Uncertainty Quantification in Machine Learning

Tom Dietterich

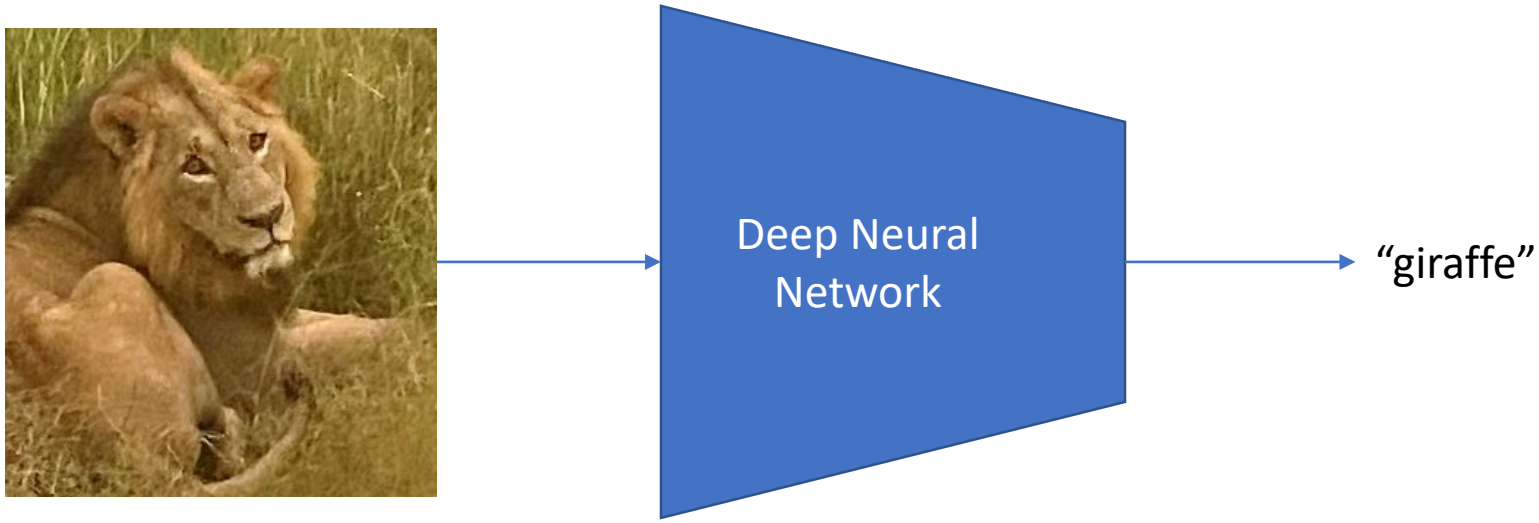
School of EECS

Oregon State University

Outline

- Goals of Uncertainty Quantification (UQ)
- Aleatoric vs. epistemic uncertainty and what causes each
- UQ as Prediction Intervals for Regression
  - Linear regression prediction intervals
  - Bayesian prediction intervals: Gaussian Processes
  - Conformal Quantile Regression intervals
- UQ for Classification
  - Calibration
  - Label sets as prediction intervals
- Local Epistemic Uncertainty
  - Outlier/Anomaly Detection
- Applications
  - Active Learning
  - Uncertainty-Aware Learning
  - Selective Prediction
  - Reducing LLM Hallucination

When should we trust a prediction?



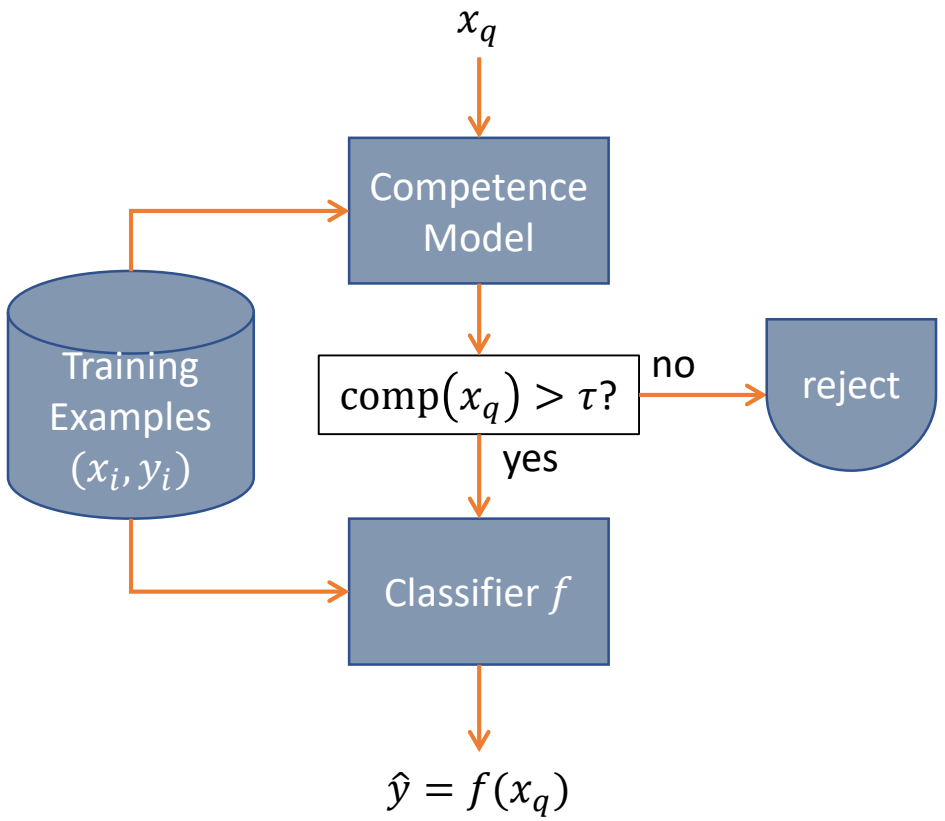
Goals of Uncertainty Quantification

- Selective Classification
  - Allow the classifier to abstain in order to guarantee high accuracy on the remaining predictions
  - Safety-critical applications
  - Reduce hallucination in large language models
  - Distribution shift/Open category settings
- System Integration
  - Communicate uncertainty to down-stream components to allow them to compute expectations, make decisions
- Provide Guidance on System Improvement
  - What components should we try to improve?
    - Active learning, better labels, better model space, better learning algorithms?
    - When do we need to retrain?
- Improving Model Accuracy
  - Down-weight training examples in uncertain regions of the input space

Selective Classification

(Also called “Classification with a reject option”)

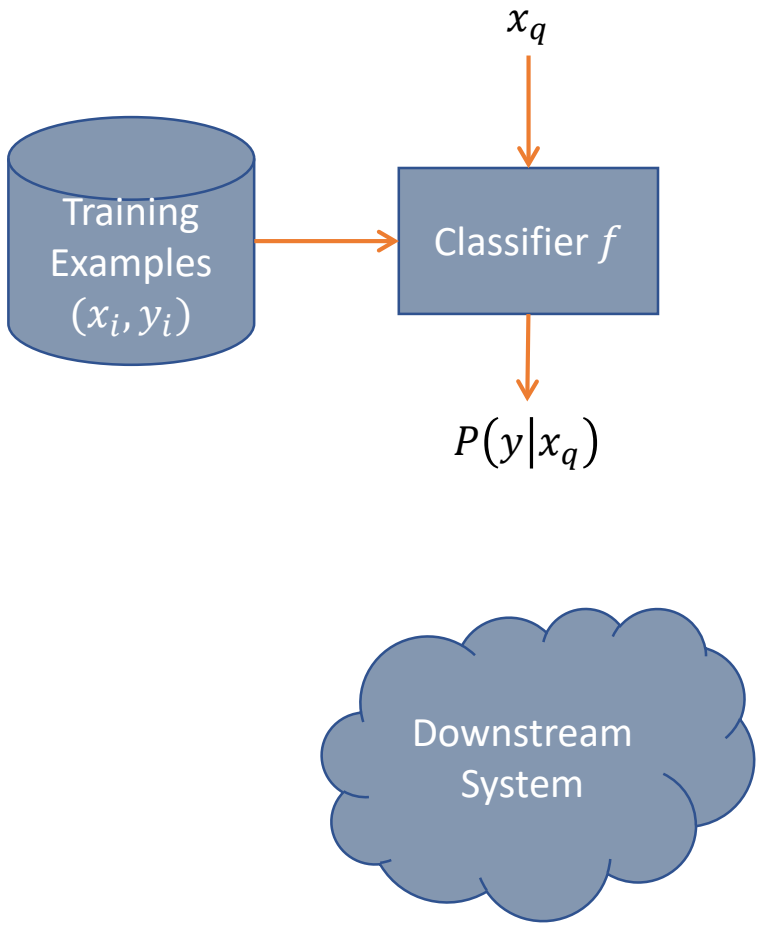
- Competence Model
  - How uncertain is $f(x_q)$?
    - “local uncertainty”
- Applies to regression models also (of course)
  - Selective prediction
  - Prediction with a reject option
- Uncertainty-aware learning: Should our uncertainty affect the learning of f itself?



System Integration: Predict a probability distribution

- Classification:
  - $[P(y_q = 1|x_q), P(y_q = 2|x_q), \ldots, P(y_q = K|x_q)]$
- Regression:
  - Probability density $P(y_q|x_q)$
  - Cumulative distribution function $F(y_q|x_q)$

- This allows a down-stream system to compute expectations and risks
  - Cost-sensitive classification
    - $\arg\min_k \sum_{k'=1}^{K} P(y = k'|x_q) C(k|k')$
    - $C(k'|k)$ is the cost of predicting class k' when the true class is k
  - Expected cost
    - $\int_y C(y) P(y|x) dy$
  - Risk of outcomes worse than u (conditional value at risk)
    - $\int_{y<u} C(y) P(y|x) dy$



Guidance for System Improvement

- Actions we might take to improve the model
  - Down-weight more uncertain training examples
  - Collect more data (active learning)
  - Reduce noise in feature measurement
  - Reduce noise in the class labels
  - Reduce sampling bias, missing values, etc.
  - Add or reduce model capacity (width and number of layers)
  - Improve the learning algorithm (e.g., optimization method; hyperparameters)

Two Different Scenarios

- Case 1: “Full Uncertainty Quantification”
  - Uncertainty in the model fitting process is important
  - Given: Training data
  - Find: Fit a model or an ensemble of models
  - Questions: Do we have enough data? Is the model class appropriate? Is the optimization working?
  - Active learning, system integration

- Case 2: “Single Model Uncertainty Quantification”
  - Uncertainty in model fitting is not important
  - Given: A fitted model f and an independent data set
  - Find: An uncertainty assessment of f
  - Question: How uncertain is my prediction $\hat{y}_q = f\left(x_q\right)$?
  - Selective classification, system integration

Outline

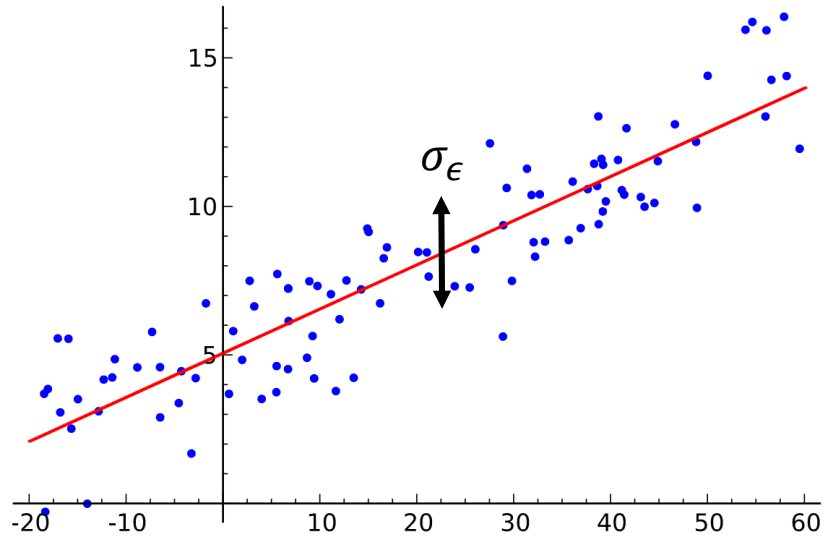
- Goals of Uncertainty Quantification (UQ)
- Aleatoric vs. epistemic uncertainty and what causes each
- UQ as Prediction Intervals for Regression
  - Linear regression prediction intervals
  - Bayesian prediction intervals: Gaussian Processes
  - Conformal Quantile Regression intervals
- UQ for Classification
  - Calibration
  - Label sets as prediction intervals
- Local Epistemic Uncertainty
  - Outlier/Anomaly Detection
- Applications
  - Active Learning
  - Uncertainty-Aware Learning
  - Selective Prediction
  - Reducing LLM Hallucination

Uncertainty Decomposition: Aleatoric and Epistemic Uncertainty

- Aleatoric: (Latin: *aleator* is a “dice player”). Random noise
- Epistemic: “lack of knowledge”
  - There are many debates about the difference between these.
- Working definition:
  - Epistemic uncertainty is uncertainty that could be removed by collecting more data (using the same features and the same data labeling process)
    - Useful for active learning, exploration in RL
  - Aleatoric uncertainty is everything else

Aleatoric Uncertainty: Regression

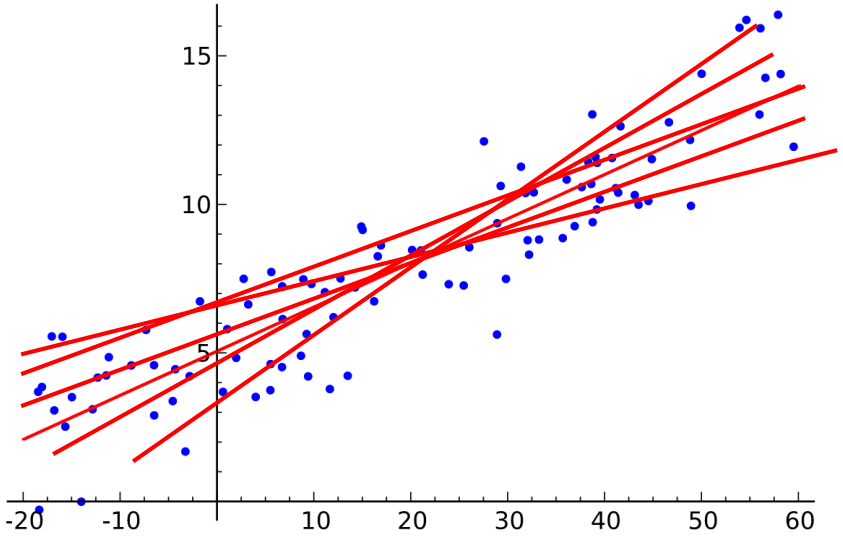
- Most models include “aleatoric” parameters that capture the mismatch between the model predictions and the labels
- Linear regression:
  - $y = \beta^{\mathsf{T}} x + \epsilon$ where $\epsilon \sim Normal(0, \sigma_\epsilon^2)$
  - σ_ϵ^2 is the aleatoric parameter. It estimates the amount of noise in the labels
  - It is estimated as the variance of the training data residuals, corrected for the fact that we fit β to those same data points



(Wikipedia: public domain)

Epistemic Uncertainty

- Uncertainty due to the finite amount of training data
- As we collect more training data, the epistemic uncertainty will drop, and all "consistent" estimation procedures will converge to the correct answer
- In a non-stationary world where the correct answer is continually changing, there is always epistemic uncertainty. We never observe enough data to converge



Measuring Epistemic Uncertainty via Ensembles

- Fix the model class $\mathcal{M}$
  - Examples: neural network architecture, decision trees of depth $\leq d$
- Fix the training data $\mathcal{D}_{train}$
- Train an ensemble of L models
- Measure the degree of disagreement at the query point x_q
  - High disagreement ➔ high epistemic uncertainty
    - Under the assumption that the learning algorithms would converge to a unique answer given infinite data
  - Measures of disagreement:
    - Regression: Variance of predictions: $\frac{1}{L}\sum_{\ell}\left(\hat{y}_{\ell} - \bar{\hat{y}}\right)^2$, where $\hat{y}_{\ell}$ is the prediction from model ℓ and $\bar{\hat{y}}$ is the average of those predictions
    - Classification: Choose a distance $\|\cdot\|$ between probability distributions (e.g., KL divergence, Total Variation distance, Hellinger distance).
      - Let α_{ℓ} be the predicted probability vector from model ℓ
      - "Disagreement": $\min_{\bar{\alpha}} \frac{1}{L}\sum_{\ell}\|\alpha_{\ell} - \bar{\alpha}\|$
      - $\bar{\alpha}$ plays the same role as the mean. But it depends on the choice of $\|\cdot\|$.

Ensemble Methods for Epistemic Uncertainty (1)

- Neural network ensemble using different random seeds
  - Train L networks. Initialize the weights using different random seeds.
  - Measure how much the fitted networks disagree on a prediction $f^{\ell}(x_q)$
  - If we have enough training data, all of the networks should give very similar answers
  - This also captures variance caused by the backpropagation algorithm:
    - Stochastic gradient descent
    - Random sampling of mini-batches
    - This may cause it to over-estimate epistemic uncertainty
  - Requires training multiple networks
  - Open question: Can we use LoRA methods to avoid full ensemble training?
    - See Babalanov & Linander (2024)

Ensemble Methods for Epistemic Uncertainty (2): Bagging Ensembles (Breiman, 1996)

- Let $\mathcal{D}$ be our training data containing n training examples: $(x_1, y_1), \ldots, (x_n, y_n)$
- A *bootstrap replicate* $\mathcal{D}^b$ is created by randomly sampling n examples *with replacement* from $\mathcal{D}$

  $\mathcal{D}^b := \{\}$

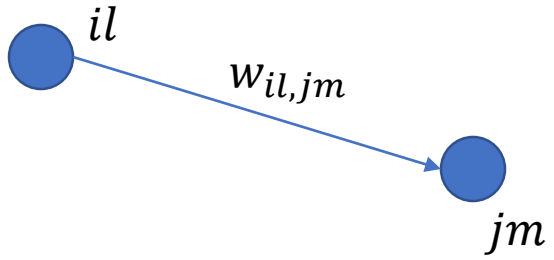
  For $i = 1, \ldots, n$

    Let (x_i, y_i) be a data point randomly sampled from $\mathcal{D}$

    Add it to $\mathcal{D}^b$
- Train a classifier on $\mathcal{D}^b$
- This simulates training on a new data set of the same size and drawn from the same distribution
- If we have low epistemic uncertainty, the classifiers trained on bootstrap replicates will be very similar

Ensemble Methods for Epistemic Uncertainty (3): Approximate Bayes via Dropout

- Dropout:
  - Let a_{il} be the activation of unit i in layer l
  - Let $w_{il,jm}$ be the weight connecting unit i in layer l to unit j in layer m
    - m is usually $l + 1$
  - During each forward pass through the network, with probability d,
    - $w_{il,jm} \coloneqq 0$
  - The contribution of a_{il} to unit j is "dropped out"



- We can think of dropout as converting one network into a huge random collection of networks
- We apply this dropout process both during training and during prediction
  - During training: Tends to make the network weights more robust (because they need to be accurate despite the dropout noise)
  - During prediction: We can obtain a probability distribution over the network predictions by doing L forward passes (e.g., 50)
  - Training is not much more expensive than standard training
  - Prediction is L times more expensive
- Gal & Ghahramani (2016) prove that dropout is a valid approximation to a full Bayesian ensemble
  - However, it changes the class of fitted models
  - It may under-estimate epistemic uncertainty

Aleatoric Uncertainty: Classification

- Multinomial Logistic Regression (softmax probability prediction)

$$P(y = k|x) = \frac{e^{\ell_k(x)}}{\sum_{k'=1}^{K} e^{\ell_{k'}(x)}}$$

  - where $\ell_k(x) = \beta_k \cdot x$ is the logit score for class $k = 1, \ldots, K$
  - To the extent $|\ell_k(x)| < \infty$, the model claims there is aleatoric uncertainty in the labels
  - Labels are determined by tossing weighted dice
  - We can quantify this using the conditional entropy:

$$H(y|x) = \sum_{k=1}^{K} -P(y = k|x) \log P(y = k|x)$$

Saharasav, CC BY-SA 4.0

Sources of Uncertainty

"Global" uncertainty of the learned classifier:

- Are the measured features sufficient to make accurate predictions?
- Is there measurement noise in the features?
- Is there selection bias in the features?
- Are there missing values in the features?
- Are the labels on the training data accurate, noisy, or biased?
- Is the optimal classifier changing over time? (data shift; novel categories)

Data Uncertainty

- Can the model class represent a good approximation of the true decision boundary?
- Do we have enough training data so that a learning algorithm can find that good approximation?
- Can the learning algorithm find that good approximation?

