CENG7880
Trustworthy and Responsible AI

Instructor: Sinan Kalkan

(https://ceng.metu.edu.tr/~skalkan)

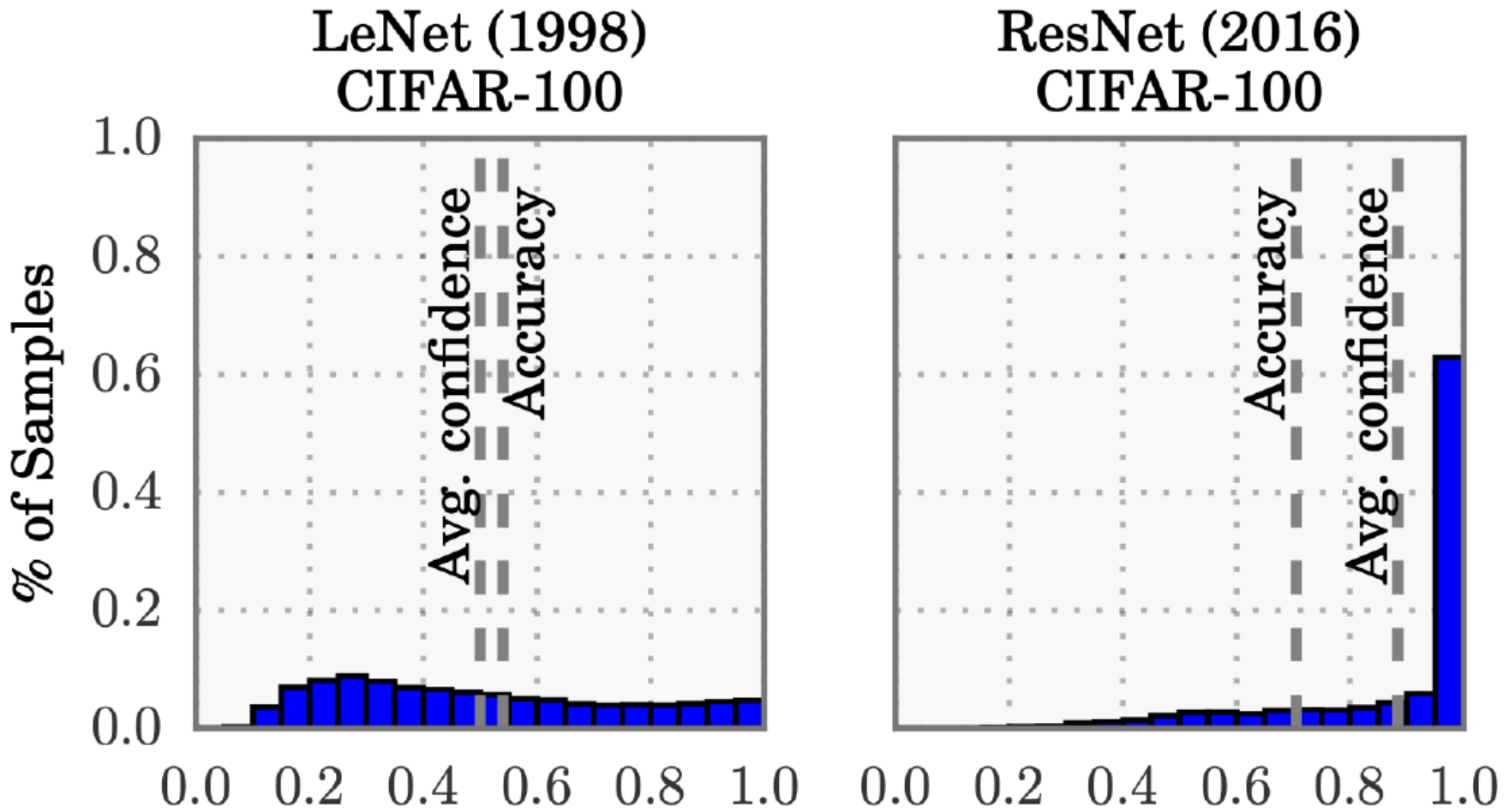
For course logistics and materials:
https://metu-trai.github.io

MIDDLE EAST TECHNICAL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Previously on CENG7880

Modern Neural Networks are Overconfident

ResNet-110

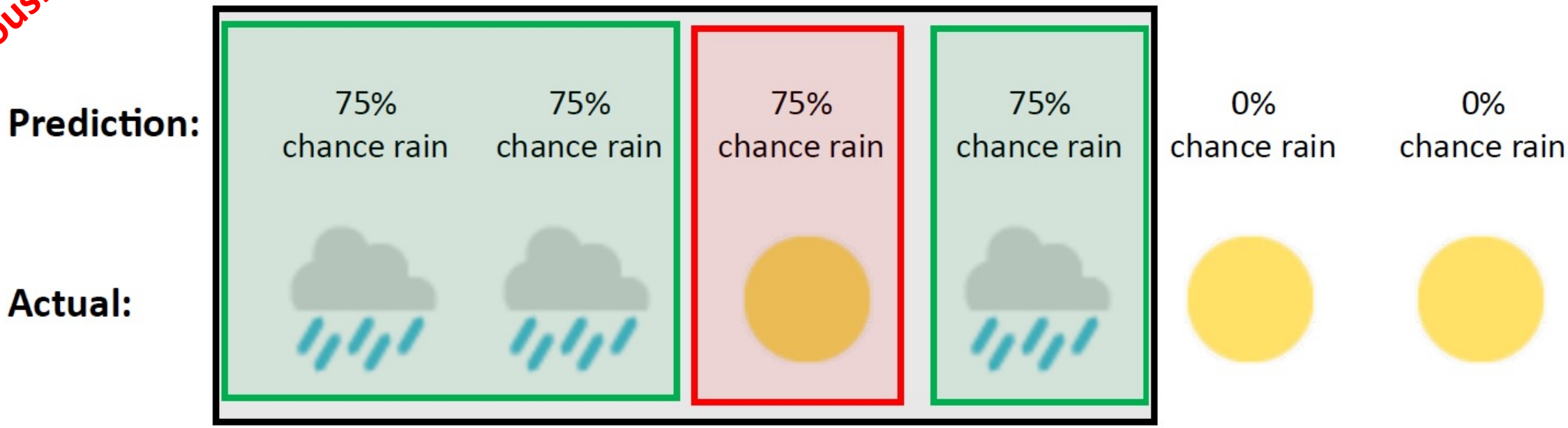


The plots are from this paper: https://arxiv.org/pdf/1706.04599

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Calibrated Prediction

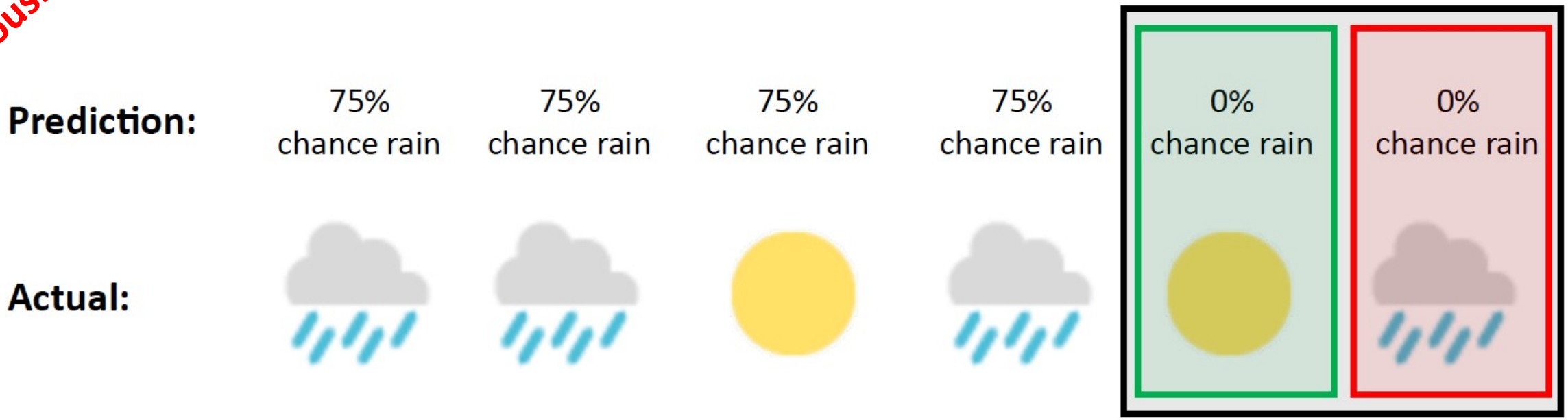


$$\Pr_{p(y^*)}[y^* = \text{rain} \mid \hat{p} = 0.75] = 0.75$$

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Calibrated Prediction



$$\Pr_{p(y^*)}[y^* = \text{rain} \mid \hat{p} = 0] = 0.5$$

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Calibration and Binning

- **Idea:** Bin probabilities into bins $P_i = \left[p_{\text{low},i}, p_{\text{high},i}\right)$ instead:

$$\Pr_{p(x,y^*)}[\hat{y}(x) = y^* \mid \hat{p}(x) \in P_i] = \text{Conf}(P_i) \qquad (\forall i \in \{1, \ldots, k\})$$

  - $\text{Conf}(P) = \mathbb{E}_{p(x,y^*)}[\hat{p}(x) \mid \hat{p}(x) \in P]$ is the average probability of bin P

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Measuring Calibration

- **Idea:** Use mean absolute error (called **expected calibration error**):

$$\mathrm{ECE}(\hat{p}) = \mathbb{E}_{p(P)}\left[\left|\Pr_{p(x,y^*)}\left[\hat{y}(x) = y^* \mid \hat{p}(x) \in P\right] - \mathrm{Conf}(P)\right|\right]$$

- On a held-out test set Z, we have the approximation

$$\mathrm{ECE}(\hat{p}; Z) = \sum_{i=1}^{k} \frac{|B_i|}{|Z|} \cdot \left| \frac{1}{|B_i|} \sum_{(x,y^*)\in B_i} 1(\hat{y}(x) = y^*) - \frac{1}{|B_i|} \sum_{(x,y^*)\in B_i} \hat{p}(x) \right|$$

  - Here, $B_i = \{(x, y^*) \in Z \mid \hat{p}(x) \in P_i\}$ is the bin in feature space

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

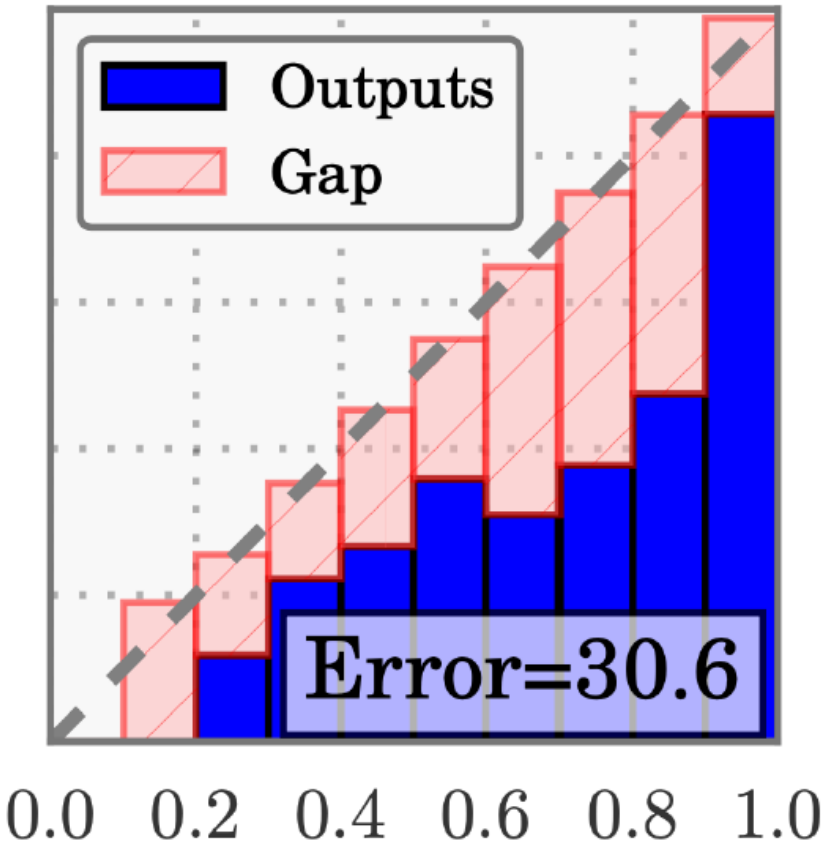
Previously on CENG7880

Reliability Diagrams

- For each bin P_i, plot accuracy

$$\Pr_{p(x,y^*)}[\hat{y}(x) = y^* \mid \hat{p}(x) \in P] \approx \frac{1}{|B_i|} \sum_{(x,y^*)\in B_i} 1(\hat{y}(x) = y^*)$$

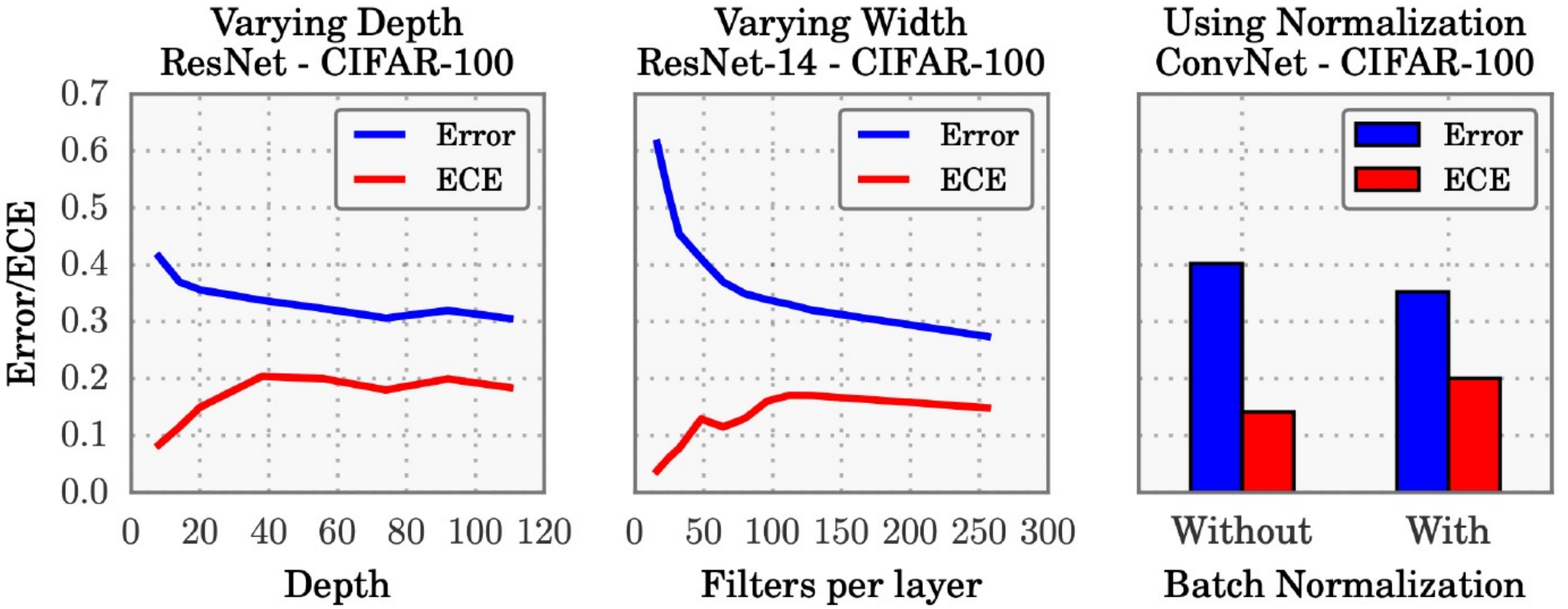
ResNet-110



The plots are from this paper: https://arxiv.org/pdf/1706.04599

Previously on CENG7880

Miscalibration and Overfitting

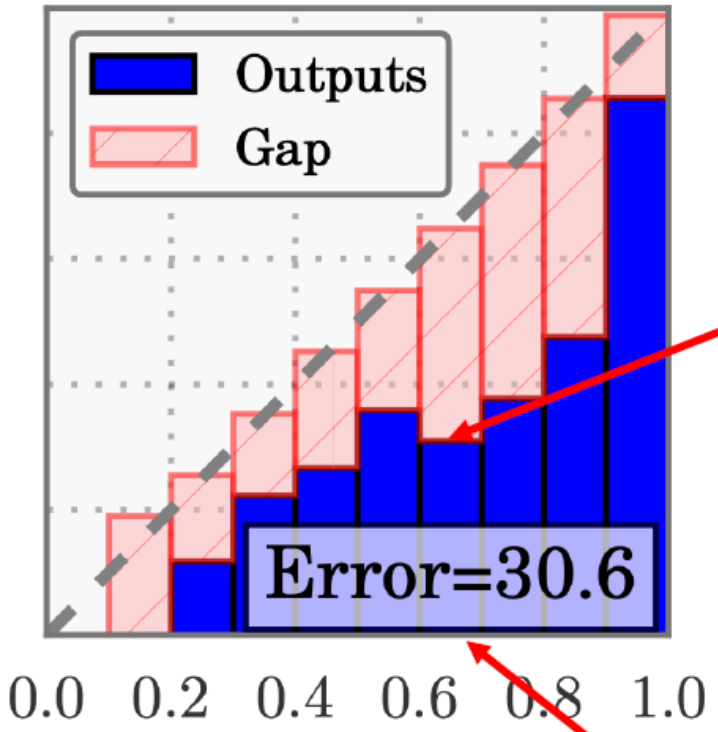


The plots are from this paper: https://arxiv.org/pdf/1706.04599

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Histogram Binning



Given x

$$\mathrm{Acc}(P) = \frac{1}{|B|} \sum_{(x',y^*)\in B} 1(\hat{y}(x') = y^*) = 0.35$$

$$\phi(x) = \mathrm{Acc}(P) = 0.35$$

$$\hat{p}(x) = 0.62 \in [0.6,0.7) := P$$

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Temperature Scaling

- Consider the model family

$$\vec{q}_\tau(x) = \text{softmax}\left(\frac{\text{logits}(x)}{\tau}\right)$$

  - Taking $\tau = 1$ recovers the original model: $\vec{q}_1(x) = \vec{p}(x)$
  - Taking $\tau > 1$ decreases confidence ($\tau \to \infty$ yields probabilities equal to $\frac{1}{k}$)
  - Taking $\tau < 1$ increases confidence ($\tau \to 0$ yields probabilities equal in $\{0,1\}$)

- Choose τ to minimize NLL of $\vec{q}_\tau$ on calibration dataset Z
  - Can use grid search to do so (i.e., search over fixed set of choices for τ)

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

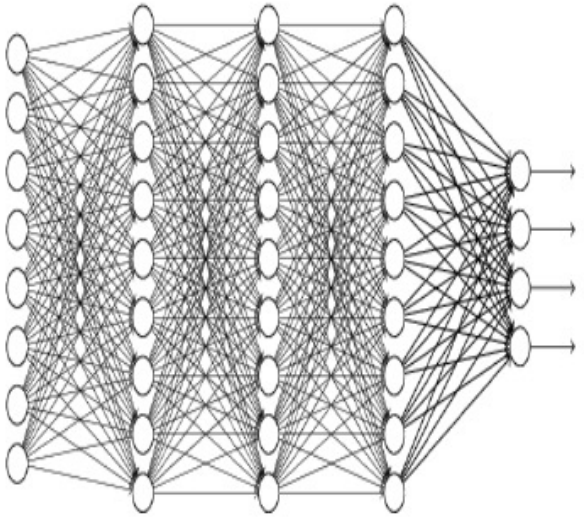
Previously on CENG7880

Conformal Prediction

Incorrect!

(Ground truth label: $y^* =$ "plunger")

Image x



DNN f

"toilet seat"

Prediction

$$\hat{y} = \max_y f(y \mid x)$$

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Conformal Prediction

Now, we have $y^* \in \tilde{f}(x)$ (**coverage**)

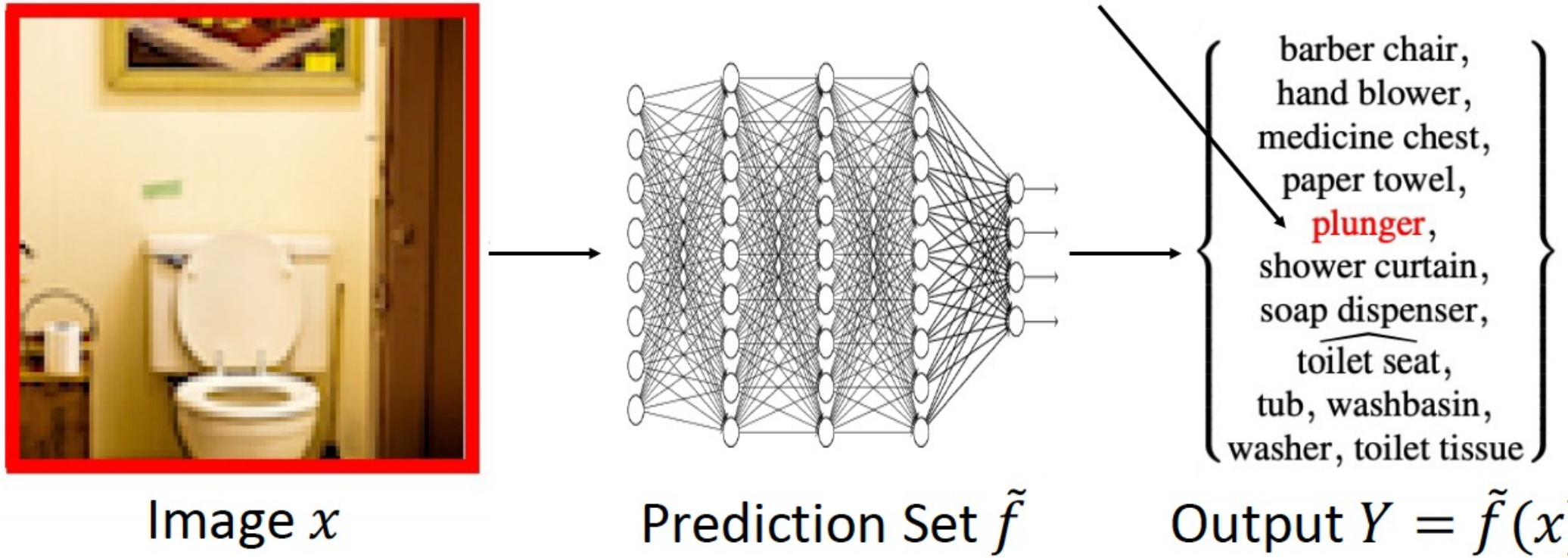


Image x | Prediction Set $\tilde{f}$ | Output $Y = \tilde{f}(x)$

Idea: Modify DNN f to predict **sets of labels**

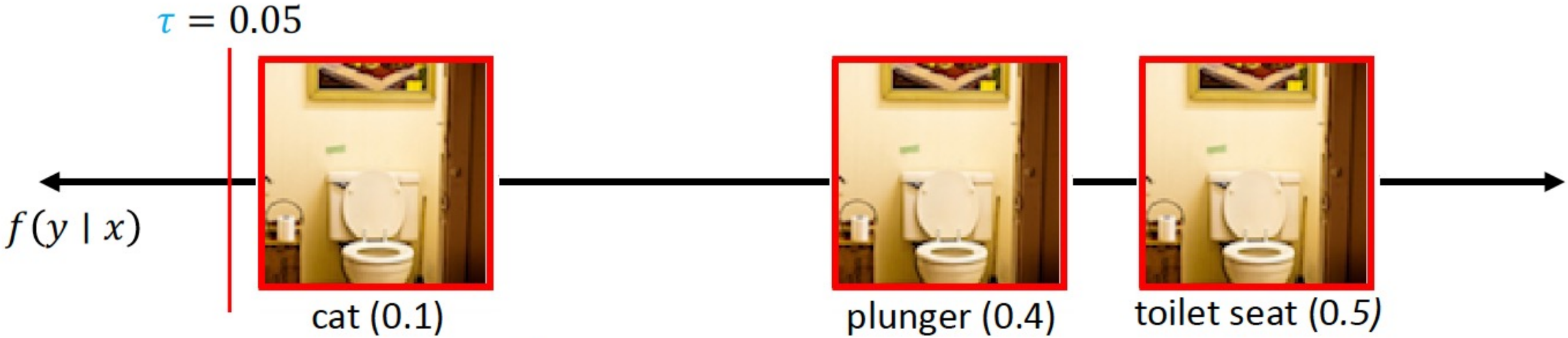
Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Conformal Prediction Problem

- **Parametric model family of prediction sets**
  - We construct prediction sets based on an **existing** DNN $f(y \mid x)$
  - Consider prediction sets that are **level sets** of f:

