query** and **task definition**, is the LLM **response** likely to be “correct”?

- Notion of “correct” depends on the application, domain, and even user.
  - There may be many correct answers.
- Given good task uncertainty scores, we can do recalibration, selection, OOD detection, etc.

Sampling and self-consistency

Repeated samples can be drawn from LLM and compared to estimate uncertainty in a given reponse.

Semantic Uncertainty

Repeated samples can be drawn from LLM and compared to estimate uncertainty in a given reponse.

For each input example:

- Produce many samples from LLM
- **Cluster samples that map to same answer**
- Compute entropy/confidence using distribution over answer clusters

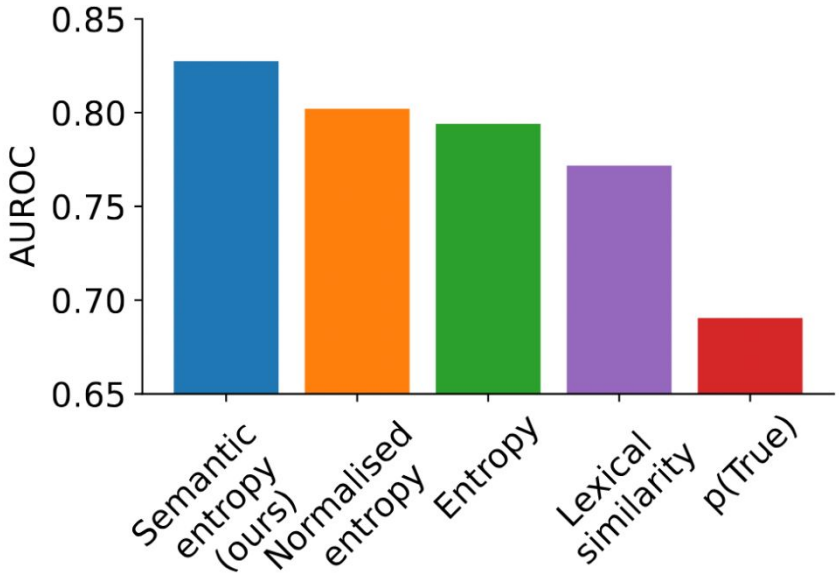
Question: What is the capital of France?

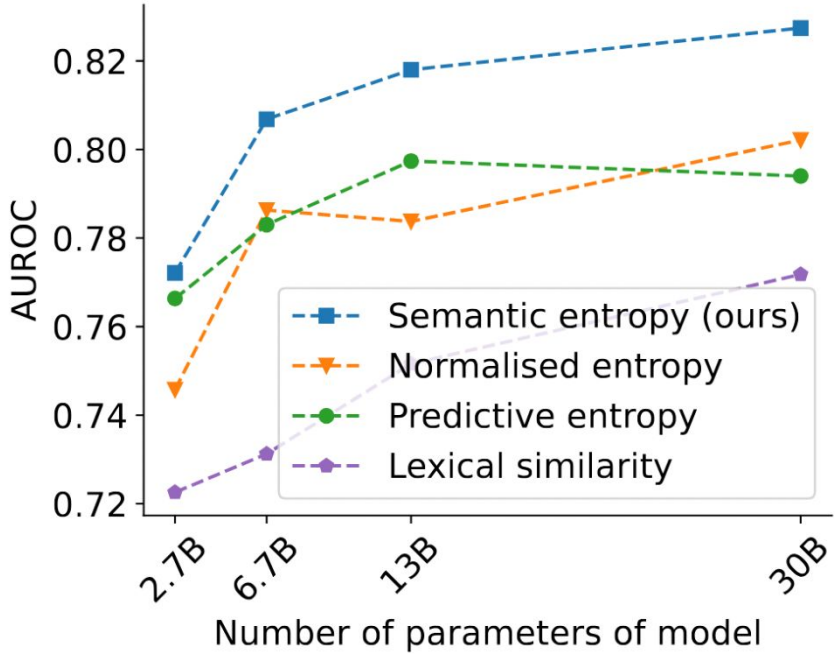
Answer $\mathbf{s}$	Likelihood $p(\mathbf{s} \mid x)$	Semantic likelihood $\sum_{\mathbf{s} \in c} p(\mathbf{s} \mid x)$
Paris	0.5	0.9
It's Paris	0.4	
London	0.1	0.1
Entropy	0.94	0.33

Semantic Entropy, Kuhn et al. 2022

src: https://arxiv.org/abs/2302.09664

Semantic Entropy



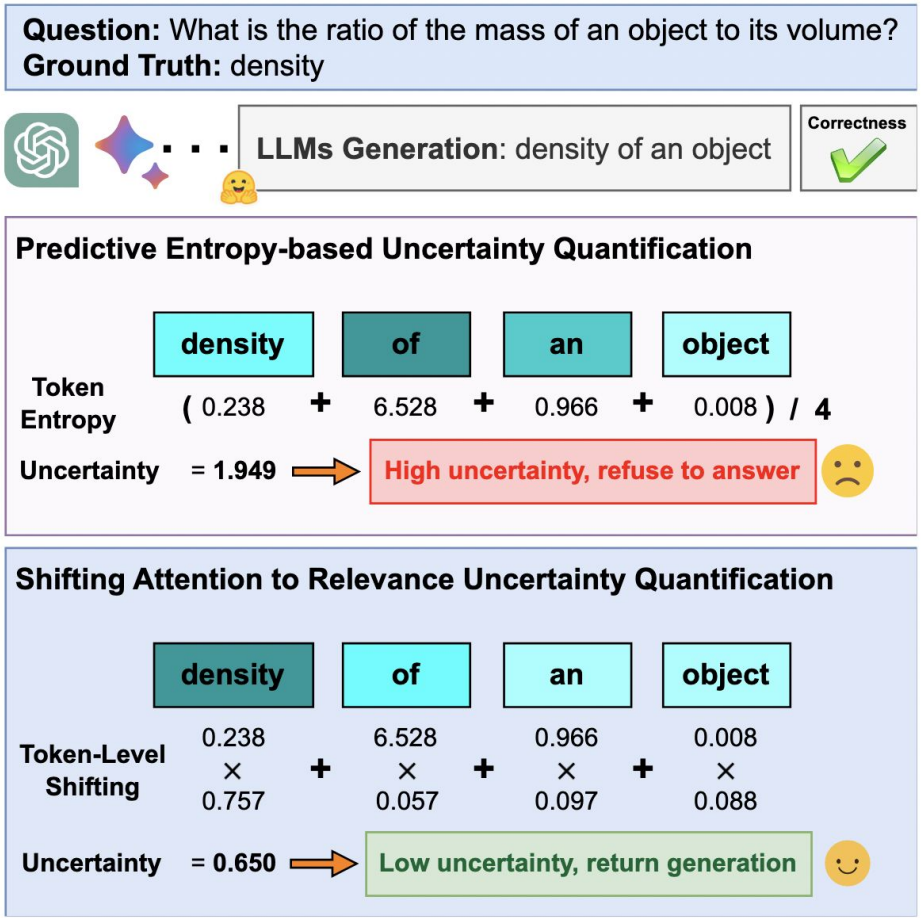


src: https://arxiv.org/abs/2302.09664

Token Relevance

Methods that use token probs (including baselines and semantic entropy) treat all tokens equally