Model Uncertainty

"Local" uncertainty with respect to a query x_q:

- All of the above, but now focused on a neighborhood around x_q

Outline

- Goals of Uncertainty Quantification (UQ)
- Aleatoric vs. epistemic uncertainty and what causes each
- UQ as Prediction Intervals for Regression
  - Linear regression prediction intervals
  - Bayesian prediction intervals: Gaussian Processes
  - Conformal Quantile Regression intervals
- UQ for Classification
  - Calibration
  - Label sets as prediction intervals
- Local Epistemic Uncertainty
  - Outlier/Anomaly Detection
- Applications
  - Active Learning
  - Uncertainty-Aware Learning
  - Selective Prediction
  - Reducing LLM Hallucination

Total Uncertainty: Prediction Intervals

- Prediction Interval:
  - Given a query x_q
  - Predict an interval $[y_{lo}(x_q), y_{hi}(x_q)]$ such that with probability at least $1-\alpha$,

$$y_{lo}(x_q) \leq y_q \leq y_{hi}(x_q)$$

- Ideally, this should capture all of our sources of uncertainty
  - Random sampling of the training data
  - Data uncertainties (measurement error)
- The width of the interval $y_{hi}(x_q) - y_{lo}(x_q)$ gives us a scalar measure of uncertainty (in units of y).
- Selective prediction:
  - If $y_{hi}(x_q) - y_{lo}(x_q) > \tau$ then reject

Prediction Intervals for Linear Models

[Gruber, et al. 2023]

- It often helps to examine linear regression models to gain insight
- Simplifying assumptions:
  - No data shift (iid samples; no sampling bias)
  - The linear model is correct (it can represent the true function and noise)
    $y = \beta^{\top} x + \epsilon$ with $\epsilon \sim Normal(0, \sigma_{\epsilon}^{2})$
  - Learning algorithm (least squares regression) finds the global optimum

Linear Regression Prediction Intervals

- Given n training examples $(x_1, y_1), \ldots, (x_n, y_n)$, the vector-matrix form is
- $\boldsymbol{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad \hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\beta}$
- Let $\hat{\beta}$ be the estimated regression coefficients
- Let $\hat{\sigma}^2$ be the estimated noise variance
- The prediction interval is

$$x_q^\top \hat{\beta} \pm t_{n-J}(1 - \alpha/2)\hat{\sigma}\sqrt{1 + x_q^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1} x_q}$$

Examining the Prediction Interval

$$x_q^\top \hat{\beta} \pm t_{n-J}(1-\alpha/2)\,\hat{\sigma}\sqrt{1 + x_q^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1} x_q}$$

Predicted value $\hat{y}$

t statistic with $n - J$ degrees of freedom
Shrinks as n grows

Training prediction error

Similarity of x_q to the training data (inverse-variance scaled distance to the mean x value)

Epistemic Uncertainty

Aleatoric Uncertainty

Local Epistemic Uncertainty

Examining the Prediction Interval

$$x_q^\top \hat{\beta} \pm t_{n-J}(1-\alpha/2)\,\hat{\sigma}\sqrt{1 + x_q^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1} x_q}$$

Notes:

- The epistemic and aleatoric components are multiplied (not added)
- The query-based (local) uncertainty depends on similarity (or distance) in the input space
- Captures all sources of uncertainty
  - *Assuming* that sampling biases, measurement noise, label noise, imputations, etc. are all iid

Bayesian Prediction Intervals

- Bayesian Recipe
  - Choose a model class: $y = x \cdot \beta + \epsilon$ with $\epsilon \sim \text{Normal}(0, \sigma_\epsilon^2)$
  - This defines a likelihood function $P(y|x, \beta)$
    - Log likelihood on training data $\mathcal{D} = \{(x_i, y_i)\}$ is

$$\log \mathcal{L}(\beta) = \sum_i \left(x_i^\top \beta - y_i\right)^2$$

  - Assume a prior distribution $P(\beta)$ over the model parameters
  - Compute the posterior distribution $P(\beta|\mathcal{D})$

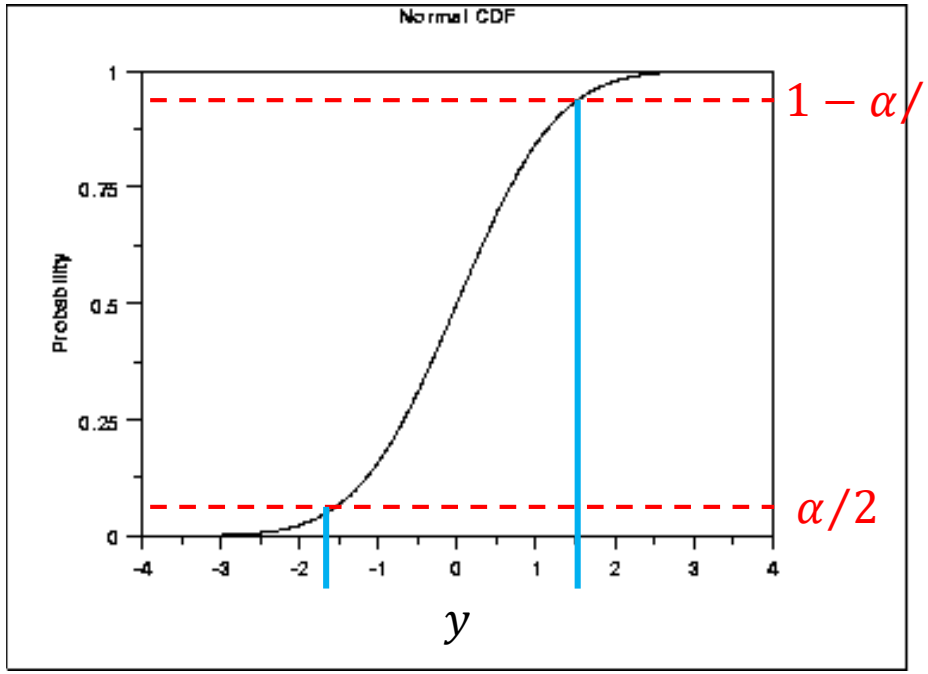
$$P(\beta|\mathcal{D}) = \frac{P(\mathcal{D}|\beta)P(\beta)}{\int_\beta P(\mathcal{D}|\beta)P(\beta)}$$

  - Posterior predictive distribution for x_q ("Bayesian model averaging")

$$P(y|x_q, \mathcal{D}) = \int_\beta P(\beta|\mathcal{D})P(y|\beta, x_q)d\beta$$

  - Compute PDF $F(y|x_q, \mathcal{D})$ and invert

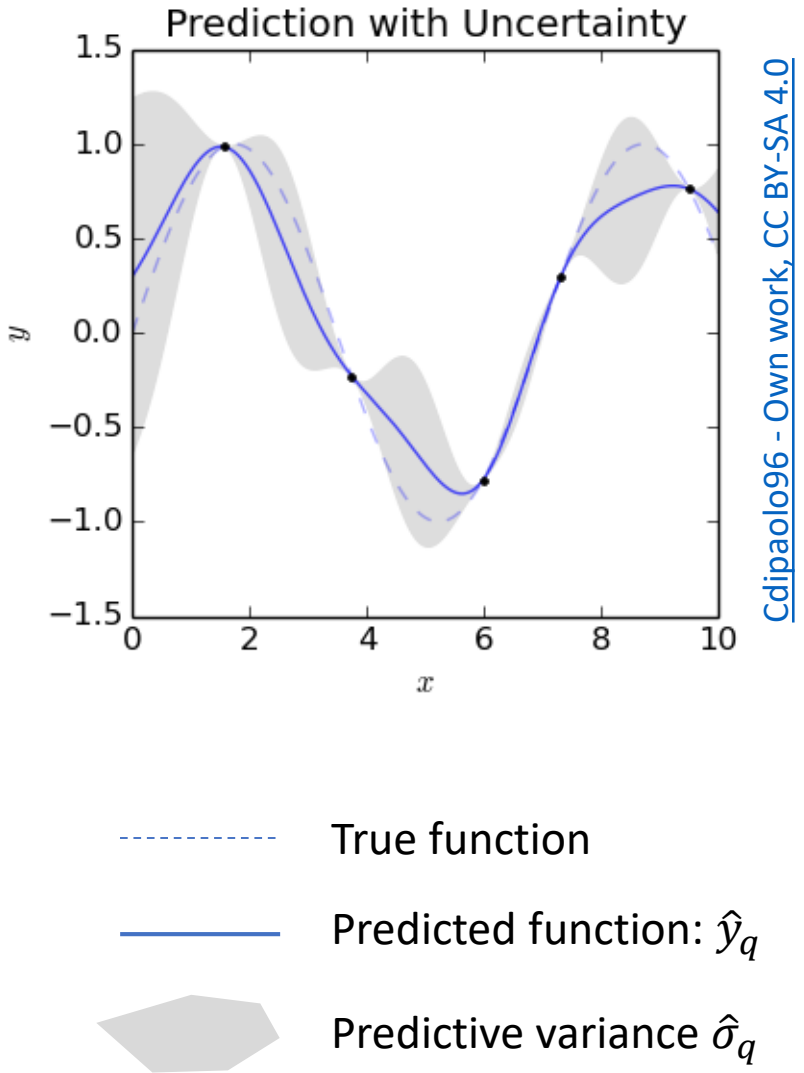
$$F^{-1}(\alpha/2\,|x_q, \mathcal{D}) \le y_q \le F^{-1}(1-\alpha/2\,|x_q, \mathcal{D})$$



$$F^{-1}(\alpha/2\,|x_q, \mathcal{D}) \le y_q \le F^{-1}(1-\alpha/2\,|x_q, \mathcal{D})$$

Gaussian Processes

- A Gaussian Process is a collection of random variables, any finite number of which have a joint Gaussian distribution
- Efficiently-computed Bayesian kernel method
- Given training data $(x_1, y_1), \ldots, (x_n, y_n)$
- At every location x_q the GP predicts a mean $\hat{y}_q$ and a variance $\hat{\sigma}_q^2$
- The variance depends on $\|x_q - x_i\|$ for all i. It captures both epistemic and aleatoric uncertainty



Gaussian Process Details

- Choose a kernel function $k(x, x')$
  - Example: Squared exponential kernel: $k(x, x') = \exp - \frac{\|x - x'\|^2}{2\ell^2}$
    - ℓ is the "kernel width"
- Given n data points $(x_1, y_1), \ldots, (x_n, y_n)$ with observation noise: $\text{Normal}(0, \sigma_\epsilon^2)$
- Form the covariance matrix $K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix} + \sigma_\epsilon^2 \mathbb{I}$
- Given a query point x_q form the covariance vector $k_q = \left[k(x_q, x_1), \ldots k(x_q, x_n)\right]$
- Let $\boldsymbol{y} = [y_1, \ldots, y_n]$
- The GP predictive distribution is

$$y_q \sim \text{Normal}\left(k_q K^{-1} \boldsymbol{y},\ \ 1 - k_q^\top K^{-1} k_q\right)$$

- If we want only the epistemic uncertainly, we can omit the $\sigma_\epsilon^2 \mathbb{I}$ term

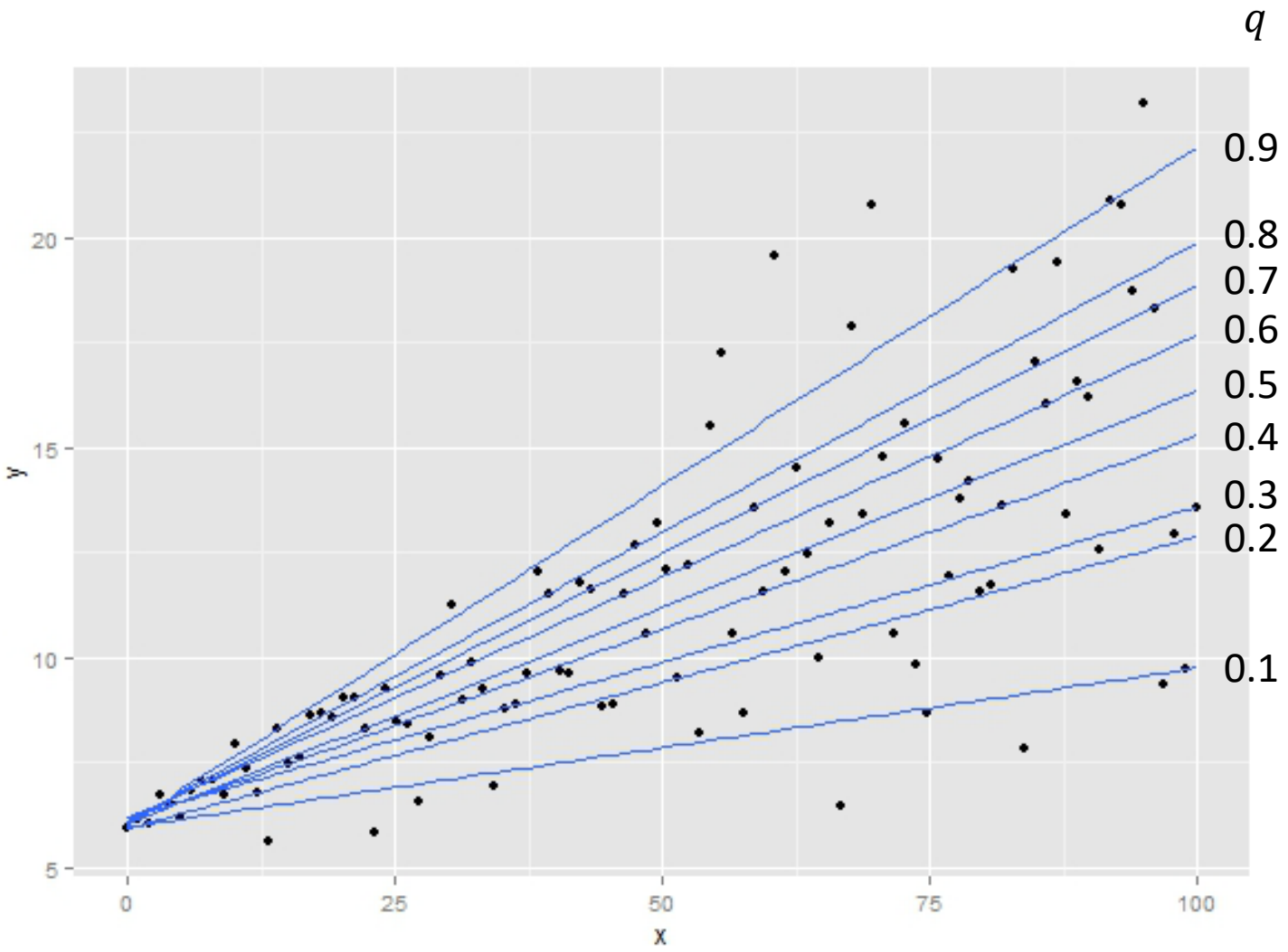
Prediction Intervals from Quantile Regression

Quantile Regression

- Suppose that for each x, there is a distribution of possible values of y: $P(y|x)$
- Quantile regression for quantile q seeks to predict the value of $\hat{y}(q) = f_q(x)$ such that $P(y \leq \hat{y}(q)|x) = q$
- The quantile function $f_q(\mathrm{x}) = F^{-1}(q|x)$, the conditional, inverse CDF
- There are several algorithms for quantile regression. I like Quantile Random Forests (Meinshausen, 2006)
- Idea:
  - Compute the $f_{\alpha/2}$ and $f_{1-\alpha/2}$ quantile regression functions
  - Output the prediction interval

$$\left[f_{\alpha/2}(x),\ f_{1-\alpha/2}(x)\right]$$

- Quantile regression does not provide any coverage guarantee
- We can get guarantees using Conformal Prediction



https://data.library.virginia.edu/files/qreg_fig_5.jpeg

Basic Idea of Conformal Prediction

(Vovk, Gammerman, Shafer, 2005)

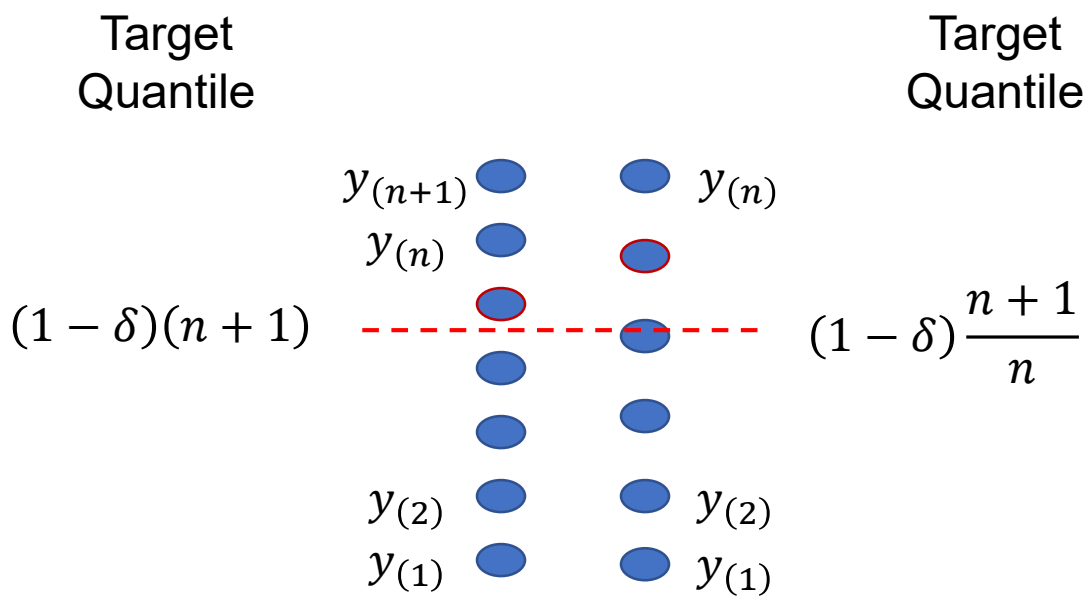
- Let's just consider an upper bound y_{hi}
- Let $y_1, \ldots, y_n, y_{n+1} \sim P(\cdot)$ $y_i \in \mathbb{R}$ "exchangeable draws"
- Define $S = \{y_1, \ldots, y_n\}$ "training data" (no classifier)
- Algorithm:
  - Let $y_{(1)}, \ldots, y_{(n)}$ be the order statistics (sorted order) of $y_1, \ldots, y_n$
  - $y_{hi}(S) \coloneqq y_{(\lceil (1-\delta)(n+1) \rceil)}$ where $\lceil x \rceil$ is the "ceiling" operator that rounds to the next larger integer
- Theorem:

$$\Pr_{y_{n+1} \sim P(\cdot)}[y_{n+1} \leq y_{hi}(S)] \geq 1 - \delta$$

for $\delta \geq \frac{1}{n+1}$

Informal Proof

- Suppose we know y_{n+1}
  - Compute $y_{(1)}, \ldots, y_{(n)}, y_{(n+1)}$
  - The rank of y_{n+1} will be **uniformly distributed** within these ranks (exchangeability)
  - The $1-\delta$ quantile estimate is element $(1-\delta)(n+1)$
  - Round up in the "safe" direction to $y_{(\lceil (1-\delta)(n+1) \rceil)}$
  - $\Pr\left[y_{n+1} \leq y_{(\lceil (1-\delta)(n+1) \rceil)}\right] \geq 1-\delta$
  - Where would the corresponding quantile be in $y_{(1)}, \ldots, y_{(n)}$?
  - Quantile estimate is element $(1-\delta)\frac{n+1}{n}$, because we now have only n points
  - Rounded up to $y_{(\lceil (1-\delta)(n+1) \rceil)}$
- This works as long as $\delta \geq \frac{1}{n+1}$
- Notice:
  - No distributional assumptions
  - Finite-sample



Extend CP to Regression: Split Conformal Prediction

- Given: $(x_1, y_1), \ldots, (x_m, y_m), (x_{m+1}, y_{m+1}), \ldots, (x_{m+n}, y_{m+n}) \sim P(X, Y)$
- Let
  - $D_1 = \{(x_1, y_1), \ldots, (x_m, y_m)\}$
  - $D_2 = \{(x_{m+1}, y_{m+1}), \ldots, (x_{m+n}, y_{m+n})\}$
- Train regression function f on D_1
- Compute prediction residuals on D_2
  $r_i := |f(x_{m+i}) - y_{m+i}|$ for $i = 1, \ldots, n$
- Sort residuals: $r_{(1)}, \ldots, r_{(n)}$
- $r^* := r_{(\lceil (1-\delta)(n+1) \rceil)}$ ("conformal correction")

Theorem: With probability $1 - \delta$ over draws $(x^*, y^*) \sim P(X, Y)$,

$$f(x^*) - r^* \leq y^* \leq f(x^*) + r^*.$$

- Weakness: The prediction interval has the same width, $2r^*$, for all x.