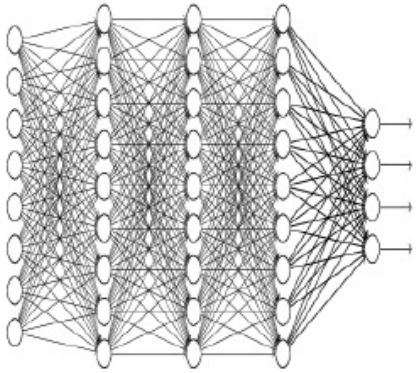
$$\tilde{f}_\tau(x) = \{y \mid f(y \mid x) \geq \tau\}$$



$$\tilde{f}_\tau(x) = \{\text{toilet seat}, \text{plunger}, \text{cat}\}$$

Previously on CENG7880

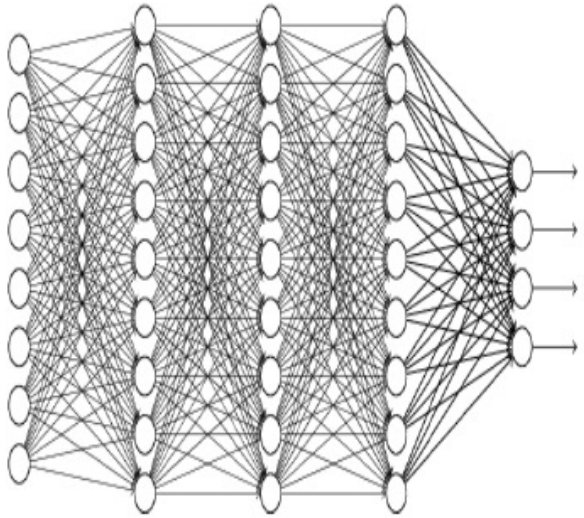
Conformal Prediction Problem



DNN f

Validation Dataset $Z_{\text{val}} = \{(x, y^*)\}$

Our Algorithm



Prediction Set $\tilde{f}$

and α: User-specified error rate threshold

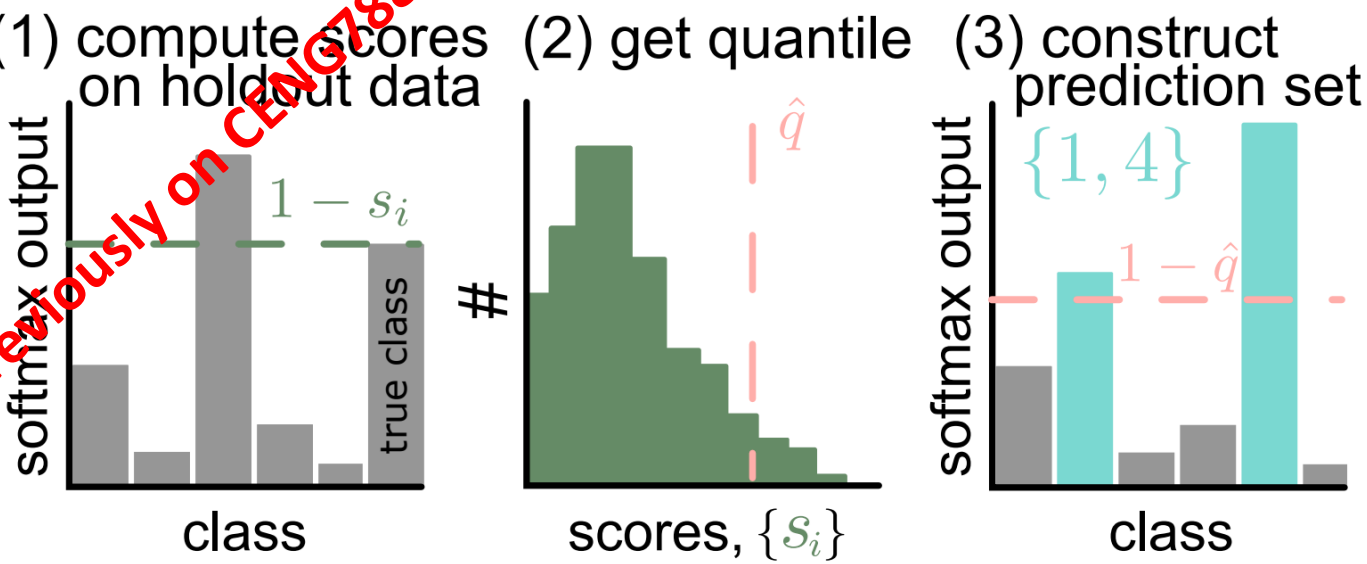
Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Conformal Prediction: Goal

- Given a predictor f, modify it to obtain a set predictor $\tilde{f}$ such that it predicts a set.
- In such a way that the correct label is within the set with $1-\alpha$ reliability/confidence:

$$P\left(Y_{test} \in \tilde{f}(X_{test})\right) \geq 1-\alpha$$




```
# 1: get conformal scores. n = calib_Y.shape[0]
cal_smx = model(calib_X).softmax(dim=1).numpy()
cal_scores = 1-cal_smx[np.arange(n),cal_labels]
# 2: get adjusted quantile
q_level = np.ceil((n+1)*(1-alpha))/n
qhat = np.quantile(cal_scores, q_level, method='higher')
val_smx = model(val_X).softmax(dim=1).numpy()
prediction_sets = val_smx >= (1-qhat) # 3: form prediction sets
```

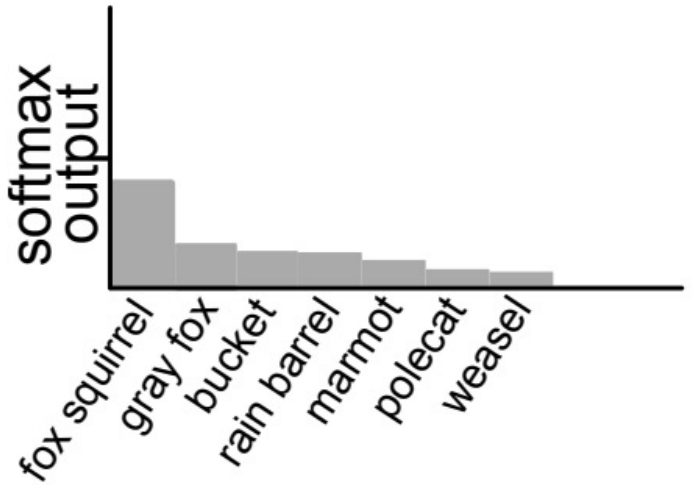
- We use a hold-out/val set since prediction probabilities on the training set are not reliable.
- Conformal score (s_i) = $1 - \hat{p}_i$
  - But we can devise other measures based on x and y
- Set threshold $\hat{q}$ as $\hat{q} = 1 - \alpha$ quantile of the scores
- Interpretation: if α = 0.1, 90% of GT softmax outputs are above 1- $\hat{q}$.

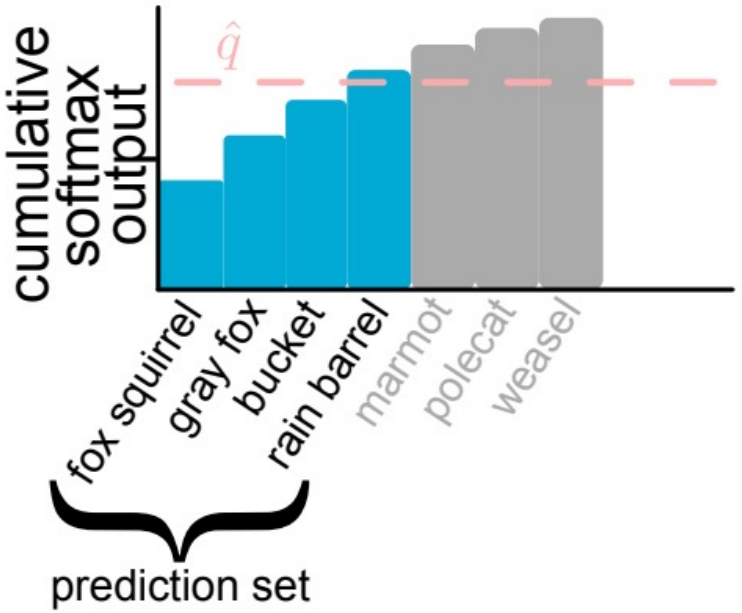
Material: https://arxiv.org/pdf/2107.07511

Previously on CENG7880

Adaptive Prediction Sets

- Problem: the current conformal predictor is not adaptive to the difficulty of the classification, providing “average” coverage.
- Goal: Conditional coverage.
- Adaptive conformal predictors approximate conditional coverage.
  - Approach: Use cumulative sum of probabilities in calculating non-conformity scores and determining the threshold.





Material: https://arxiv.org/pdf/2107.07511

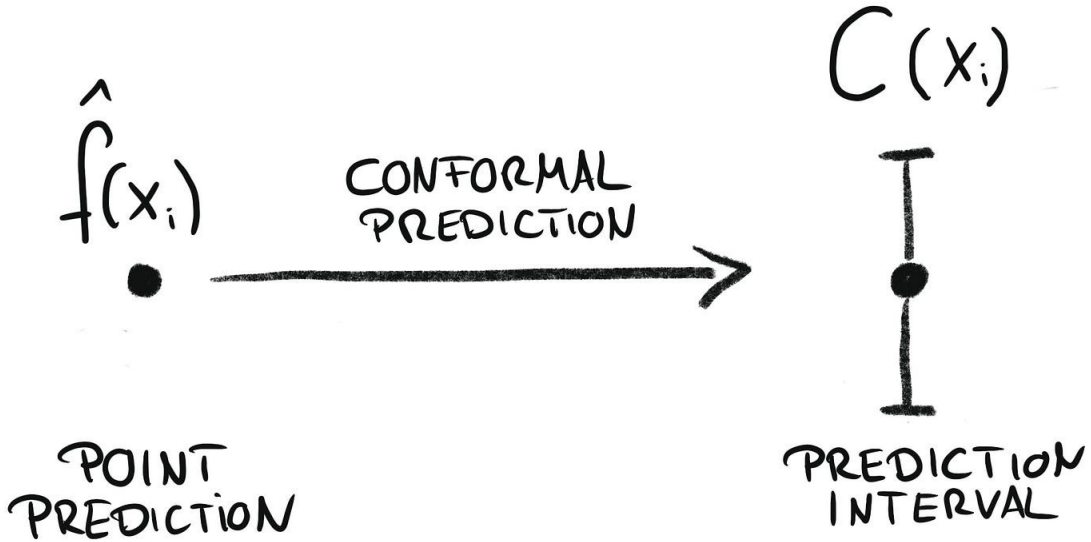
Previously on CENG7880

Conformal Prediction for Regression

Goal: Predict an interval around model's prediction where half-width is α.

- Split data into training and calibration
- Train model on training data
- Compute non-conformity scores
- Find threshold q
- Form prediction intervals for new data

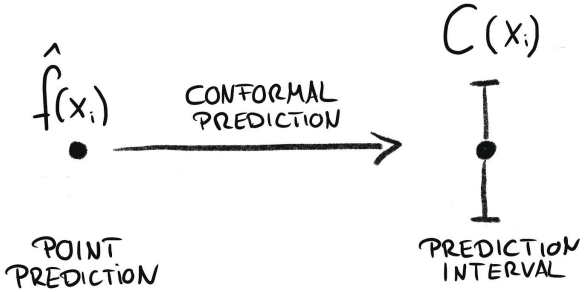
as $\left[\hat{f}(x_i) - q, \hat{f}(x_i) + q\right]$



https://mindfulmodeler.substack.com/p/week-3-conformal-prediction-for-regression

Conformal Prediction for Regression

Previously on CENG7880



The main difference between classification and regression is the non-conformity score that we use:

$$s(y, x) = |y - \hat{f}(x)|$$

This function, when using the true y, computes the absolute residual of the prediction. Conformalizing this score means finding where to cut off so that 1-α of the predictions have a score below and α a score above. In our rent index case, this value is q=1.97. This means that all intervals have a width of 2 * 1.97 = 3.94.

To compute the prediction interval for a new data point, we include all possible y's that produce a score below q.

https://mindfulmodeler.substack.com/p/week-3-conformal-prediction-for-regression

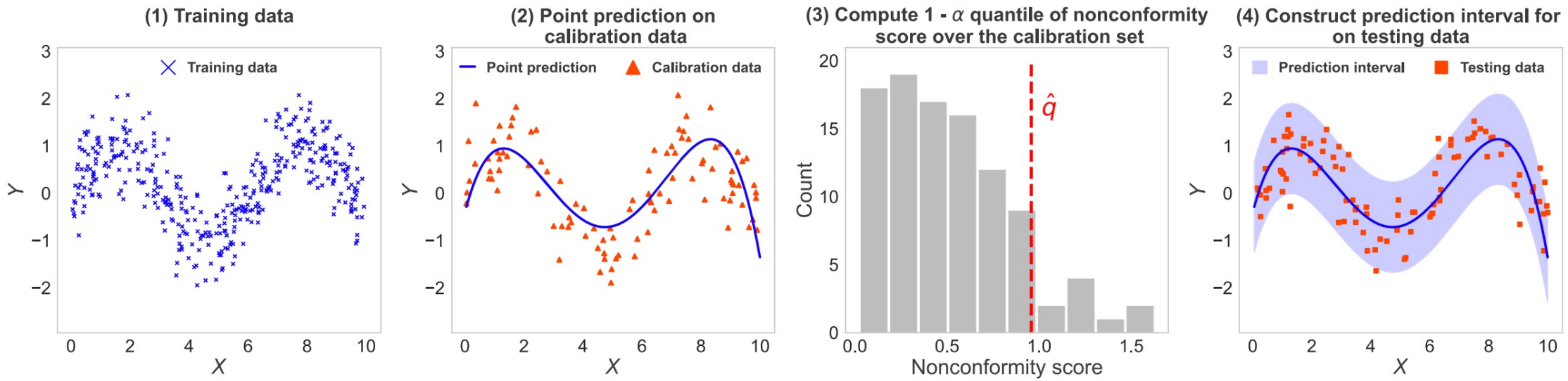


Figure 3. Demonstration of conformal prediction on top of a machine learning model fitted with polynomial regression over a 1D problem. (1) Training data: 400 samples are randomly generated from $Y = \sin(X) + \epsilon$, where $X \sim \mathcal{U}[0, 10]$ and $\epsilon \sim \mathcal{N}(0, 0.5^2)$. (2) Point prediction: Polynomial regression $f(x)$ with a degree of 4 is used to fit the 400 training points. The trained model is then used to make predictions on the 100 samples in the calibration set $\{(x_j, y_j)\}_{j=1}^{j=100}$. (3) Nonconformity score: we derive the nonconformity score $s(x, y) = |y - f(x)|$ for each sample in the calibration set. Given the miscoverage rate of $\alpha = 0.1$, we compute $1 - \alpha$ quantile of the nonconformity score over the calibration set denoted by $\hat{q}$. (4) Prediction interval construction: The point prediction $f(x)$ (blue line) for 100 randomly generated testing points are converted into valid prediction intervals (shaded light blue) $[f(x) - \widehat{q}, f(x) + \widehat{q}]$ by conformal prediction.

https://iopscience.iop.org/article/10.1088/3050-2454/adcef2

Previously on CENG7880

Types of Uncertainty

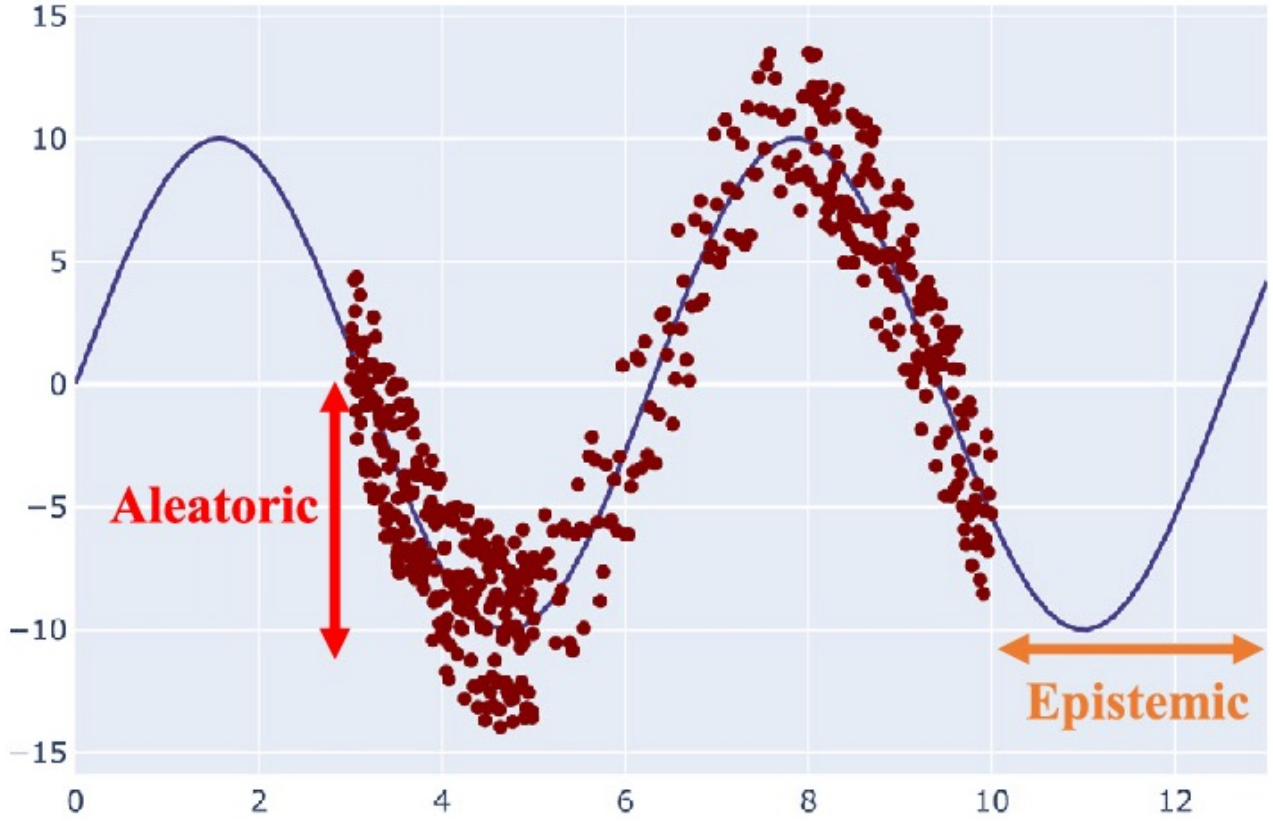


Fig: Abdar et al., A review of uncertainty quantification in deep learning: Techniques, applications and challenges, 2021

Sources of model (epistemic) uncertainty

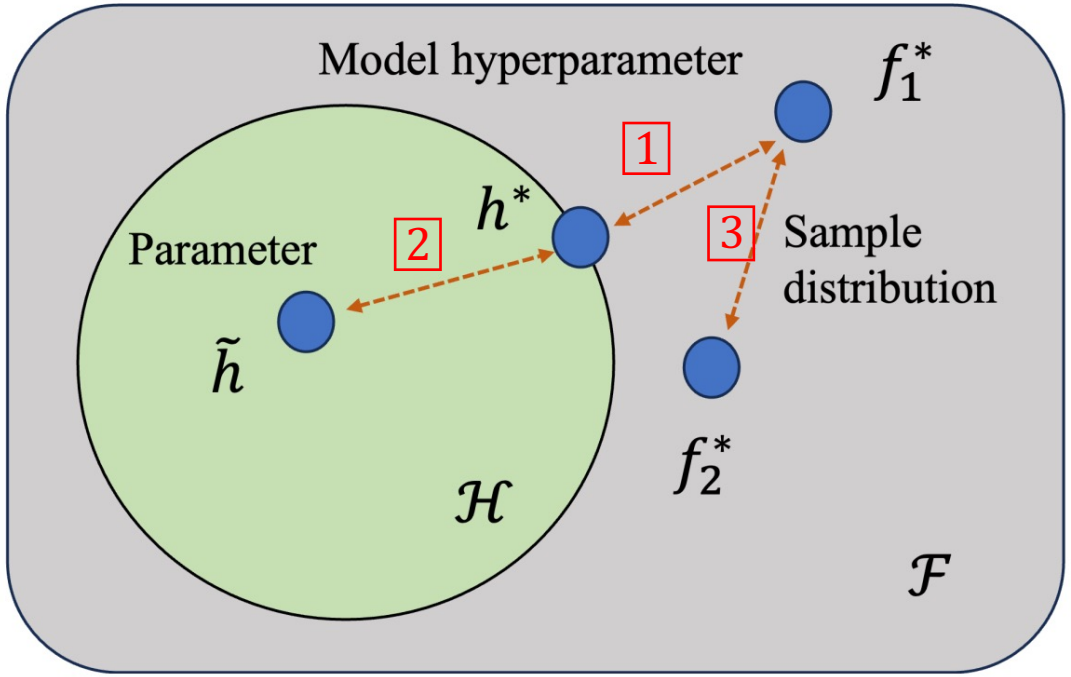
Previously on CENG7880

Table 1. Comparison of different types of model uncertainty in the supervised learning setting

	Model uncertainty sources	Corresponding notation in supervised learning
1	Choice of model hyper-parameter	Optimal solution h^* within $\mathcal{H}$ does not align with theoretical optimal f^* in $\mathcal{F}$
2	Model parameter learning	Learned solution $\tilde{h}$ does not align with optimal h^* in $\mathcal{H}$
3	Different sample distributions in learning and inference	Theoretical optimum f_1^* and f_2^* mismatch under different sample distribution $p(\boldsymbol{x}, y)$

Fig, table and content: A survey on uncertainty quantification methods for deep learning, 2023. arxiv.org/pdf/2302.13425

https://arxiv.org/pdf/2302.13425



- Due to different sample distributions between training and inference.
- A model trained on $p_1(\boldsymbol{x}, y)$ will make errors on samples from $p_2(\boldsymbol{x}, y)$.
  - Out of distribution samples.
  - Samples far from training set.

Previously on CENG7880

Sources of data (aleatoric) uncertainty

- Owing to inherent data randomness, noise or class/label confusion
- Irreducible even with more training data

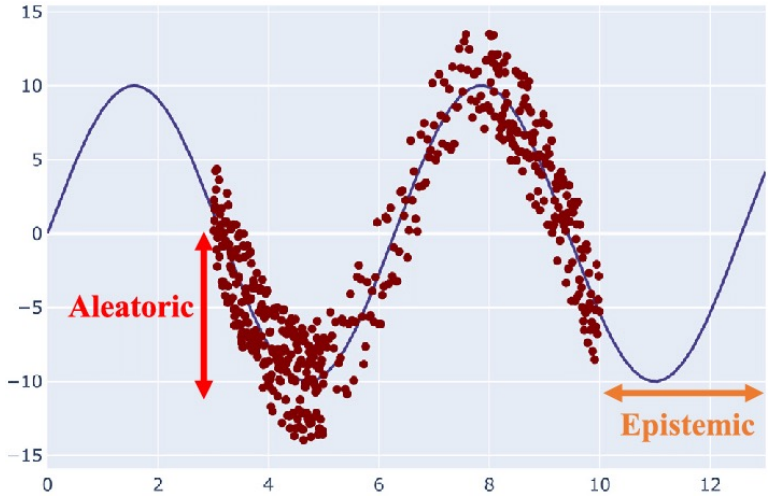
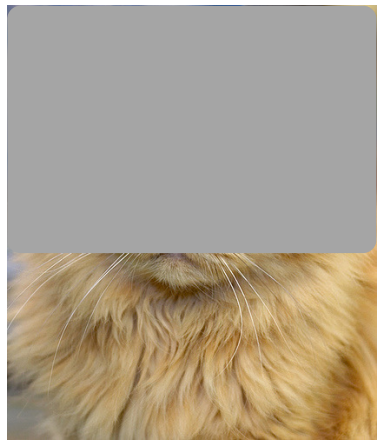


Fig: Abdar et al., A review of uncertainty quantification in deep learning: Techniques, applications and challenges, 2021



Sample from Imagenet

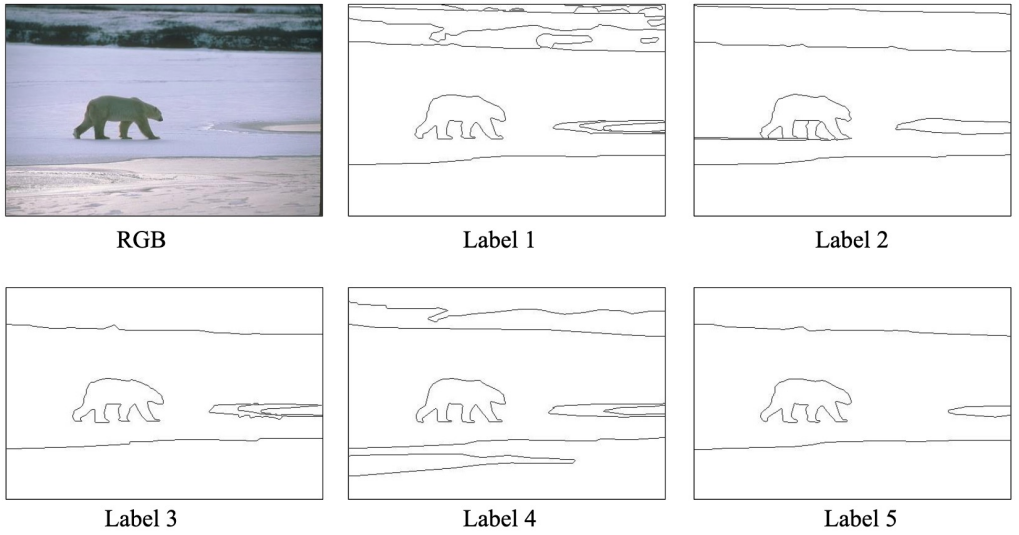


Figure 1.3: An example of the multi-label challenge in edge detection: a single input image and its five different ground truth edge maps annotated by five different human annotators. The image and labels are taken from the BSDS dataset [1].

