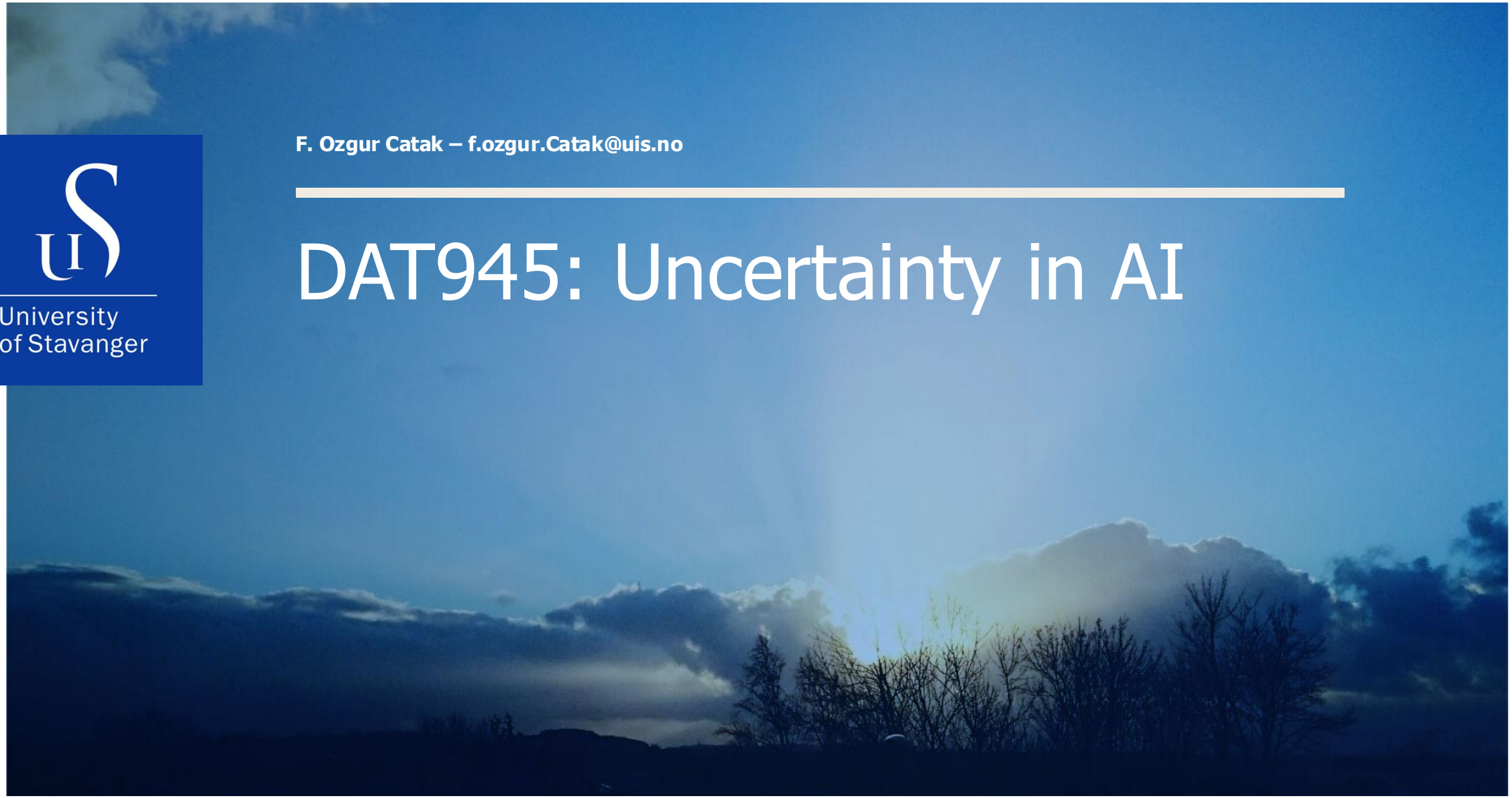
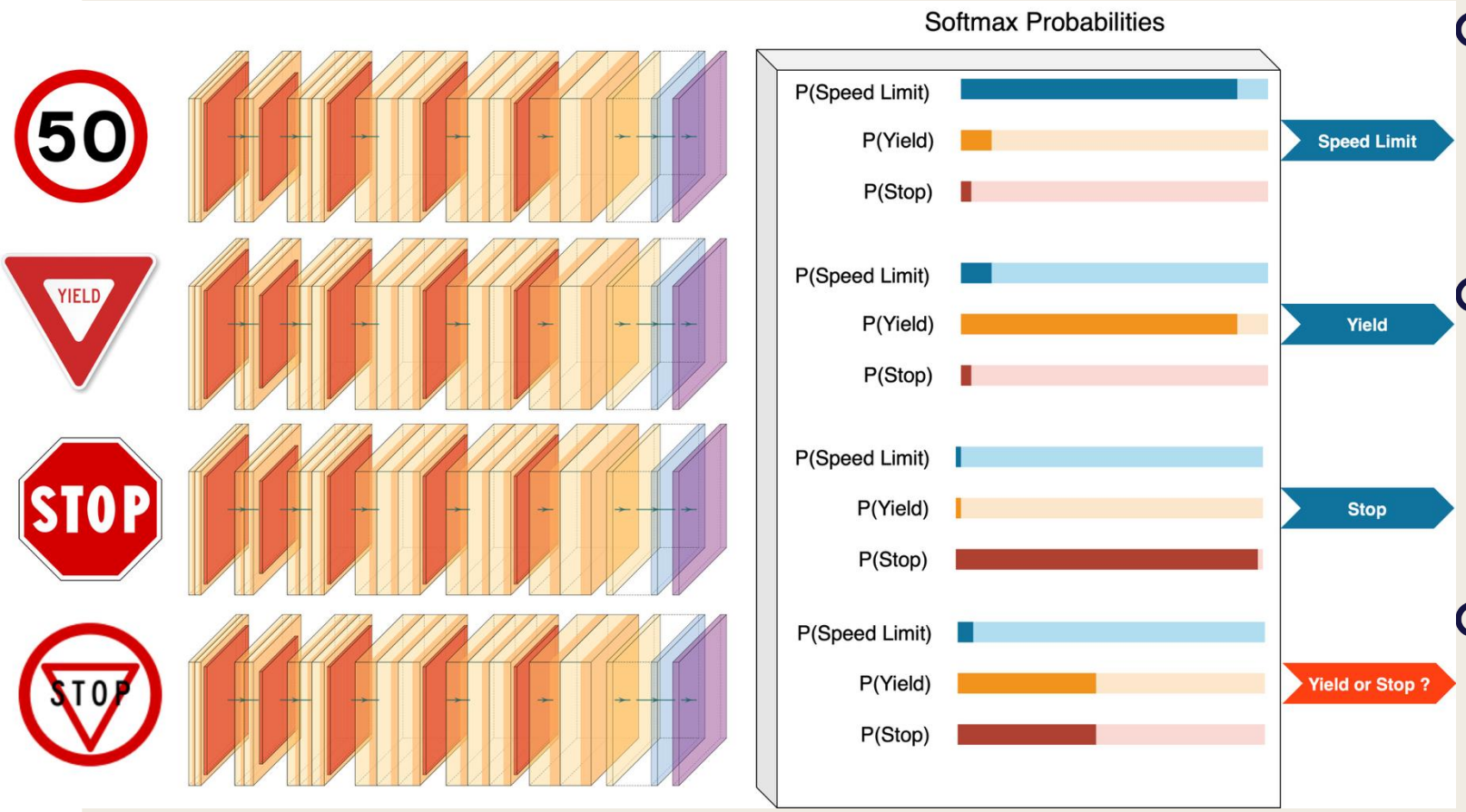
F. Ozgur Catak – f.ozgur.Catak@uis.no