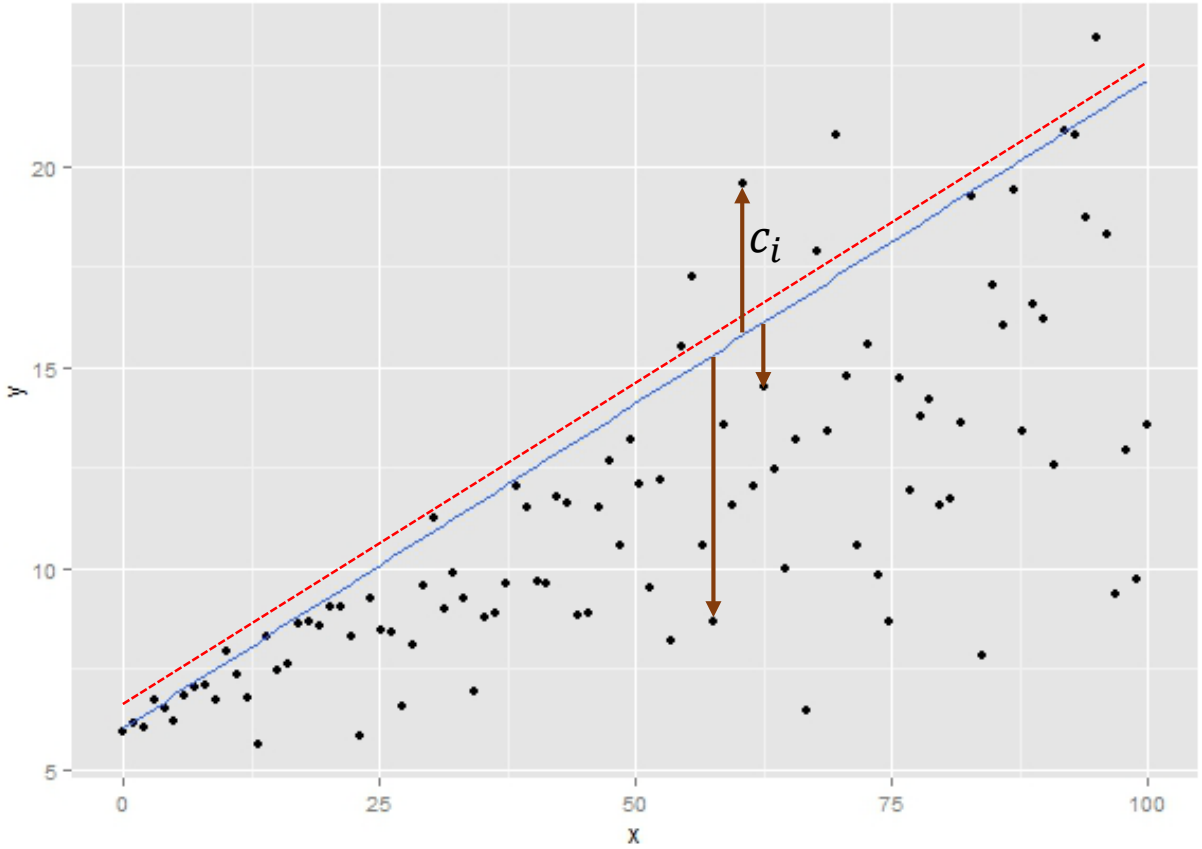
Back to Quantile Regression: Adding CP Correction

- Two data sets:
  - D_1: used for fitting quantile regressions
  - D_2: used for conformalization
- Fit quantile functions $f_{\delta/2}$ and $f_{1-\delta/2}$ to D_1
- Compute "residuals" on D_2

$$c_i := \max\{f_{\delta/2}(x_i) - y_i, y_i - f_{1-\delta/2}(x_i)\}$$

- Sort to obtain $c_{(1)}, \ldots, c_{(n)}$
  - $r^* := c_{(\lceil (1-\delta)(n+1) \rceil)}$
- Let (x_q, y_q) be a new data point
  - $c_q := \max\{f_{\delta/2}(x_q) - y_q,\ y_q - f_{1-\delta/2}(x_q)\}$
- Claim: The c_i values are exchangeable ➔ rank of c_q will be uniformly distributed in $c_{(1)}, \ldots, c_{(n+1)}$
- Therefore, $P[c_q \leq r^*] \geq 1 - \delta$
- Bounds:
  - $lo(x_q) := f_{\delta/2}(x_q) - r^*$
  - $hi(x_q) := f_{1-\delta/2}(x_q) + r^*$
- Theorem:

$$P(f_{\delta/2}(x_q) - r^* \leq y_q \leq f_{1-\delta/2}(x_q) + r^*) \geq 1 - \delta$$



Note

- By splitting our data (split conformal), we are using $\mathcal{D}_2$ to quantify the predictive uncertainty of the single model that was fit to $\mathcal{D}_1$.
- We are not quantifying the epistemic uncertainty of fitting that model.
- We are converting “Case 1” (full uncertainty quantification) into “Case 2” (uncertainty quantification for a single model)

What sources of uncertainty are captured by conformal quantile regression interval?

Captured

- Missing features
- Noisy feature measurement
- Noisy labels (response variable)
- Insufficient model class
- Imputed missing values (as long as missing at random)
- Bad learning algorithm

Not Captured

- Lack of sufficient data (epistemic uncertainty)
- Local epistemic uncertainty
- Data shift

Outline

- Goals of Uncertainty Quantification (UQ)
- Aleatoric vs. epistemic uncertainty and what causes each
- UQ as Prediction Intervals for Regression
  - Linear regression prediction intervals
  - Bayesian prediction intervals: Gaussian Processes
  - Conformal Quantile Regression intervals
- UQ for Classification
  - Calibration
  - Label sets as prediction intervals
- Local Epistemic Uncertainty
  - Outlier/Anomaly Detection
- Applications
  - Active Learning
  - Uncertainty-Aware Learning
  - Selective Prediction
  - Reducing LLM Hallucination

Uncertainty Quantification for Classifiers

1. Ensure the accuracy of $\hat{p}(y|x) = [\hat{p}(y = 1|x), \dots, \hat{p}(y = K|x)]$
2. Output a prediction interval
   - For classifiers, a prediction interval is a set $Y(x_q) = \{k_1, k_2, \dots, k_L\}$ such that with probability $1 - \delta$, $y_q \in Y(x_q)$

- Let's consider a single classifier $\hat{p}$ and ignore epistemic uncertainty

Calibrated Classifiers: The Ideal

- A classifier is well-calibrated if the predicted probability $\hat{p}(y = k|x_q)$ is equal to the true conditional probability $P\left(y = k\middle|x_q\right)$ for all classes $k \in \{1, \dots, K\}$

- This is a point-wise statement for a specific x_q. It cannot be achieved in practice unless we have several training examples that exactly match x_q

Practical Definition of Calibration

- We will use a *set* of points $C_q \subseteq \mathcal{C}$ to compute an estimate $\hat{P}(y = k|x_q)$:

$$\hat{P}(y = k|x_q) = \frac{1}{|C_q|} \sum_{x_i \in C_q} \mathbb{I}[y_i = k]$$

- The classifier is calibrated if

$$\hat{p}(y = k|x_q) \approx \hat{P}(y = k|x_q)$$

Defining C_q in terms of $f(x_q)$

- Confidence Calibration: Only calibrate the predicted class $\hat{y}$
  - Let $\alpha = \hat{p}(\hat{y}|x_q)$
  - Let $C_\alpha = \{x_i \in \mathcal{C}: \hat{p}(y_i = \hat{y}|x_i) = \alpha\}$
    - The set of all points $x \in \mathcal{C}$ where the predicted class $\hat{y}$ is assigned the same predicted probability α
  - $\hat{P}(y = \hat{y}|x_q) = \frac{1}{|C_\alpha|}\sum_{x_i \in C_\alpha} \mathbb{I}[y_i = \hat{y}]$
  - The classifier is calibrated if $\hat{P}(y = \hat{y}|x_q) = \alpha$
  - "The weather forecast is well-calibrated if on all days where the forecast says 80% chance of rain ($X_{0.80}$), it rains 80% of the time"

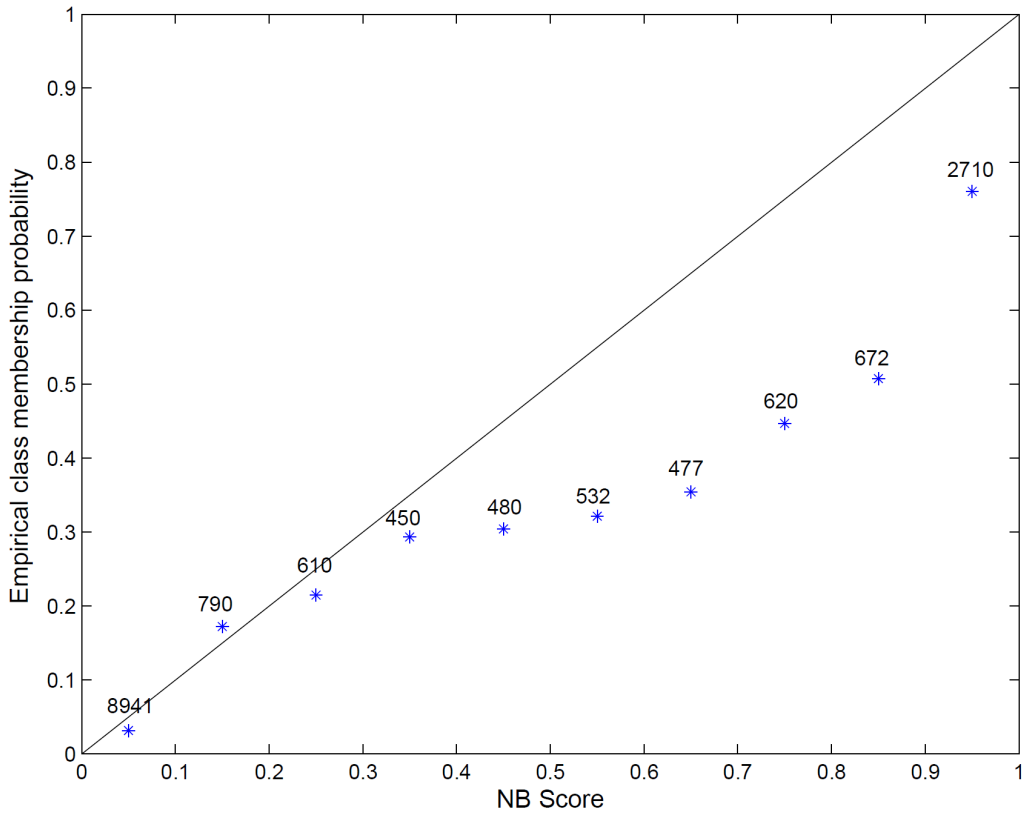
Defining C_q in terms of $f(x_q)$

- Multi-class Calibration ("Full Calibration")
  - Let $\vec{\alpha} = f(x_q)$
  - Let $C_{\vec{\alpha}} = \{x_i \in \mathcal{C} \mid \hat{p}(y_i|x_i) = \vec{\alpha}\}$
    - The set of all points $x_i \in \mathcal{C}$ where the predicted probability vector is exactly $\vec{\alpha}$
  - $\hat{P}(y|x_q) = \frac{1}{|C_{\vec{\alpha}}|} \sum_{x_i \in C_{\vec{\alpha}}} y_i$
    - where y is one-hot encoded: $(\mathbb{I}[y = 1], \ldots, \mathbb{I}[y = K])$
  - The classifier is calibrated if $\hat{P}(y|x_q) = \vec{\alpha}$
- Issue: C_α and $C_{\vec{\alpha}}$ may be very small

Visualizing Calibration with a Reliability Diagram

- Consider the 2-class case
  - Define a set of M bins $C_1, \ldots, C_M \subset C$ based on $\hat{p}(y = 1|x)$
  - Compute $\hat{\alpha}$ and $\hat{P}$ for each bin
  - Plot $(\hat{\alpha}, \hat{P})$ pairs
  - Let $P_x(b)$ be the fraction of $x_i \in C_b$
- Expected Squared Calibration Error
  - $\sum_{b=1}^{M} P_x(b)\left[\hat{p}(C_b) - \hat{P}(C_b)\right]^2$ expected squared calibration error
- Expected Calibration Error (ECE)
  - $\sum_{b=1}^{M} P_x(b)\left|\hat{p}(C_b) - \hat{P}(C_b)\right|$ expected absolute calibration error

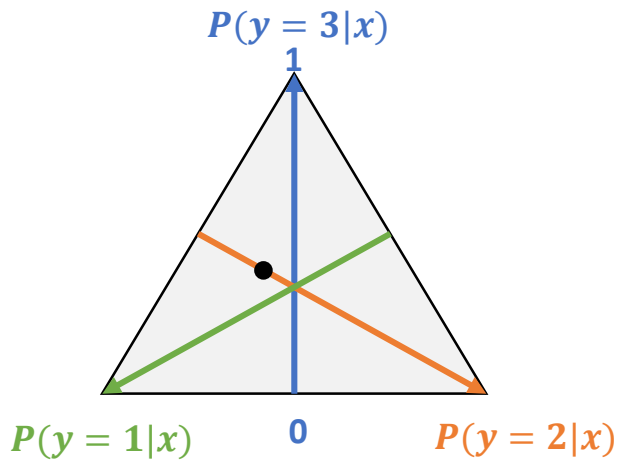
Reliability Diagram (Naïve Bayes; ADULT)



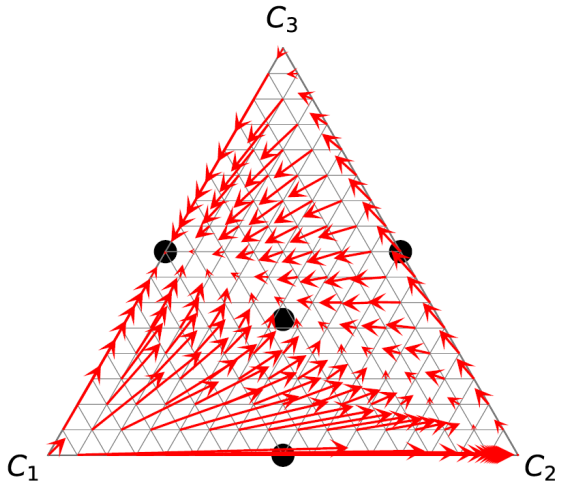
Zadrozny & Elkan, 2002

Ensuring the accuracy of $P(y|x)$ via calibration

- Let $f(x)$ be a vector-valued function that outputs $[\hat{p}(y = 1|x), \ldots, \hat{p}(y = K|x)]$.
  - This probability vector can be viewed as a point in the $K - 1$ dimensional simplex
- Let $\mathcal{C}$ be a "calibration set" of data points $(x_1, y_1), \ldots, (x_n, y_n)$ drawn iid from the same distribution as the test data
- We can use these data points to learn a *calibration map* g that maps from Δ^{K-1} to Δ^{K-1} such that $g(f(x))$ is well-calibrated



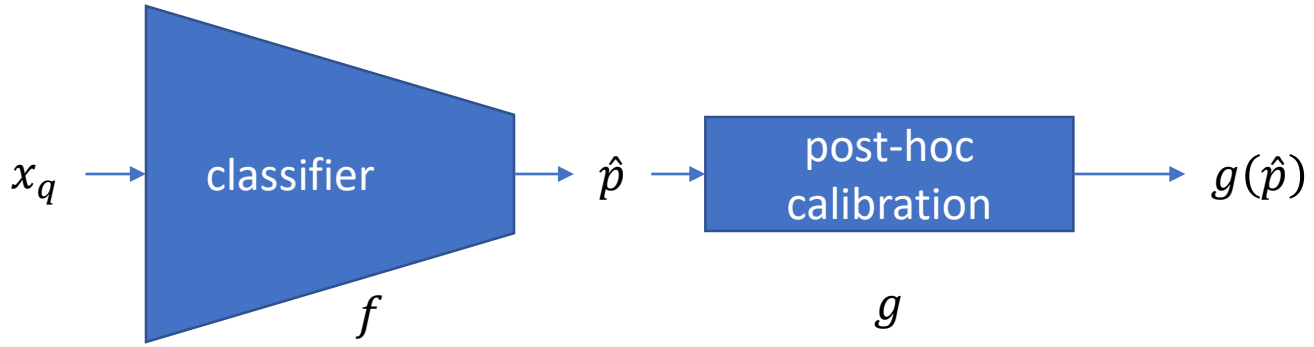
- $(0.375, 0.250, 0.375)$

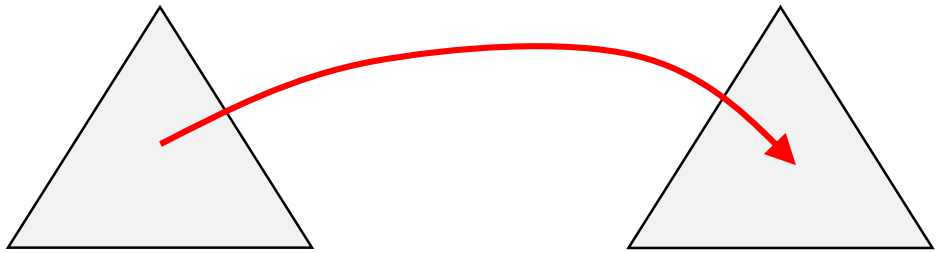


[Kull, et al., 2019]