➔ However, can we do better by **focusing on the probabilities of important tokens?**



src: https://arxiv.org/abs/2307.01379

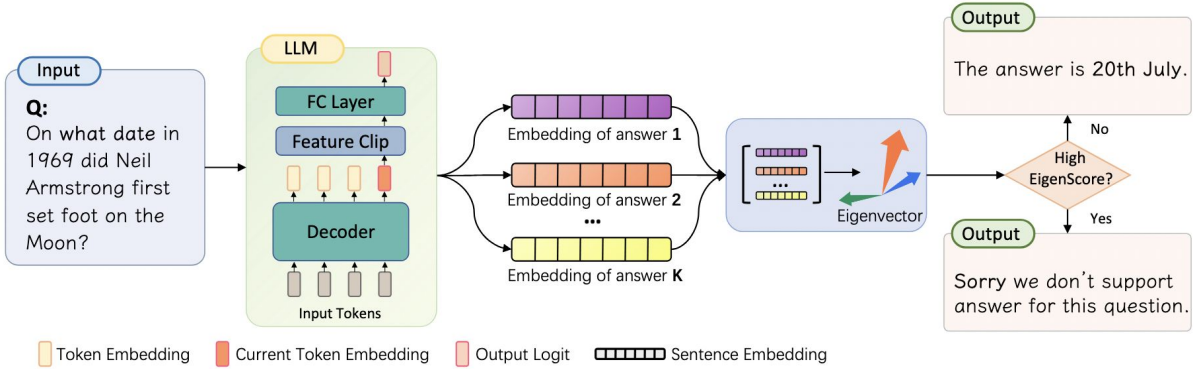
Shifting Attention to Relevance

Models & Datasets	*LS*	*PE*	*LN-PE*	*SE*	TOKEN*SAR* (ΔSE)	SENT*SAR*(ΔSE)	*SAR*(ΔSE)
Vicuna-13b w./ 5 generations are generated for each question							
Trivia QA	0.560	0.690	0.624	0.630	0.692 (+6.2%)	0.745 (+11.5%)	**0.749** (+11.9%)
SciQ	0.589	0.708	0.668	0.675	0.706 (+3.1%)	**0.745** (7.0%)	0.741 (+6.6%)
Vicuna-33b w./ 5 generations are generated for each question							
Trivia QA	0.565	0.644	0.639	0.651	0.652 (+0.1%)	**0.715** (+6.4%)	0.710 (5.9%)
SciQ	0.584	0.665	0.668	0.674	0.665 (-0.9%)	**0.717** (+4.3%)	0.710 (+3.6%)
WizardLM-13b w./ 5 generations are generated for each question							
Trivia QA	0.519	0.647	0.615	0.634	0.657 (+2.3%)	0.743 (+10.9%)	**0.744** (+11.0%)
SciQ	0.574	0.677	0.638	0.649	0.681 (+3.2%)	**0.719** (+7.0%)	0.707 (+5.8%)
LLaMA-2-13b-chat w./ 5 generations are generated for each question							
Trivia QA	0.504	0.647	0.615	0.622	0.654 (+3.2%)	0.698 (+7.6%)	**0.704** (+8.2%)
SciQ	0.578	0.718	0.688	0.692	0.718 (+2.6%)	**0.737** (+4.5%)	0.725 (+3.3%)
Average	0.555	0.675	0.644	0.653	0.678 (+2.5%)	**0.727** (+7.4%)	0.724 (+7.1%)

Looking Inside the Model

We've mostly focused on logits and token probs, what about embeddings?

- Compute the covariance matrix of embeddings of multiple generated answers to the same query.
- Use the log determinant (EigenScore) of this covariance matrix to capture semantic divergence.



src: https://arxiv.org/abs/2402.03744

Looking Inside the Model

We've mostly focused on logits and token probs, what about embeddings?

- Compute the covariance matrix of embeddings of multiple generated answers to the same query.
- Use the log determinant (EigenScore) of this covariance matrix to capture semantic divergence.

Models	Datasets / Methods	CoQA			SQuAD			NQ			TriviaQA		
		AUC_s	AUC_r	PCC	AUC_s	AUC_r	PCC	AUC_s	AUC_r	PCC	AUC_s	AUC_r	PCC
LLaMA-7B	Perplexity	64.1	68.3	20.4	57.5	60.0	10.2	74.0	74.7	30.1	**83.6**	83.6	54.4
	Energy	51.7	54.7	1.0	45.1	47.6	-10.7	64.3	64.8	18.2	66.8	67.1	29.1
	LN-Entropy	68.7	73.6	30.6	70.1	70.9	30.0	72.8	73.7	29.8	83.4	83.2	54.0
	Lexical Similarity	74.8	77.8	43.5	74.9	76.4	44.0	73.8	75.9	30.6	82.6	**84.0**	55.6
	EigenScore	**80.4**	**80.8**	**50.8**	**81.5**	**81.2**	**53.5**	**76.5**	**77.1**	**38.3**	82.7	82.9	**57.4**
LLaMA-13B	Perplexity	63.2	66.2	20.1	59.1	61.7	14.2	73.5	73.4	36.3	**84.7**	**84.5**	56.5
	Energy	47.5	49.2	-5.9	36.0	39.2	-20.2	59.1	59.8	14.7	71.3	71.5	36.7
	LN-Entropy	68.8	72.9	31.2	72.4	74.0	36.6	74.9	75.2	39.4	83.4	83.1	54.2
	Lexical Similarity	74.8	77.6	44.1	77.4	79.1	48.6	74.9	76.8	40.3	82.9	84.3	57.5
	EigenScore	**79.5**	**80.4**	**50.2**	**83.8**	**83.9**	**57.7**	**78.2**	**78.1**	**49.0**	83.0	83.0	**58.4**
OPT-6.7B	Perplexity	60.9	63.5	11.5	58.4	69.3	8.6	76.4	77.0	32.9	**82.6**	**82.0**	**50.0**
	Energy	45.6	45.9	-14.5	41.6	43.3	-16.4	60.3	58.6	25.6	70.6	68.8	37.3
	LN-Entropy	61.4	65.4	18.0	65.5	66.3	22.0	74.0	76.1	28.4	79.8	80.0	43.0
	Lexical Similarity	71.2	74.0	38.4	72.8	74.0	39.3	71.5	74.3	23.1	78.2	79.7	42.5
	EigenScore	**76.5**	**77.5**	**45.6**	**81.7**	**80.8**	**49.9**	**77.9**	**77.2**	**33.5**	80.3	80.4	0.485

src: https://arxiv.org/abs/2402.03744

Calibration in LLMs

- Baselines and semantic entropy do not give probabilistic scores
- Could draw many samples and measure self-consistency, but:
  - Expensive
  - Difficult in free-form settings
- How can we effectively and efficiently get probabilistic estimates of task correctness?