University of Stavanger

DAT945: Uncertainty in AI



Uncertainty in AI



- The computer saying, "**I'm doing my best, but I can't be 100% sure about everything.**"
- This honesty about not being perfect is important, mainly when the computer is used in things like self-driving cars.
- It helps us understand when the computer might need extra help or when we should be more cautious.

Uncertainty in AI

- **Uncertainty**: could potentially lead to unreliable (e.g., unsafe) behaviors of Apps,
  - if such uncertainty is not properly dealt with.

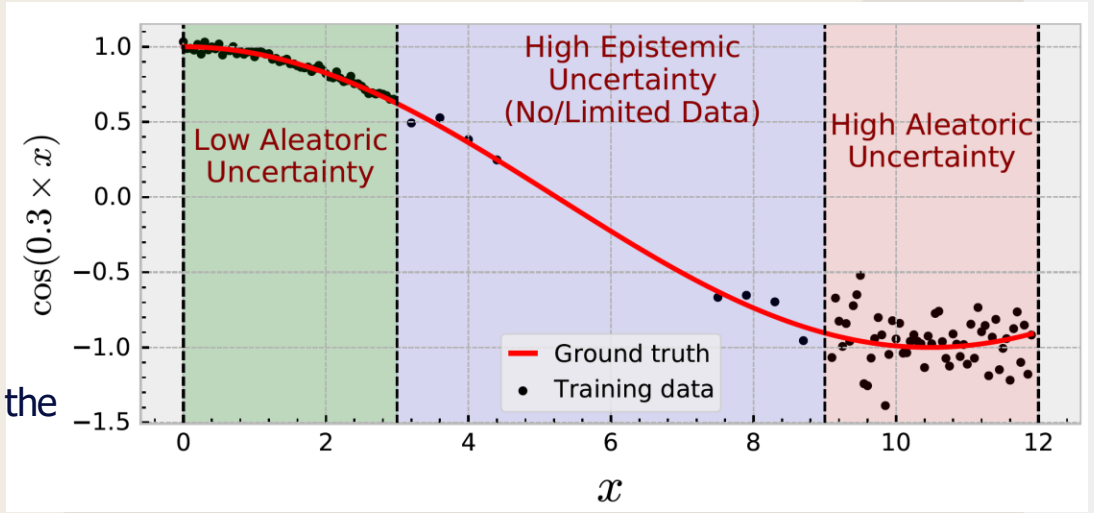
- **DNNs**: black box models (multilayered nonlinear structures)
  - non-transparent
  - Predictions not identifiable by humans