From Bedrettin Cetinkaya's PhD Thesis

Previously on CENG7880

Sources of data (aleatoric) uncertainty

- Owing to inherent data randomness, noise or class/label confusion
- Irreducible even with more training data

Two types:

- Homoscedastic (homogeneous variances) noise: Constant observation noise for all samples.
- Heteroscedastic (heterogeneous variances) noise: Input dependent noise.

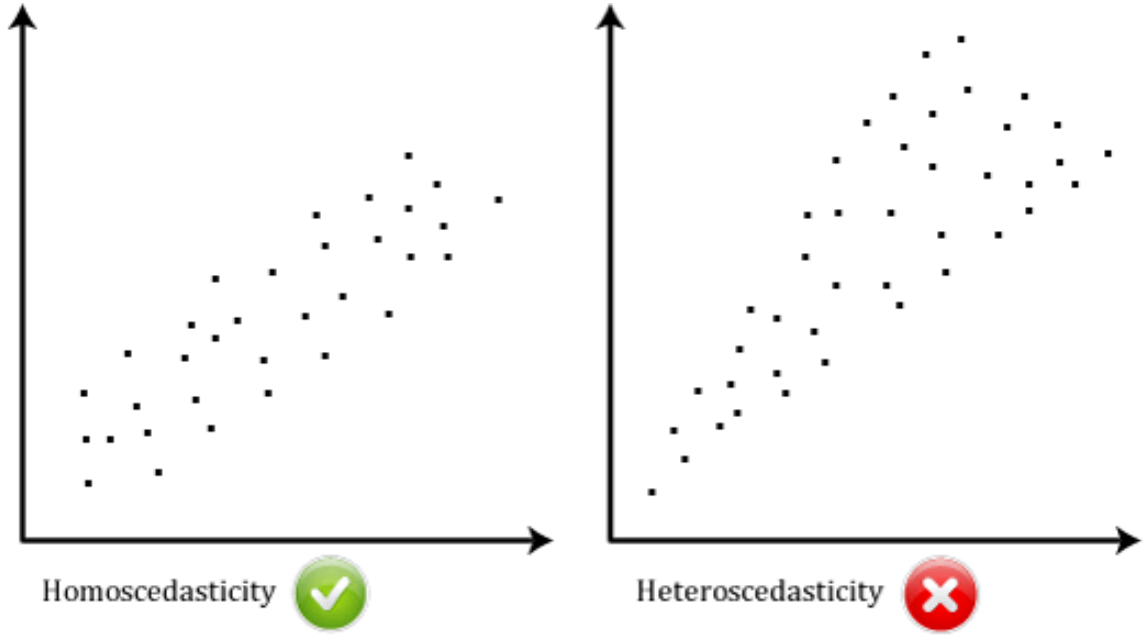


Fig: https://stats.stackexchange.com/questions/76151/what-is-an-intuitive-explanation-of-why-we-want-homoskedasticity-in-a-regression

Previously on CENG7880

Aleatoric vs. Epistemic Uncertainty

- In general, the **residual error** decomposes as

$$y - f_{\hat{\beta}(Z)}(x) = \underbrace{\left(y - f_{\beta^*}(x)\right)}_{\text{Aleatoric uncertainty}} + \underbrace{\left(f_{\beta^*}(x) - f_{\hat{\beta}(Z)}(x)\right)}_{\text{Epistemic uncertainty}}$$

  - **Aleatoric uncertainty:** Error of best possible model f_{β^*}
  - **Epistemic uncertainty:** Error of our model $f_{\hat{\beta}(Z)}$ vs. f_{β^*}

- How can we disentangle the two?

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Previously on CENG7880

Uncertainty Quantification in Deep/Machine Learning

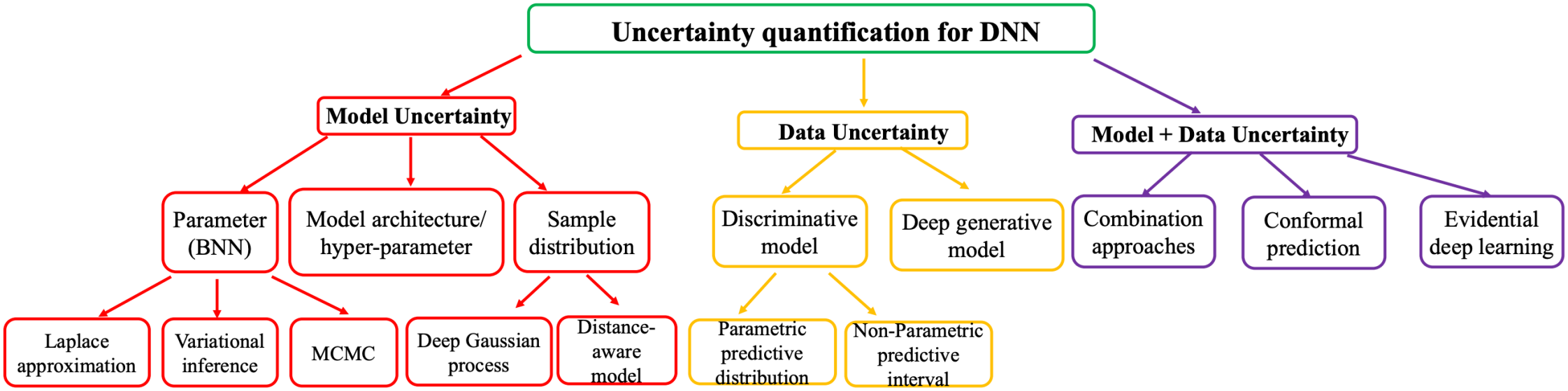


Fig. 5. A taxonomy for existing literature on UQ for DNN.

Fig, table and content: A survey on uncertainty quantification methods for deep learning, 2023. arxiv.org/pdf/2302.13425

Agenda

- (Predictive) Uncertainty Quantification
- Explainability
  - Feature Attribution Methods (LIME, SHAP)
  - Saliency Methods

Administrative Notes

- Final Exam:
  - 13 January 16:30
- Paper selection finalized except for two projects
- Project milestones
  - **1. Milestone (November 23, midnight):**
    - Read & understand the paper
    - Download the datasets
    - Prepare the Readme file excluding the results & conclusion
  - **2. Milestone (December 7, midnight)**
    - The results of the first experiment
  - **3. Milestone (January 4, midnight)**
    - Final report (Readme file)
    - Repo with all code & trained models

Aleatoric vs. Epistemic Uncertainty

- In general, we have

$$y - f_{\hat{\beta}}(x) = \left(y - f_{\beta^*}(x)\right) + \left(f_{\beta^*}(x) - f_{\hat{\beta}(Z)}(x)\right)$$

- **Hard to disentangle**
  - We directly observe the predictive uncertainty $y - f_{\hat{\beta}}(x)$
  - But we don't know β^*

- **General strategy (statistics):** Pretend $\hat{\beta} = \beta^*$, and disentangle
  - Works in practice even if it feels circular

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Aleatoric vs. Epistemic Uncertainty

- The **epistemic uncertainty** is

$$\text{Epistemic}(x) = f_{\beta^*}(x) - f_{\hat{\beta}(Z)}(x)$$

  - Here, $\text{Epistemic}(x)$ is a random function of the random variable $Z \sim p^n$
  - Thus, $\text{Epistemic}(x)$ is itself a random variable

- **Goal:** Estimate the distribution of $\text{Epistemic}(x)$

  But we don't know β^*

- **Assumption:** Our model is **unbiased**: $\mathbb{E}_Z\left[f_{\hat{\beta}(Z)}(x)\right] = f_{\beta^*}(x)$

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Typo here with the sign since the definition is the other way around: $\text{Epistemic}(x) = f_{\beta^*}(x) - f_{\hat{\beta}(Z)}(x)$

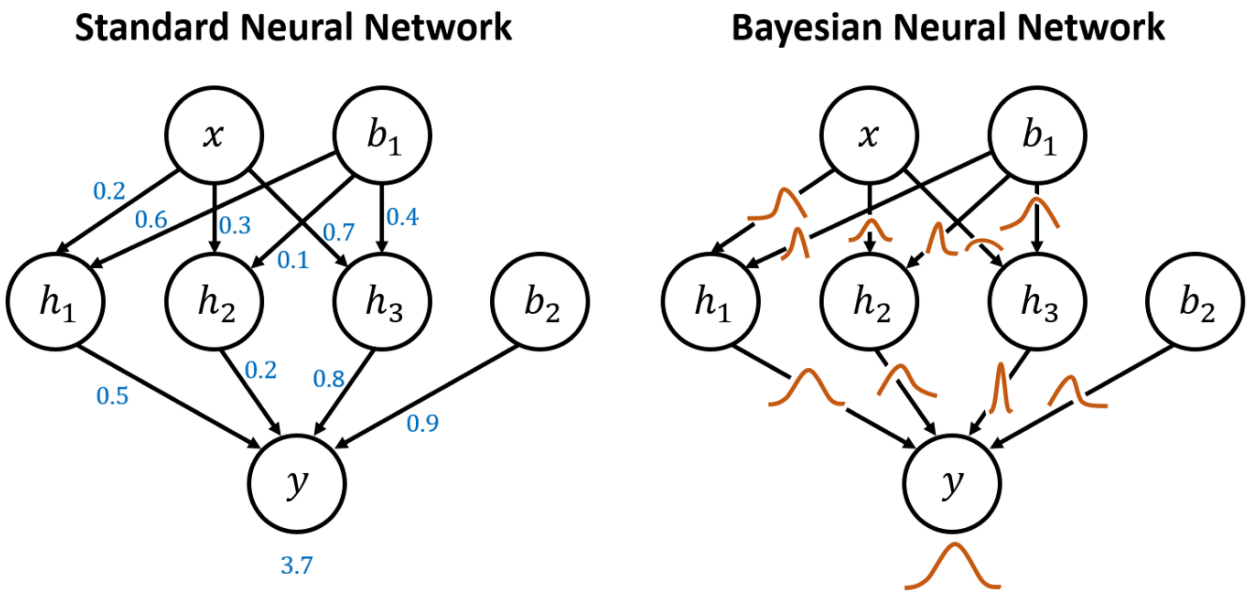
- $\{\hat{f}_i(x) - f_{\beta^*}(x)\}_{i=1}^{k}$ are i.i.d. samples from $\text{Epistemic}(x)$
- By our unbiasedness assumption:

$$f_{\beta^*}(x) = \mathbb{E}_Z\left[f_{\hat{\beta}(Z)}(x)\right] \approx k^{-1}\sum_{i=1}^{k}\hat{f}_i(x) := \hat{\mu}(x)$$

Same architecture trained on different subsets/versions of Z.

- $\{\hat{f}_i(x) - \hat{\mu}(x)\}_{i=1}^{k}$ are approximately i.i.d. samples from $\text{Epistemic}(x)$
  - **Problem:** We cannot take unlimited samples from P
  - Only have a single training dataset Z!

Method for Estimating Epistemic Uncertainty



- With a Bayesian Neural Network, we can sample different "models" from a distribution over the weights.

Fig: https://medium.com/@costaleirbag/a-first-insight-into-bayesian-neural-networks-bnn-c767551e9526

Method for Estimating Epistemic Uncertainty

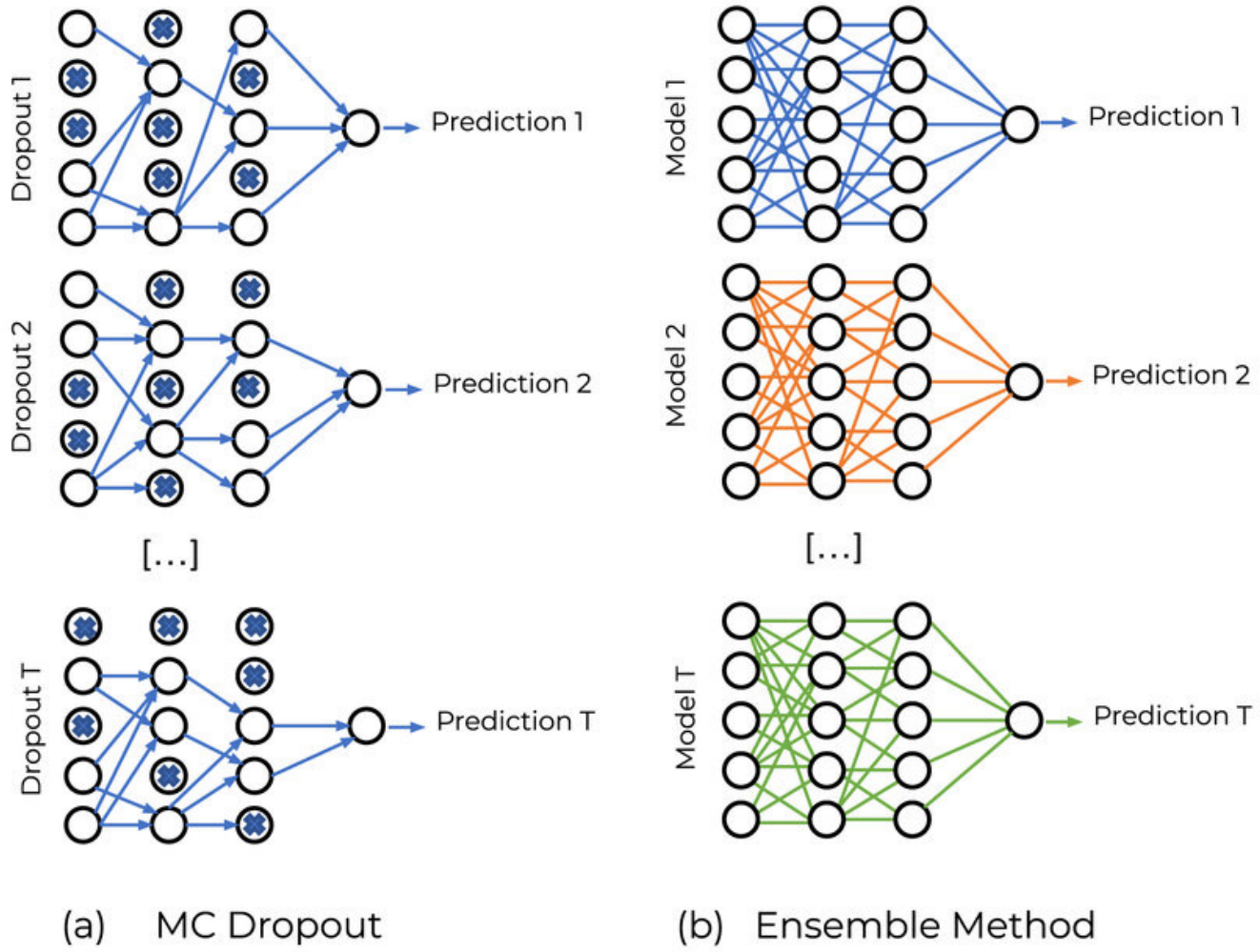


Fig: https://www.researchgate.net/figure/a-MC-Dropout-at-inference-time-The-T-predictions-are-obtained-from-Dropout-of_fig10_359970124

Method for Estimating Epistemic Uncertainty: Monte Carlo Dropout

For classification problems:

For classification this can be approximated using Monte Carlo integration as follows:

$$p(y = c|\mathbf{x}, \mathbf{X}, \mathbf{Y}) \approx \frac{1}{T} \sum_{t=1}^{T} \text{Softmax}(\mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x})) \tag{3}$$

with T sampled masked model weights $\widehat{\mathbf{W}}_t \sim q_\theta^*(\mathbf{W})$, where $q_\theta(\mathbf{W})$ is the Dropout distribution [6]. The uncertainty of this probability vector $\mathbf{p}$ can then be summarised using the entropy of the probability vector: $H(\mathbf{p}) = -\sum_{c=1}^{C} p_c \log p_c$. For regression this epistemic uncertainty is captured

"What uncertainties do we need for computer vision?"
https://arxiv.org/pdf/1703.04977

Method for Estimating Epistemic Uncertainty: Monte Carlo Dropout

For regression problems:

probability vector: $H(\mathbf{p}) = -\sum_{c=1}^{C} p_c \log p_c$. For regression this epistemic uncertainty is captured by the predictive variance, which can be approximated as:

$$\text{Var}(\mathbf{y}) \approx \sigma^2 + \frac{1}{T}\sum_{t=1}^{T} \mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x})^T \mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x}_t) - E(\mathbf{y})^T E(\mathbf{y}) \quad (4)$$

with predictions in this epistemic model done by approximating the predictive mean: $E(\mathbf{y}) \approx \frac{1}{T}\sum_{t=1}^{T} \mathbf{f}^{\widehat{\mathbf{W}}_t}(\mathbf{x})$. The first term in the predictive variance, σ^2, corresponds to the amount of noise inherent in the data (which will be explained in more detail soon). The second part of the predictive variance measures how much the model is uncertain about its predictions – this term will vanish when we have zero parameter uncertainty (i.e. when all draws $\widehat{\mathbf{W}}_t$ take the same constant value).

"What uncertainties do we need for computer vision?"
https://arxiv.org/pdf/1703.04977

Methods for Estimating Aleatoric Uncertainty

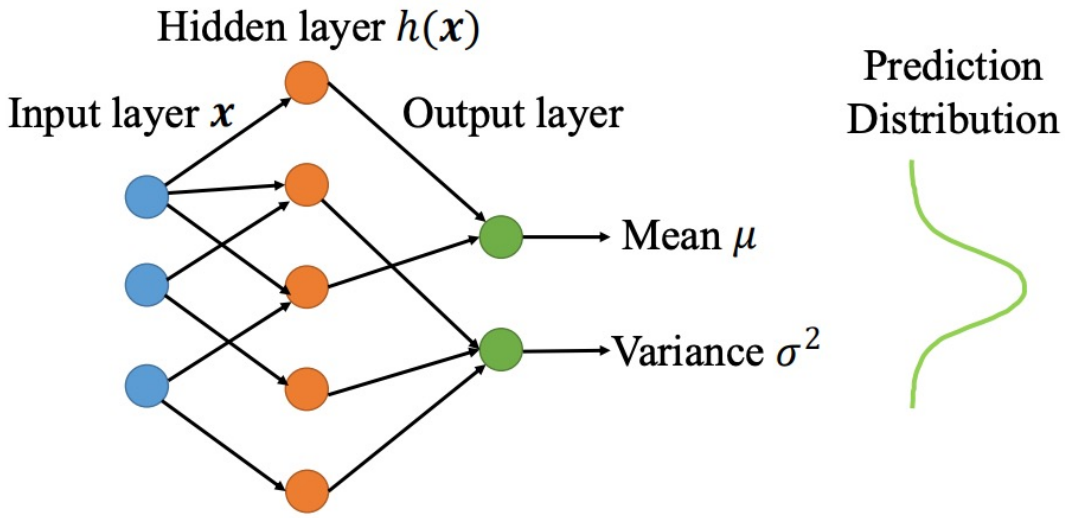
For regression problems:

- Aleatoric uncertainty is modeled by the distribution:

$$p(y|\boldsymbol{x}, \bar{\boldsymbol{\theta}})$$

- In a regression problem, we can obtain this as:

$$p(y|\boldsymbol{x}, \boldsymbol{\theta}) = \mathcal{N}(f_{\boldsymbol{\theta}}(\boldsymbol{x}), \sigma_{\boldsymbol{\theta}}(\boldsymbol{x}))$$

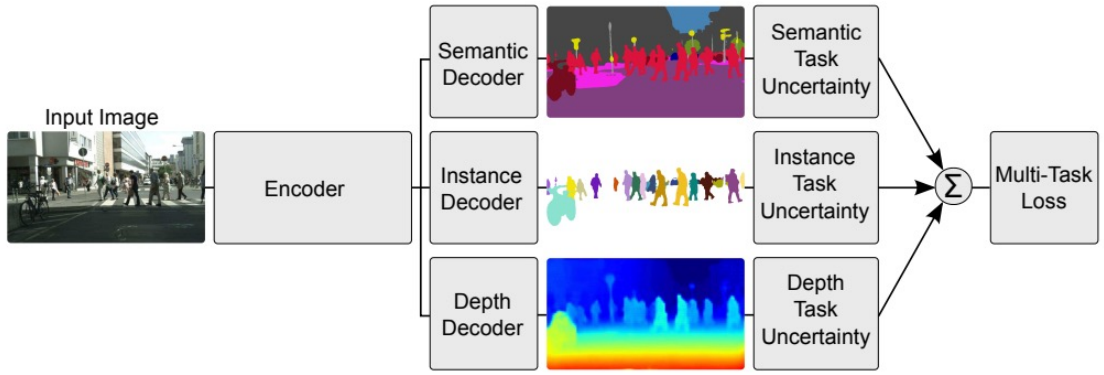


(a) Prediction distribution example.

$$\mathcal{L}_{\mathrm{NN}}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2\sigma_{\boldsymbol{\theta}}^2(\boldsymbol{x}_i)} ||y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)||^2 + \frac{1}{2}\log \sigma_{\boldsymbol{\theta}}^2(\boldsymbol{x}_i).$$

https://arxiv.org/pdf/2302.13425

Side Note: Multi-task Learning



$$\frac{1}{2\sigma_1^2}\mathcal{L}_1(\mathbf{W}) + \frac{1}{2\sigma_2^2}\mathcal{L}_2(\mathbf{W}) + \log\sigma_1\sigma_2$$

Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7482-7491).

$$\mathcal{L}_{\beta\text{-VAE}} = -E_{q(\mathrm{z}|\mathrm{x})}\Big[\log \underbrace{p(\mathrm{x}|\mathrm{z})}_{Decoder}\Big]$$

$$+\beta \cdot D_{KL}\Big(\underbrace{q(\mathrm{z}|\mathrm{x})}_{Encoder} \parallel p(\mathrm{z})\Big),$$

$$\mathcal{L}_{\text{L-VAE}} = -\frac{1}{\sigma_0^2}E_{q(\mathrm{z}|\mathrm{x})}\Big[\log p(\mathrm{x}|\mathrm{z})\Big] + \frac{1}{\sigma_1^2}\mathcal{D}_{KL}\Big(q(\mathrm{z}|\mathrm{x}) \parallel \mathrm{z}\Big) + \sum_{i=0,1}\sigma_i^2.$$

H. Mogultay Ozcan, S. Kalkan, F. Y. Vural, "L-VAE: Variational Auto-Encoder with Learnable Beta for Disentangled Representation", Machine Vision and Applications, 36(104):1-14, 2025.

Methods for Estimating Aleatoric Uncertainty

For classification problems:

(passing inputs through the model to get logits). We only need to sample from the logits, which is a fraction of the network's compute, and therefore does not significantly increase the model's test time. We can rewrite the above and obtain the following numerically-stable stochastic loss:

$$\hat{\mathbf{x}}_{i,t} = \mathbf{f}_i^{\mathbf{W}} + \sigma_i^{\mathbf{W}} \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I)$$

$$\mathcal{L}_x = \sum_i \log \frac{1}{T} \sum_t \exp(\hat{x}_{i,t,c} - \log \sum_{c'} \exp \hat{x}_{i,t,c'}) \tag{12}$$

with $x_{i,t,c'}$ the c' element in the logit vector $\mathbf{x}_{i,t}$.