Calibration in LLMs

- Baselines and semantic entropy do not give probabilistic scores
- Could draw many samples and measure self-consistency, but:
  - Expensive
  - Difficult in free-form settings
- How can we effectively and efficiently get probabilistic estimates of task correctness?

Just ask the LLM!!!

Verbalizing confidence

An LLM can be prompted to verbalize its confidence in a given response, for example by instructing model to:

- Multiple choice: e.g., predict if a given answer is true or false.

```
Question: Who was the first president of the United States?
Proposed Answer: George Washington
Is the proposed answer:
 (A) True
 (B) False
The proposed answer is:
```

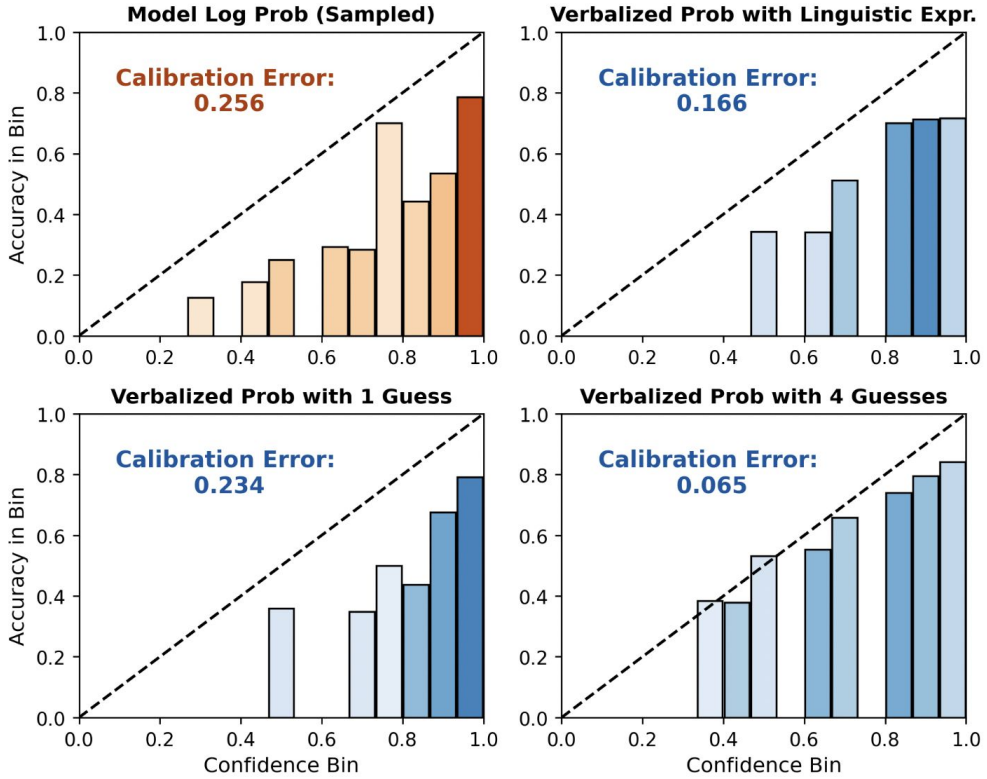
Kadavath et al., 2022

- Verbalization: estimate its confidence using marker words (e.g. probably, definitely) or numbers (e.g. 70%, 1-5 scale).

src: https://arxiv.org/abs/2207.05221

Verbalizing Confidence

- **Top left:** baseline
- **Top right:** linguistic markers associated with confidence levels
- **Bottom:** direct confidence verbalization, with one (left) or multiple (right) guess(es)



src: https://arxiv.org/abs/2305.14975

Limitations to Verbalizing Confidence

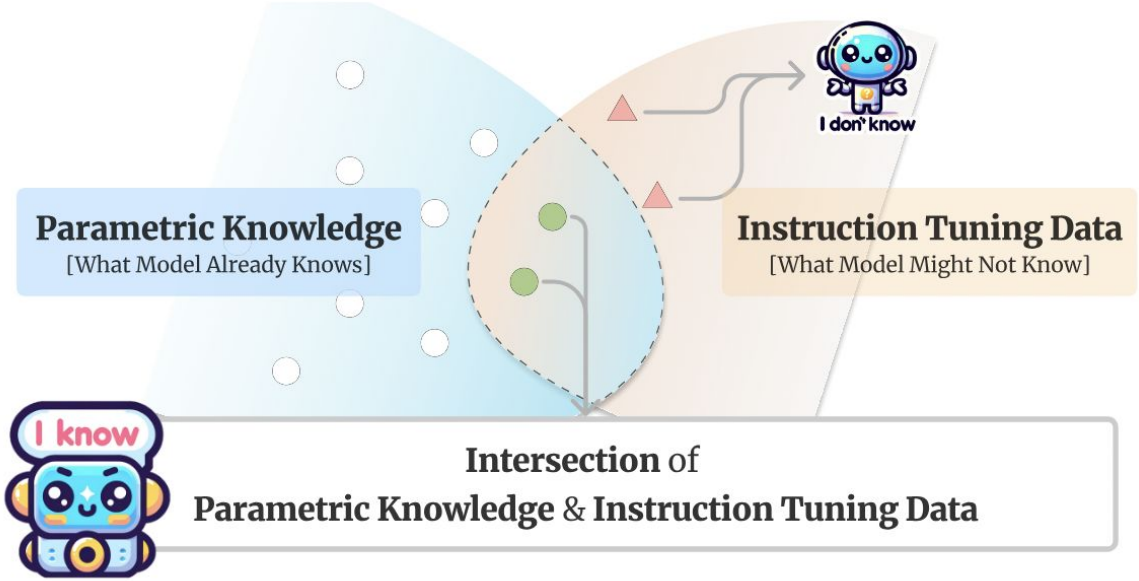
An LLM can be prompted to verbalize its confidence in a given response, however:

- One study suggests LLMs mimic observed language use, rather than truly reflecting epistemic uncertainty. (Zhou, Jurafsky, Hashimoto, 2023)
- Calibration achieved via verbalized confidence is often trivial
    - If a model is 85% accurate, and predicts 90-95% confidence for every example, ECE is low but confidence estimates are not useful

Learning to Reject

Build selective prediction into the post-trianing process

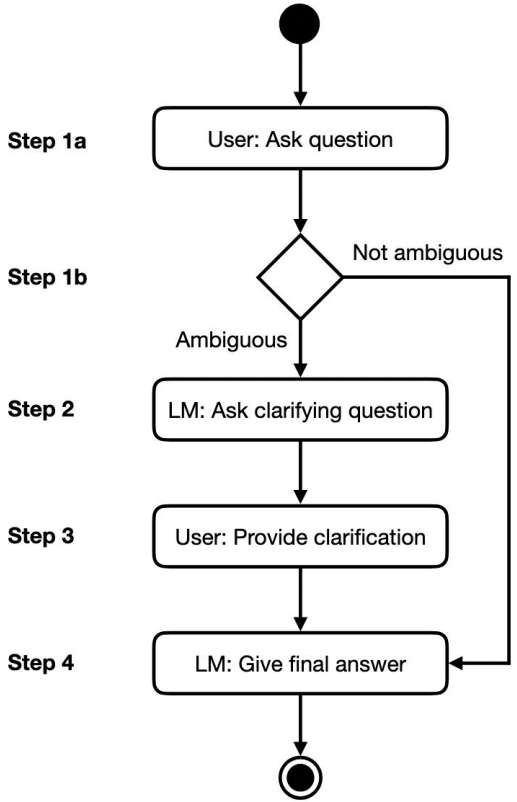
- Identify instruction tuning data examples with facts unknown to the LLM
- Replace preferred response with “I don’t know”

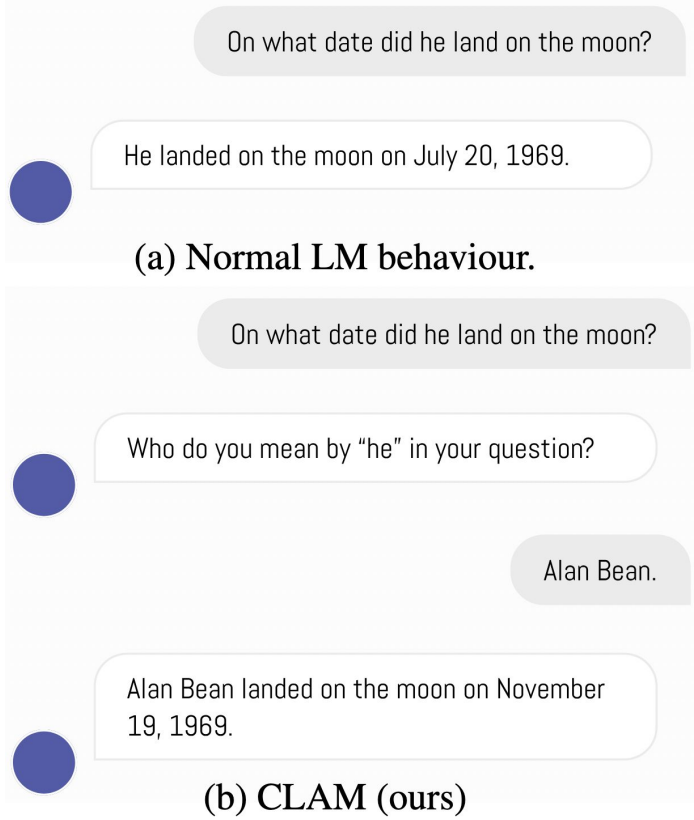


src: https://arxiv.org/abs/2311.09677

Handling Ambiguous Queries

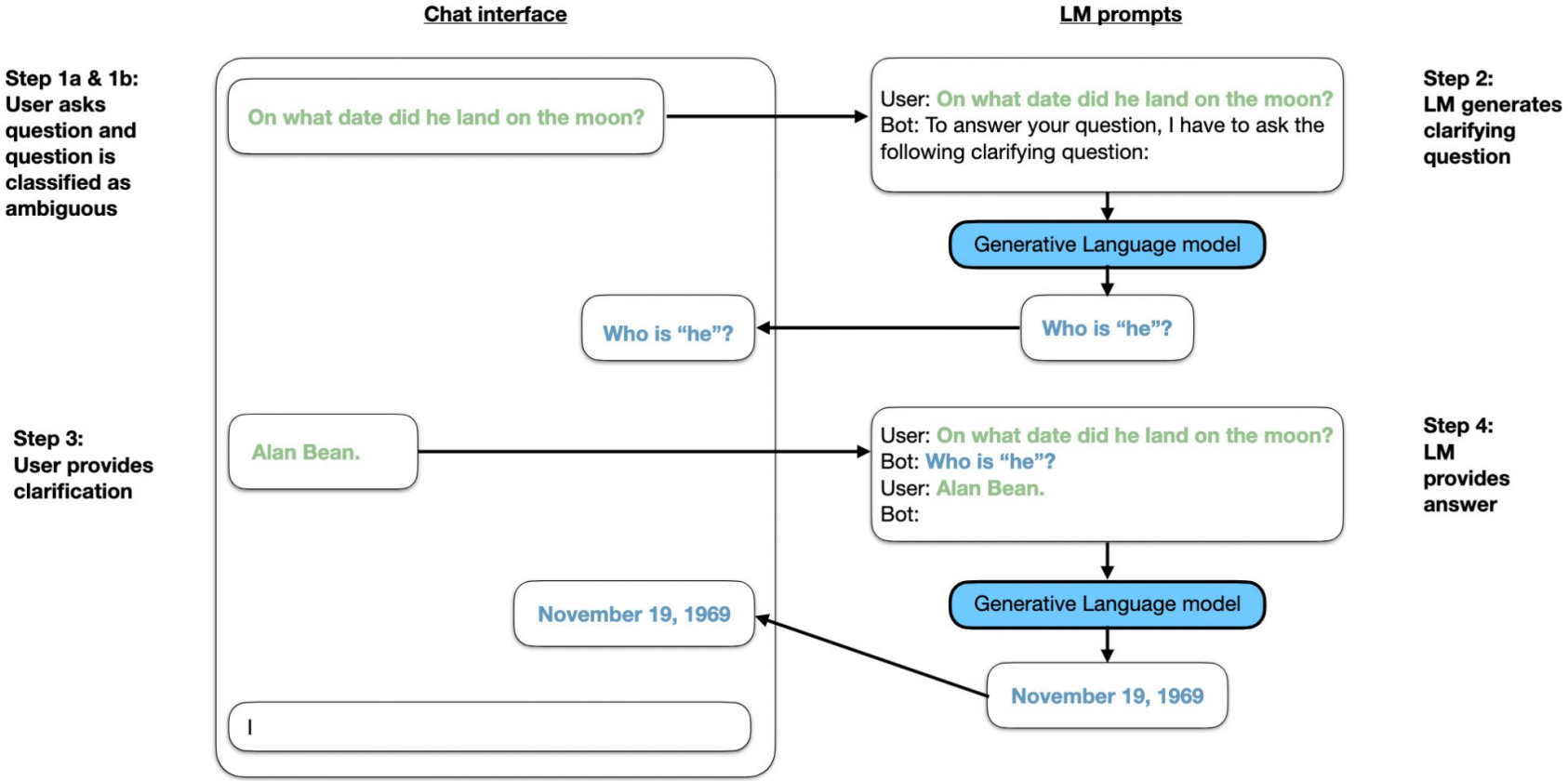
Unlike image classification models, **LLMs can ask for more information when they are uncertain!**