Fitting a Calibration Map g

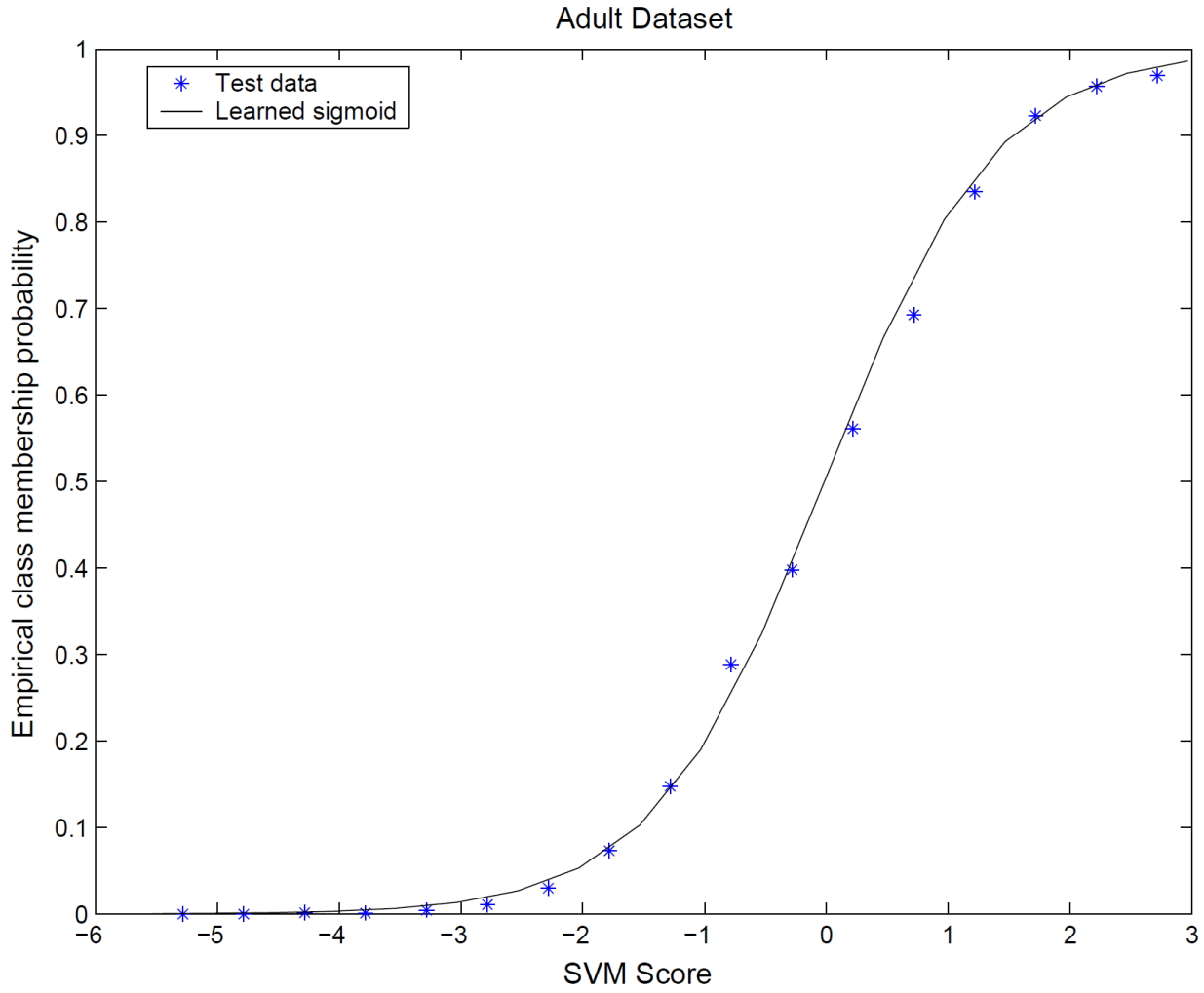
- Learn a "calibration map" g that transforms the classifier's output probabilities into well-calibrated probabilities:





Calibration for the predicted class: Platt Scaling
(Platt, 1999)

- Create labeled "training data" consisting of
$$(\hat{p}(y = \hat{y}|x_i), \mathbb{I}[y_i = \hat{y}])$$
pairs
- Fit the calibration map:
$$g(\hat{p}; a, b) = \frac{1}{1 + e^{a+b\hat{p}}}$$
- This is logistic regression with a single "feature" ($\hat{p}$)
- Minimizes the log loss



Full calibration via Softmax Temperature Scaling

(Guo et al, 2017)

- Let $\ell = (\ell_1, \ldots, \ell_K)$ be the logits of a softmax classifier
- Scale the logits by dividing by a temperature T:

$$\hat{p}(y = k|x) = \frac{\exp \frac{\ell_k}{T}}{\sum_{k'} \exp \frac{\ell_{k'}}{T}}$$

- Adjust T to fit the calibration data

Boosted Trees after Platt Scaling

(Niculescu-Mizil & Caruana, 2005)

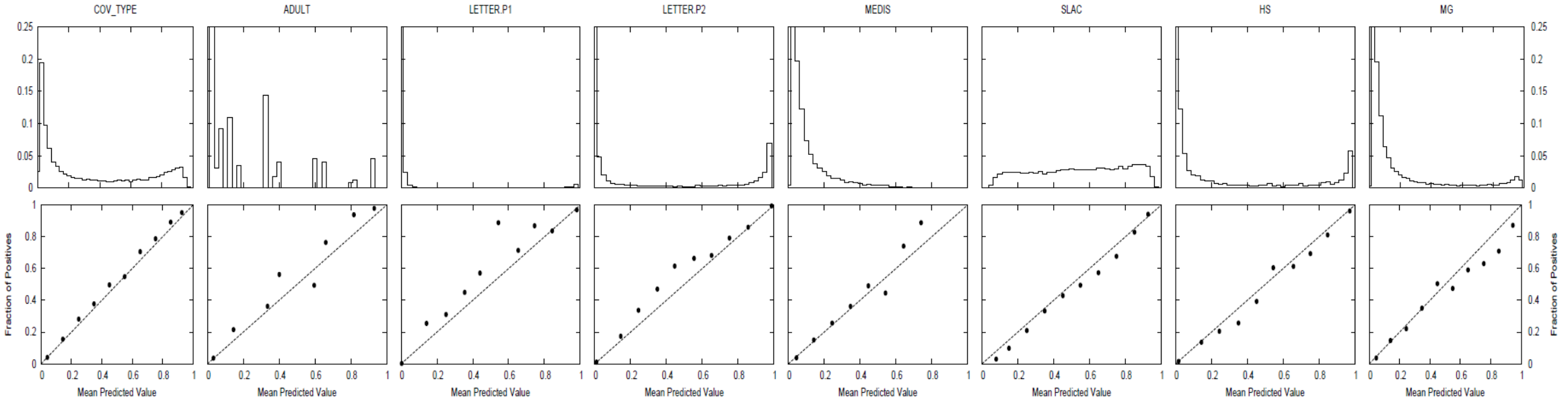
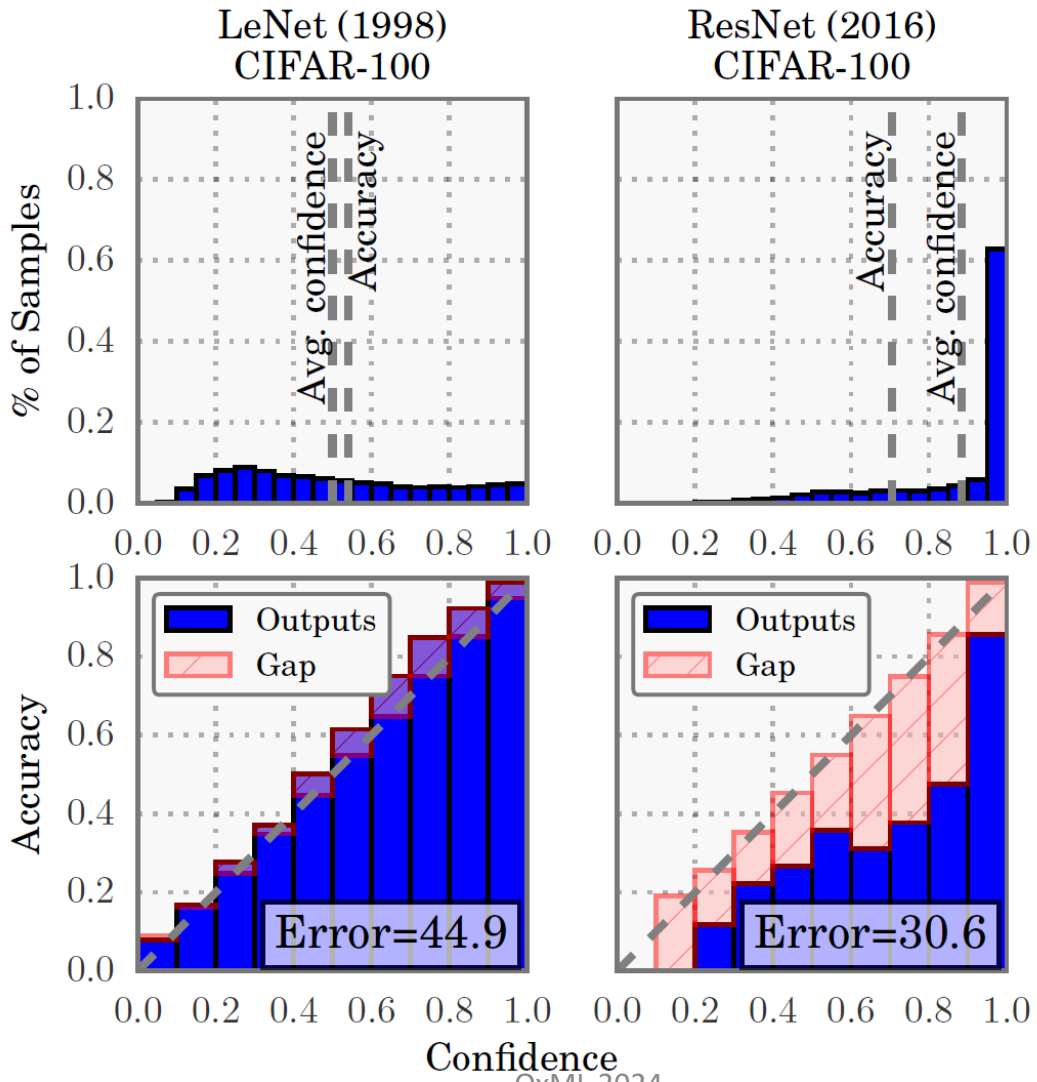


Figure 2. Histograms of predicted values and reliability diagrams for boosted trees calibrated with Platt's method.

ResNet-110 with Temperature Scaling

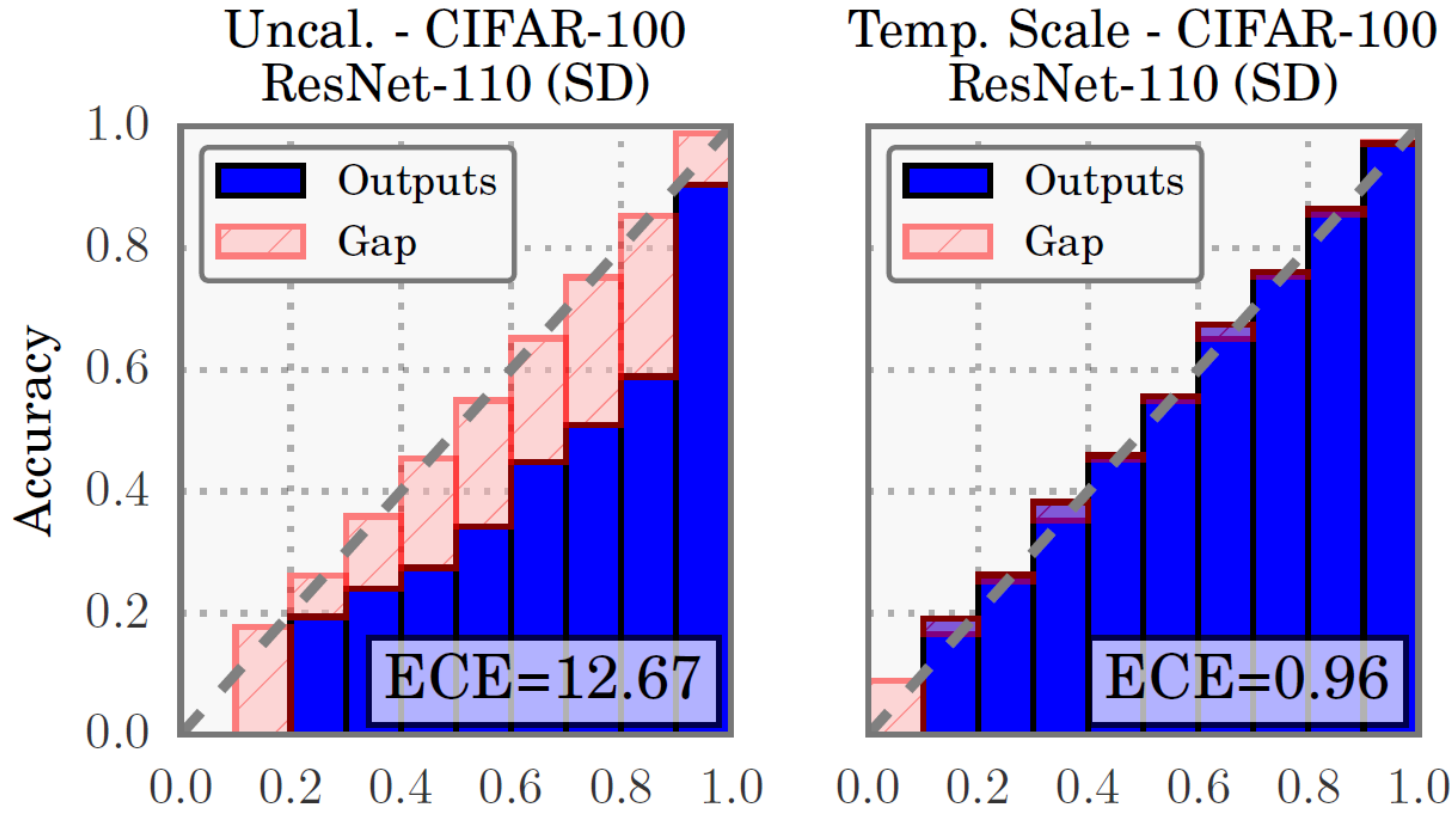
(Guo, Pleiss, Sun & Weinberger, 2017)



ResNet is much more confident

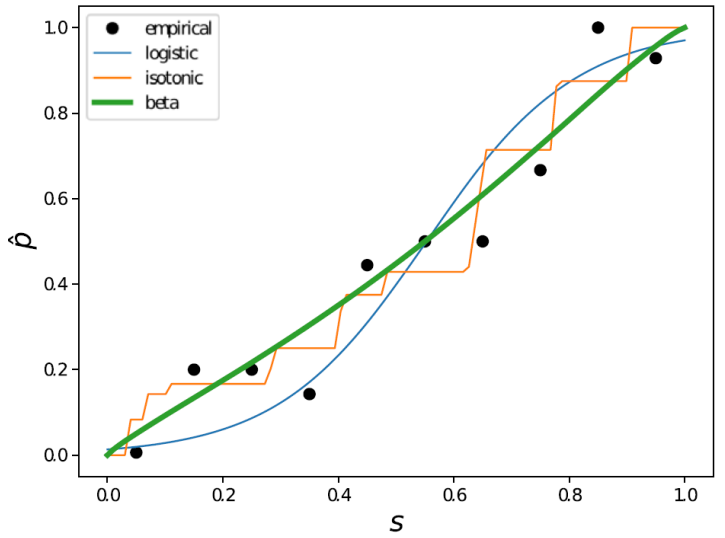
ResNet over-confident!