"What uncertainties do we need for computer vision?"
https://arxiv.org/pdf/1703.04977

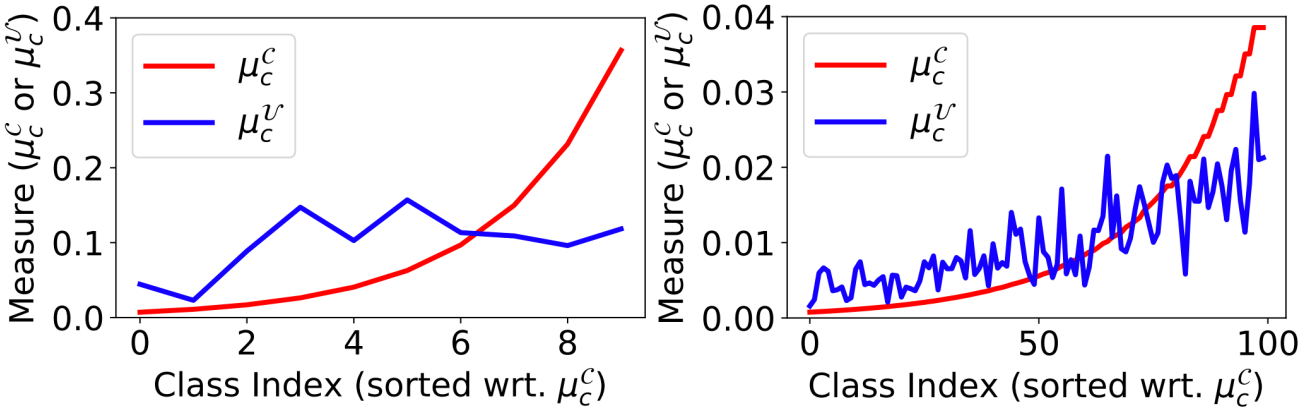
Using Uncertainty [from our research]

Class Uncertainty:
A Measure to Mitigate Class Imbalance

Zeynep Sonat Baltaci[1,2], Kemal Oksuz[3], Selim Kuzucu[1], Kivanc Tezoren[1], Berkin Kerim Konar[1], Alpay Ozkan[1], Emre Akbas[1,4,†], Sinan Kalkan[1,4,†]
[1]Dept. of Computer Engineering, METU, Ankara, Turkey
[2]LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
[3]Five AI Ltd., United Kingdom
[4]Center for Robotics and Artificial Intelligence (ROMER), METU, Ankara, Turkey
https://arxiv.org/pdf/2311.14090

- Measuring data imbalance using epistemic and aleatoric uncertainty

- Alternative:
  - Class cardinality
  - Loss values
  - Prediction probability



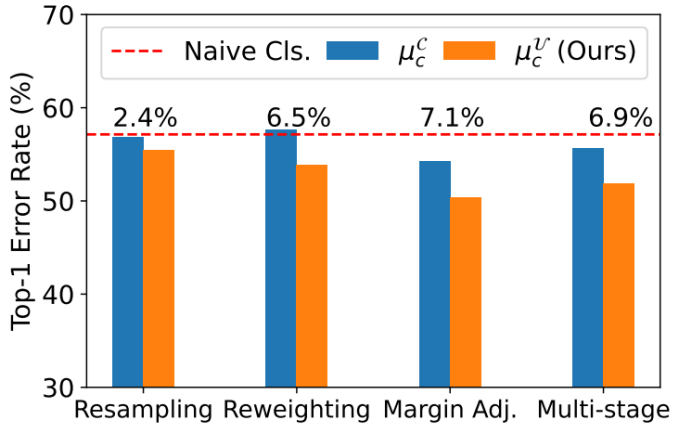
(a) μ^C vs. μ^U on CIFAR-10-LT (b) μ^C vs. μ^U on CIFAR-100-LT

Using Uncertainty [from our research]

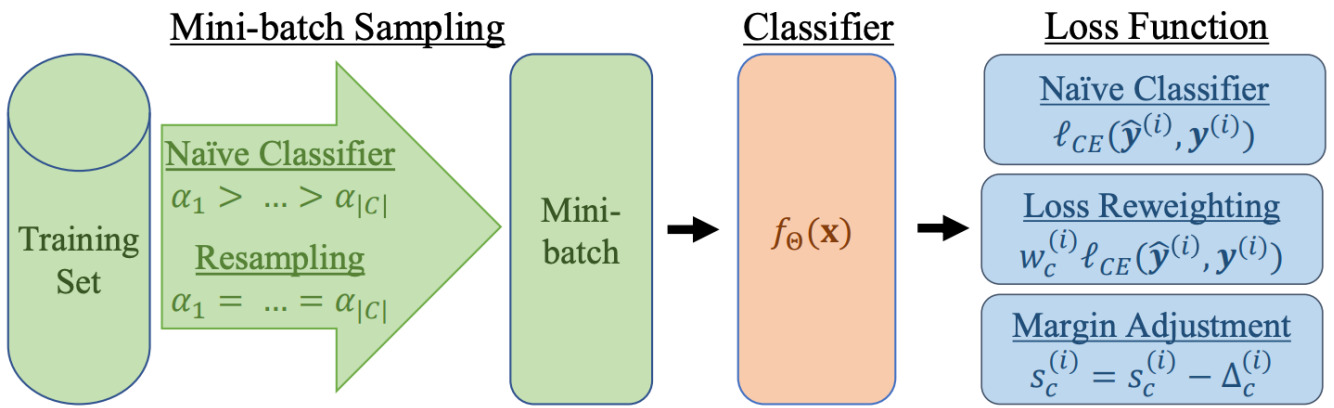
Class Uncertainty:
A Measure to Mitigate Class Imbalance

Zeynep Sonat Baltaci[1,2], Kemal Oksuz[3], Selim Kuzucu[1], Kivanc Tezoren[1], Berkin Kerim Konar[1], Alpay Ozkan[1], Emre Akbas[1,4,†], Sinan Kalkan[1,4,†]
[1]Dept. of Computer Engineering, METU, Ankara, Turkey
[2]LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
[3]Five AI Ltd., United Kingdom
[4]Center for Robotics and Artificial Intelligence (ROMER), METU, Ankara, Turkey

https://arxiv.org/pdf/2311.14090



(a) Gains of our CLASS UNCERTAINTY



(b) Naïve classifier vs. different ways of handling imbalance

Fig. 1. **(a)** Using our CLASS UNCERTAINTY improves top-1 error rate of all aforementioned methods (CIFAR-100-LT with an IR of 50 using ResNet-32). In particular, we obtain (i) the sampling probability of each class in resampling methods; (ii) the weights of the loss for each class; (iii) the margins to be enforced around each class; or (iv) again sampling probability in the second-stage of multi-stage training strategy using CLASS UNCERTAINTY. The numbers on top of the histograms show relative gain over cardinality-based methods. Baselines: Progressively-balanced sampling [2], class-balanced Focal Loss [3], LDAM with reweighting [7] and deferred resampling. **(b)** Naïve classifier is not robust to class imbalance, giving rise to a plethora of mitigation methods: (1) "resampling" samples a balanced set of examples; (2) "reweighting" assigns a weight for each class ($w_c^{(i)}$); (3) "margin-adjustment" methods assign different margins ($\Delta_c^{(i)}$) to the logits ($s_c^{(i)}$) of different classes; and (4) "multi-stage training" first trains a naïve classifiers followed by a resampling or reweighting method to re-train or fine-tune the classifier. These methods generally rely on the cardinality of the classes.

Using Uncertainty [from our research]

Journal of Artificial Intelligence Research 80 (2024) 12-15 Submitted 02/2024; published 08/2024

Uncertainty as a Fairness Measure

Selim Kuzucu SELIM.KUZUCU@METU.EDU.TR
Department of Computer Engineering
Middle East Technical University
06800 Ankara, Turkiye

Jiaee Cheong JC2208@CAM.AC.UK
Department of Computer Science
University of Cambridge
Cambridge, CB3 0FD, United Kingdom
The Alan Turing Institute
London, NW1 2DB, United Kingdom

Hatice Gunes HG410@CAM.AC.UK
Department of Computer Science
University of Cambridge
Cambridge, CB3 0FD, United Kingdom

Sinan Kalkan SKALKAN@METU.EDU.TR
Department of Computer Engineering &
ROMER Robotics-AI Center
Middle East Technical University
06800 Ankara, Turkiye

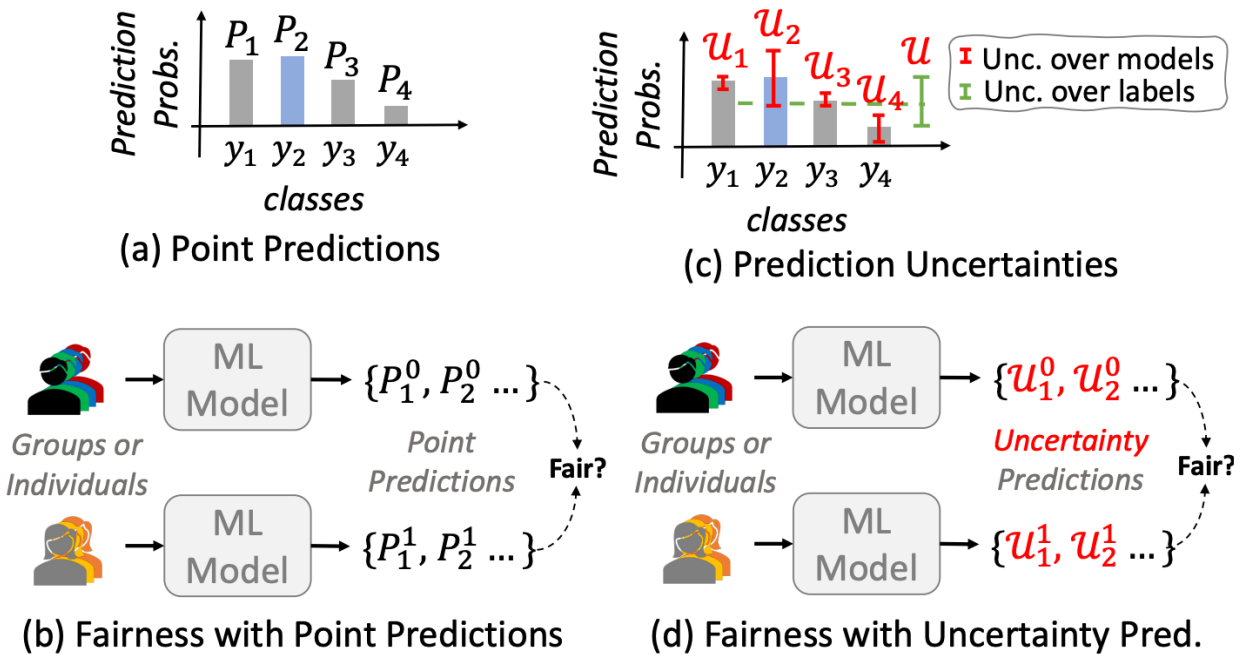


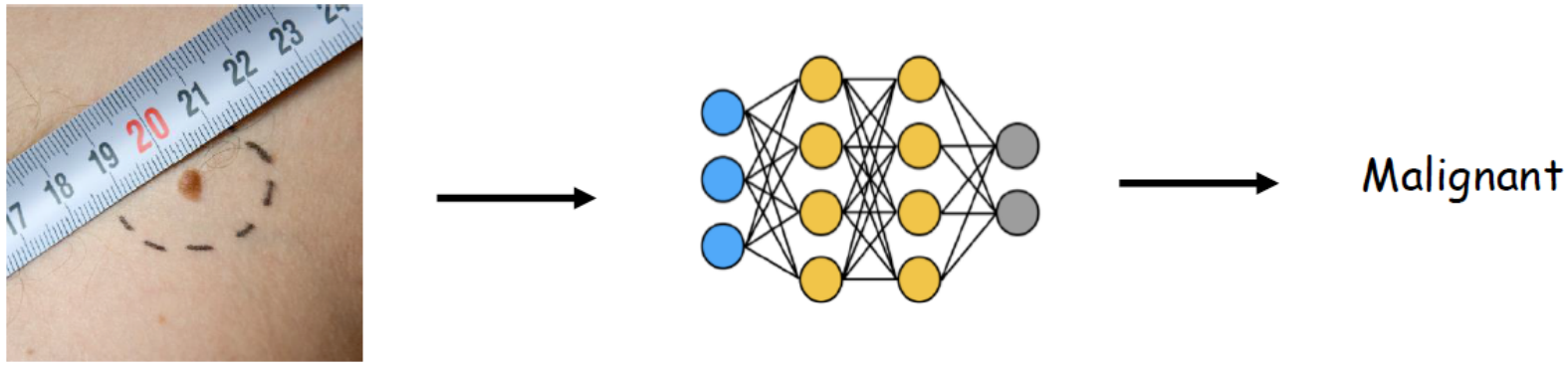
Figure 1: Existing fairness measures utilize point predictions for quantifying fairness, which ignores the uncertainty (variance) of the predictions **(a-b)**. We fill this gap by using uncertainty instead for measuring fairness **(c-d)**.

Explainable AI

Agenda

- Interpretable models vs. posthoc explanations
- Local vs. global explainability
- Feature attribution methods
- Saliency methods
- Causality
- Concept Bottleneck Models

Beyond Accuracy



Why did the model make this prediction?

"… the algorithm appeared more likely to label images with rulers as malignant … "

2

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Goals of Explainable ML

- Explain why the model made a particular prediction on a specific input
- Explain how the model makes predictions across all inputs
- Explain how the training data affects model predictions
- Explain what changes to the input can cause the model make a different decision

3

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Explainability and Emerging AI Policy

EU General Data Protection Regulation (2018)

Right to explanation

…

In any case, such processing should be subject to suitable safeguards, which should include specific information to the data subject and the right to obtain human intervention, to express his or her point of view, to obtain an explanation of the decision reached after such assessment and to challenge the decision.

7

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Explaining predictions

- Facilitates debugging

(A) **Cow: 0.99**, Pasture: 0.99, Grass: 0.99, No Person: 0.98, Mammal: 0.98

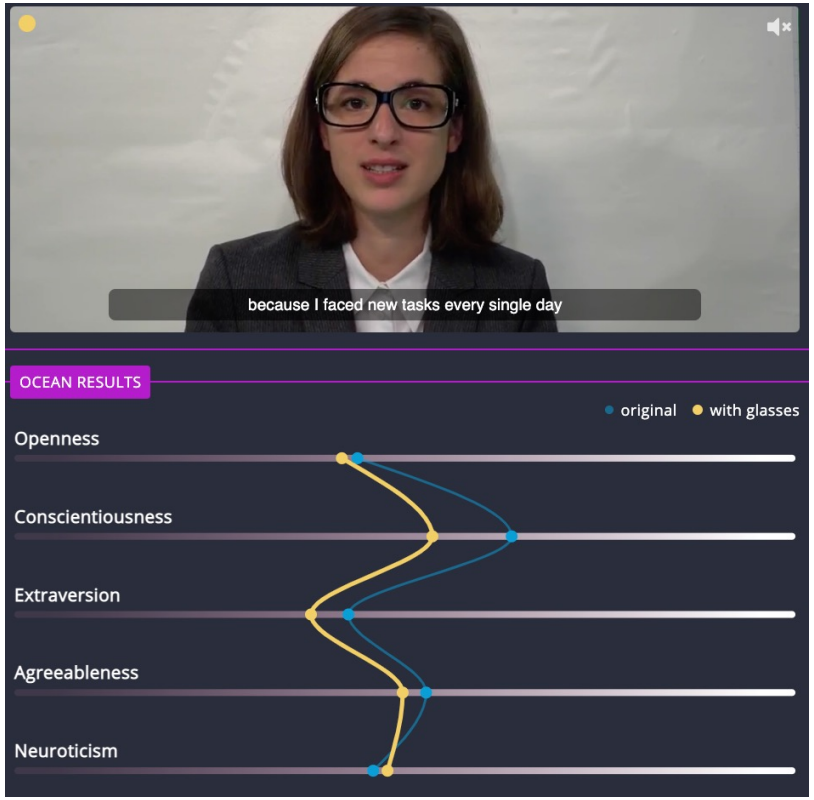
(B) No Person: 0.99, Water: 0.98, Beach: 0.97, Outdoors: 0.97, Seashore: 0.97

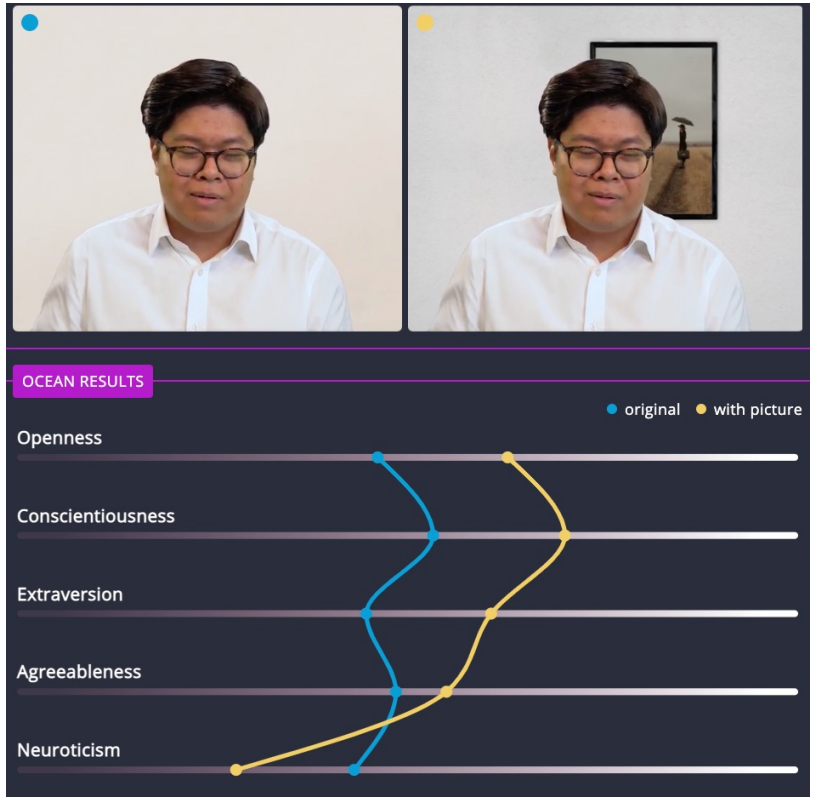
(C) No Person: 0.97, **Mammal: 0.96**, Water: 0.94, Beach: 0.94, Two: 0.94

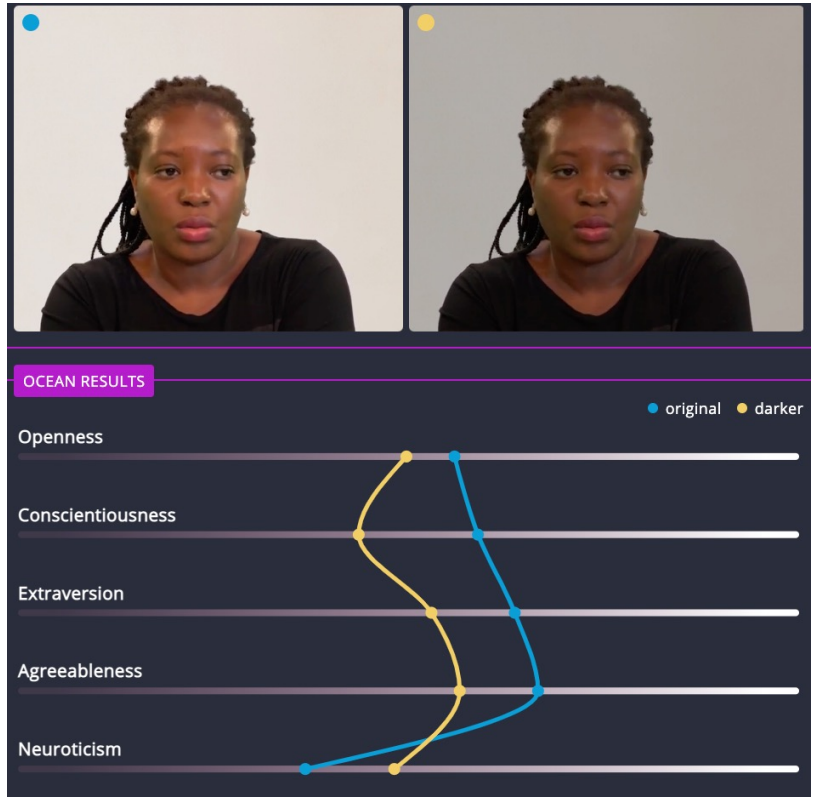
Beery, Sara, Grant Van Horn, and Pietro Perona. "Recognition in terra incognita." Proceedings of the European Conference on Computer Vision (ECCV). 2018.

Explaining predictions

- Helps identifying biases



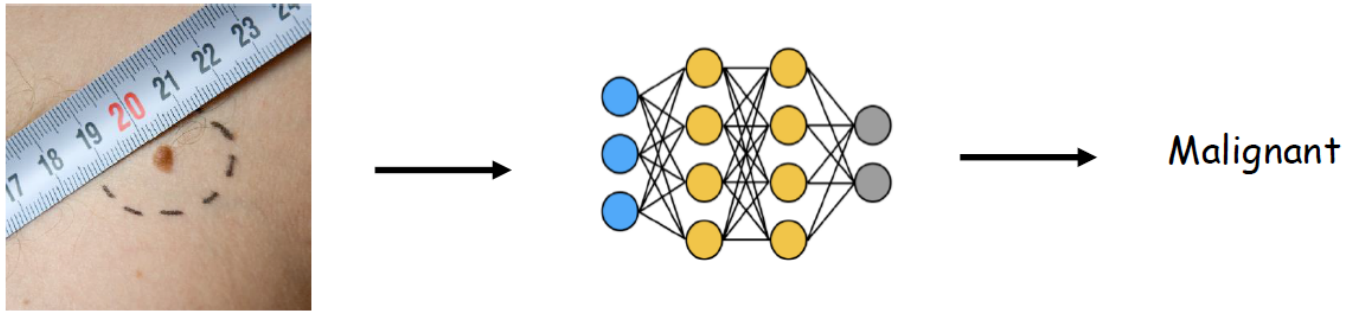




https://web.br.de/interaktiv/ki-bewerbung/en/

Explaining predictions

- Helps whether/if model predictions can be trusted

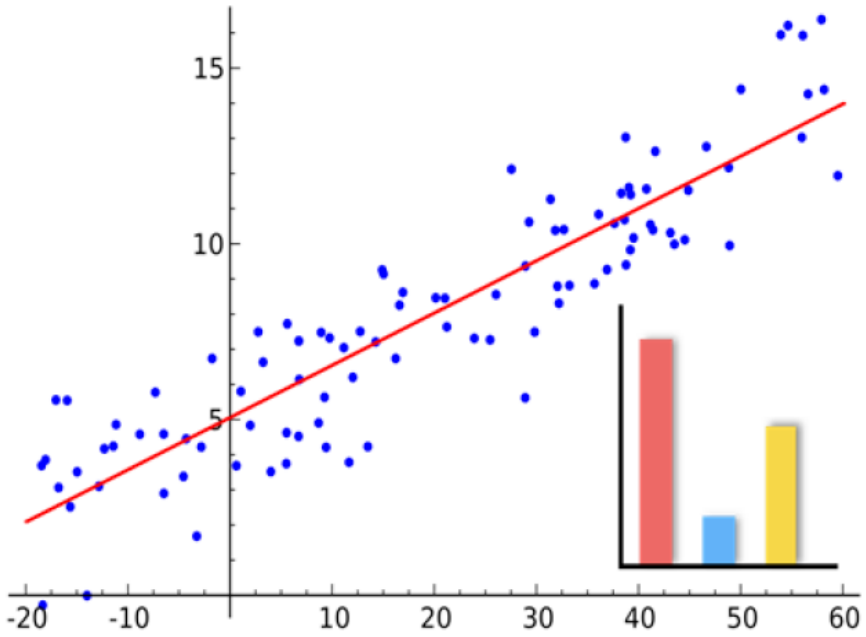


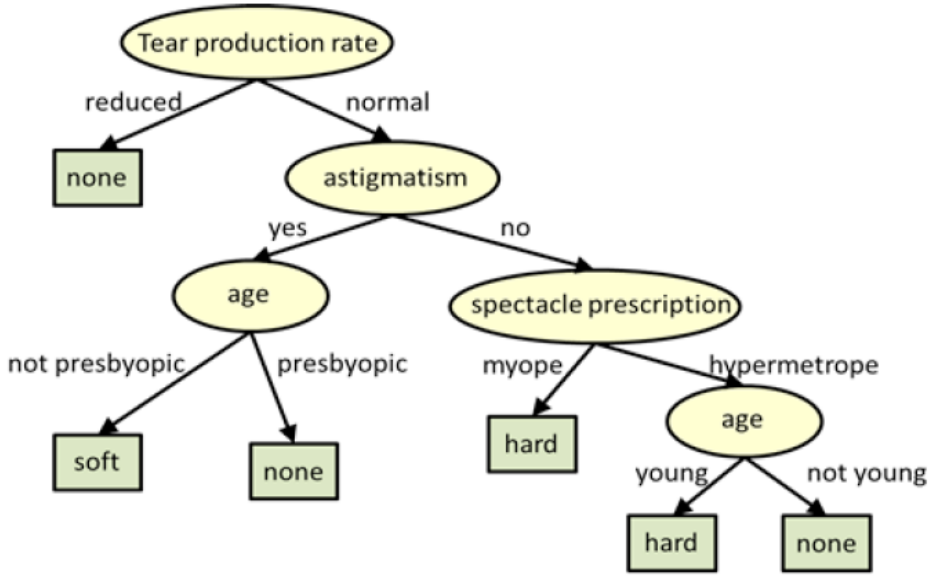
Why did the model make this prediction?