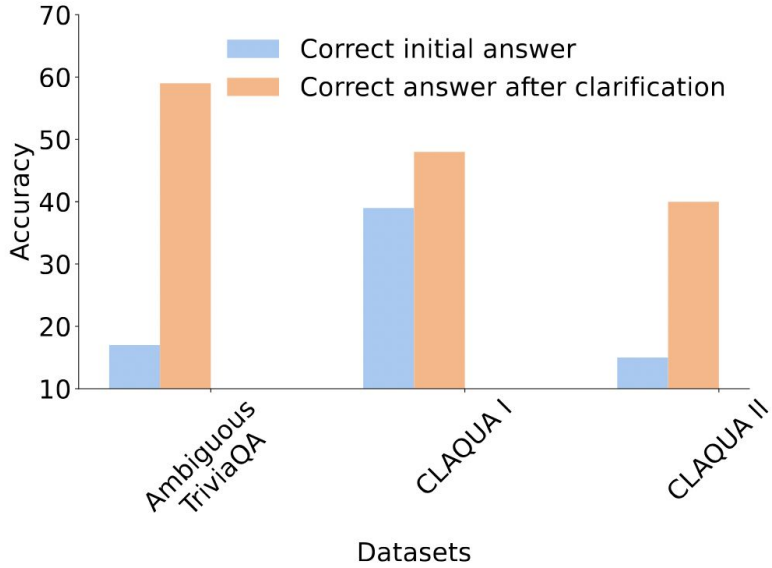
src: https://arxiv.org/abs/2212.07769

Selective Clarification for Ambiguous Questions



Selective Clarification for Ambiguous Questions

Empirically, this is helpful...



Can think of this as another sort of test-time compute...

Figure 7. **Prompting the model to ask the user for clarification improves QA accuracy on ambiguous questions.** ClariQ does

LLMs and Distribution-Free Uncertainty Quantification

Lots of interest in LLMs

Lots of interest in DFUQ

LLMs and Distribution-Free Uncertainty Quantification

Lots of interest in LLMs

+ Lots of interest in DFUQ

Lots of interest in LLMs+DFUQ!

LLMs and Distribution-Free Uncertainty Quantification

Lots of interest in LLMs

+ Lots of interest in DFUQ

Lots of interest in LLMs+DFUQ!

But it’s not always an easy fit…

Conformal LLMs

- Conformal Language Modeling
  - Generate samples until a stopping rule is met, filter for diversity and quality
  - https://arxiv.org/abs/2306.10193
- Language Models with Conformal Factuality Guarantees
  - Ensure high probability of factuality in a single generation using back-off
  - https://arxiv.org/abs/2402.10978
- Mitigating LLM Hallucinations via Conformal Abstention
  - Score with self-consistency, use conformal techniques to calibrate rejecting threshold
  - https://arxiv.org/abs/2405.01563

Prompt Risk Control

Prompt selection is a leverage point for managing performance/behavior trade-offs.

- **Prompt risk control** is a lightweight framework for selecting a prompt based on high-probability bounds on families of informative risk measures.

Goals:

- Move beyond selecting prompts based on an average empirical performance.
- Manage trade-offs inherent in LLM behavior (e.g., helpfulness vs. harmlessness, usefulness vs. safety) in a principled and rigorous manner.

src: https://arxiv.org/abs/2212.07769

Prompt Risk Control

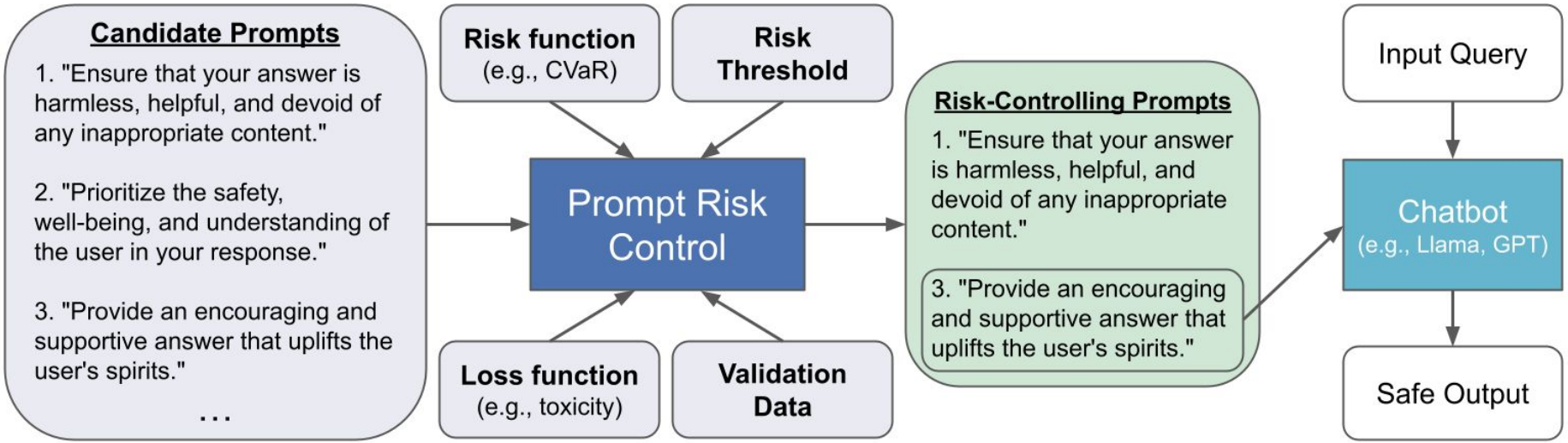
Prompt Risk Control → lightweight framework to select a prompt based on rigorous upper bounds on families of informative risk measures in order to:

- Move beyond selecting prompts based on an average empirical score.
- Manage trade-offs inherent in LLM behavior, like helpfulness vs. harmlessness, or usefulness vs. safety.
    - First guarantee safety, and then pick the most useful prompt left standing

Prompt Risk Control

Input: Candidate prompts, validation data (i.i.d.), loss function, risk measure, threshold

Output: Set of prompts that (with high probability) bound risk at acceptable level



2-Step Pipeline Including PRC

Framework fits naturally as the initial step in a 2-stage prompt selection pipeline.

1. **Prompt Risk Control** uses validation data to identify prompts unlikely to incur an unacceptably bad outcome according to risk and loss fns.
2. **Using the same data,** risk controlling prompts can be scored on performance metric for final prompt selection.

Note: because PRC treats LLM as black box and only requires outputs from the model, this framework **can be used with a proprietary model held behind an API (e.g., GPT-4).**

PRC Example Application

An organization plans to deploy an LLM chat application, where the goal is to provide helpful answers to user-provided queries.

- They have concerns about the model including toxic content in its output, and decide that:
  - with 95% likelihood ($\delta = 0.05$)
  - at least 90% of generations (Value-at-Risk, $\beta = 0.90$)
  - must have toxicity score less than $\alpha = 0.05$.
- Want to consider 5 system prompts and 5 one-shot exemplars ➜ 25 candidate prompts

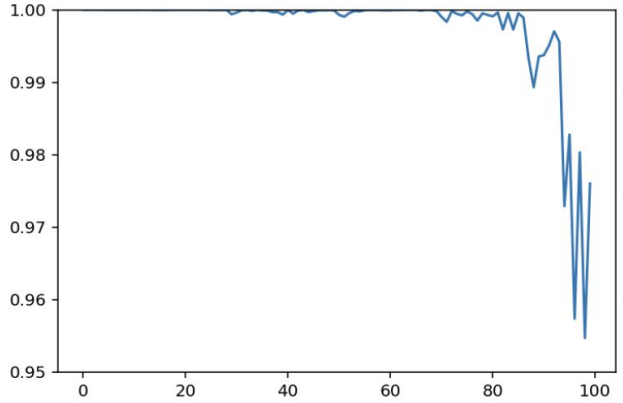
Using validation set, LLM generations, and toxicity scores, PRC will return the prompts that control the risk at an acceptable level.

- Then, using the same validation data and the set of prompts returned by PRC, the final prompt might be chosen according to the average helpfulness score (often known as the “reward”) across the validation set.

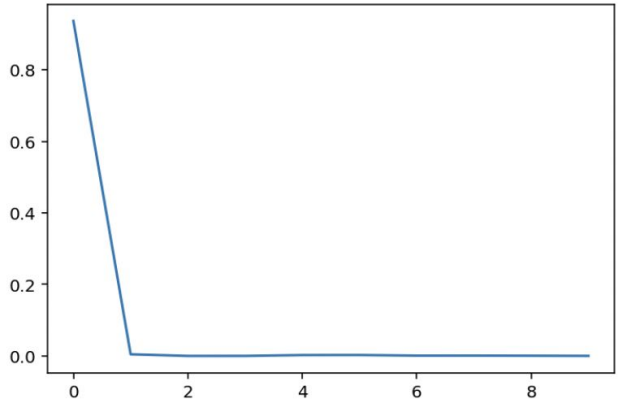
Epistemic Uncertainty in LLMs

To Believe or Not to Believe Your LLM https://arxiv.org/abs/2406.02543

- Iterative prompting to measure epistemic uncertainty



Q: *What is the capital of the UK?* A: *London* ($\approx$ 1.0) and *Paris* (1.29×10^{-10}).



Q: *What is the national instrument of Ireland?* A: *The harp* (0.936) and *Uilleann pipes* (0.063).

Identifying Epistemic Uncertainty in LLMs

- Fine-Tuning Language Models via Epistemic Neural Networks
  - Train epistemic NN to quantify uncertainty of LLM, use to select fine-tuning data
  - https://arxiv.org/abs/2211.01568
- Decomposing Uncertainty for LLMs through Input Clarification Ensembling
  - Uses re-prompting to separate epistemic and aleotoric uncertainty
  - https://arxiv.org/abs/2311.08718