Results before/after Temperature Scaling

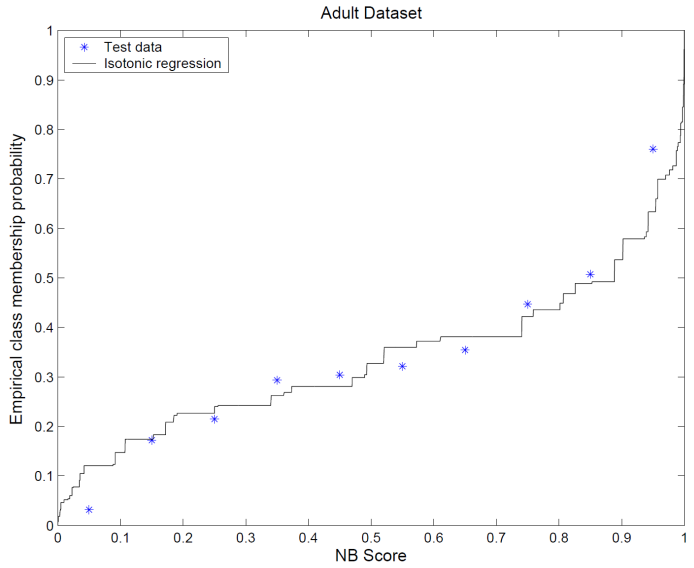


Other probability maps

- Beta and Dirichlet calibration. Similar to Platt scaling, but with a slightly more expressive model
  - Kull, et al. 2019
  - Key advantage: Contains the identity function, unlike Platt Scaling. This allows it to correctly handle classifiers that are already well calibrated.
- Isotonic Regression: Fits variable-width histogram bins to minimize the squared calibration error
  - Ayer, et al. (1955)
  - Robertson, Wright, & Dykstra (1988)
  - Risk of overfitting



(b) Naive Bayes – vowel



Computing Prediction Intervals from Calibrated Probabilities

- Let $\hat{p}$ now denote the calibrated probabilities
- Sort in descending order of $\hat{p}(y = k|x_q)$
  - $\hat{p}_{(1)}(y = k_{(1)}|x_1), \hat{p}_{(2)}(y = k_{(2)}), \ldots$
- Let m be the smallest value such that
  - $\sum_{j=1}^{m} \hat{p}_{(j)}\left(y = k_{(j)}|x_q\right) \geq 1 - \alpha$
- Let the prediction interval be

$$Y(x_q) = \{k_{(j)} : j \leq m\}$$

- With a slight modification, we can obtain conformal guarantees

j	$k_{(j)}$	$\hat{p}_{(j)}(y = k_{(j)}\|x_q)$
1	3	0.60
2	1	0.31
3	5	0.05
4	4	0.03
5	2	0.01

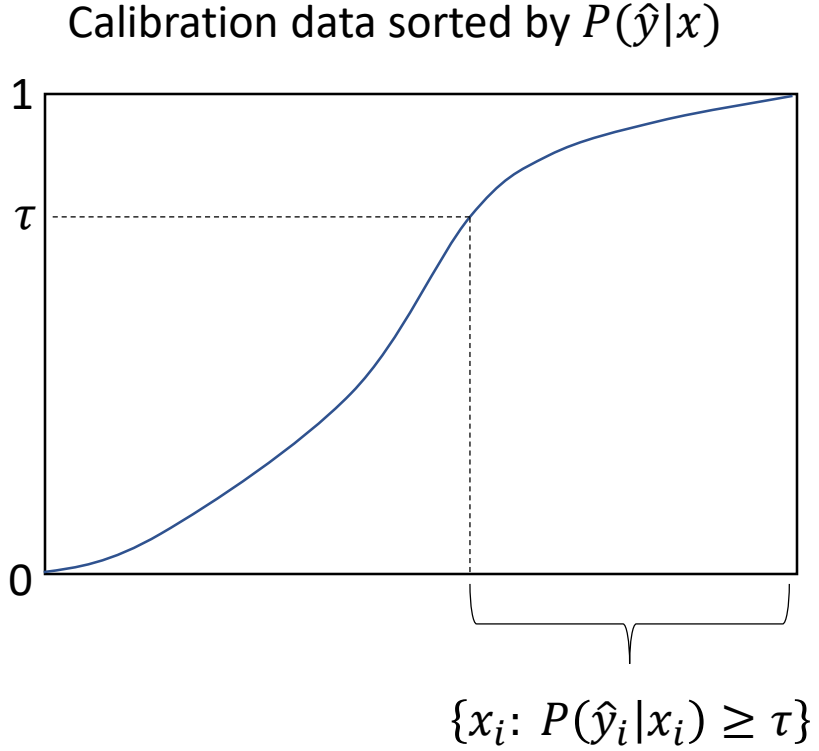
$\Sigma = 0.91$

For $1 - \alpha = 0.9$, $Y(x_q) = \{3,1\}$

Selective Classification with Calibrated Probabilities

- Given x, let $\hat{y}$ be the predicted class, and $P(\hat{y}|x)$ be the calibrated probability from the classifier.
- Let $0 \leq \tau \leq 1$ be a probability threshold
- If we require $P(\hat{y}|x) \geq \tau$, we know our classification accuracy will be at least τ.
- But it will usually be bigger. Our actual accuracy over N test points $\{x_i\}_{i=1}^{N}$ will be

$$\frac{1}{N}\sum_{i=1}^{N} P(\hat{y}_i|x_i)\mathbb{I}[P(\hat{y}_i|x_i) \geq \tau]$$



Outline

- Goals of Uncertainty Quantification (UQ)
- Aleatoric vs. epistemic uncertainty and what causes each
- UQ as Prediction Intervals for Regression
  - Linear regression prediction intervals
  - Bayesian prediction intervals: Gaussian Processes
  - Conformal Quantile Regression intervals
- UQ for Classification
  - Calibration
  - Label sets as prediction intervals
- Local Epistemic Uncertainty
  - Outlier/Anomaly Detection
- Applications
  - Active Learning
  - Uncertainty-Aware Learning
  - Selective Prediction
  - Reducing LLM Hallucination

Local Epistemic Uncertainty

- Method 1: Fit an ensemble and measure the variability at x_q
  - Unfortunately, many learning algorithms tend to predict a constant value (e.g., $1/K$) far from the training data
  - As a result, ensemble disagreement fails to accurately measure epistemic uncertainty
- Method 2: Measure some form of distance from x_q to the training data (anomaly detection)
  - Many anomaly detectors are essentially measures of distance
  - Isolation Forest is very popular; it approximates the L_1 distance from x_q to the nearest training data point
  - This requires an appropriate representation and distance metric
  - Hand-engineered features work well
  - Features learned by deep learning do not work well

Distance-Based Anomaly Detection

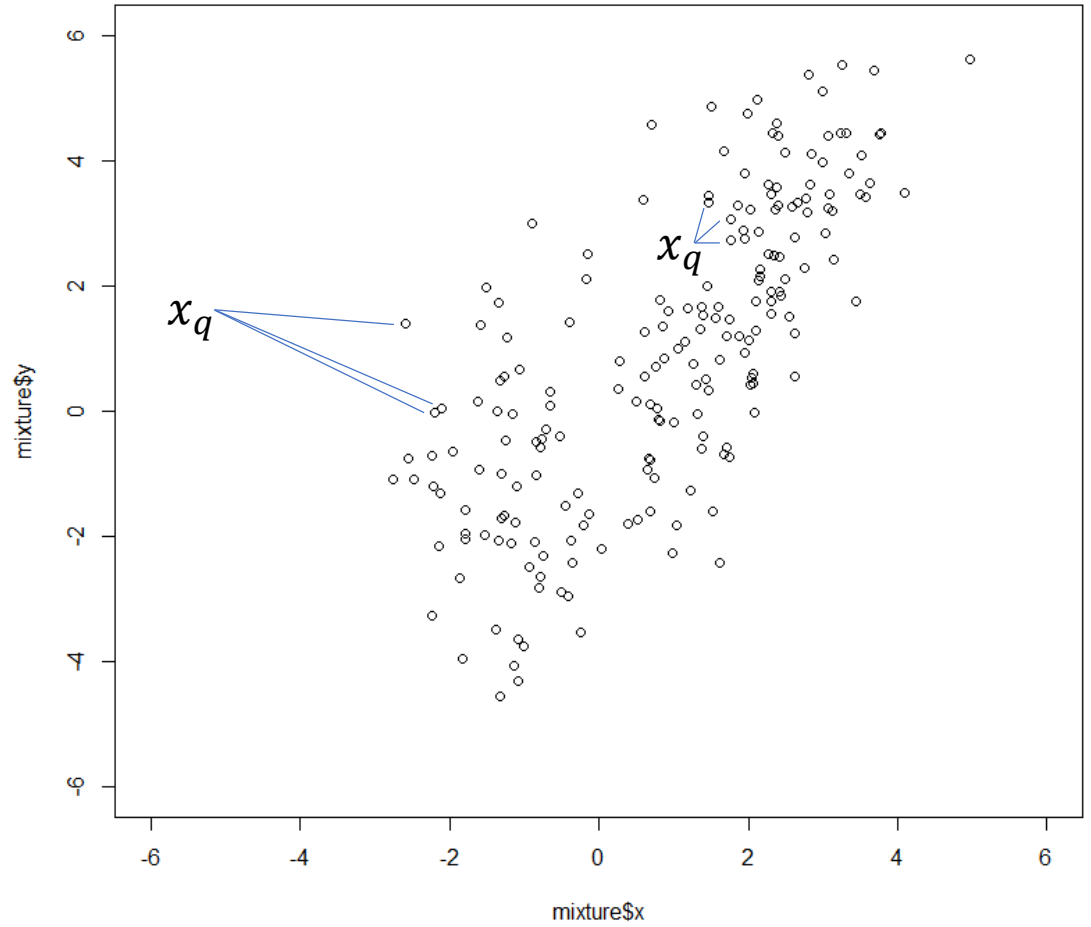
- Key challenge: Defining a good distance metric
- Case 1: Hand-Engineered Feature Vectors
- Case 2: Features learned by Deep Learning

Distance-Based Anomaly Detection

- Define a distance $d(x_i, x_j)$
  - Transform all features to have zero mean and unit standard deviation
  - Apply PCA to perform dimensionality reduction and de-correlate features
- Anomaly score:

$$A(x_q) = \min_{x \in D} d(x_q, x)$$

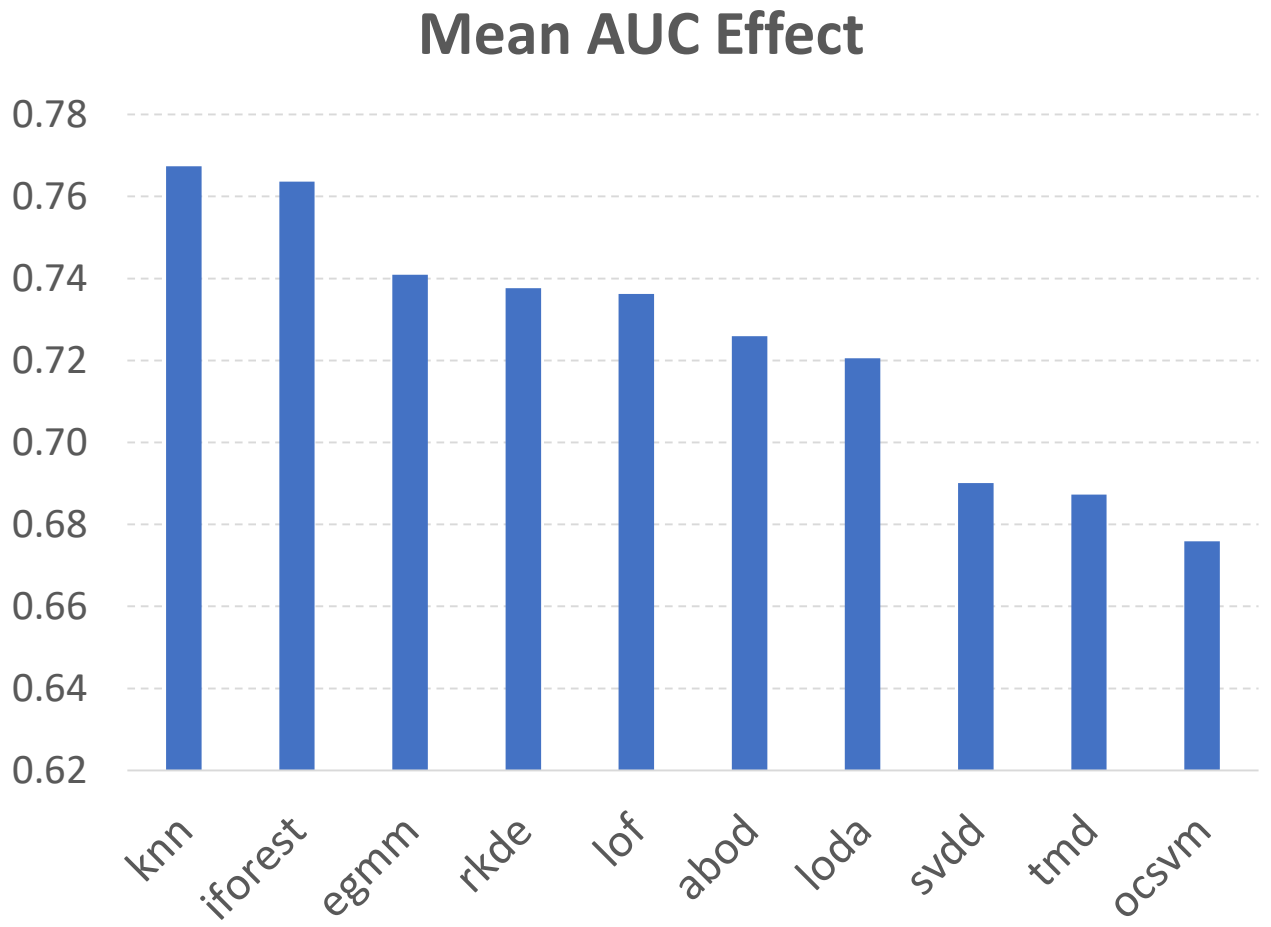
- This can be made more robust by looking at the average distance to the k-nearest points
  - "k-nn anomaly detection"
- Improved and efficient methods
  - LOF (Local Outlier Factor; Breunig, et al., 2000)
  - Isolation Forest (Liu, Ting, Zhou, 2011)



Benchmarking Study

Andrew Emmott

- 19 UCI Datasets
- 8 Leading “feature-based” algorithms
- 11,888 non-trivial benchmark datasets
- Mean AUC effect for “nominal” vs. “anomaly” decisions
  - Controlling for
    - Parent data set
    - Difficulty of individual queries
    - Fraction of anomalies
    - Irrelevant features
    - Clusteredness of anomalies
- Baseline method: Distance to nominal mean (“tmd”)
- Best methods: K-nearest neighbors and Isolation Forest
- Worst methods: Kernel-based OCSVM and SVDD



[Emmott, Das, Dietterich, Fern, Wong, 2013; KDD ODD-2013]
[Emmott, Das, Dietterich, Fern, Wong. 2016; arXiv 1503.01158v2]
[Emmott, MS Thesis. 2020]

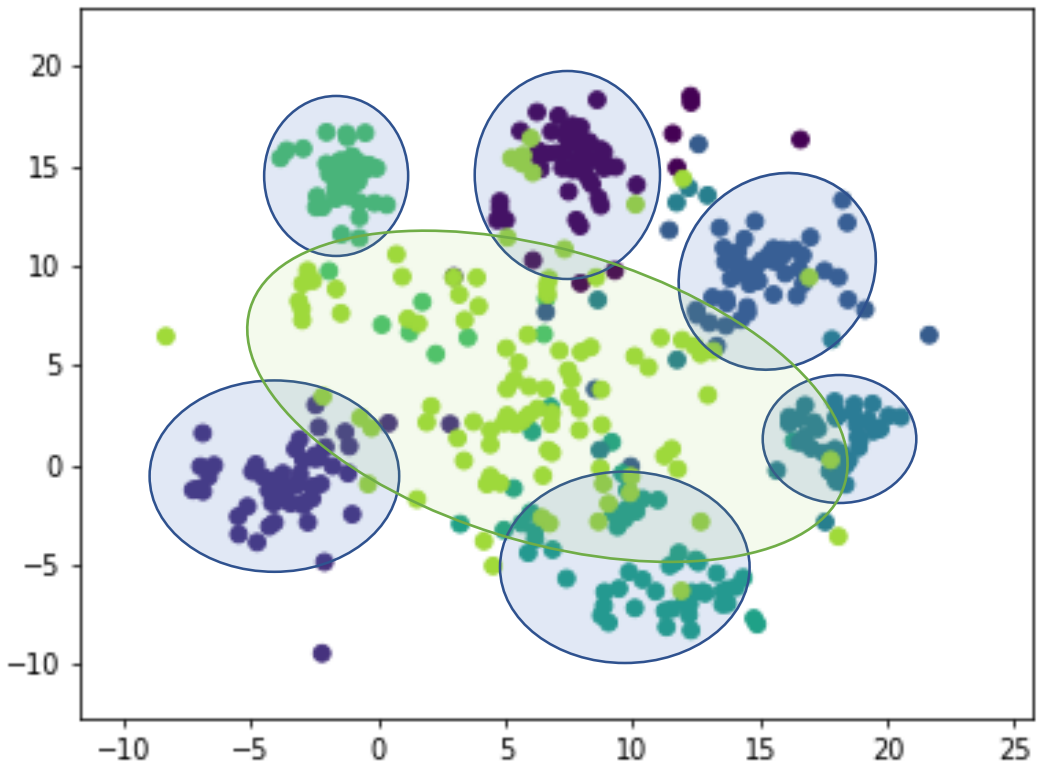
Deep Anomaly Detection

- An important advantage of deep learning is that it learns its own internal features
  - Euclidean distance in pixel space is not useful
- Problem: Deep learning only learns features that it needs for the training task. These features may not separate out-of-distribution queries x_q

Deep Learned Features in Computer Vision

- DenseNet with 384-dimensional latent space.
- CIFAR-10: 6 known classes, 4 novel classes
- Light green: novel classes
- Darker greens: known classes

- Images from known classes are “pulled out” from the center of the space
- Most novel-class images stay toward the center of the space; others overlap with known classes
- Novel images are “inliers”



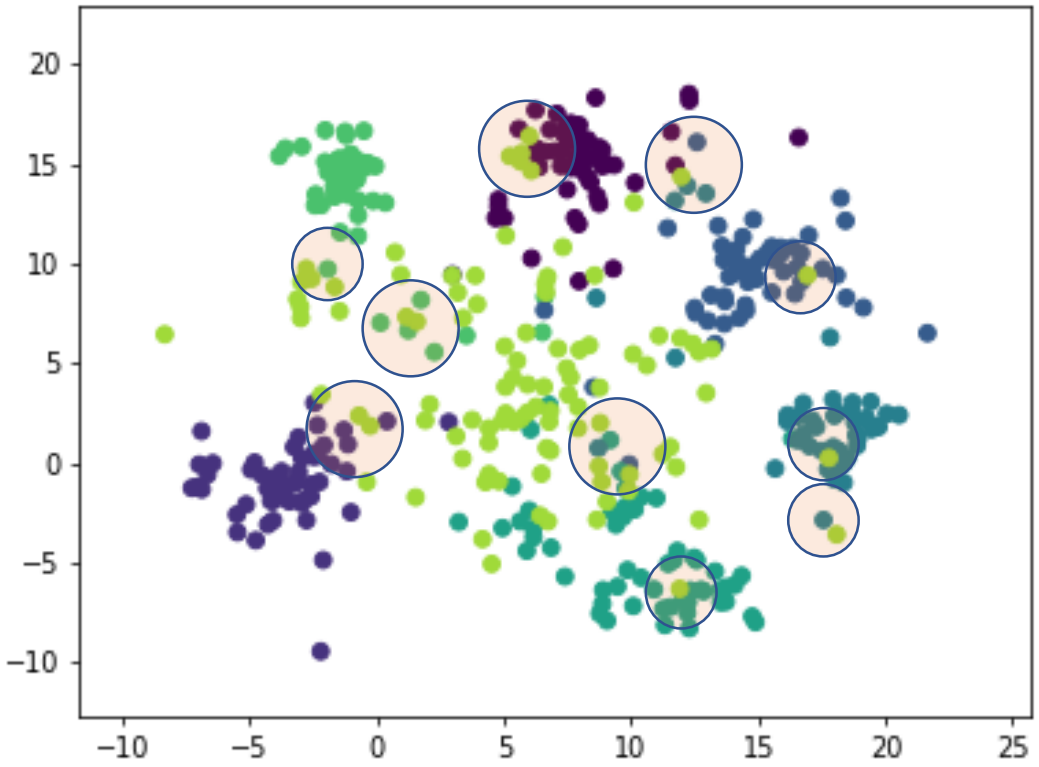
6 Known Classes

4 Novel Classes

Dietterich & Guyer, 2022

The Learned Representation is Promising But Not a Complete Solution

- Many novel-class images are mapped into clusters of known images
- ➔The learned representation can't detect the novelty



How can we learn better features?

- Foundation Model Approach:
  - Train on more data
  - Train on additional classes
- Artificially introduce variation through augmentations
  - Rotations, flips, simulated snow, rain, pixel noise, etc.
- Synthetic data?

Training on Auxiliary Classes

- CIFAR-100
  - 10 known classes
  - 10 novel classes
  - 80 auxiliary classes
- Train on “known” + “auxiliary”
- Test on “known” + “novel”

Configuration	AUC ± s.d.
Normal	0.757 ±0.033
Auxiliary	0.886 ±0.036

- More effective than pre-training + fine-tuning?