"... the algorithm appeared more likely to label images with rulers as malignant ... "

Fig: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Achieving Explainability: Inherently Interpretable Models



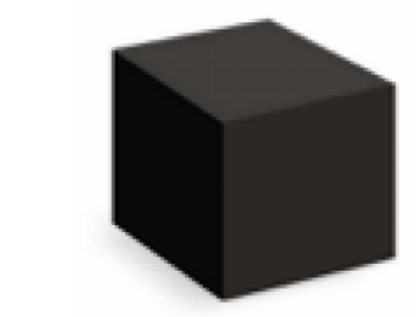


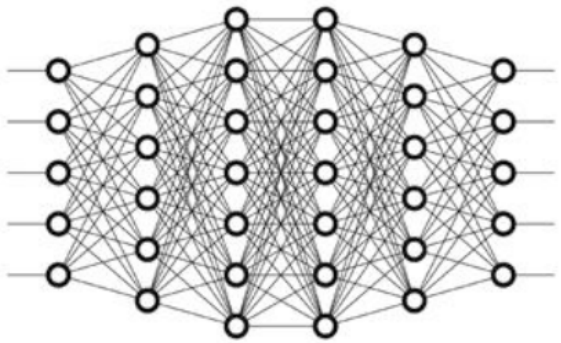
if $(age = 18 - 20)$ **and** $(sex = male)$ **then predict** yes
else if $(age = 21 - 23)$ **and** $(priors = 2 - 3)$ **then predict** yes
else if $(priors > 3)$ **then predict** yes
else predict no

12

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

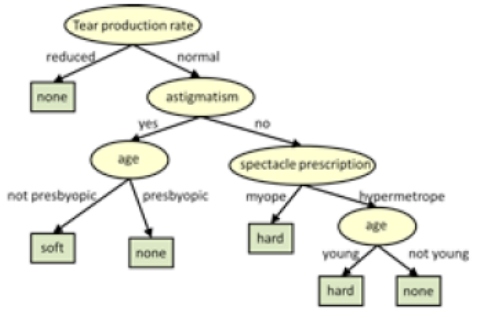
Achieving Explainability: Post-hoc Explanations





Explainer

if $(age = 18 - 20)$ **and** $(sex = male)$ **then predict** yes
else if $(age = 21 - 23)$ **and** $(priors = 2 - 3)$ **then predict** yes
else if $(priors > 3)$ **then predict** yes
else predict no



14

Interpretability vs. Accuracy

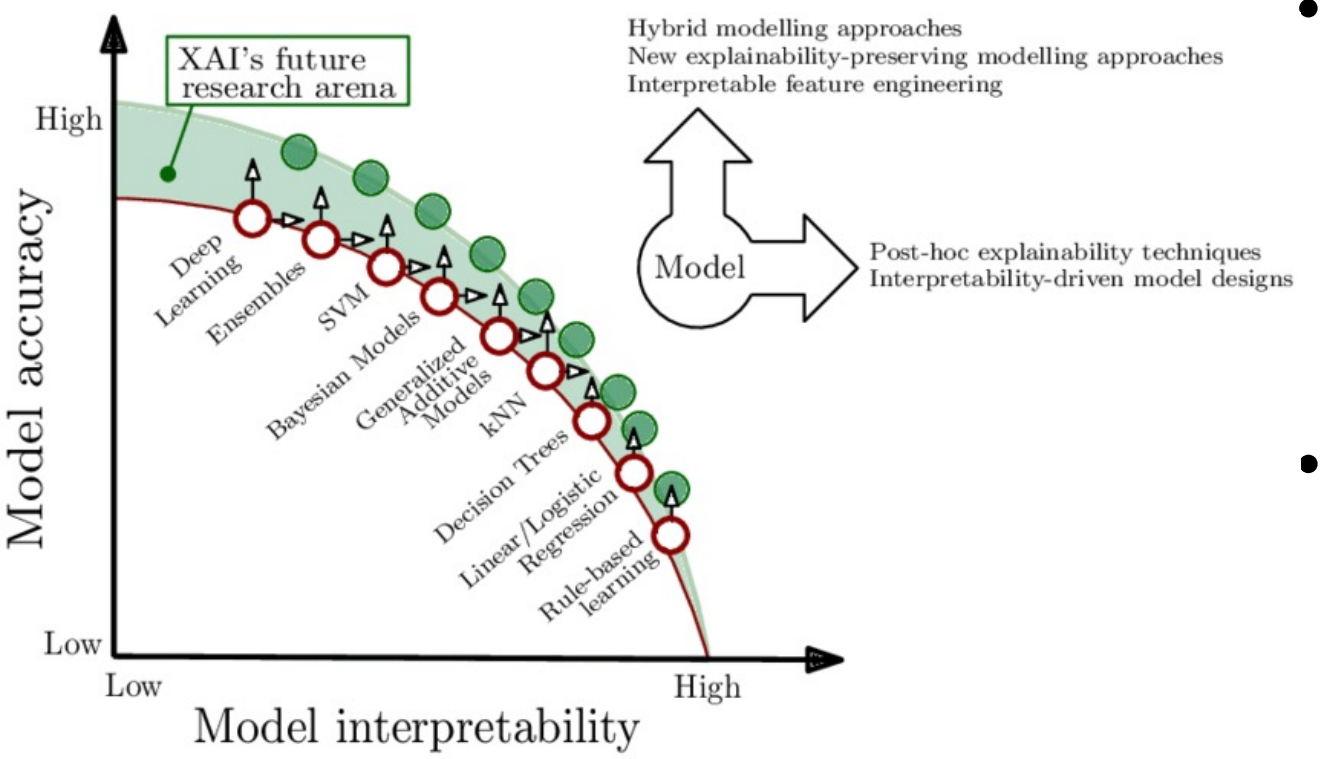
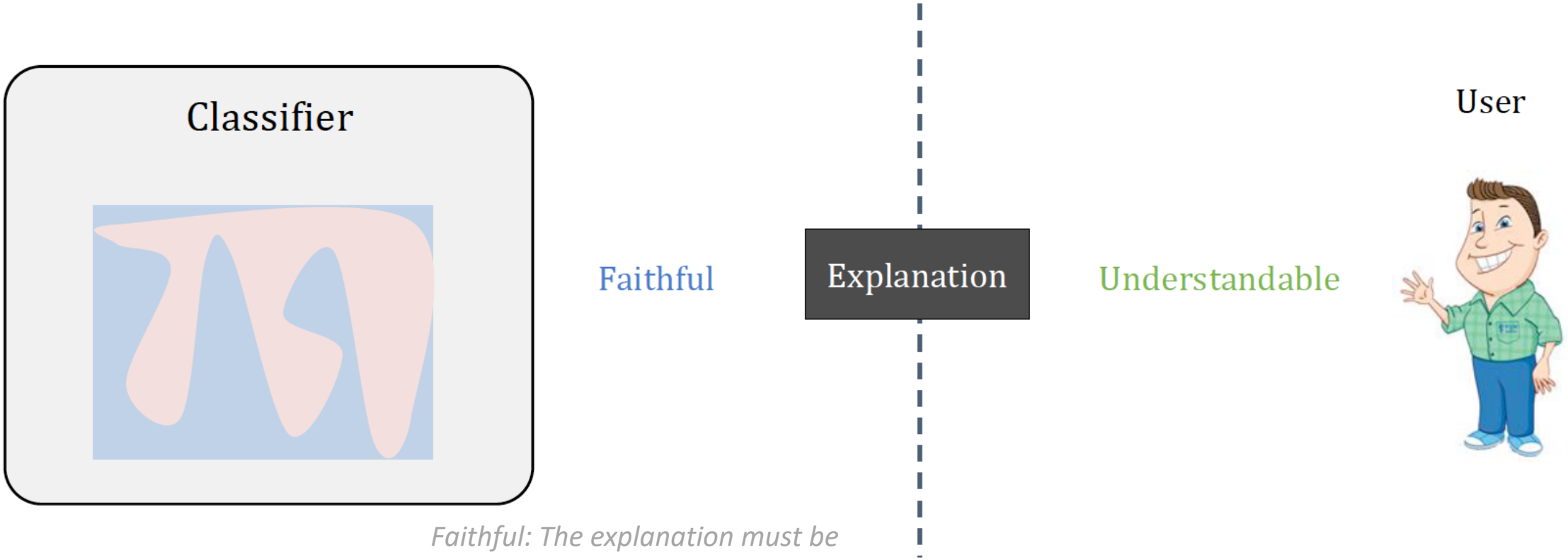


Fig: https://www.sciencedirect.com/science/article/abs/pii/S1566253519308103

- If you have an interpretable model that is sufficiently accurate for your application, use it
- Otherwise, use posthoc explanations for a non-interpretable model

What is an Explanation ?

Ideally, interpretable description of the model behavior



Faithful: The explanation must be true to the model's operation

16

Local vs. Global Explanations

Local	Global
Explain individual predictions	Explain complete behavior of the model
Help unearth biases in the *local neighborhood* of a given instance	Help shed light on *big picture biases* affecting larger subgroups
Help vet if individual predictions are being made for the right reasons	Help vet if the model, at a high level, is suitable for deployment

18

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Feature Attribution Methods

Formalizing Feature Attribution Problem

- Given an input x and model f, select a subset (of specified size) of features of x that contribute the most to the prediction f(x)

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Formalizing Feature Attribution Problem

- Given an input x and model f, select a subset (of specified size) of features of x that contribute the most to the prediction f(x)
- First attempt: if $x \in \mathbb{R}^d$ output should be d-dim vector over {0,1} specifying for each feature whether it is selected or not x
- Problem: Features in representation of x may not be interpretable to humans

  e.g. an input image is a tensor with three color channels per pixel

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Formalizing Feature Attribution Problem

- If input x is d-dimensional, first define a simpler d'-dimensional "interpretable" representation

- Output of explanation method g, for given input x and model f, is d'-dim vector over {0,1} that selects a subset of interpretable features (possibly with weights)

- Desired properties
  - Model agnostic: Method works for any model f
  - Interpretability: Minimize complexity of g (e.g. select at most 25% features)
  - Local fidelity: g approximates f well in the vicinity of x

  Note: to formalize local fidelity, need a way to map x and g to d-dim vectors

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1135-1144).

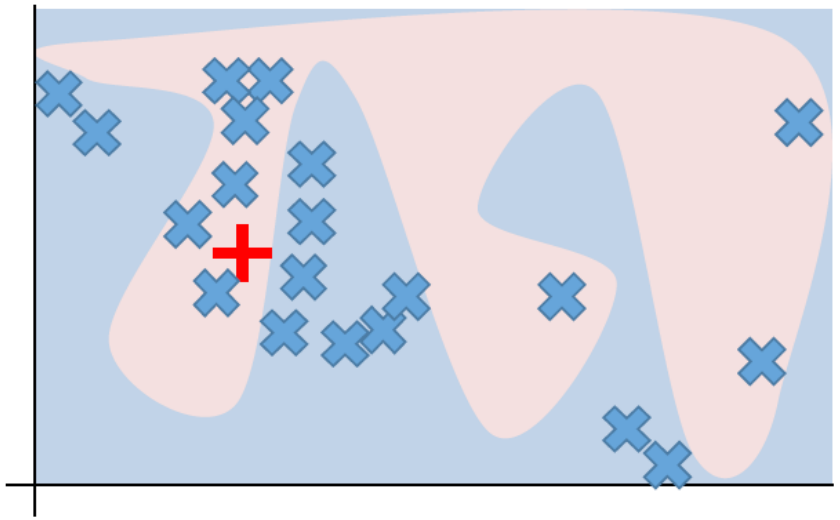
Local Interpretable Model-agnostic Explanations (LIME)

Resources:

https://christophm.github.io/interpretable-ml-book/lime.html

LIME: Local Interpretable Model-agnostic Explanations

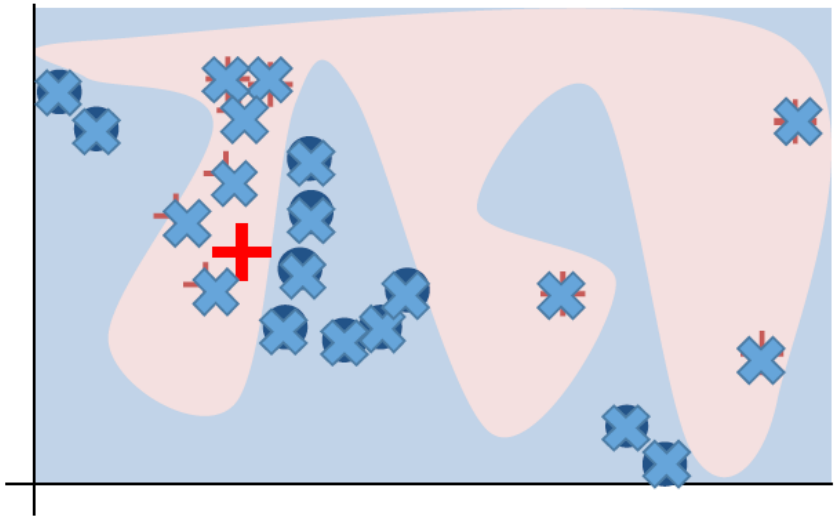
1. Sample points around x



Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

LIME Algorithm

1. Sample points around x
2. Use model to predict labels for each sample

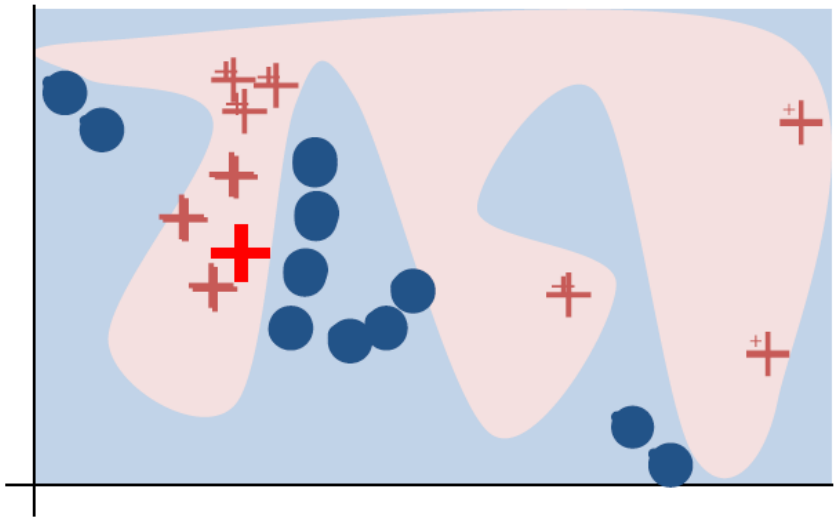


27

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

LIME Algorithm

1. Sample points around x
2. Use model to predict labels for each sample
3. Weigh samples according to distance to x



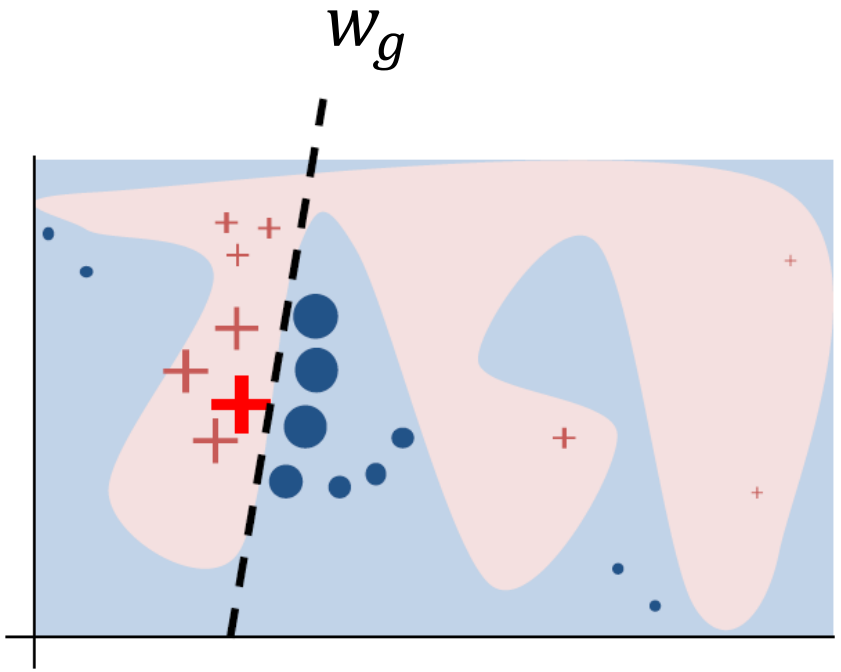
$$\pi_x(z) = \exp\left(-\frac{D(x,z)^2}{\sigma^2}\right)$$

where $D(x,z)$ can be e.g., cosine distance for text and L2-distance for images.

28

LIME Algorithm

1. Sample points around x
2. Use model to predict labels for each sample
3. Weigh samples according to distance to x
4. Learn simple linear model on weighted samples



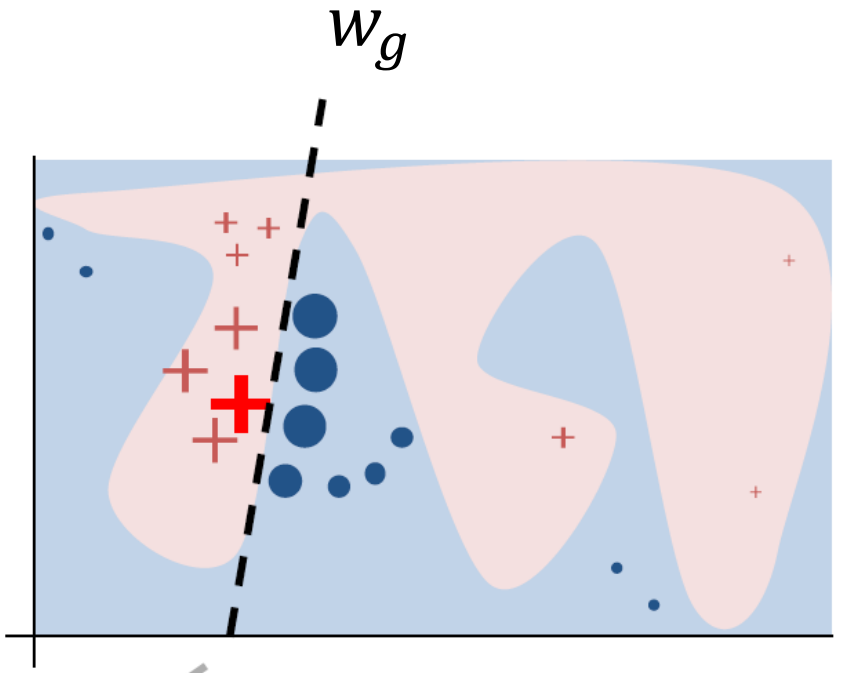
$$g(z') = w_g \cdot z'$$

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z) \left(f(z) - g(z')\right)^2$$

29

LIME Algorithm

1. Sample points around x
2. Use model to predict labels for each sample
3. Weigh samples according to distance to x
4. Learn simple linear model on weighted samples
5. Use simple linear model to explain



$$g(z') = w_g \cdot z'$$

w_g of this model includes one coefficient (importance) for each feature.

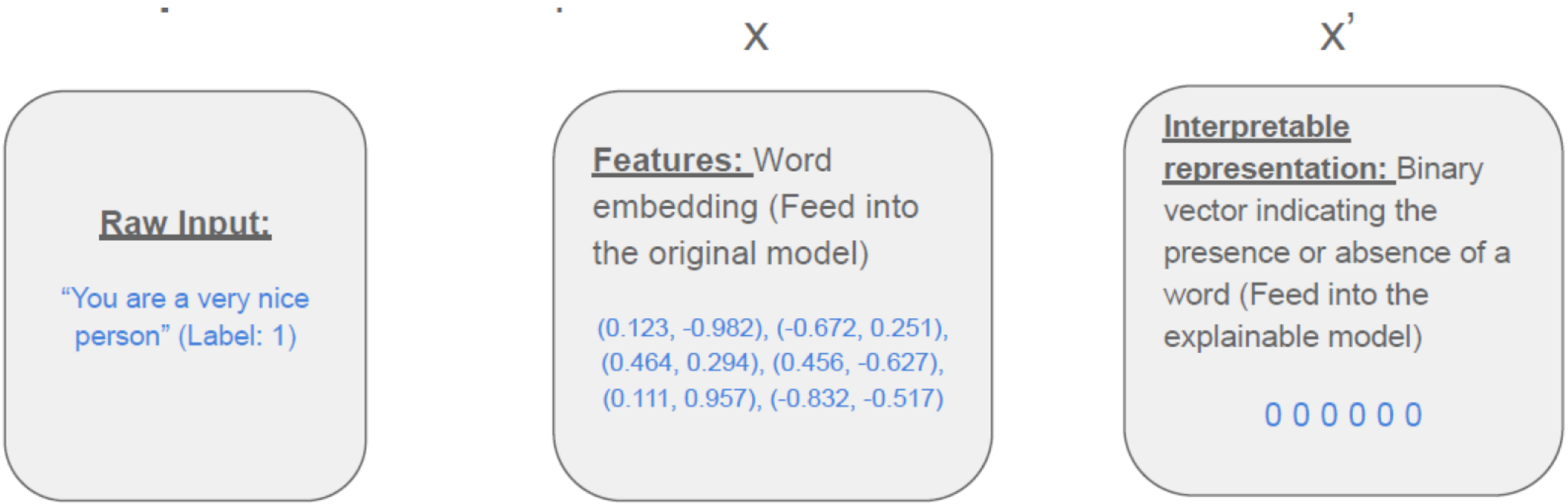
LIME in more detail

- Consider a (black-box) classifier that labels an input sentence as good (label 1) or bad (label 0)
- Suppose it labels “You are a very nice person” as 1
- We want as an explanation which 3 words contributed the most to this prediction

31

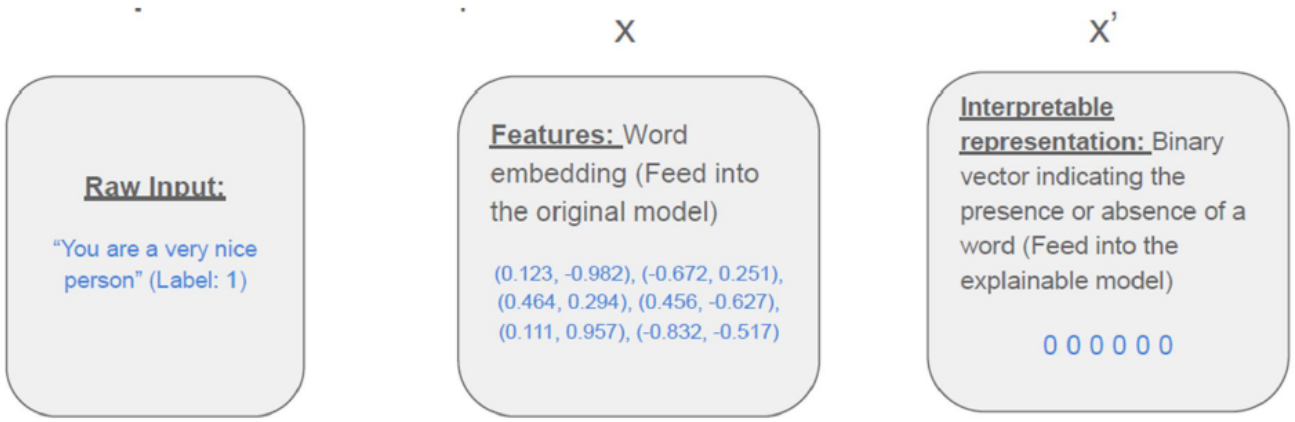
Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Interpretable Data Representation



Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

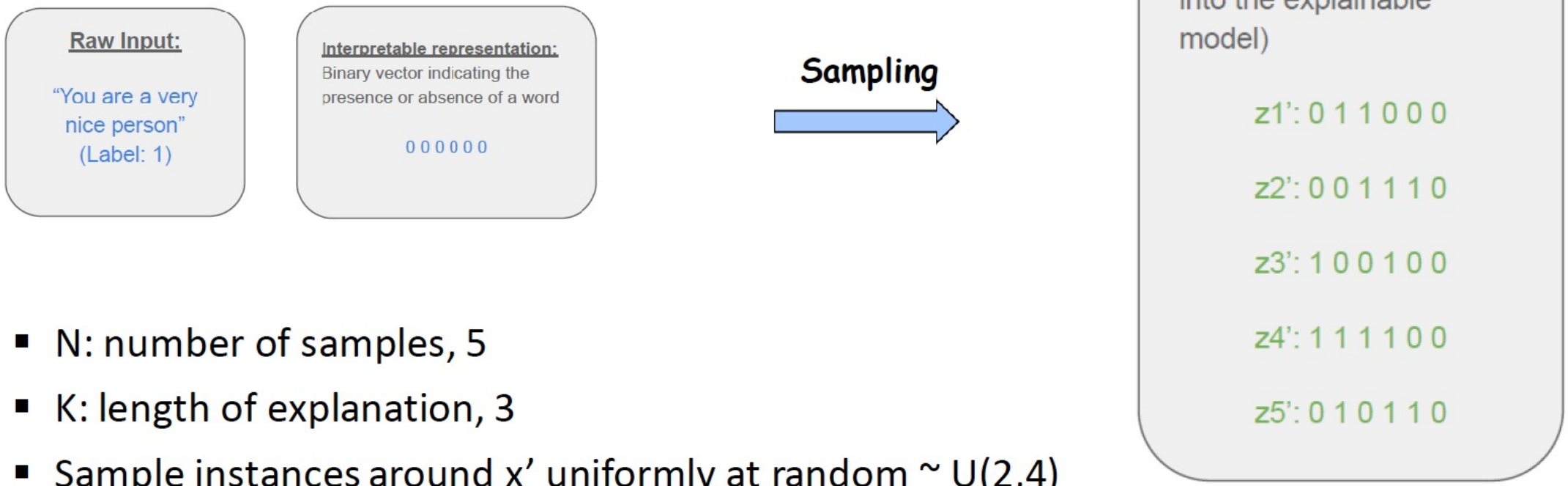
Sampling



- N: number of samples, say, 5
- K: length of explanation, say, 3
- Sample instances around x' by drawing nonzero elements uniformly at random, say, ~U(2,4)

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Sampling



- N: number of samples, 5
- K: length of explanation, 3
- Sample instances around x' uniformly at random ~ U(2,4)

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Analyzing Samples

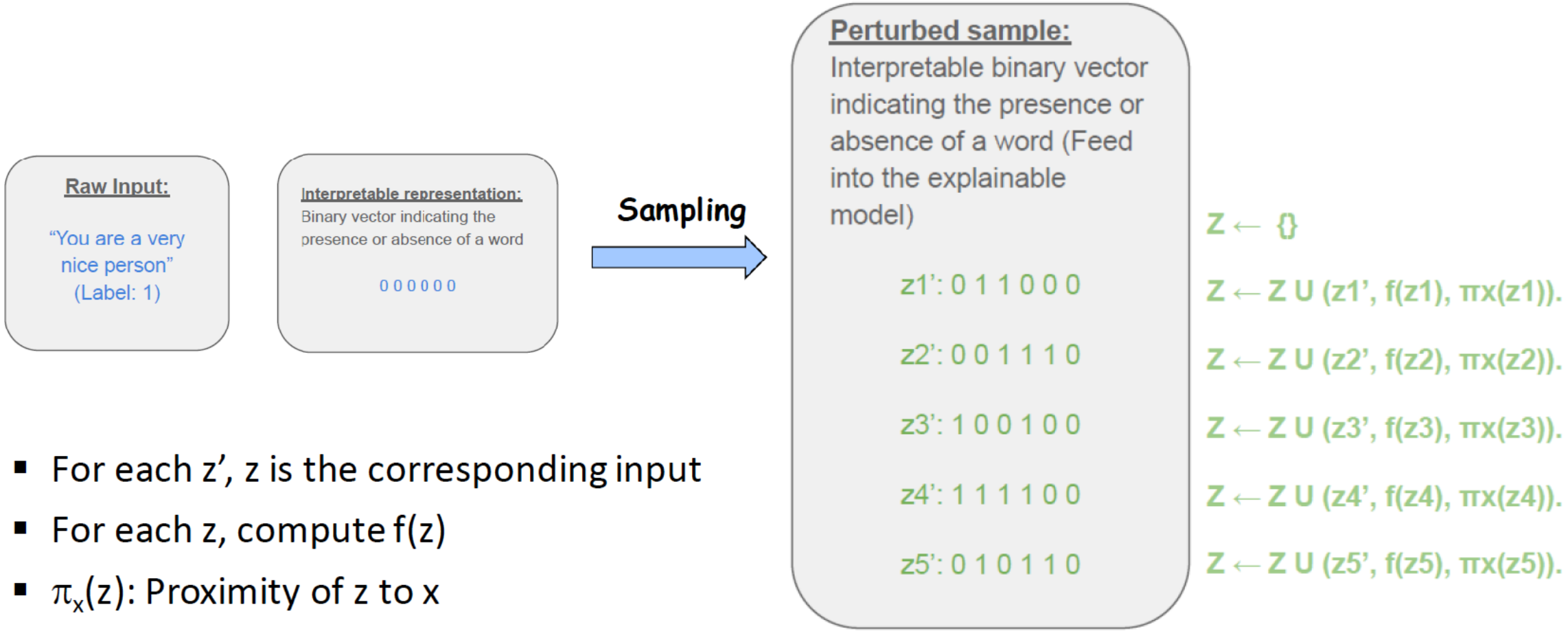
- For each z', z is the corresponding input

  z'= 111100 maps to word vector of "You are a very"
- For each z, compute f(z)
- $\pi_x(z)$: Proximity measure between an instance z to x

35

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Analyzing Samples



- For each z', z is the corresponding input
- For each z, compute f(z)
- $\pi_x(z)$: Proximity of z to x

36

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Finding Explanable Model

Explainable model g:

Choose linear models here

Dataset:

Z (contains data & label & additional distance metric)

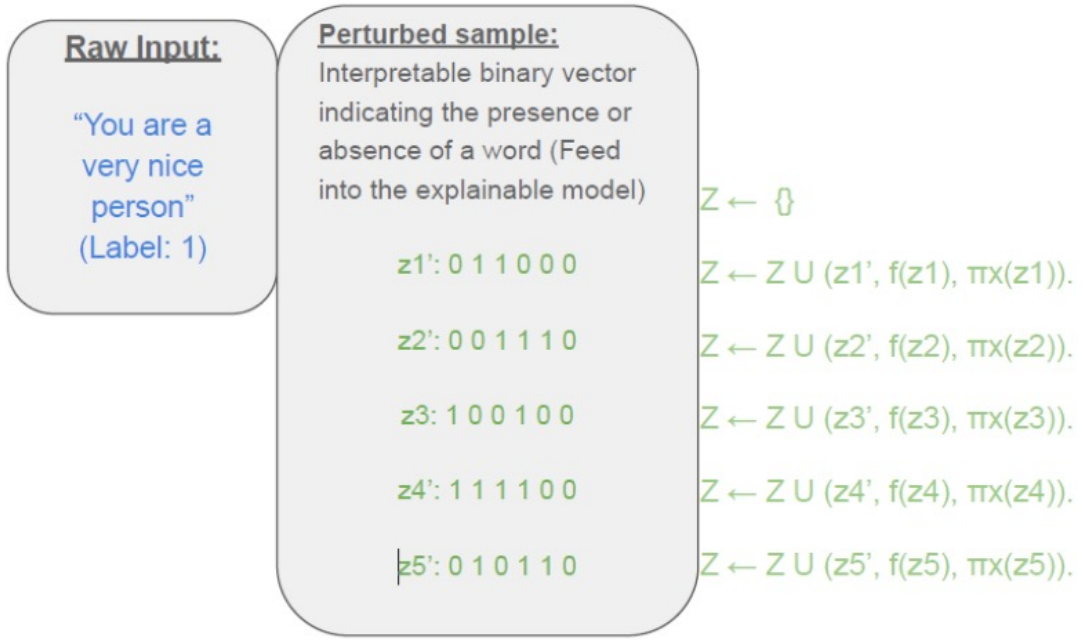
Objective function:

Local fidelity Interpretability

$$\min \ \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

Ω(g): A measure of complexity (as opposed to interpretability)

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z) \left(f(z) - g(z')\right)^2$$



Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Finding Explanable Model

Explainable model g:

Choose linear models here

Dataset:

Z (contains data & label & additional distance metric)

Objective function:

Local fidelity Interpretability

$$\min \quad \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

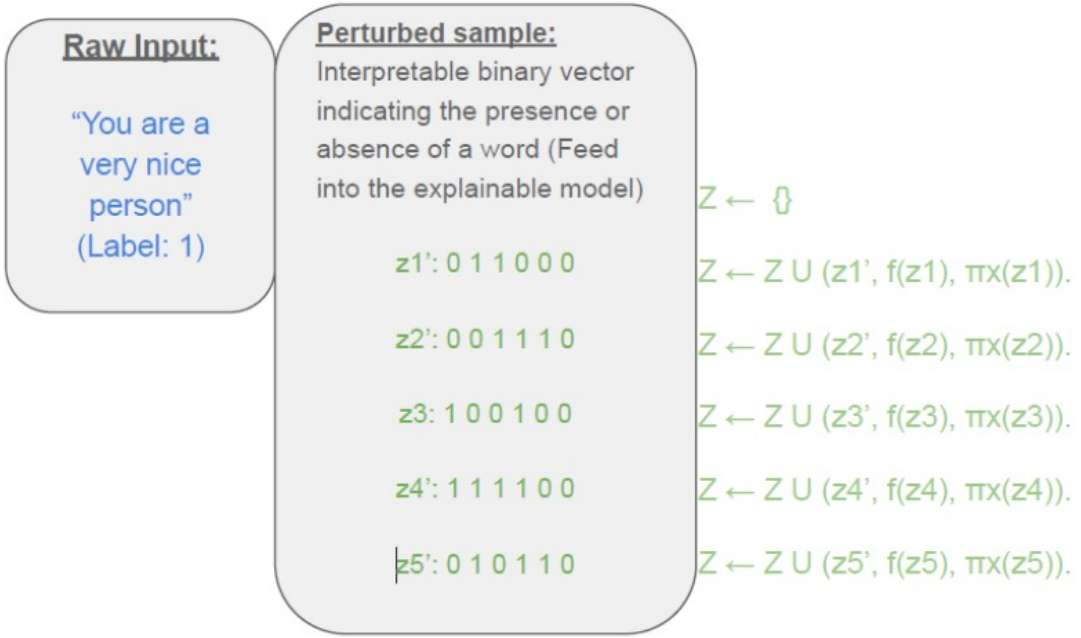
Ω(g): A measure of complexity (as opposed to interpretability)

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z) \left(f(z) - g(z')\right)^2$$

Explanation:

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \quad \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

(Corresponding weight for each feature)



37

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Finding Explanable Model

Explainable model g:

Choose linear models here

Dataset:

Z (contains data & label & additional distance metric)

Objective function:

Local fidelity Interpretability

$$\min \ \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

Ω(g): A measure of complexity (as opposed to interpretability)

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z) \left(f(z) - g(z')\right)^2$$

Explanation:

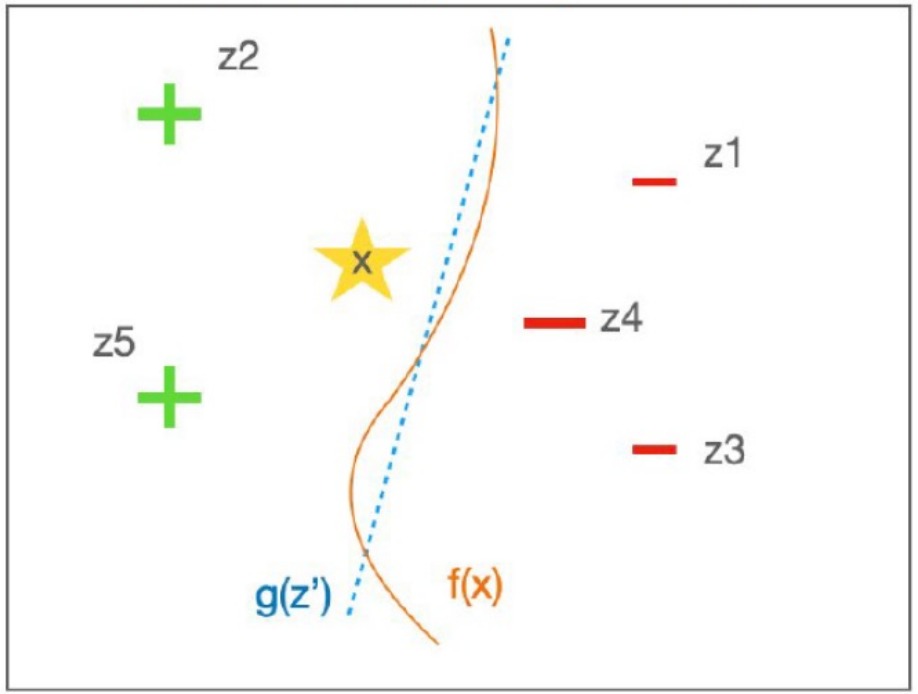
$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \ \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

(Corresponding weight for each feature)

For the toy example:

Choose K-Lasso to limit # of explanations (K=3), i.e. we can only choose up to 3 words here for explanation

Explainable model g vs original model f



Explanation:

Nice 0.96
Very 0.56
You 0.43

37

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

K-Lasso

(Least Absolute Shrinkage and Selection Operator)

- Linear regression with L1 regularization

$$\min_w \frac{1}{2n_{\text{samples}}} ||Xw - y||_2^2 + \alpha ||w||_1$$

- K-Lasso: Select K features in X, i.e., select K non-zero elements in w.
  - Start with high α => strong regularization => all weights are close to zero.
  - Slowly decrease α => weaker regularization => non-zero weights emerge.
  - When # of non-zero dimensions reach K, stop.

LIME Algorithm

Algorithm 1 Sparse Linear Explanations using LIME

Require: Classifier f, Number of samples N
Require: Instance x, and its interpretable version x'
Require: Similarity kernel π_x, Length of explanation K

$\mathcal{Z} \leftarrow \{\}$
for $i \in \{1, 2, 3, ..., N\}$ **do**
  $z'_i \leftarrow sample\_around(x')$
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup \langle z'_i, f(z_i), \pi_x(z_i) \rangle$
end for
$w \leftarrow$ K-Lasso$(\mathcal{Z}, K)$ ▷ with z'_i as features, $f(z)$ as target
return w

39

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Interpretable Features: From Pixels to Superpixels

- Superpixel is a technique to segment an image into regions by considering similarity according to perceptual features
- Segmentation is dependent on specific input
- Well-known algorithms based on graph partitioning and clustering

24

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Application of LIME to Images

Interpretable Features: From Pixels to Superpixels

25

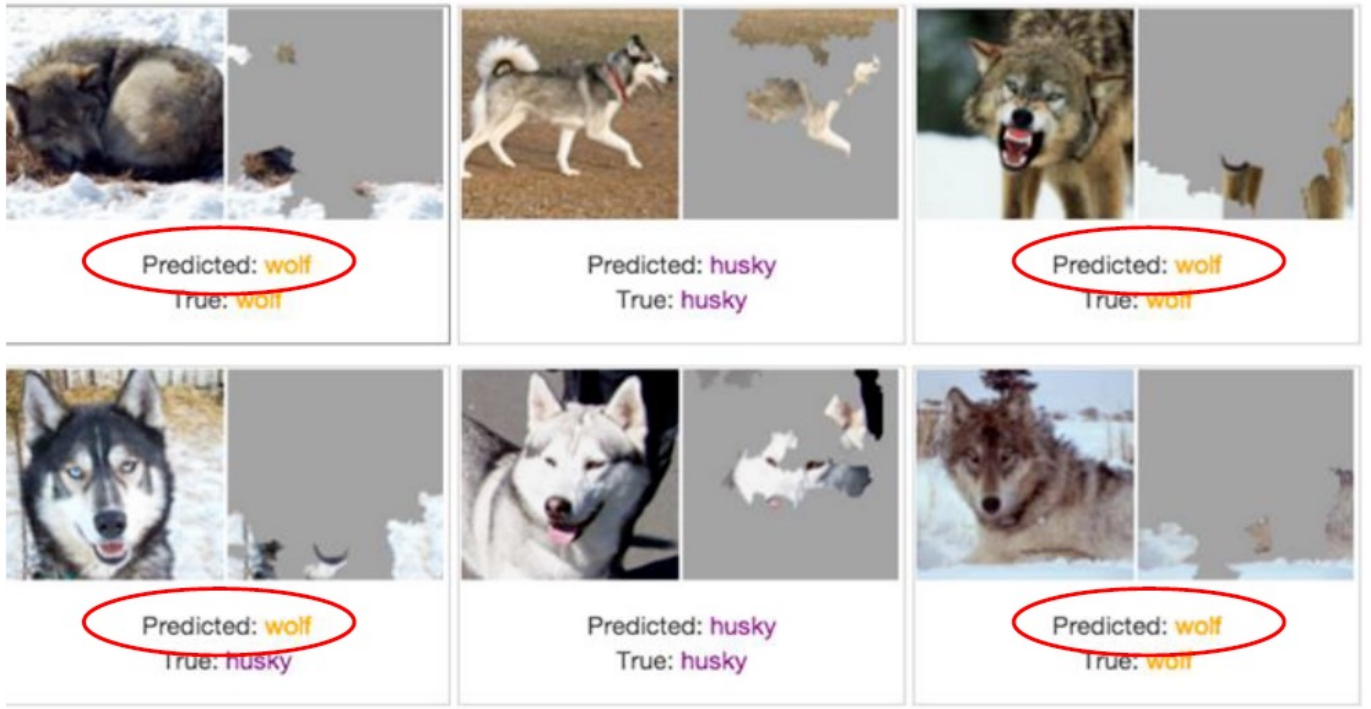
Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Image Classifier: Wolf vs Husky

Only 1 mistake!

40

Check Explanations with LIME



We've built a great snow detector…

41

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Explanations with LIME

(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 4: Explaining an image classification prediction made by Google's Inception neural network. The top 3 classes predicted are "Electric Guitar" ($p = 0.32$), "Acoustic guitar" ($p = 0.24$) and "Labrador" ($p = 0.21$)

42

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

SHapley **A**dditive Ex**P**lanations (SHAP)

- Resources:
  - Talk slides by Su-In Lee (U. Washington)
  - A unified approach to interpreting model predictions
    Lundberg and Lee; NeurIPS 2017

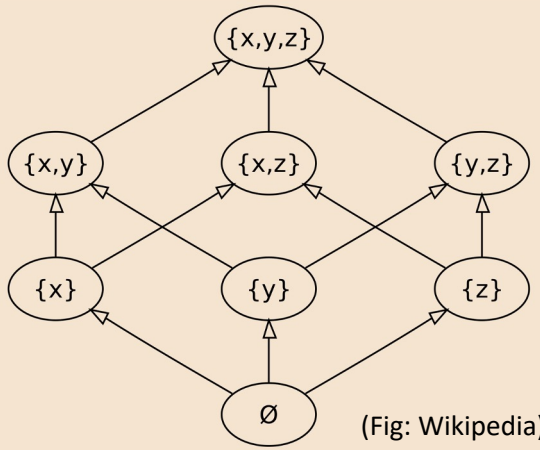
Cooperative game notation

- Set of *players* $D = \{1, \ldots, d\}$
- A *game* is given by specifying a value for every coalition $S \subseteq D$
- Mathematically represented by a *characteristic function*:

$$v: 2^D \mapsto \mathbb{R}$$

- Grand coalition value $v(D)$, null coalition $v(\emptyset)$, arbitrary coalition $v(S)$

2^D : Power set; i.e., set of all subsets.
For three variables:



(Fig: Wikipedia)

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Shapley Values: Introduction and Intuition

- Relies on Shapley values
  - "In cooperative game theory, the Shapley value is a method (solution concept) for fairly distributing the total gains or costs among a group of players who have collaborated." – Wikipedia
  - Players form coalitions based on their strategies and get "profit" based on their contributions.

- In ML, game = prediction, features = players.
  - For a certain input $\mathbf{x}$,
    - treat each feature dimension value $\mathbf{x}_i$ as a "player"
    - prediction of the model is the "payout"
    - "fairly" distribute the payout among the features (players)

Shapley Values: Introduction and Intuition

- Example: Predict an apartment's price given its features:
  - Area, floor, park-proximity, cats-allowed
  - Goal: Explain the importance/contribution of each feature to the price (or in comparison to an average price)

- Potential answer:
  - park-nearby: €30,000
  - area: €10,000
  - floor-2nd: €0
  - cat-banned: - €50,000.
  - When added, these can explain the price of the apartment in comparison to other apartments

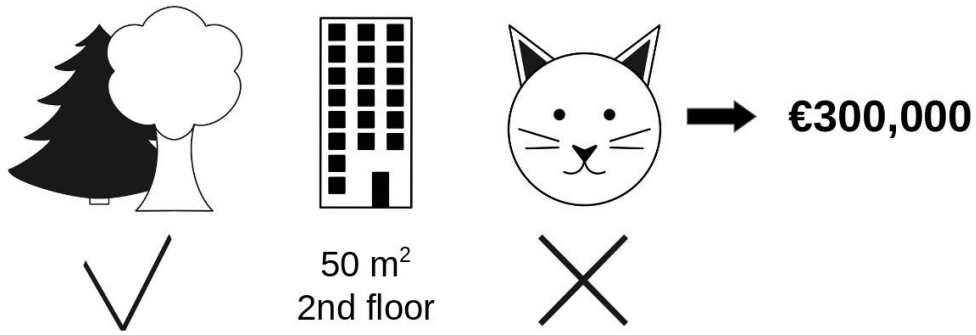


Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

Lloyd Shapley

- Won 2012 Nobel Memorial Prize in economics

Shapley Values: Introduction and Intuition

- Shapley value for a feature is the average contribution of a feature across all possible coalitions
- How to calculate the contribution of e.g. cat-banned?
  - Compare two coalitions with cat-banned and not cat-banned.
  - The difference is the contribution of cat-banned.
- Repeat this for more samples & coalitions.

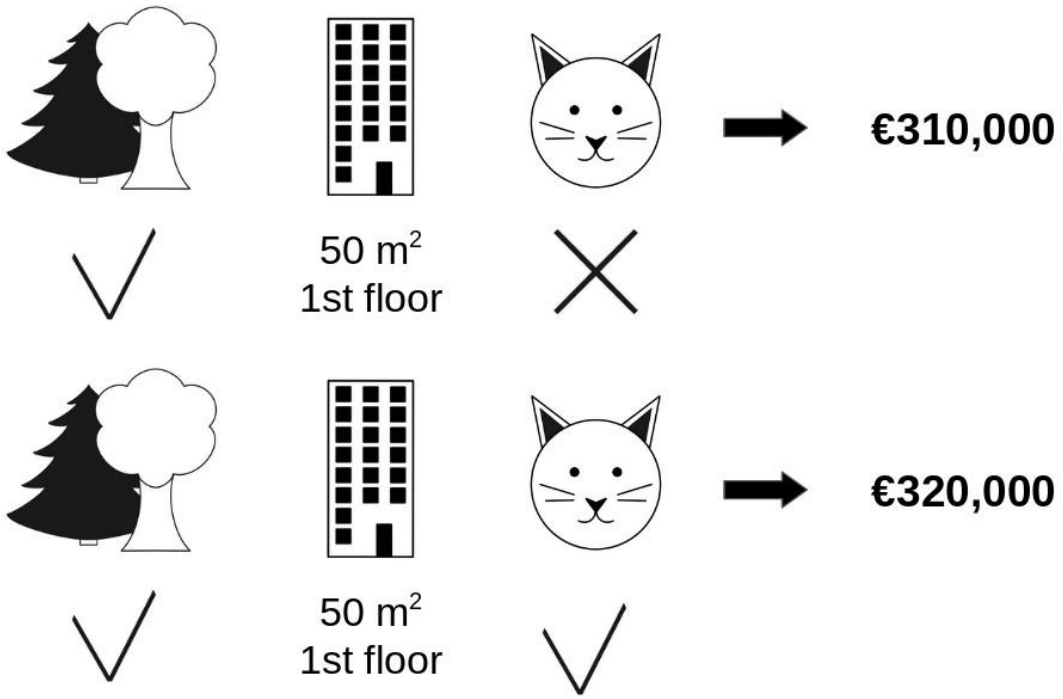
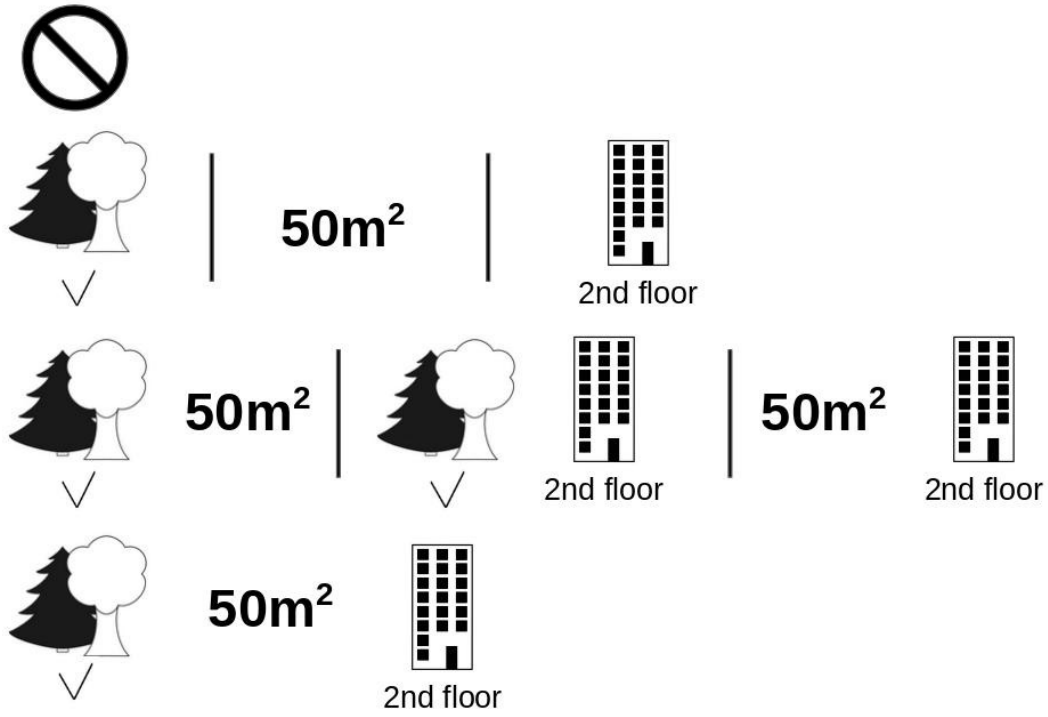


Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

Shapley Values: Introduction and Intuition

- Shapley value for j-th feature is:
  - The value of the j-th feature's contribution to the prediction of this particular instance compared to the average prediction for the dataset.