Reducing Uncertainty Using Natural Language

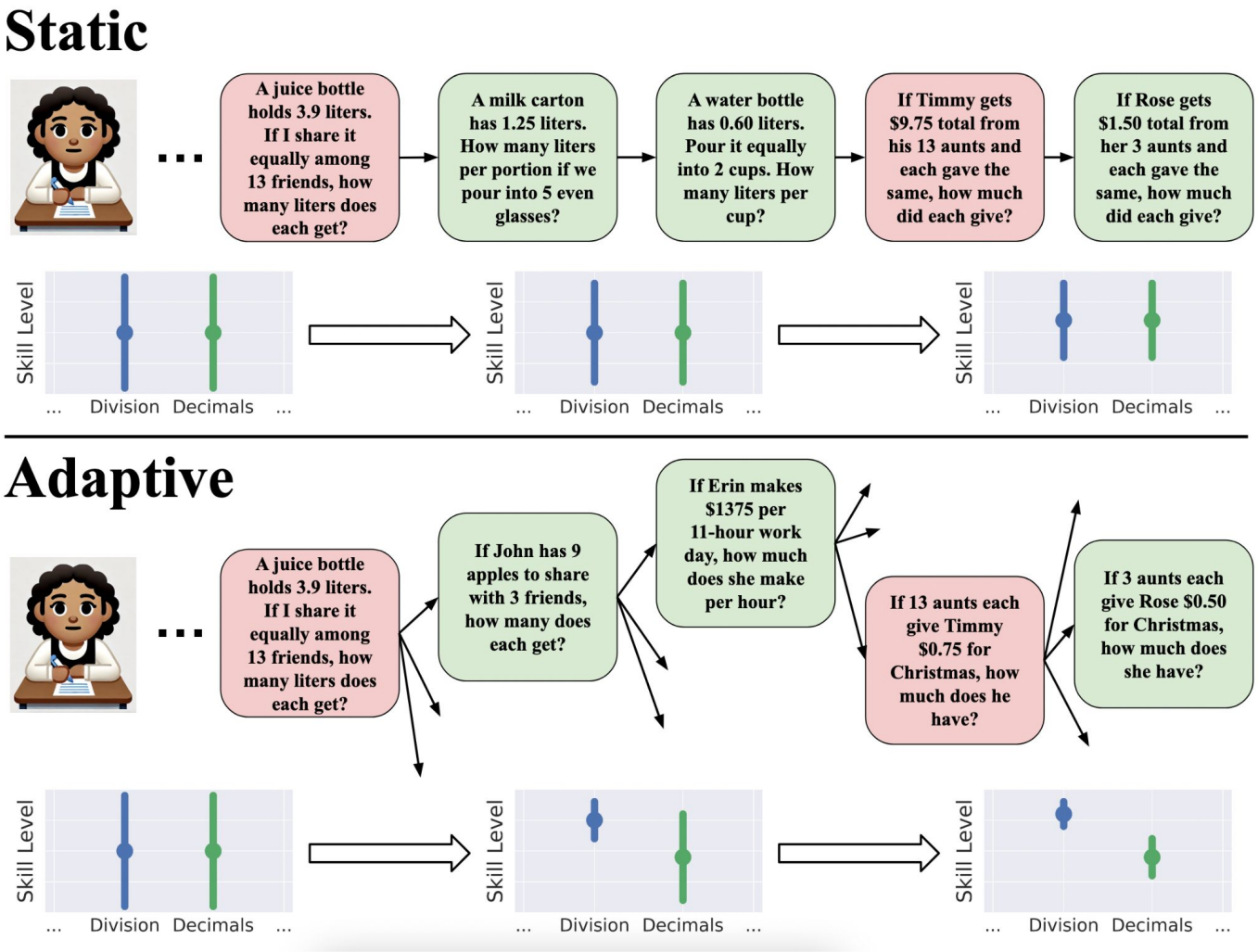
Adaptive Elicitation of Latent Information Using Natural Language

Jimmy Wang, Tom Zollo, Rich Zemel, Hongseok Namkoong

https://openreview.net/forum?id=63c2erbMoc

- The performance of many valuable services and systems depends on the ability to efficiently **elicit information and reduce uncertainty about a previously unseen latent entity.**
  - Assess a new student's skills
  - Understand underlying health of patient upon intake
- To achieve efficiency, these strategies must be **adaptive**, dynamically tailoring subsequent queries based on the information gained so far.
  - For student assessment might start with a broad question covering multiple skills
  - If the student gets a question wrong, the system would then drill down into each skill individually, asking questions of varying difficulty to determine the limits of their proficiency

Reducing Uncertainty Using Natural Language



Reducing Uncertainty Using Natural Language

Can LLMs help with this? Not out of the box!

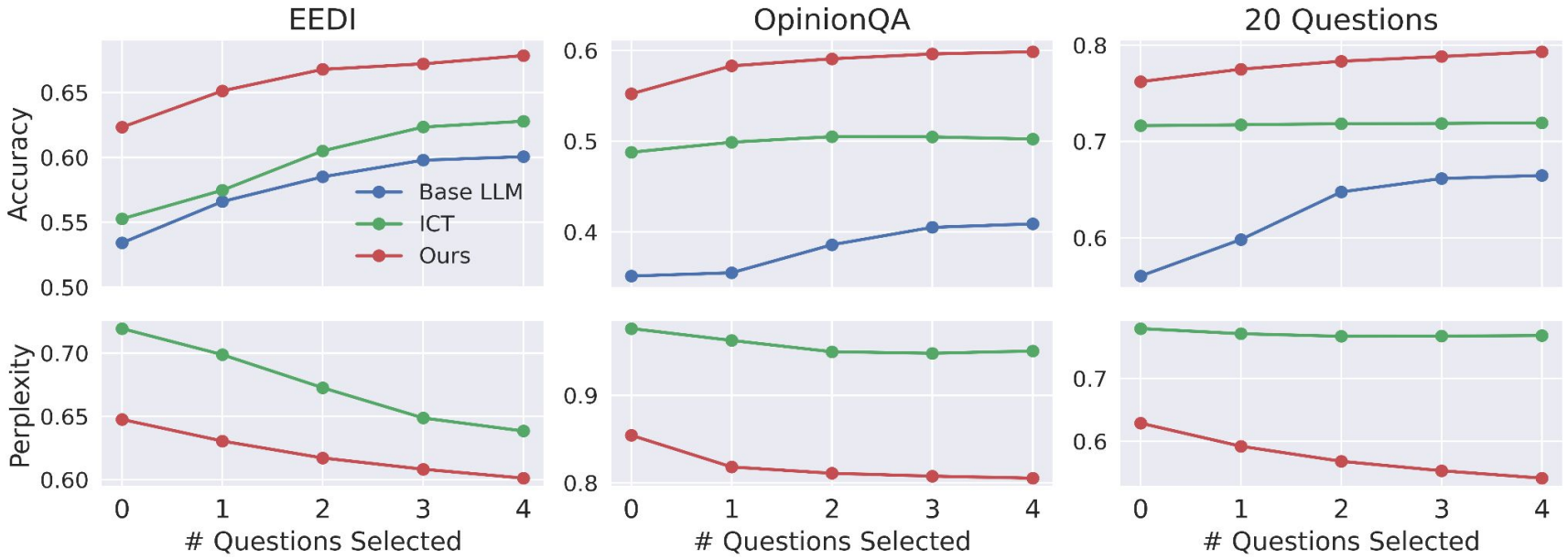
- UQ is not a top priority in LLM training and fine-tuning algorithms
- Lack mechanisms for strategically gathering information

Our algorithm

- Meta-learns a predictive language model from historical question–answer data
- Uses this model to quantify uncertainty about future or unobserved answers
- Dynamically adapts question selection to reduce uncertainty about the latent entity