Dietterich & Guyer, 2022)

Outline

- Goals of Uncertainty Quantification (UQ)
- Aleatoric vs. epistemic uncertainty and what causes each
- UQ as Prediction Intervals for Regression
  - Linear regression prediction intervals
  - Bayesian prediction intervals: Gaussian Processes
  - Conformal Quantile Regression intervals
- UQ for Classification
  - Calibration
  - Label sets as prediction intervals
- Local Epistemic Uncertainty
  - Outlier/Anomaly Detection
- Applications
  - Active Learning
  - Uncertainty-Aware Learning
  - Selective Prediction
  - Reducing LLM Hallucination

Applications of Uncertainty Quantification

- Active Learning
- Uncertainty-Sensitive Learning
- Selective Classification
  - Reducing Hallucination in Large Language Models

Application: Active Learning

- Assume we have a large set of unlabeled data points $\mathcal{U}$
- Find the data point $x \in \mathcal{U}$ with maximum epistemic uncertainty
- Query human expert to obtain y
- Add (x, y) to training data and update the classifier (retrain if necessary)
- This can be done in batches (of course).
- When the epistemic uncertainty is measured using an ensemble, this is known as “Query-by-Committee”
- Active learning using aleatoric uncertainty is called “Uncertainty Sampling”
- Current research suggests that active learning methods should take a more global view of the data (e.g., construct a clustering and then use it for sampling), so minimizing epistemic uncertainty alone is not state-of-the-art

Uncertainty-Sensitive Learning

- Data points (x_i, y_i) with high aleatoric uncertainty should be down-weighted during learning
- Loss function:

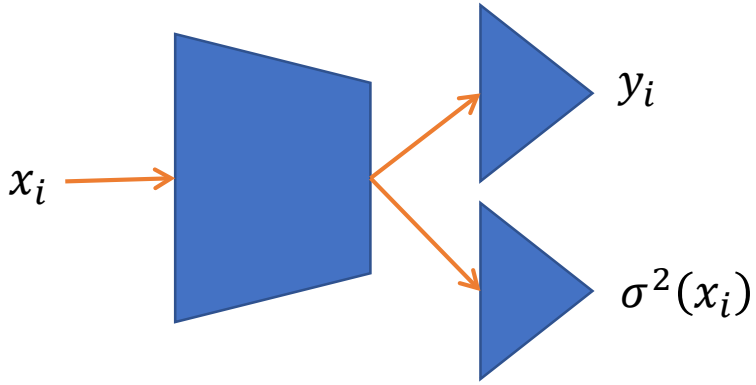
$$\sum_i \frac{1}{\sigma^2(x_i)} (f(x_i) - y_i)^2 + \log \sigma^2(x_i)$$

Implementation: Two-Headed Neural Network

(Kendall & Gal, 2017)

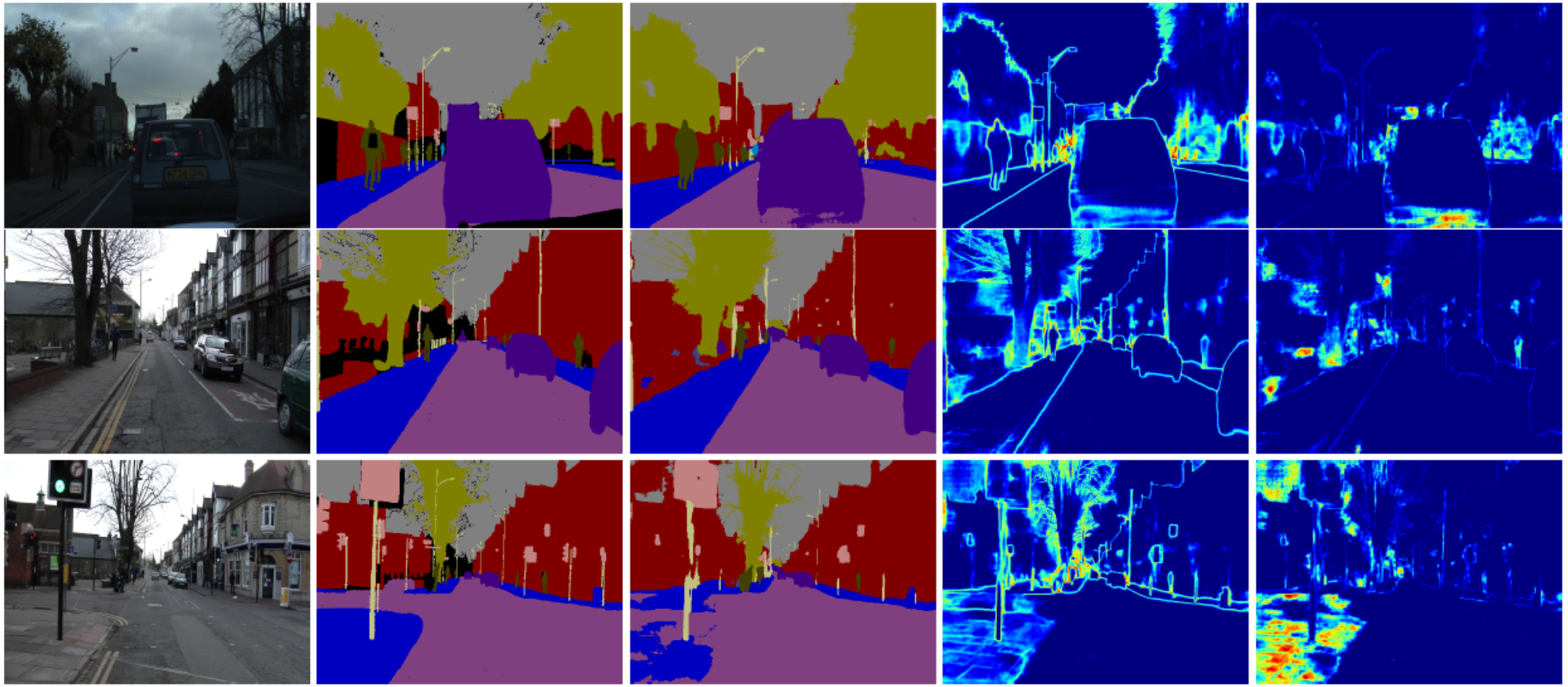
- The $x_i \rightarrow y_i$ training via backpropagation
- θ are the weights of the network
- $\sigma^2(x_i)$ is not supervised. It is just constrained by the log likelihood

$$\mathcal{L}(\theta) = \sum_i \log \sigma_\theta(x_i) + \frac{(f_\theta(x_i) - y_i)^2}{2\sigma_\theta(x_i)^2}$$



- Other regularization terms are not shown

Kendall & Gal: Semantic Segmentation and Depth Estimation



(a) Input Image (b) Ground Truth (c) Semantic Segmentation (d) Aleatoric Uncertainty (e) Epistemic Uncertainty

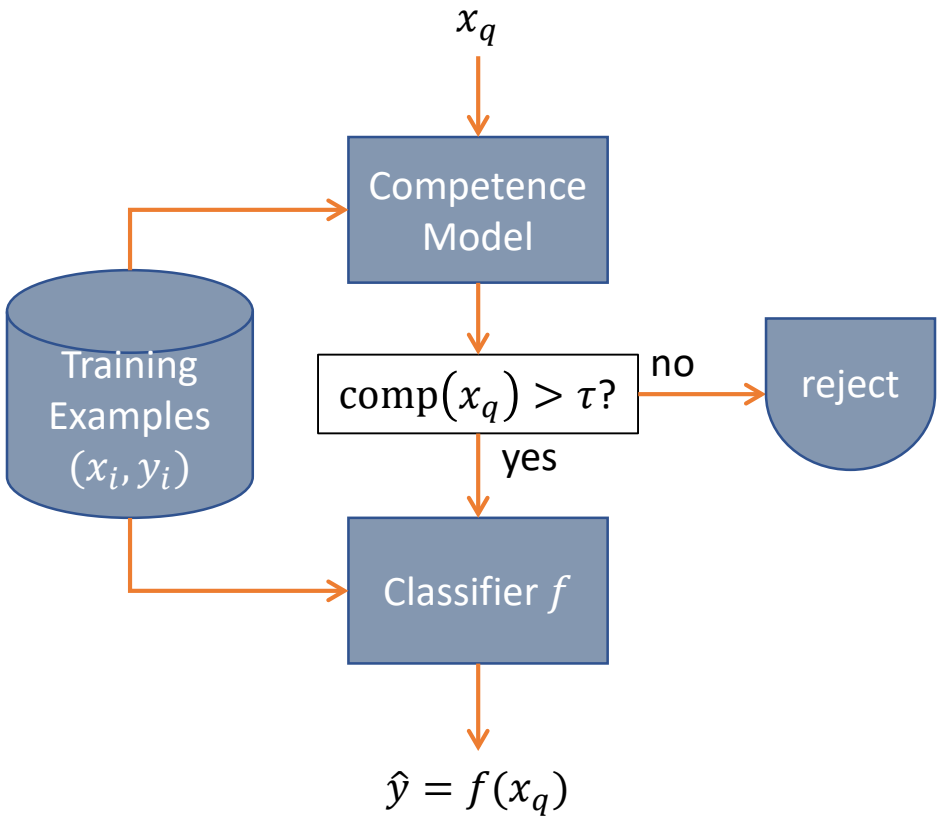
Kendall & Gal Results

Task	Metric	Standard Method	Uncertainty Aware Method
Semantic Segmentation	IOU	67.1	67.4
Indoor Scenes (NYUv2)	Accuracy	70.1	70.4
	IOU	36.5	37.1
Make 3D Depth	Relative Error	0.167	0.149
NYU v2 Depth	Relative Error	0.117	0.112

Small but useful improvements

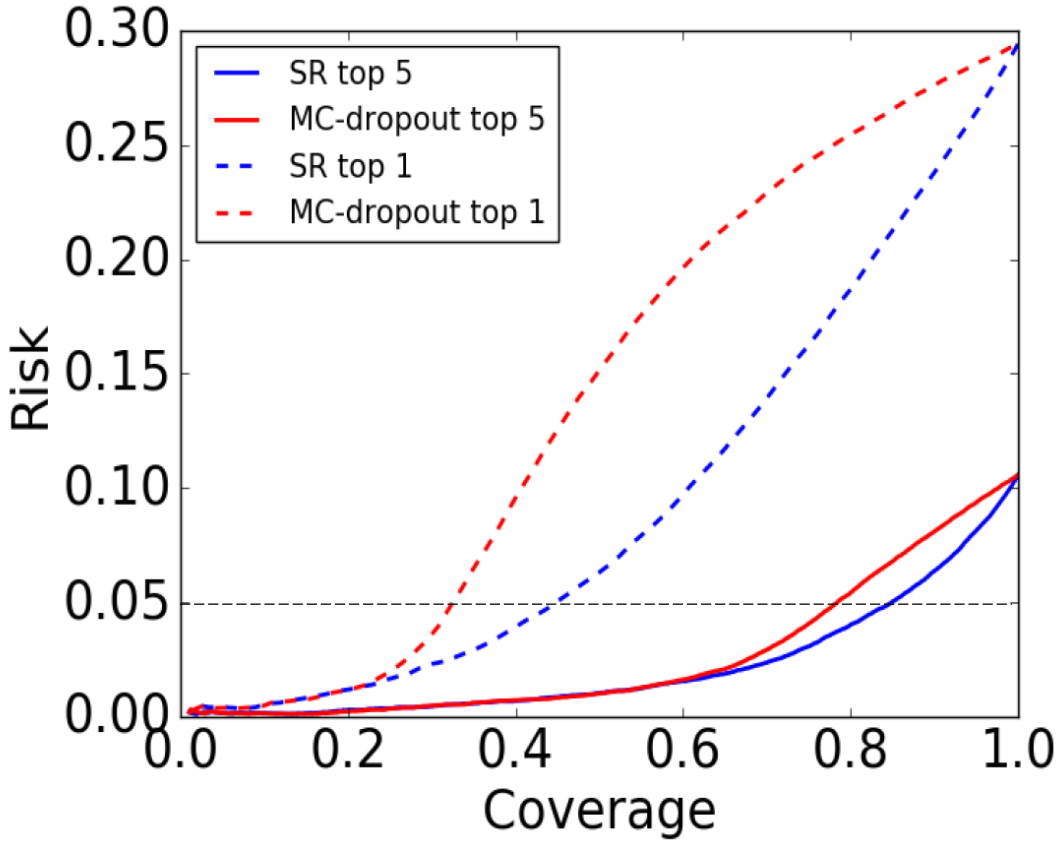
Application: Selective Prediction

- Competence model $\mathrm{comp}(x_q)$
  - Should combine epistemic and aleatoric uncertainty
- Combined prediction intervals:
  - Width of linear regression prediction interval $< \tau$
  - Width of Bayesian prediction interval $< \tau$
- Separate epistemic and aleatoric UQ
  - Anomaly score $A(x_q) < \tau_e$
  - Regression: Width of conformal prediction interval $< \tau_a$
  - Classification: Calibrated probability of predicted class $\hat{p}(\hat{y}|x_q) > \tau_a$
    - See Fisch, Jaakola & Barzilay 2022: Calibrated selective classification



Selective Prediction: Rejection Curves

- Comparison of two uncertainty signals
  - "SR": $\hat{p}(\hat{y}|x_q)$ probability assigned to the predicted class label (aleatoric uncertainty)
  - "MC-dropout": variance of dropout predictions for x_q (epistemic uncertainty)
- For each setting of the threshold τ, compute
  - Coverage: Fraction of test data points that are not rejected
  - Risk: Error rate on the test data points that are not rejected
- Aleatoric uncertainty gave better results than epistemic uncertainty
- We can achieve 5% error rate (for top-5) while rejecting only about 15% of the test queries



(c) Image-Net

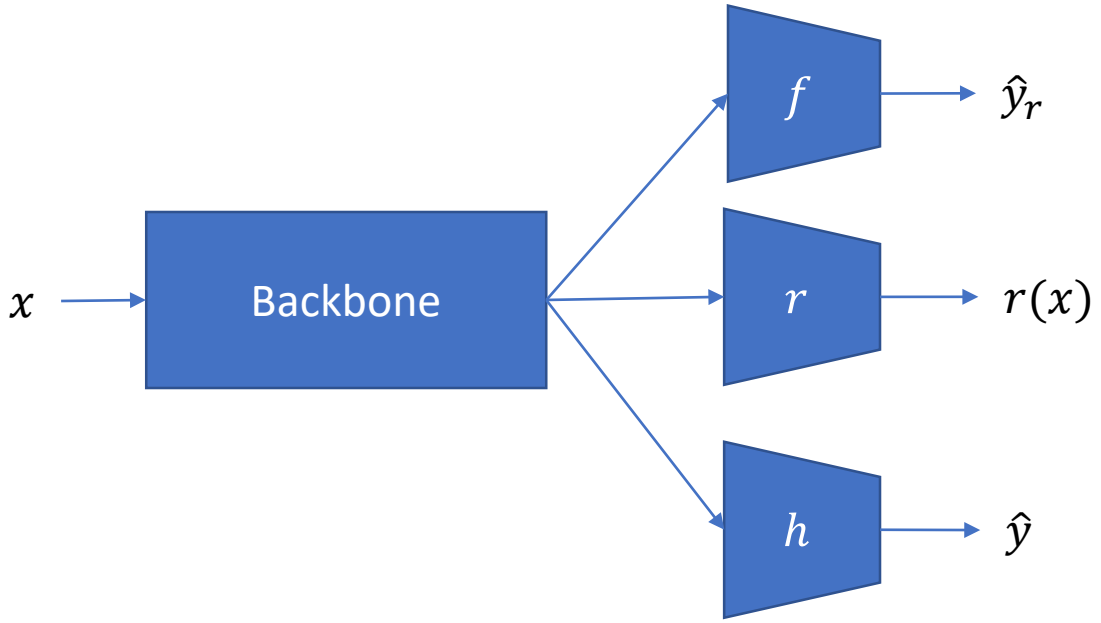
(Geifman & El Yaniv, 2017)

Learning under selective prediction

- Suppose we know we are going to be doing selective prediction
- Idea: Specify the desired coverage $1 - \alpha$ at training time; then maximize accuracy subject to this coverage
  - The learning algorithm can choose which α training examples to ignore
  - This may make it easier to learn a more accurate classifier on the non-rejected examples

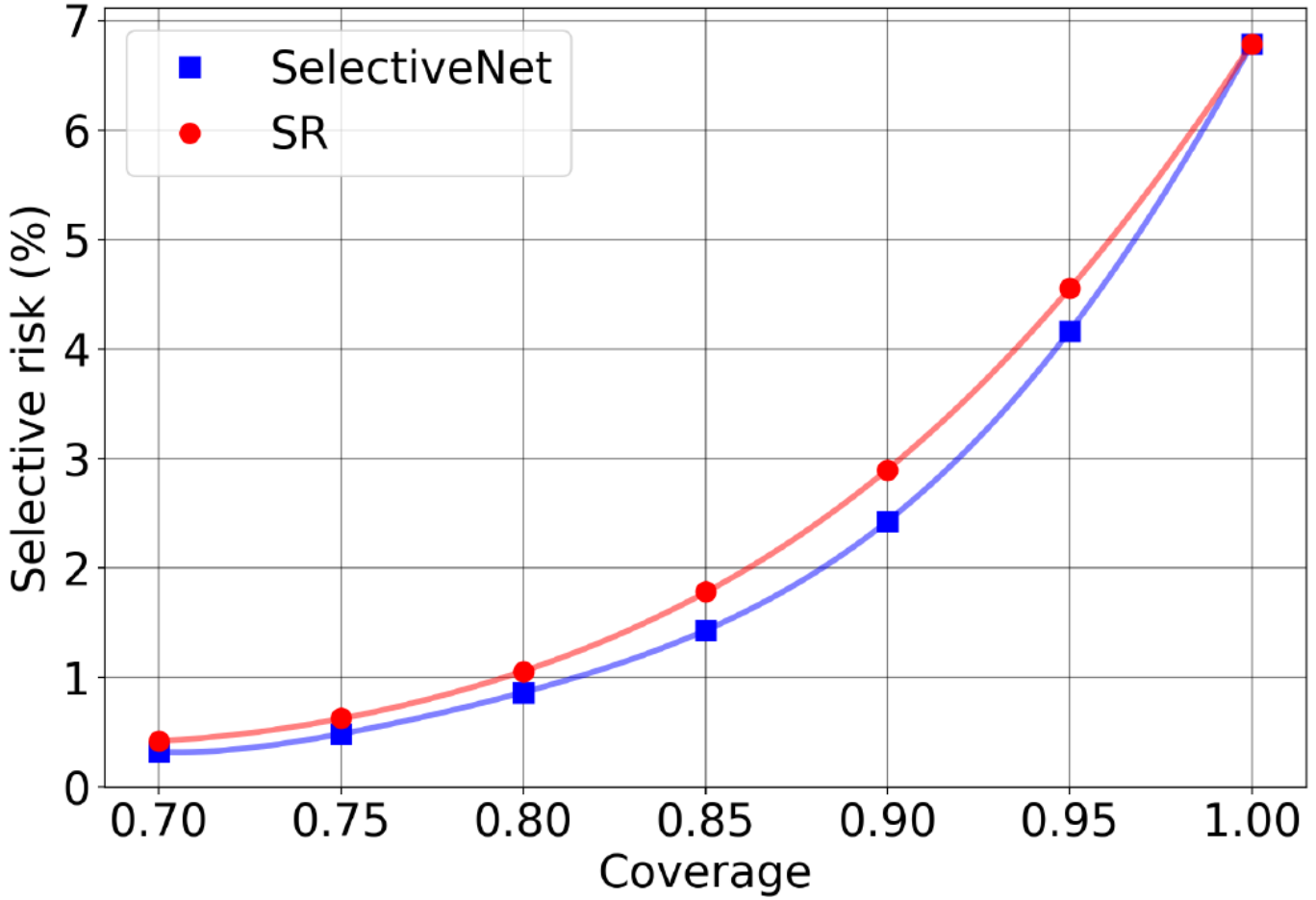
SelectiveNet

- Geifman & El Yaniv (2019)
  - Minimize the error on the non-rejected images subject to a constraint on the coverage (fraction of images not rejected)
  - User must specify c, the target coverage, rather than ϵ, the target error rate
- Network has three "heads"
  - f and h are both classification heads trained with cross-entropy loss
  - r is the rejection classifier (reject if $r(x) \geq 0.5$)
  - h encourages the backbone to learn a latent representation that can classify *all* of the examples
- Loss function:
  - $\alpha\mathcal{L}_f + (1-\alpha)\mathcal{L}_h + \lambda([c - \phi(r)]_+)^2$
    - $\mathcal{L}_f$ classification loss on training examples for which $r(x) < 0.5$
    - $\mathcal{L}_h$ classification loss on all training examples
    - $\phi(r)$: fraction of training examples for which $r(x) < 0.5$; (i.e., not rejected)