All possible (8) coalitions without cat-banned.

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

Shapley Value

The Shapley value of a feature value is its contribution to the payout, weighted and summed over all possible feature value combinations:

$$\phi_j(val) = \sum_{S \subseteq \{1,\ldots,p\} \setminus \{j\}} \frac{|S|!\,(p - |S| - 1)!}{p!} (val\,(S \cup \{j\}) - val(S))$$

where S is a subset of the features used in the model, $\mathbf{x}$ is the vector of feature values of the instance to be explained, and p is the number of features. $val_{\mathbf{x}}(S)$ is the prediction for feature values in set S that are marginalized over features X_C, which are all the features not included in S ($S \cap C = \emptyset$ and $S \cup C = \{1, \ldots, p\}$):

$$val_{\mathbf{x}}(S) = \int \hat{f}(x_1, \ldots, x_p) d\mathbb{P}_{X_C} - \mathbb{E}[\hat{f}(\mathbf{X})]$$

You actually perform multiple integrations for each feature that is not contained in S. A concrete example: The machine learning model works with 4 features X_1, X_2, X_3, and X_4, and we evaluate the prediction for the coalition S consisting of feature values x_1 and x_3:

$$val_{\mathbf{x}}(S) = val_{\mathbf{x}}(\{1,3\}) = \int_{\mathbb{R}} \int_{\mathbb{R}} \hat{f}(x_1, X_2, x_3, X_4) d\mathbb{P}_{X_2 X_4} - \mathbb{E}[\hat{f}(\mathbf{X})]$$

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley value is the only attribution method that satisfies the properties **Efficiency**, **Symmetry**, **Dummy**, and **Additivity**, which together can be considered a definition of a fair payout.

Efficiency The feature contributions must add up to the difference of prediction for $\mathbf{x}$ and the average.

$$\sum_{j=1}^{p} \phi_j = \hat{f}(\mathbf{x}) - \mathbb{E}[\hat{f}(\mathbf{X})]$$

Symmetry The contributions of two feature values j and k should be the same if they contribute equally to all possible coalitions. If

$$val(S \cup \{j\}) = val(S \cup \{k\})$$

for all

$$S \subseteq \{1, \ldots, p\} \setminus \{j, k\}$$

then

$$\phi_j = \phi_k$$

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

Dummy A feature j that does not change the predicted value – regardless of which coalition of feature values it is added to – should have a Shapley value of 0. If

$$val(S \cup \{j\}) = val(S)$$

for all

$$S \subseteq \{1, \ldots, p\}$$

then

$$\phi_j = 0$$

Additivity For a game with combined payouts $val + val^+$ the respective Shapley values are as follows:

$$\phi_j + \phi_j^+$$

Suppose you trained a random forest, which means that the prediction is an average of many decision trees. The Additivity property guarantees that for a feature value, you can calculate the Shapley value for each tree individually, average them, and get the Shapley value for the feature value for the random forest.

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

Estimating Shapley Values

- Calculations with all possible coalitions can be expensive for large dimensions
- Alternative: Monte-Carlo Sampling

"Explaining Prediction Models and Individual Predictions with Feature Contributions." *2014.*

$$\hat{\phi}_j = \frac{1}{M} \sum_{m=1}^{M} \left(\hat{f}(\mathbf{x}_{+j}^{(m)}) - \hat{f}(\mathbf{x}_{-j}^{(m)}) \right)$$

where $\hat{f}(\mathbf{x}_{+j}^{(m)})$ is the prediction for $\mathbf{x}$, but with a random number of feature values replaced by feature values from a random data point $\mathbf{z}$, except for the respective value of feature j. The feature vector $\mathbf{x}_{-j}^{(m)}$ is almost identical to $\mathbf{x}_{+j}^{(m)}$, but the value $x_j^{(m)}$ is also taken from the sampled $\mathbf{z}$. Each

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

From Shapley values to SHAP

SHAP: Lundberg, Scott M., and Su-In Lee. 2017. "A Unified Approach to Interpreting Model Predictions." In NeurIPS.

- Shapley values were introduced in 2011 and 2014:
  - Štrumbelj, Erik, and Igor Kononenko. 2011. "A General Method for Visualizing and Explaining Black-Box Regression Models.
  - Štrumbelj, Erik, and Igor Kononenko. 2014. "Explaining Prediction Models and Individual Predictions with Feature Contributions."
- However, these approaches did not become popular.
- SHAP simplified Shapley value estimation as a linear regression problem,
  - connecting Shapley values to LIME and other post-hoc methods
  - providing Shapley theory
  - introduced new ways of estimating Shapley values
  - introduced how Shapley values can be estimated for text and image models

SHAP

SHAP: Lundberg, Scott M., and Su-In Lee. 2017. “A Unified Approach to Interpreting Model Predictions.” In NeurIPS.

table is that the Shapley value explanation is represented as an additive feature attribution method, a linear model. That view connects LIME and Shapley values. SHAP specifies the explanation as:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$$

where g is the explanation model, $\mathbf{z}' = (z'_1, \ldots, z'_M)^T \in \{0,1\}^M$ is the coalition vector, M is the maximum coalition size, and $\phi_j \in \mathbb{R}$ is the feature attribution for feature j, the Shapley values.

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

SHAP

The KernelSHAP estimation has five steps:

- Sample coalition vectors $\mathbf{z}'_k \in \{0,1\}^M, \quad k \in \{1,\ldots,K\}$ (1 = feature present in coalition, 0 = feature absent).
- Get prediction for each $\mathbf{z}'_k$ by first converting $\mathbf{z}'_k$ to the original feature space and then applying model $\hat{f}$: $\hat{f}(h_{\mathbf{x}}(\mathbf{z}'_k))$.
- Compute the weight for each coalition $\mathbf{z}'_k$ with the SHAP kernel.
- Fit weighted linear model.
- Return Shapley values ϕ_k, the coefficients from the linear model.

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

SHAP

We have the data, the target and the weights; Everything we need to build our weighted linear regression model:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$$

We train the linear model g by optimizing the following loss function L:

$$L(\hat{f}, g, \pi_{\mathbf{x}}) = \sum_{\mathbf{z}' \in \mathbf{Z}} [\hat{f}(h_{\mathbf{x}}(\mathbf{z}')) - g(\mathbf{z}')]^2 \pi_{\mathbf{x}}(\mathbf{z}')$$

where $\mathbf{Z}$ is the training data. This is the good old boring sum of squared errors that we usually optimize for linear models. The estimated coefficients of the model, the ϕ_j's, are the Shapley values.

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

SHAP

the features. To achieve Shapley compliant weighting, Lundberg and Lee (2017) proposed the SHAP kernel:

$$\pi_{\mathbf{x}}(\mathbf{z}') = \frac{(M-1)}{\binom{M}{|\mathbf{z}'|}|\mathbf{z}'|(M-|\mathbf{z}'|)}$$

Here, M is the maximum coalition size and $|\mathbf{z}'|$ the number of present features in instance $\mathbf{z}'$. Lundberg and Lee (2017) show that linear regression with this kernel weight yields Shapley values. If you were to use the SHAP kernel with LIME on the coalition data, LIME would also estimate Shapley values!

Fig & Material: https://christophm.github.io/interpretable-ml-book/shapley.html

Other SHAP variants

- TreeSHAP for tree-based learners

- Permutation SHAP:
  - Sample from coalitions by incrementally creating permutations

SHAP: Strengths and Limitations

Strengths

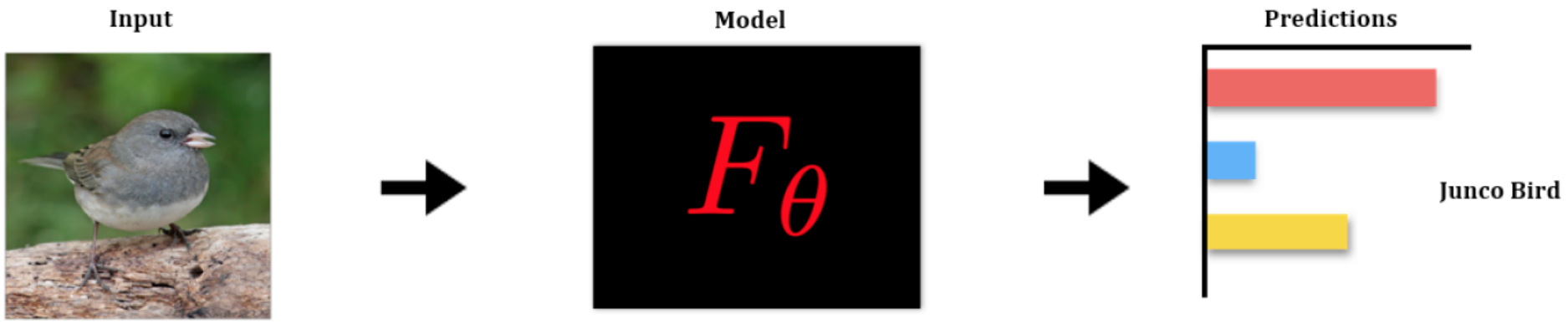
- Fair distribution of prediction
- Allows contrastive explanations
  - You can select a subset of data and compare a prediction in contrast to the predictions on the subset
- Solid theory

Limitations

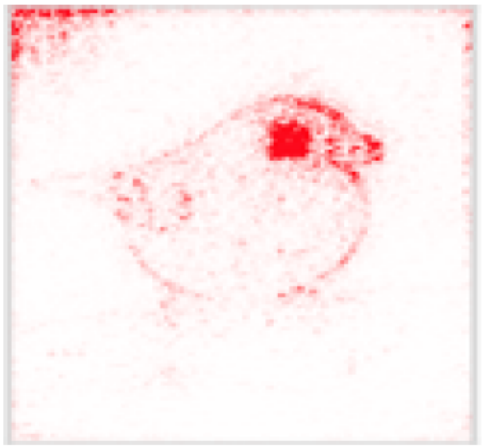
- Only approximations are feasible/practical
- Easy to misinterpret as the contribution of a feature if removed from the dataset.
- Wrong method if “sparse” explanations (few features) are needed
- Sensitive to correlation between features

Feature Attribution (Saliency) Methods

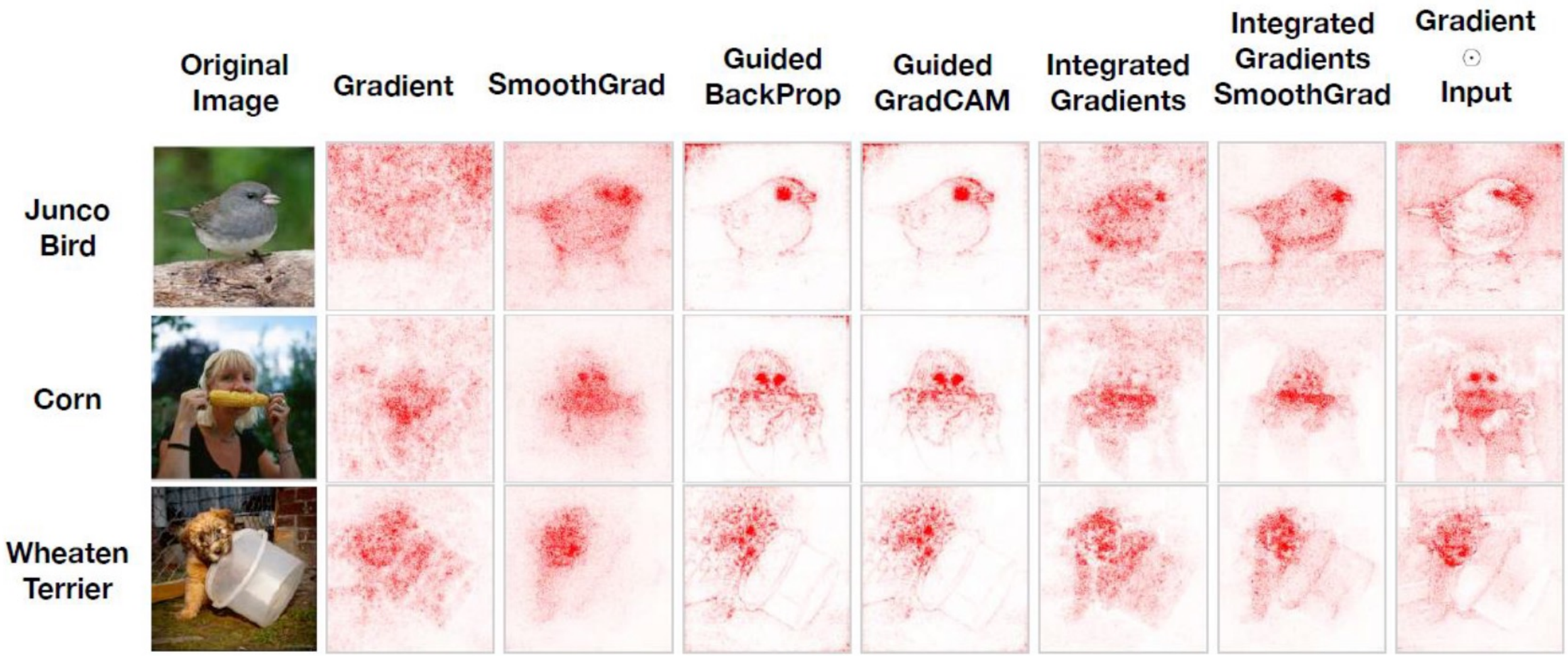
Saliency Maps



What parts of the input are most relevant for the model's prediction: 'Junco Bird'?



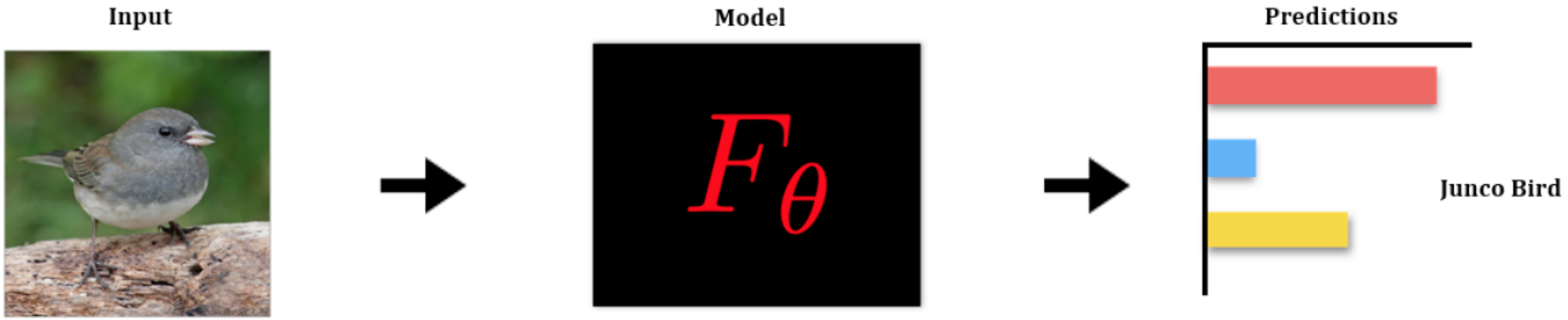
3



4

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Saliency Maps for Deep Neural Networks

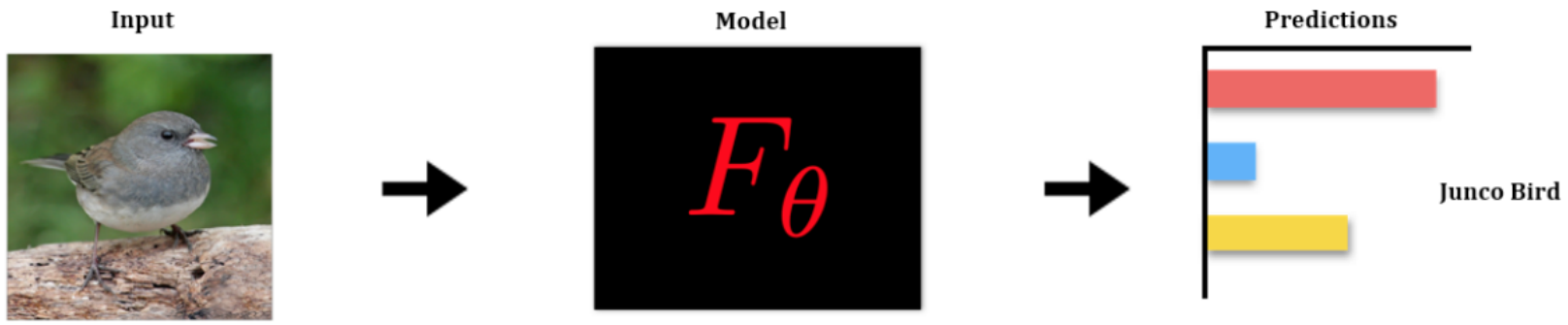


$$F : \mathbb{R}^d \rightarrow \mathbb{R}^c$$ **Model**

$$F_i : \mathbb{R}^d \rightarrow \mathbb{R}$$ **class specific logit**

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Input Gradient



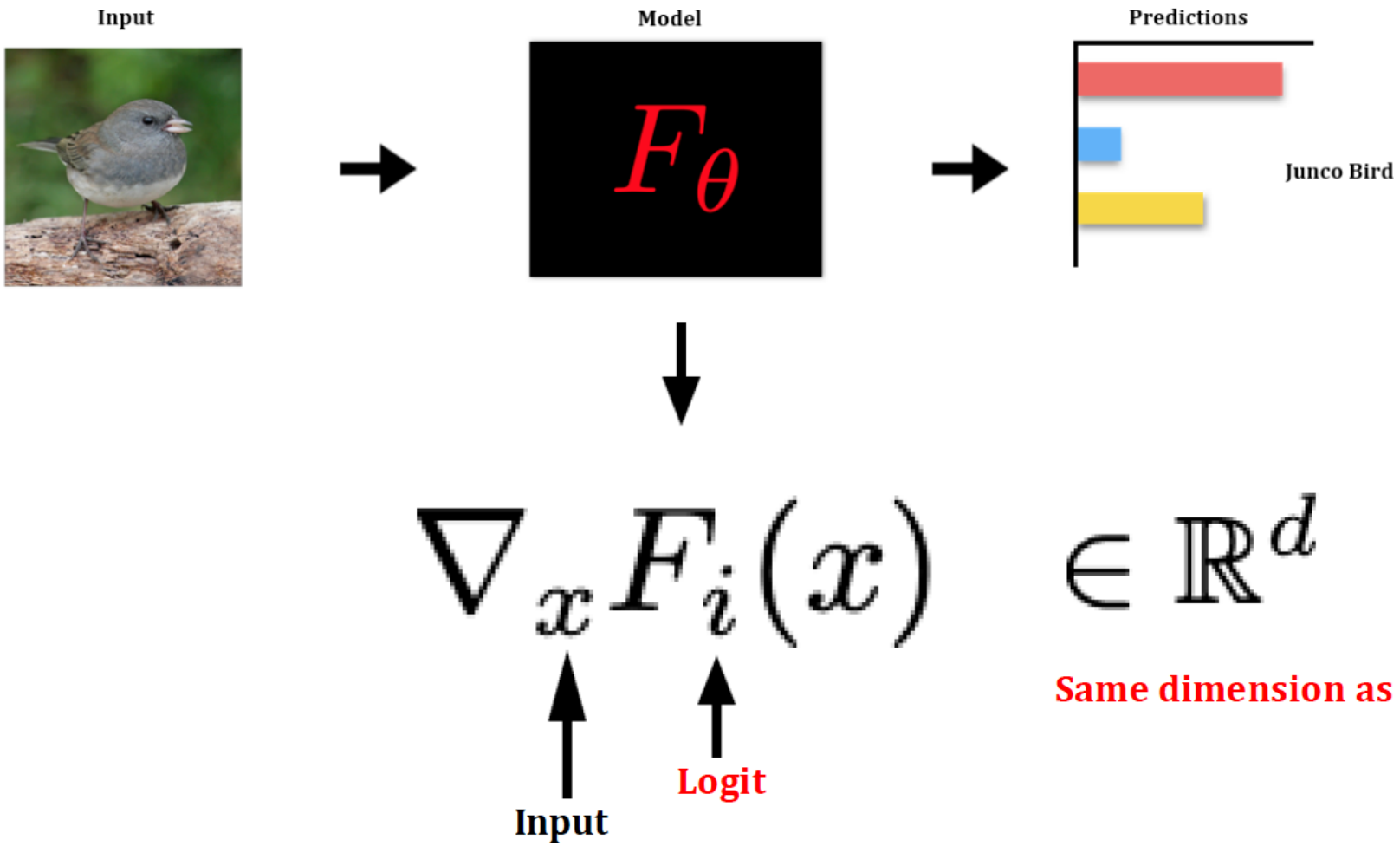
What's the contribution of a pixel in input image to the prediction "Junco bird"

Solution: Compute the gradient of the output logit w.r.t. the pixel

6

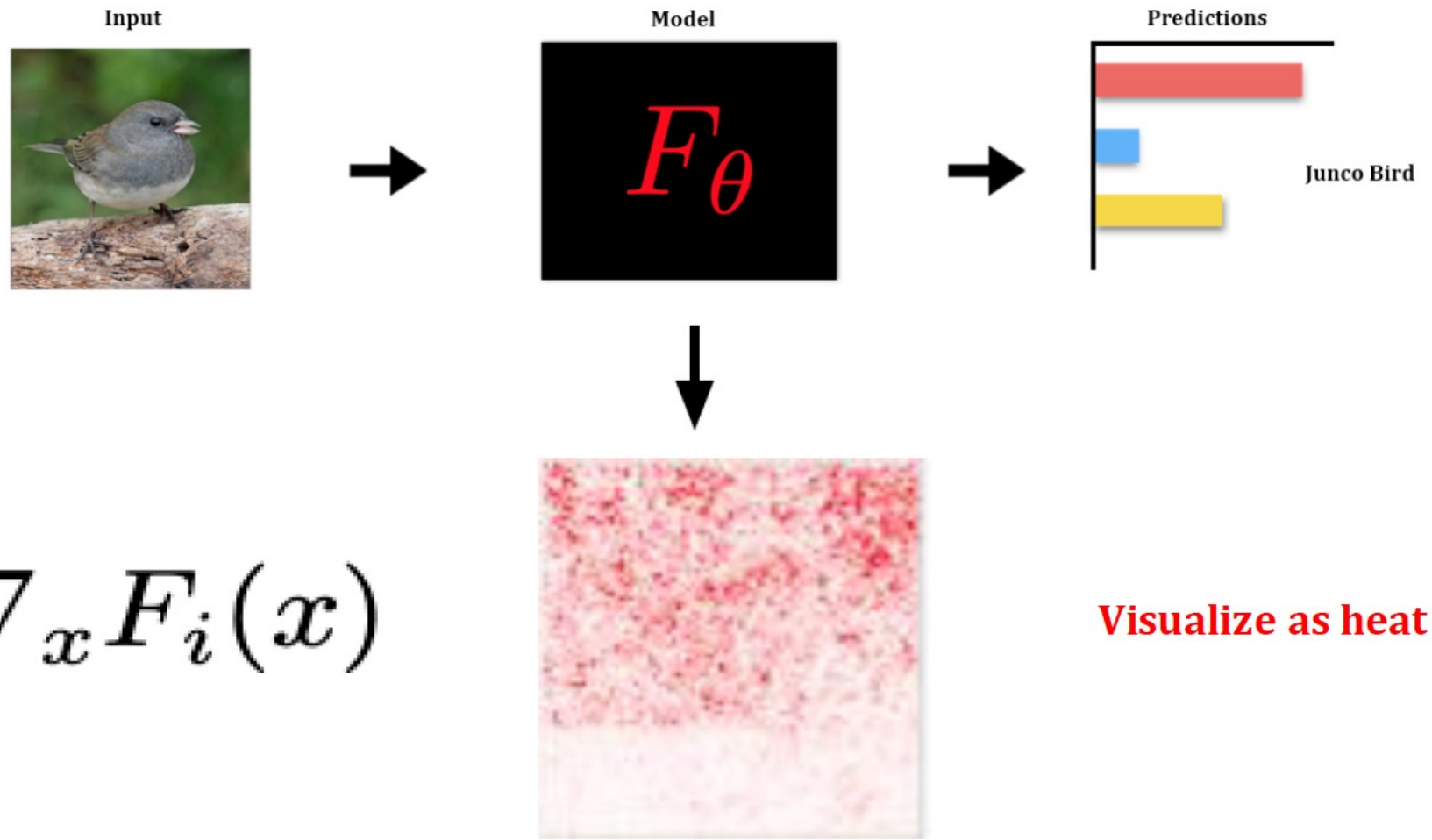
Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Input Gradient



Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Input Gradient

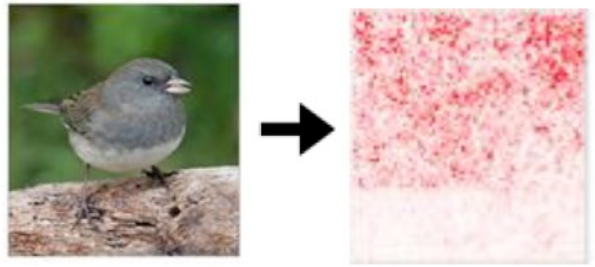


$$\nabla_x F_i(x)$$

Visualize as heat map

8

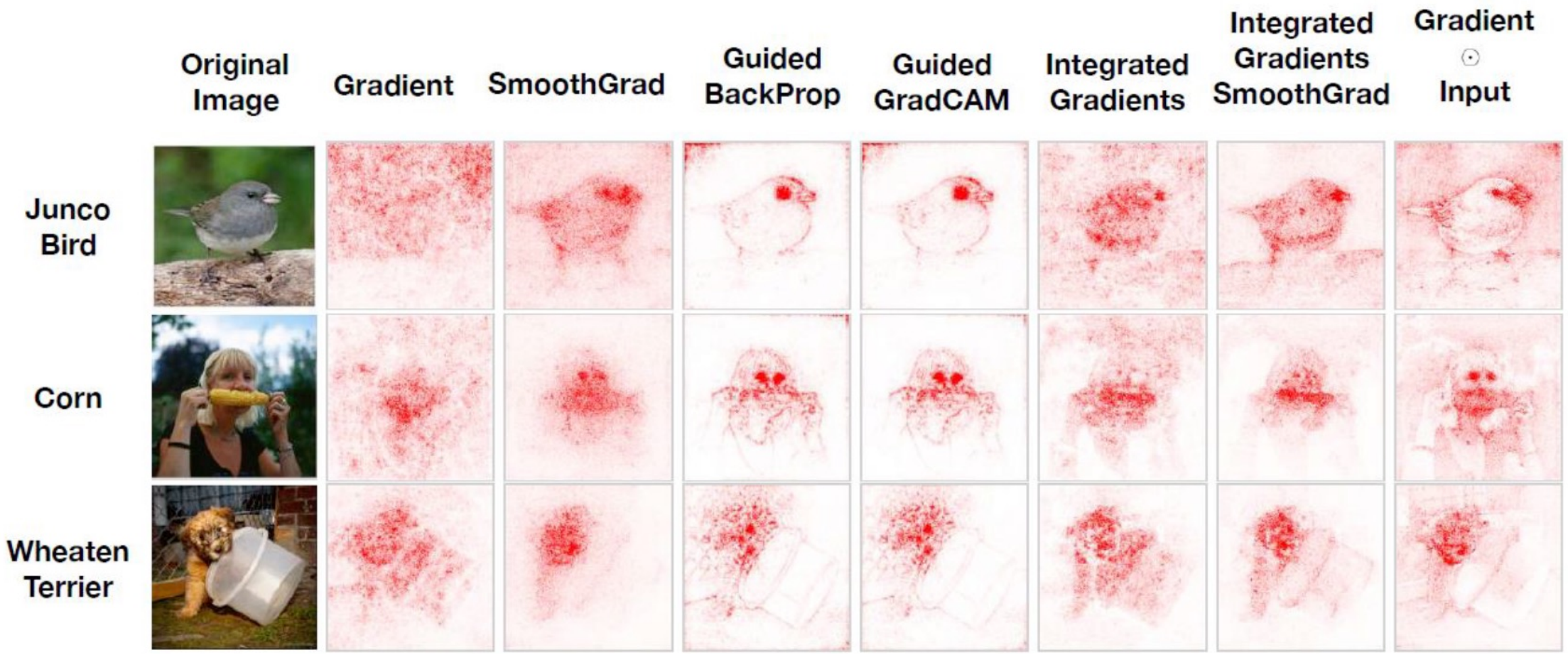
Beyond Input Gradients



Input gradients capture sensitivity to individual pixels/features well, but …
Raw gradients are visually noisy

Many improvements proposed in literature

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

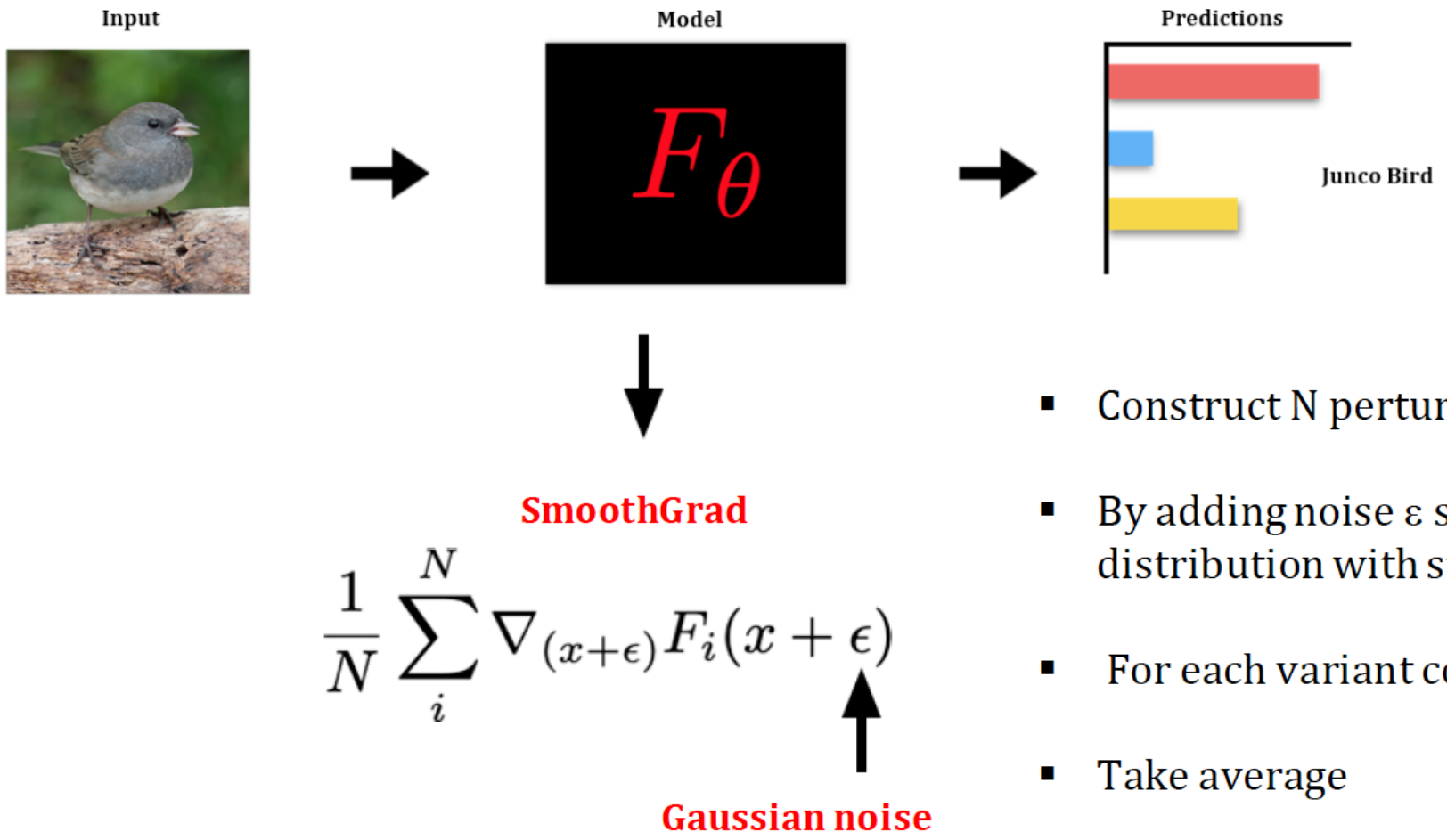


10

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

SmoothGrad

Smilkov, D., Thorat, N., Kim, B., Viégas, F., & Wattenberg, M. (2017). Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.

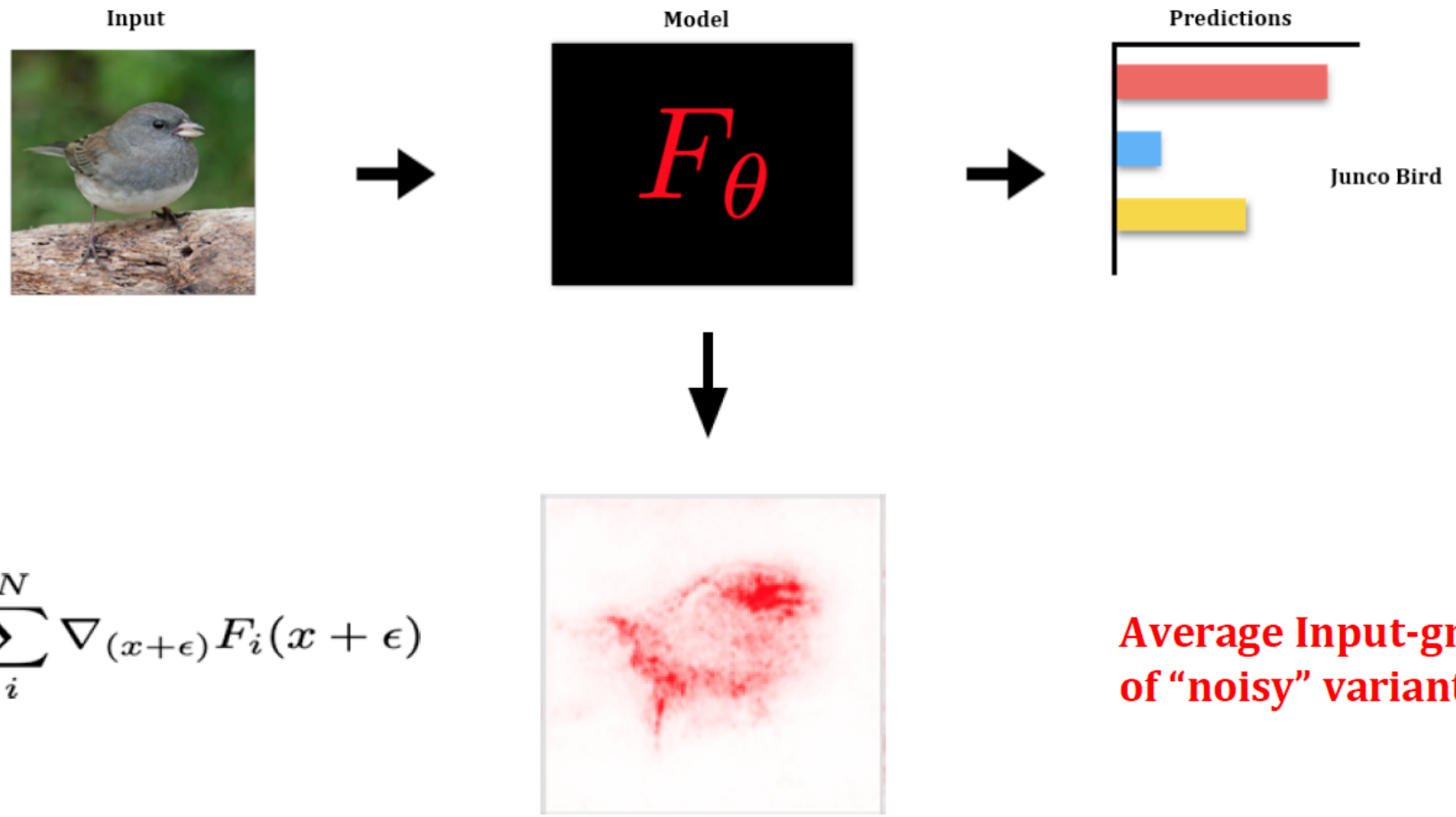


- Construct N perturbations of input x
- By adding noise ε sampled from Gaussian distribution with standard deviation σ
- For each variant compute input gradient
- Take average

11

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

SmoothGrad



$$\frac{1}{N}\sum_{i}^{N}\nabla_{(x+\epsilon)}F_i(x+\epsilon)$$

Average Input-gradients of "noisy" variants

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

SmoothGrad: Effect of noise

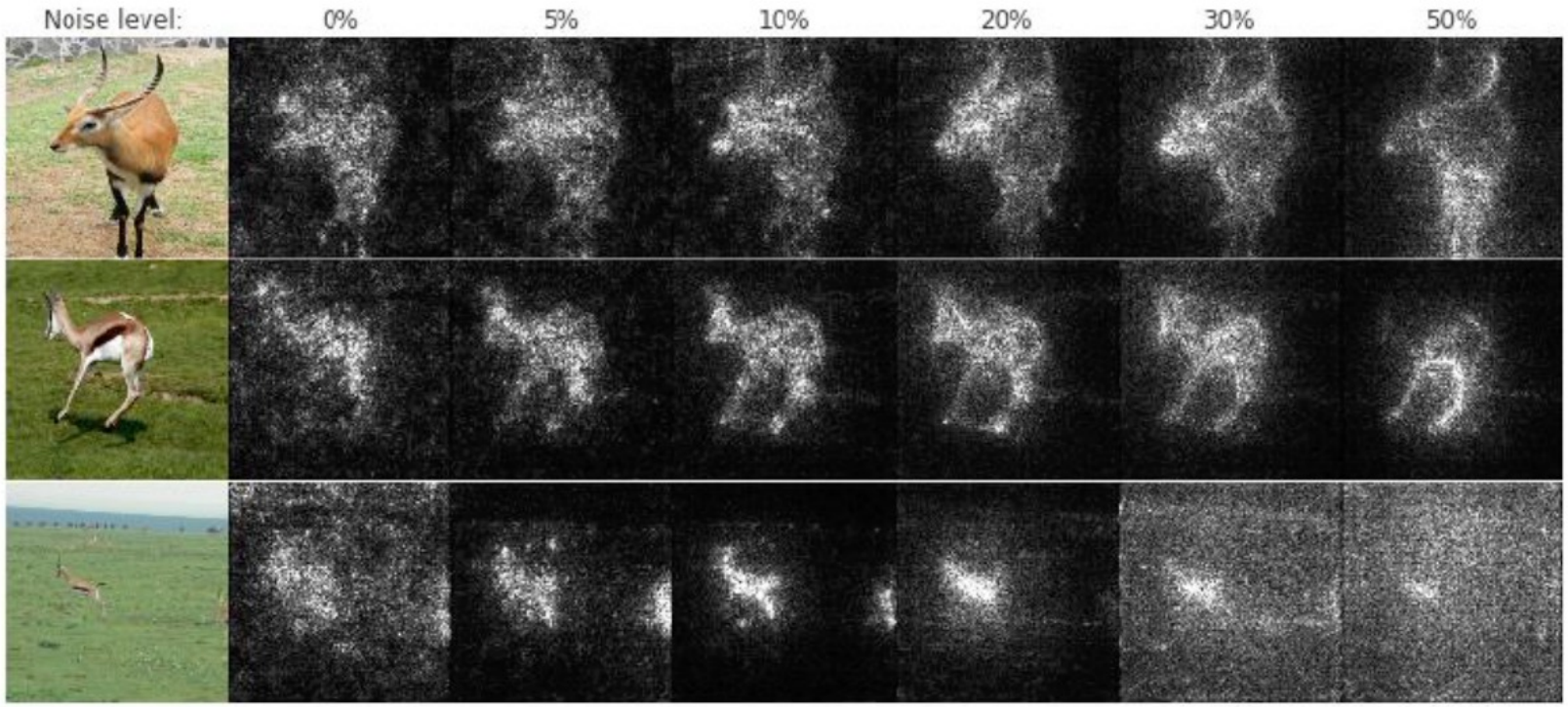
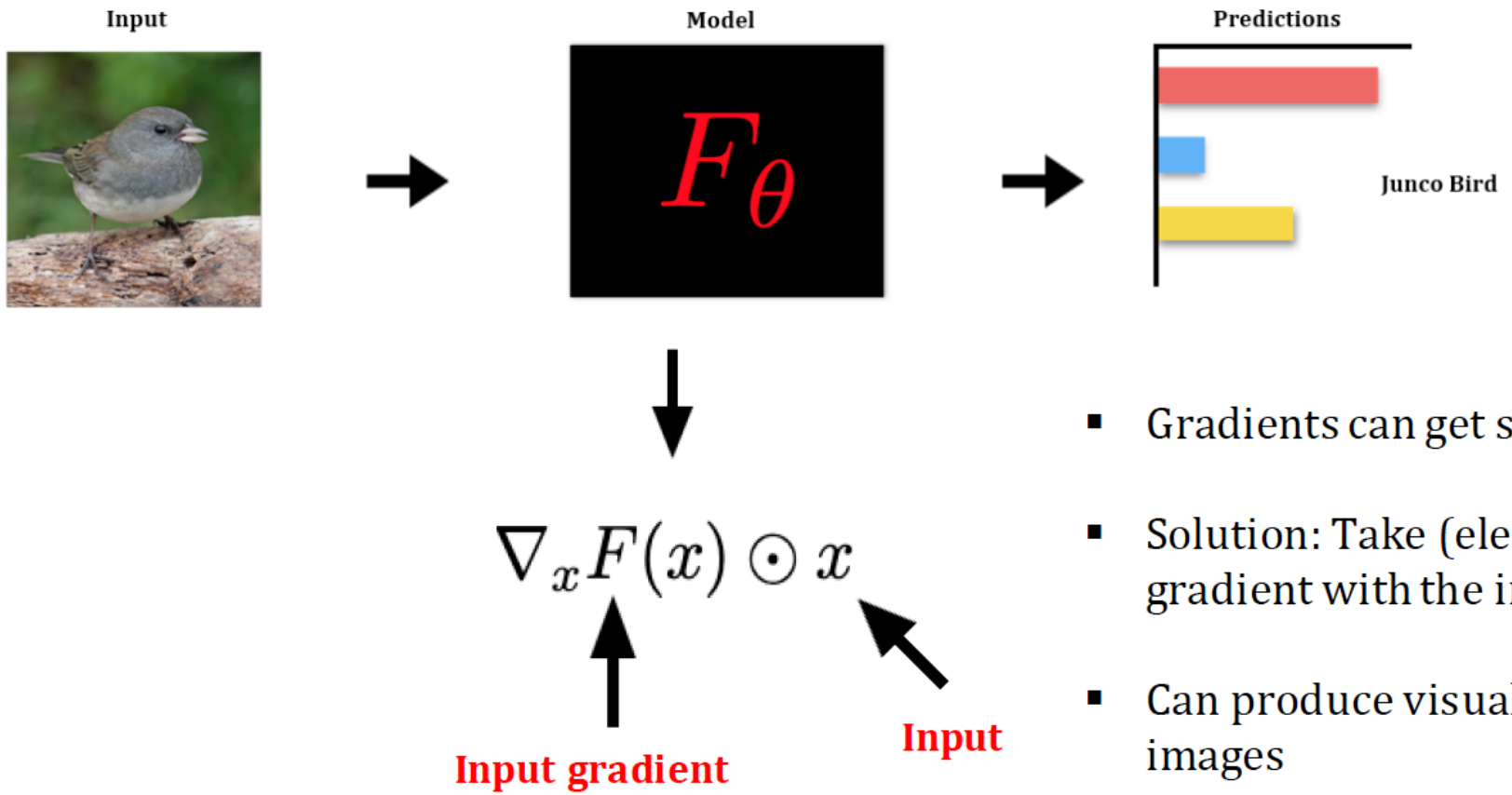


Figure 3. Effect of noise level (columns) on our method for 5 images of the gazelle class in ImageNet (rows). Each sensitivity map is obtained by applying Gaussian noise $\mathcal{N}(0, \sigma^2)$ to the input pixels for 50 samples, and averaging them. The noise level corresponds to $\sigma/(x_{max} - x_{min})$.

13

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Gradient-Input

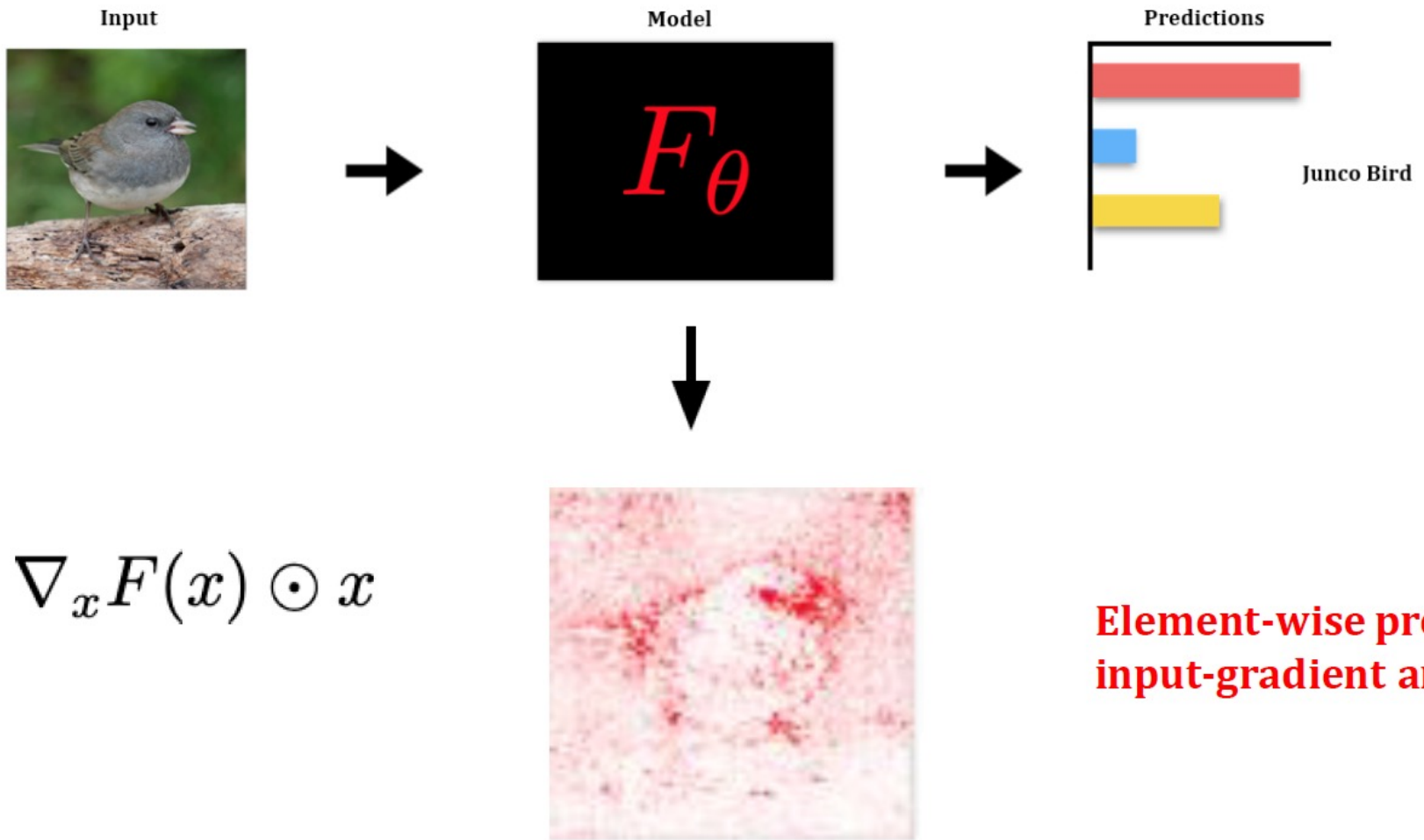


- Gradients can get saturated
- Solution: Take (element-wise) product of the gradient with the input itself
- Can produce visually simpler/sharper images
- Can be used in conjunction with any method

14

Slide: Uni of Pennsylvania, Trustworthy ML - CIS 7000

Gradient-Input



$$\nabla_x F(x) \odot x$$

Element-wise product of input-gradient and input

15