Reducing Uncertainty Using Natural Language

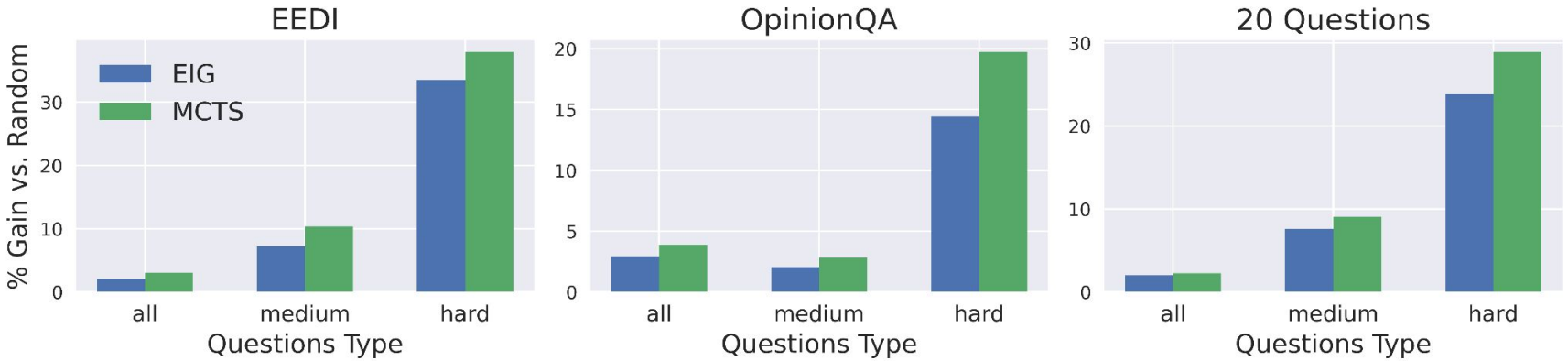
Goal: Select questions (and observe answers) that will allow the model to predict answers to held-out questions



Reducing Uncertainty Using Natural Language

When does this help? identifying features of the latent which are relatively rare in the population.

e.g., while many students may have overlapping weaknesses, it can be harder to learn that a particular student is struggling where other students generally do not.



Part 2: Summary

Uncertainty quantification in LLMs

- Not as easy as classification
- Baseline: Token probabilities
- Advanced approaches to UQ in LLMs
  - Semantic Entropy
  - Verbalized Confidence
  - Learning to reject
  - Asking for more information
  - LLMs+DFUQ
  - “Bayesian” approaches
  - Reducing latent concept uncertainty with LLMs

Roadmap

- Introduction
  - Why quantify uncertainty?
  - Types and sources of uncertainty
- Uncertainty quantification in (image) classification models
  - Estimating uncertainty
  - Calibration
  - Selective classification, DFUQ, Bayesian approaches
- Uncertainty quantification in LLMs
  - Why is it harder?
  - Baselines
  - Advanced methods
  - Asking for more information
- Future directions and closing thoughts

Part 3:

Future Directions and Closing Thoughts

Limitations

- Experiments in limited settings.
  - Experiments are usually performed on tasks like trivia question answering, which can be answered via a single token, word, or short phrase
  - Tasks under study also often assume that there is only one right answer
- Expensive
  - Inference costs are already soaring, and many of these methods require repeated sampling
- Verbalized uncertainty is suspect (at least for now)
  - Good evidence exists that any correlation between accuracy and verbalized expressions of confidence is simply a result of spurious features in training data
- Largely unsuitable for long-form and open-domain problems

Current Status

- Overall, it is not clear that any advanced method for quantifying LLM uncertainty in the zero-shot setting robustly outperforms a baseline normalized sequence entropy score calculated using token probabilities.
- However, these scores are often not available for black-box LLMs held behind an API.
- It is difficult to imagine how best to exploit probabilities taken directly from the language model, as these probabilities do not necessarily relate to the task at hand.

Future Directions

- Explore **how uncertainty can be better quantified and addressed across the entire model development and deployment pipeline**, and how interventions and measurements at different points in the pipeline interact and affect downstream outcomes.
- Understand **how techniques for selecting, mixing, and filtering training data affect the ability to accurately estimate the model's confidence** on downstream tasks, whether via token probabilities or verbalizations.
- As new architectures and pretraining recipes emerge, they should be **benchmarked for calibration, not only accuracy.**
- Fine-tuning algorithms, whether supervised or RL, have been shown to worsen the UQ characteristics of models, and **this phenomena must be kept in focus as the community iterates on these methods.**

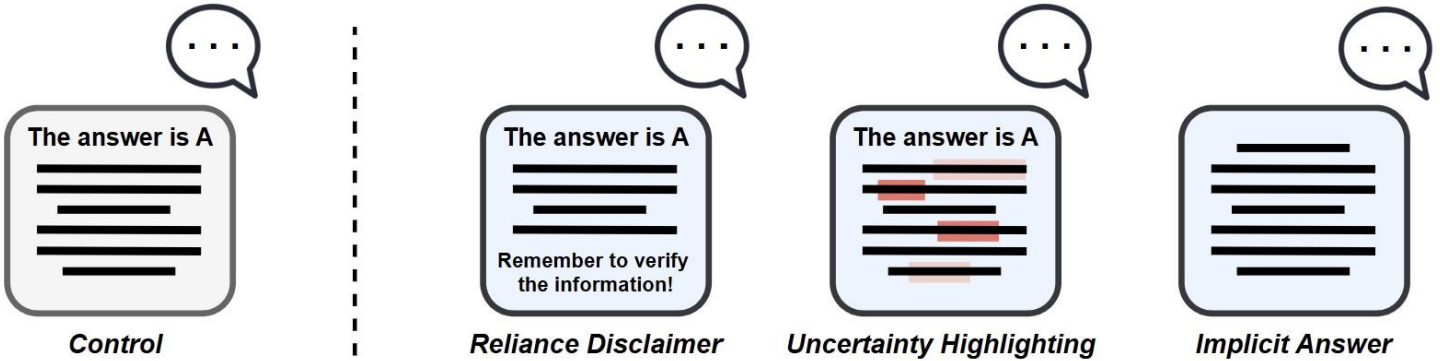
Future Directions

To Rely or Not to Rely? Evaluating Interventions for Appropriate Reliance on Large Language Models

Jessica Y. Bo
jbo@cs.toronto.edu
University of Toronto
Toronto, Canada

Sophia Wan
sophia.wan@mail.utoronto.ca
University of Toronto
Toronto, Canada

Ashton Anderson
ashton@cs.toronto.edu
University of Toronto
Toronto, Canada



src: https://arxiv.org/abs/2412.15584

Future Directions - My Bet

Robustness has always haunted deep learning

Future Directions - My Bet

Robustness has always haunted deep learning

This is a missing data problem...

Future Directions - My Bet

Robustness has always haunted deep learning

This is a missing data problem...

...but LLMs (and other modern models/systems) can be used to seek out their own missing data!

Closing Thought(s)...

This problem is not getting scaled away

Algorithmic innovation is needed

Thank you!!!!!!

reach me at tpz2105@columbia.edu