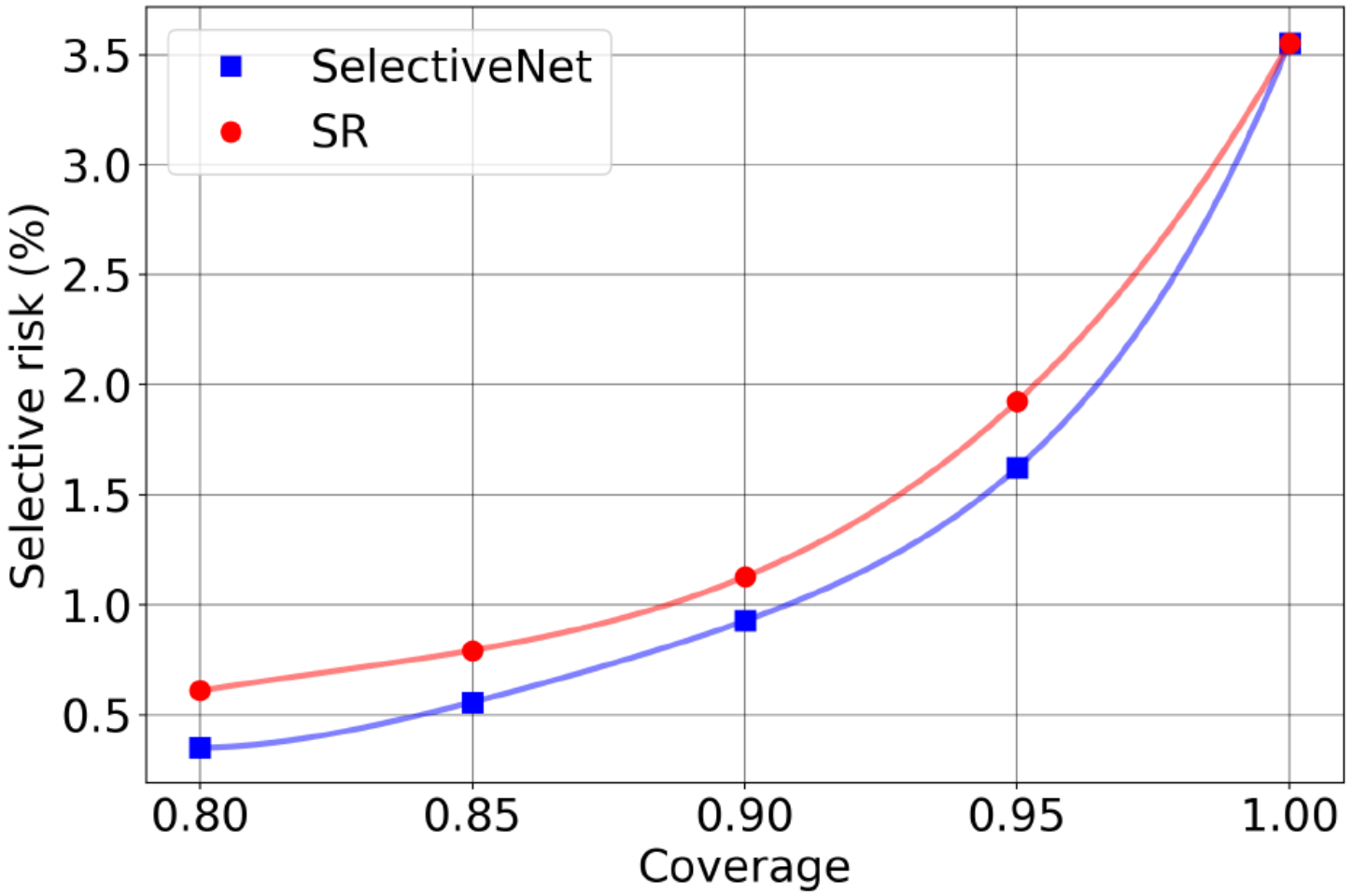
Results: CIFAR10

- "SR": $\hat{p}(\hat{y}|x_q)$ probability assigned to the predicted class label (aleatoric uncertainty)
- SelectiveNet gives a slight improvement

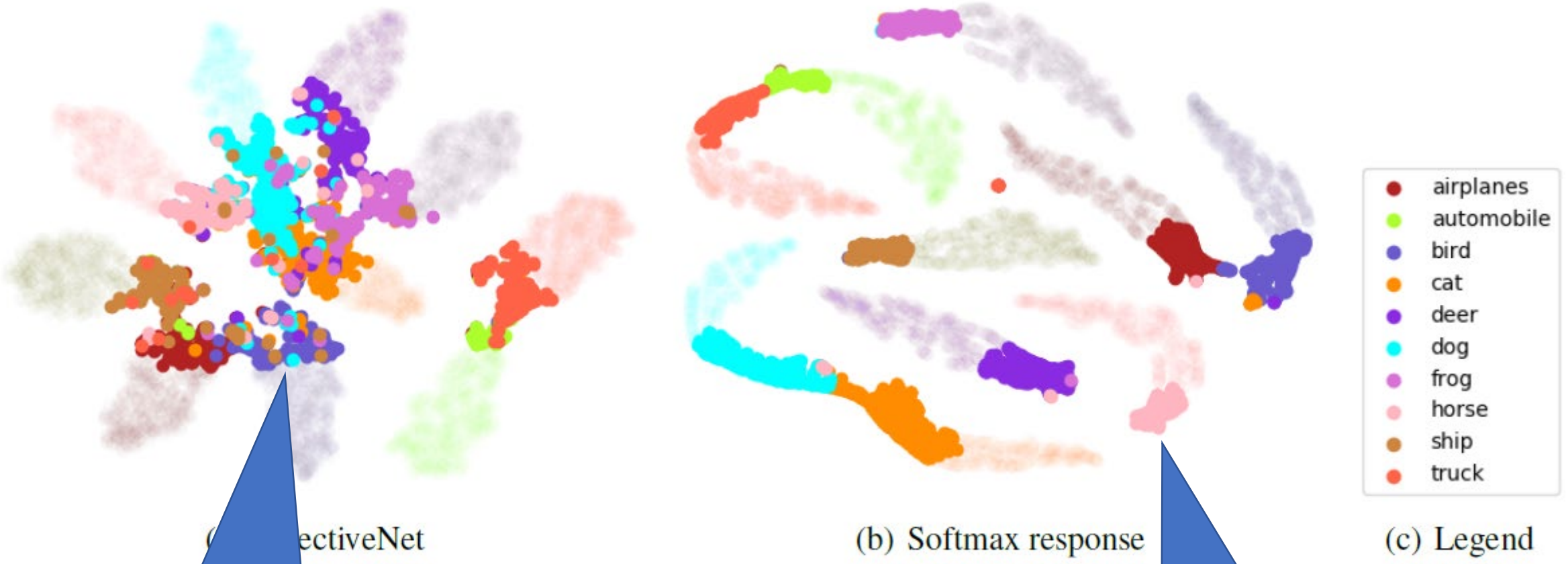


Results: Dogs vs Cats

- Same trend: SelectiveNet gives a small improvement



CIFAR-10 Visualization of Learned Representation



(a) ectiveNet (b) Softmax response (c) Legend

visualization of the embedding representation of (a) Se

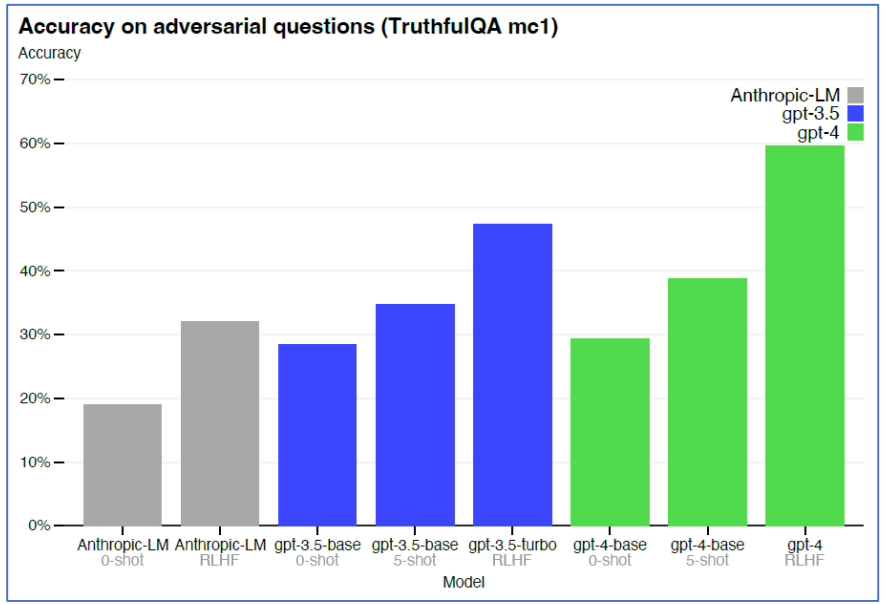
SelectiveNet does not try to discriminate among images that will be rejected

Softmax tries to separate all of the classes (of course)

Application: Improving Large Language Models

- Hallucinations

GPT-4 Hallucination Rate is 40% on adversarial questions

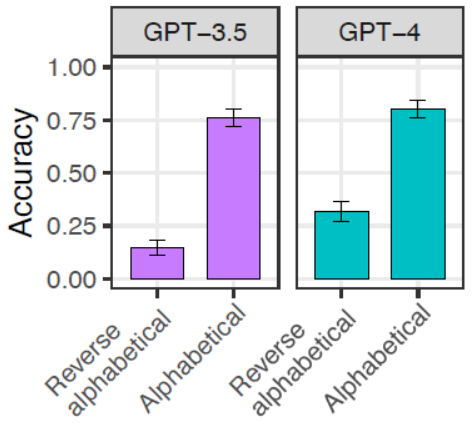


[Open AI (2023) GPT-4 Technical Report]

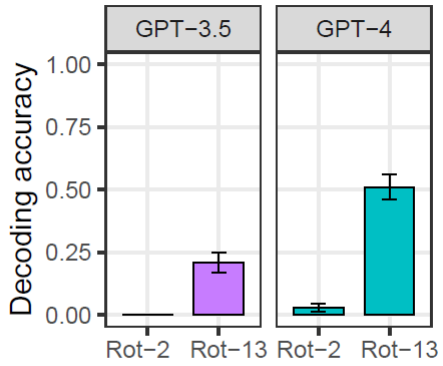
Sensitivity to Input and Output Probabilities

- LLMs perform much worse on rare tasks

Sorting



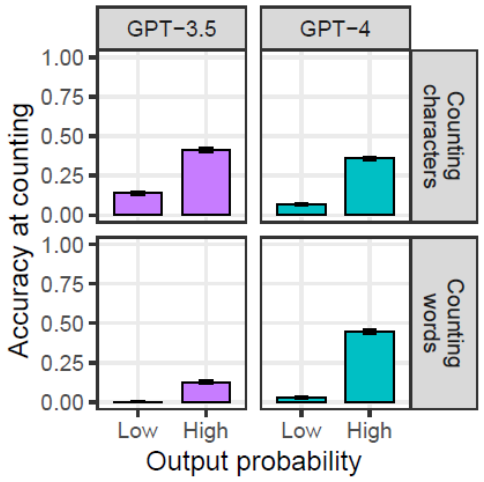
Rotation Ciphers

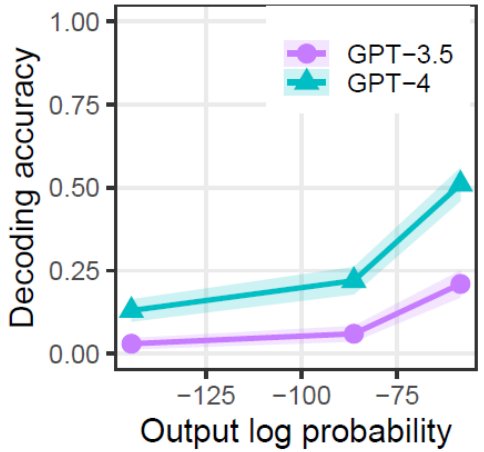


Note: In Internet text, rot-13 is about 60 times more common than rot-2.

- LLMs perform much worse on rare outputs
  - If the true answer is unusual, LLMs will substitute a higher probability answer instead
  - "auto-correcting the world"

Counting





McCoy, R. T., et al. (2023). Embers of Autoregression: Understanding Large Language Models Through the Problem They are Trained to Solve.

Hypotheses:

- H1: Queries with high epistemic uncertainty cause errors
  - LLM doesn’t know the right answer, so it makes something up
  - Solution: Estimate Epistemic Uncertainty and reject when high
  - Several new papers on arXiv in May and June
- H2: Queries with high aleatoric uncertainty cause errors
  - There are multiple valid yet conflicting answers, the model chooses “the wrong one” of them at random
  - Solution: Estimate Aleatoric Uncertainty and reject
  - More mature: LM-Polygraph study (Vashurin, et al., 2024)

Some Measures of Aleatoric Uncertainty of LLMs

- Suppose x is the prompt and $s = (z_1, \ldots, z_K)$ is the output sequence of tokens

$$P(x|s) = \prod_{j=1}^{K} P(z_i|z_{<i}, x)$$

- We usually operate in log space:

$$\log P(x|s) = \sum_{j=1}^{K} \log P(z_i|z_{<i}, x)$$

- We can convert this to an uncertainty measure by subtracting from 1:

$$MSP(x|s) = 1 - \log P(x|s)$$

- We can negate this to obtain the "surprisal". Large values ➔ large aleatoric uncertainty

$$-\log P(x|s) = \sum_{j=1}^{K} -\log P(z_i|z_{<i}, x)$$

- Length normalization gives us the "perplexity":

$$\text{perplexity}(x|s) = \frac{1}{K}\sum_{j=1}^{K} -\log P(z_i|z_{<i}, x)$$

- Mean token entropy:

$$MTE(x|s) = \frac{1}{K}\sum_{j=1}^{K} -P(z_i|z_{<i}, x) \log P(z_i|z_{<i}x)$$

Aleatoric Uncertainty from Generative Variety

- Given a prompt x ask the LLM to generate K answers:

$$S = \{s_1, \dots, s_K\}$$

- Measure the diversity of these answers

- Method 1: LexSim (Fomicheva, et al., 2020) the average pairwise similarity $\text{sim}(s_i, s_j)$ for $i \neq j$ computed using the Rouge-L metric (based on longest common word subsequences)

- Method 2: Semantic Entropy (Kuhn, et al., 2023; Farquhar, et al., 2024) Apply Natural Language Inference (NLI) to cluster the answers
  - $NLI(s_i|s_j)$ indicates whether the answer in s_i can be inferred from s_j
  - Put two sentences into the same cluster, if they can be inferred from each other. Let C be the set of clusters, and each $c \in C$ be the set of sentences in cluster c

$$SE(x) = -\sum_{c \in C} P(c|x) \log P(c|x) = -\sum_{c \in C} \left[\sum_{s \in c} P(s|x)\right] \log \left[\sum_{s \in c} P(s|x)\right]$$

Method 3: Shifting Attention to Relevance (SAR)

[Duan, et al., 2023]

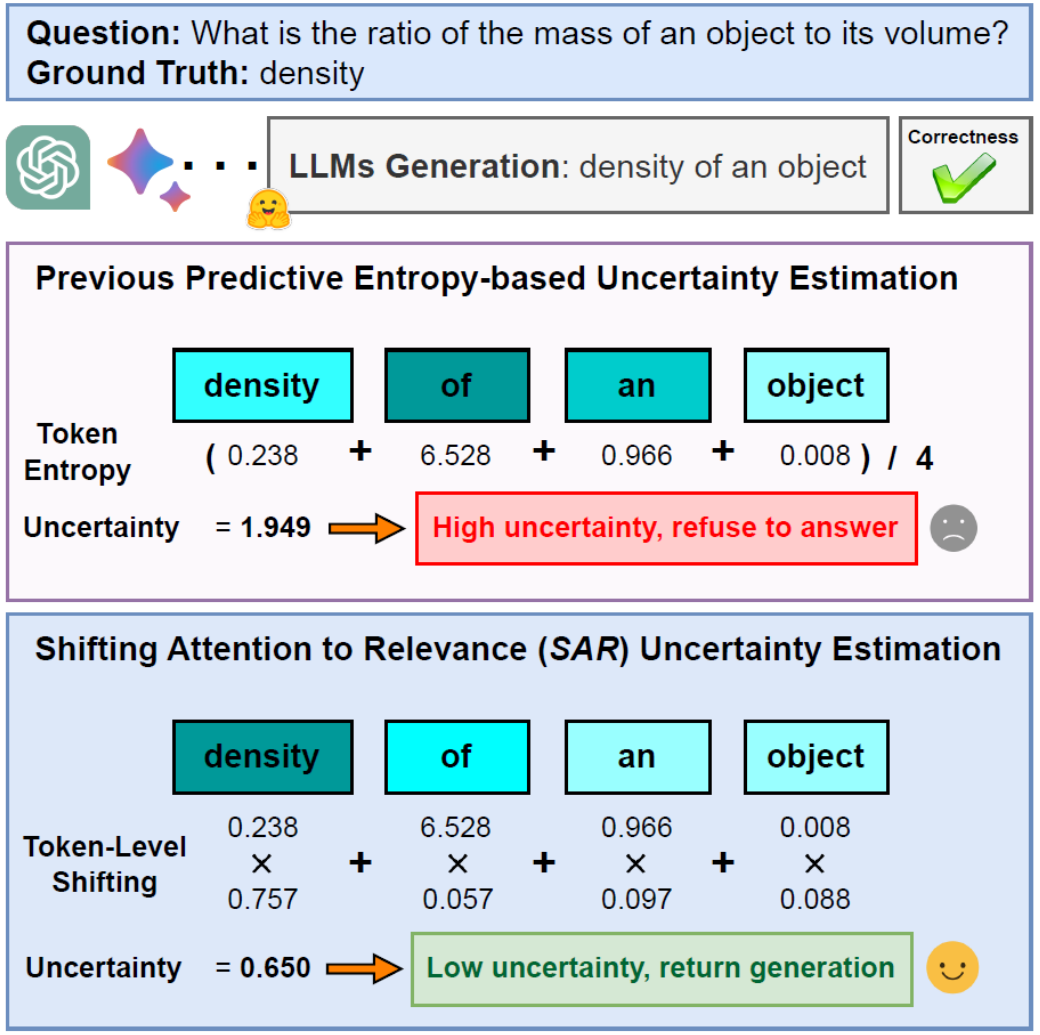
- Relevance of token z_t

$$R_T(z_t, s, x) = 1 - \text{sim}(x \cup s, x \cup s \setminus \{z_t\})$$

- Normalized relevance
- $\tilde{R}_T(z_t, s, x) = \frac{R_T(z_t, s, x)}{\sum_{t'} R_T(z_{t'}, s, x)}$
- TokenSAR

$$\text{TokenSAR}(s|x) = \sum_{t=1}^{T} -\tilde{R}_T(z_t, s, x) \log P(z_t | z_{<t}, s, x)$$

- Idea: Down-weight tokens that have low semantic "relevance"



Sentence SAR

$$S = \{s_1, \ldots, s_K\}$$

- Relevance of sentence s_j is the probability of the other sentences, weighted by their similarity to s_j

$$E_S(s_j, S, x) = -\log\left[P(s_j|x) + \lambda \sum_{j' \neq j} \text{sim}(s_j, s_{j'}) P(s_j|x)\right]$$

- Intuitions:
  - If s_j is semantically different from all other $s_{j'} \in S$, then $E_S(s_j, S, x) = -\log P(s_j|x)$
  - If all $s_j \in S$ are semantically identical, then $E_S(s_j, S, x) = -\log KP(s_j|x)$

$$SENTSAR(S, x) = \frac{1}{K} \sum_j E_S(s_j, S, x)$$

Combined SAR

Let $P'(s|x) = \exp[-\text{Token}SAR(s|x)]$

$$E_{S,T}(s_j, S, x) = -\log\left[P'(s_j|x) + \lambda \sum_{j' \neq j} \text{sim}(s_j, s_{j'})P'(s_j|x)\right]$$

$$SAR(S|x) = \frac{1}{K}\sum_{j=1}^{K} E_{S,T}(s_j, S, x)$$

Method 4: Degree Matrix NLI Score $Ddeg$

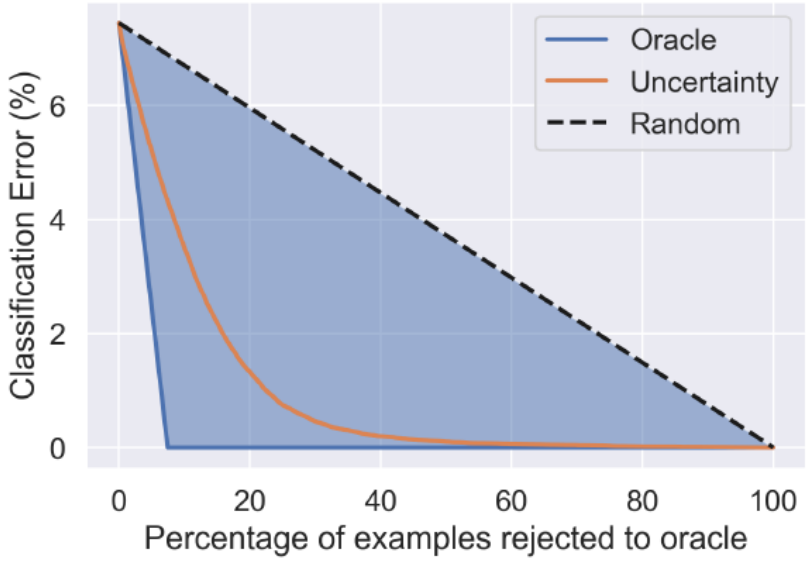
- Define $\mathrm{sim}(s_i, s_j) = \frac{1}{2}\left[NLI(s_i \to s_j) + NLI(s_j \to s_i)\right]$
  - Measures the extent to which s_j follows from s_i and vice versa
- Let $Cdeg(s_i, S, x) = \frac{1}{K}\sum_j \mathrm{sim}(s_i, s_j)$

LM-Polygraph Comparison Study
[Vashurin, et al., 2024]

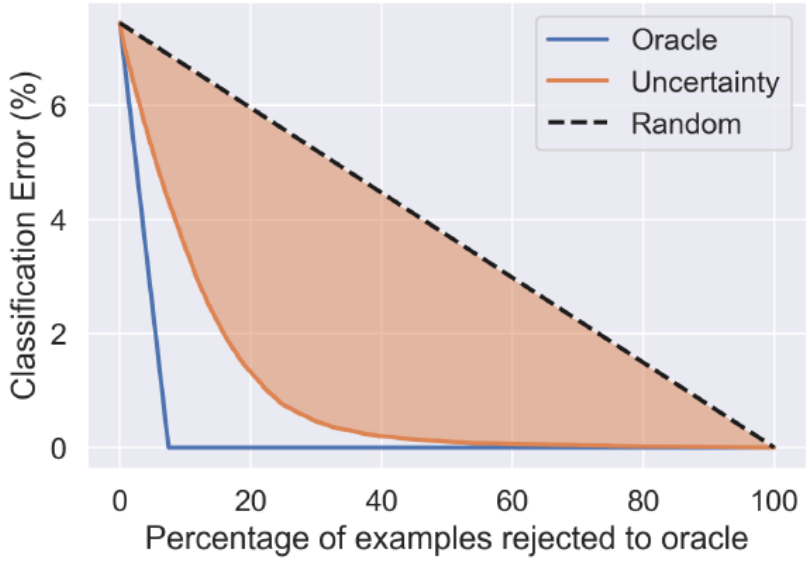
- Evaluation Benchmarks
  - CoQA: free-form answers about conversations
  - TriviaQA: * complex and compositional questions (no context)
  - MMLU * (multiple choice QA)
  - GSM8K: * grade school math word problems
  - WMT-14 French->English
  - WMT-19 German->English
- Prompting
  - CoQA: few shot prompt using all preceding questions for the conversation
  - * indicates 5-shot prompt
  - Translation tasks: zero-shot

Evaluation Metric: Prediction Rejection Ratio

[Malinin & Gales, ICLR 2020]



(a) Shaded area is $AR_{\mathtt{orc}}$.

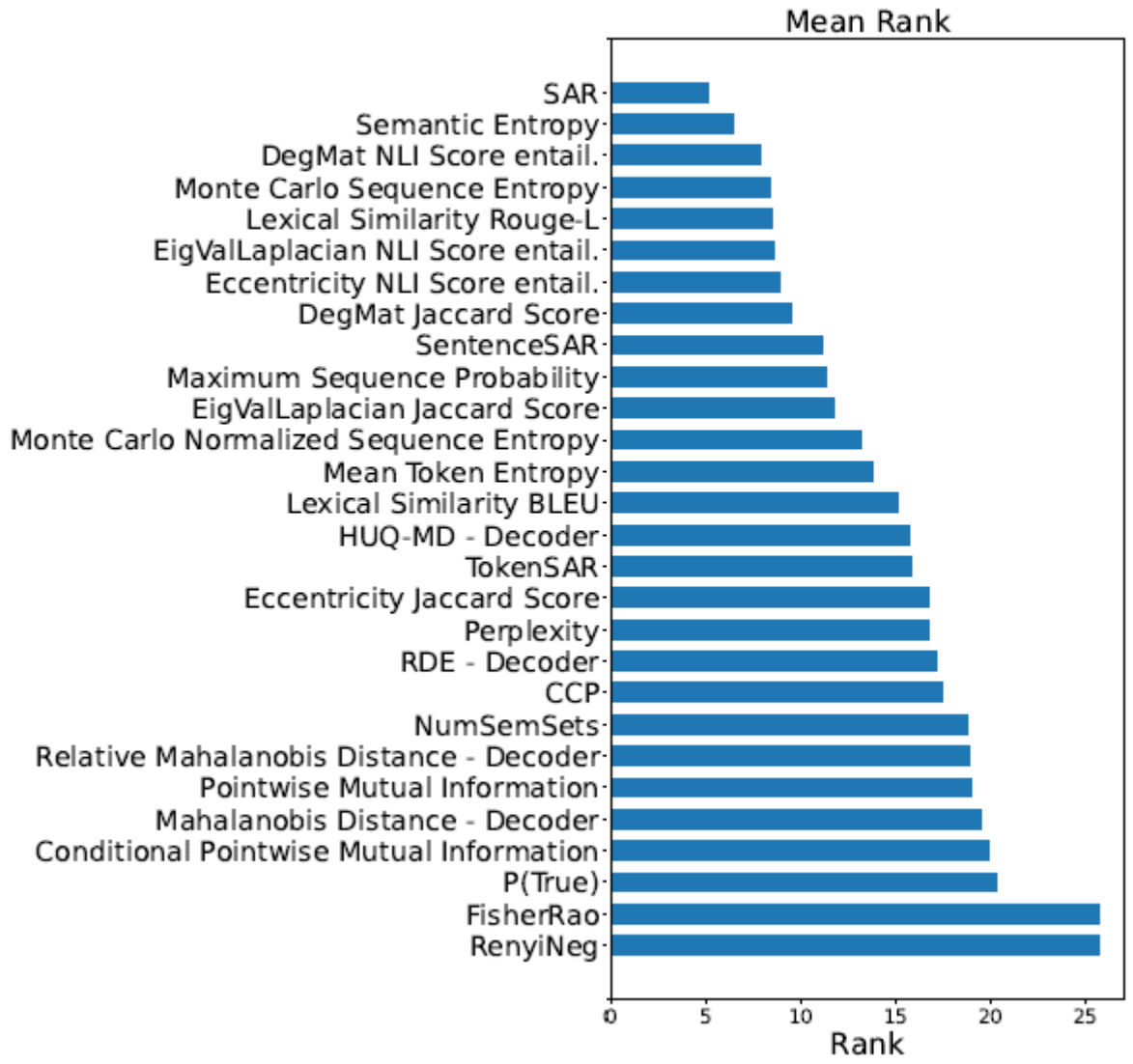


(b) Shaded area is $AR_{\mathtt{uns}}$.

- Blue curve ("oracle"): sort the test examples with all misclassified examples before all correctly-classified examples. This curve rejects all of the mistakes before rejecting the correctly-classified examples.
- Orange curve ("uncertainty"): sort the test examples in increasing order of estimated uncertainty.
- $PRR = \frac{Area(orange)}{Area\ (blue)}$
- Measures how close the system comes to the oracle

LM-Polygraph Results

- SAR
- Semantic entropy
- DegMat NLI Score entail
  - $Cdeg$
- Monte Carlo Sequence Entropy
  - Same as Perplexity
- Lexical Similarity Rouge-L
  - LexSim



Assessment

- The evaluations do not tell us how effective the methods will be in practice
- Better metric:
  - Rejection rate @90% accuracy
  - Rejection rate @95% accuracy
  - Rejection rate @99% accuracy
  - Rejection rate @100% accuracy
- What is achievable in practice?

Summary

- Goals of UQ:
  - Selective prediction, active learning, system integration
- Decomposing uncertainty into epistemic and aleatoric
- Sources of uncertainty: Data uncertainty, Model uncertainty
- Regression
  - Prediction intervals: linear regression, Bayesian Gaussian Process regression, Conformalized Quantile Regression
- Classification
  - Calibrating $\hat{p}(y = \hat{y}|x_q)$
  - Full calibration
  - Constructing prediction sets from calibrated probabilities
- Local epistemic uncertainty via outlier detection
  - Measure distance from x_q to the training data
  - Requires a good representation; this is a challenge for deep learning
- Applications
  - Active Learning
  - Uncertainty-Aware Learning
  - Selective prediction
  - Reducing Hallucination in LLMs

References

- Balabanov, O., & Linander, H. (2024). Uncertainty quantification in fine-tuned LLMs using LoRA ensembles. *ArXiv*, *2402.12264*(v1).
- Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140.
- Breunig, M. M., Kriegel, H., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. *ACM SIGMOD 2000 International Conference on Management of Data*, 1–12.
- Dietterich, T. G., & Guyer, A. (2022). The Familiarity Hypothesis: Explaining the Behavior of Deep Open Set Methods. *ArXiv*, *2203.02486*(v1). http://arxiv.org/abs/2203.02486
- Duan, J., Cheng, H., Wang, S., Zavalny, A., Wang, C., Xu, R., Kailkhura, B., & Xu, K. (2023). Shifting Attention to Relevance: Towards the Uncertainty Estimation of Large Language Models. ArXiv, 2307.01379(v2), 1–15.
- Emmott, A., Das, S., Dietterich, T., Fern, A., & Wong, W.-K. (2016). A Meta-Analysis of the Anomaly Detection Problem. *ArXiv*, *1503.01158*(v2), 1–35.
- Farquhar, S., Kossen, J., Kuhn, L., & Gal, Y. (2024). Detecting hallucinations in large language models using semantic entropy. *Nature*, *625–630*(July 2023). https://doi.org/10.1038/s41586-024-07421-0

References (2)

- Fisch, A., Jaakkola, T., & Barzilay, R. (2022). Calibrated Selective Classification. *ArXiv*, *2208.12084*(v1), 1–23.
- Fomicheva, M., Sun, S., Yankovskaya, L., Guzm, F., Fishel, M., Aletras, N., Chaudhary, V., & Specia, L. (2020). Unsupervised Quality Estimation for Neural Machine Translation. Transactions of the Association for Computational Linguistics, 8, 539–555.
- Gal, Y., & Ghahramani, Z. (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. *Proceedings of The 33rd International Conference on Machine Learning*, *48*.
- Geifman, Y., & El-Yaniv, R. (2017). Selective Classification for Deep Neural Networks. *ArXiv*, 1–12.
- Geifman, Y., & El-Yaniv, R. (2019). SelectiveNet: A deep neural network with an integrated reject option. *36th International Conference on Machine Learning, ICML 2019*, *2019-June*, 3768–3776.
- Gruber, C., Schenk, P. O., Schierholz, M., Kreuter, F., & Kauermann, G. (2023). Sources of Uncertainty in Machine Learning - A Statisticians ' View. *ArXiv*, *2305.16703*(v1).
- Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. *ArXiv*, *1706.04599*(v1). http://arxiv.org/abs/1706.04599
- Kendall, A., & Gal, Y. (2017). What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? *31st Conference on Neural Information Processing Systems (NIPS 2017)*.

References (3)

- Kuhn, L., Gal, Y., & Farquhar, S. (2023). Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. *ICLR 2023*, 1–19.
- Kull, M., Silva Filho, T. M., & Flach, P. (2017). Beyond Sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration. *Electronic Journal of Statistics*, *11*(2), 5052–5080. https://doi.org/10.1214/17-EJS1338SI
- Kull, M., Perello-Nieto, M., Kängsepp, M., Filho, T. S., Song, H., & Flach, P. (2019). Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with Dirichlet calibration. *Advances in Neural Information Processing Systems (NeurIPS 2019)*, 14. http://arxiv.org/abs/1910.1265
- Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation Forest. *2008 Eighth IEEE International Conference on Data Mining*, 413–422. https://doi.org/10.1109/ICDM.2008.17
- Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2012). Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data*, *6*(1), 1–39. https://doi.org/10.1145/2133360.2133363
- Malinin, A., Mlodozeniec, B., & Gales, M. (2020). Ensemble Distribution Distillation. *ICLR 2020*, 1–22.
- McCoy, R. T., Yao, S., Friedman, D., Hardy, M., & Griffiths, T. L. (2023). Embers of Autoregression: Understanding Large Language Models Through the Problem They are Trained to Solve. *ArXiv*, *2309.13638*(v1). http://arxiv.org/abs/2309.13638
- Meinshausen, N. (2006). Quantile regression forests. *Journal of Machine Learning Research*, *7*, 983–999.

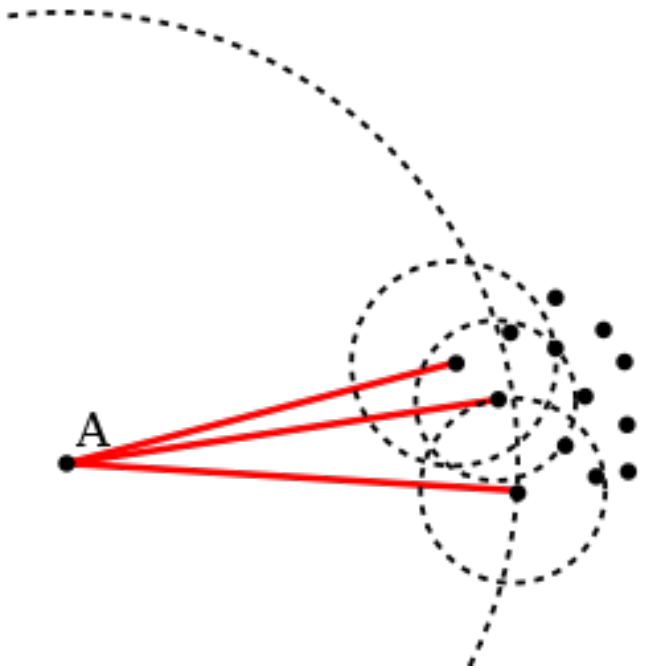
References (4)

- Niculescu-Mizil, A., & Caruana, R. (2005). Obtaining calibrated probabilities from boosting. *Proceedings of the Proceedings of the Twenty-First Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, 413–420.
- Niculescu-Mizil, A., & Caruana, R. (2005). Predicting good probabilities with supervised learning. *Proceedings of the 22nd International Conference on Machine Learning - ICML '05*, 625–632. https://doi.org/10.1145/1102351.1102430
- OpenAI (2023). GPT-4 Technical Report.
- Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. J. Smola, P. Bartlett, B. Schoelkopf, & D. Schuurmans (Eds.), *Advances in Large Margin Classifiers* (pp. 61-74). MIT Press.
- Vashurin, R., Fadeeva, E., Vazhentsev, A., Tsvigun, A., Vasilev, D., Xing, R., Sadallah, A. B., Rvanova, L., Petrakov, S., Panchenko, A., Baldwin, T., Nakov, P., Panov, M., & Shelmanov, A. (2024). Benchmarking Uncertainty Quantification Methods for Large Language Models with LM-Polygraph. *ArXiv*, *2406.15627*(v1). http://arxiv.org/abs/2406.15627
- Vovk, V., Gammerman, A., & Shafer, G. (2005). *Algorithmic Learning in a Random World*. Springer.
- Zadrozny, B., & Elkan, C. (2001). Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. *International Conference on Machine Learning (ICML-2001)*, *1*, 609–616.

Backup Slides

LOF: Local Outlier Factor
(Breunig, et al., 2000)

- Distance from x to its k-th nearest neighbor divided by the average distance of each of those neighbors to their k-th nearest neighbors
- [The actual calculation is slightly more complex.]



Breunig, et al.,

Isolation Forest [Liu, Ting, Zhou, 2011]

- Approximates the L_1 distance between the x_q and the training data (Guha et al., 2016)
  - Does not require standardizing the features
- Construct a fully random binary tree
  - choose attribute j at random
  - choose splitting threshold θ_1 uniformly from $[\min(x_{.j}), \max(x_{.j})]$
  - until every data point is in its own leaf
  - let $d(x_i)$ be the depth of point x_i
- repeat 100 times
  - let $\bar{d}(x_i)$ be the average depth of x_i
  - $score(x_i) = 2^{-\left(\frac{\bar{d}(x_i)}{r(x_i)}\right)}$
    - $r(x_i)$ is the expected depth