- **Apps**: black-box DL models have been used to make critical predictions

Uncertainty Sources

Epistemic Uncertainty

- Uncertainty in models (not in data)
- "Epistemic": Greek "episteme" = knowledge
- **Reducible: more data helps**
- There are two types Epistemic uncertainty
  - Model Uncertainty
    - Neural network model's neuron weights are not optimized well for the domain
  - Approximation uncertainty
    - Model structure (# of layers, activation functions (ReLU, Tanh, Sigmoid etc), optimizer functions (SGD, RMSProp, Adam, Adamax etc)



Aleatoric Uncertainty

- because of the noise input dataset.
- "**Aleatoric**": Latin "aleator" = dice players
- Noise in the training data
- **Stochastic**, **irreducible** in data (noise)
  - **More data doesn't help**
- For instance; noise sensor readings for a CPS application

Role of Probabilistic Models in Uncertainty Quantification

- **Capturing Uncertainty**: Probabilistic models inherently capture the uncertainty in predictions by providing a distribution of possible outcomes rather than a single point estimate. This allows for a more comprehensive understanding of the range of potential results and their likelihood.

- **Predictive Distributions**: Probabilistic models generate predictive distributions instead of deterministic outputs. For instance,
  - in regression, instead of predicting a single value for a response variable, the model predicts a distribution over possible values, capturing both the mean and the variance.

MC dropout/Ensemble Predictions

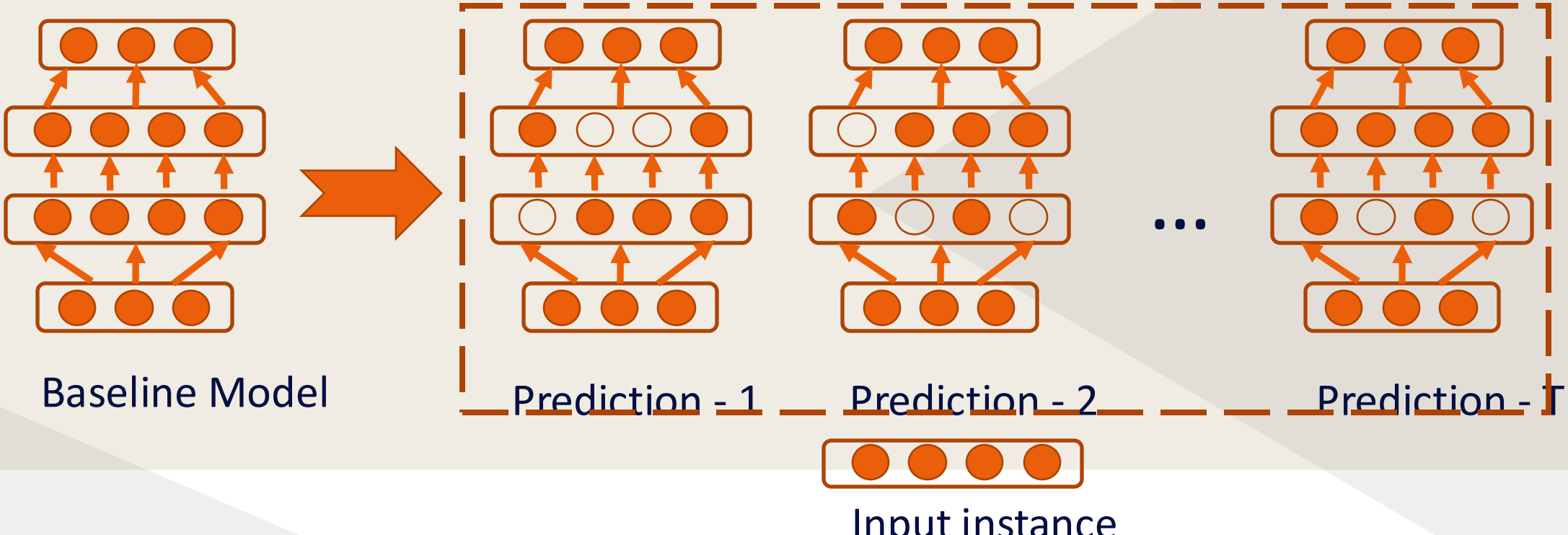
Probabilistic model instead of deterministic – Jupyter Notebook

Important parameters:

- **T**: the number of predictions
- **p**: dropout ratio

$$\begin{bmatrix} y_{11} & y_{21} & y_{31} & y_{41} & \vdots & y_{T1} \\ y_{12} & y_{22} & y_{32} & y_{42} & \vdots & y_{T2} \\ y_{13} & y_{23} & y_{33} & y_{43} & \vdots & y_{T3} \end{bmatrix}$$

Softmax output vectors for each prediction (i.e. 3 classes, T predictions)



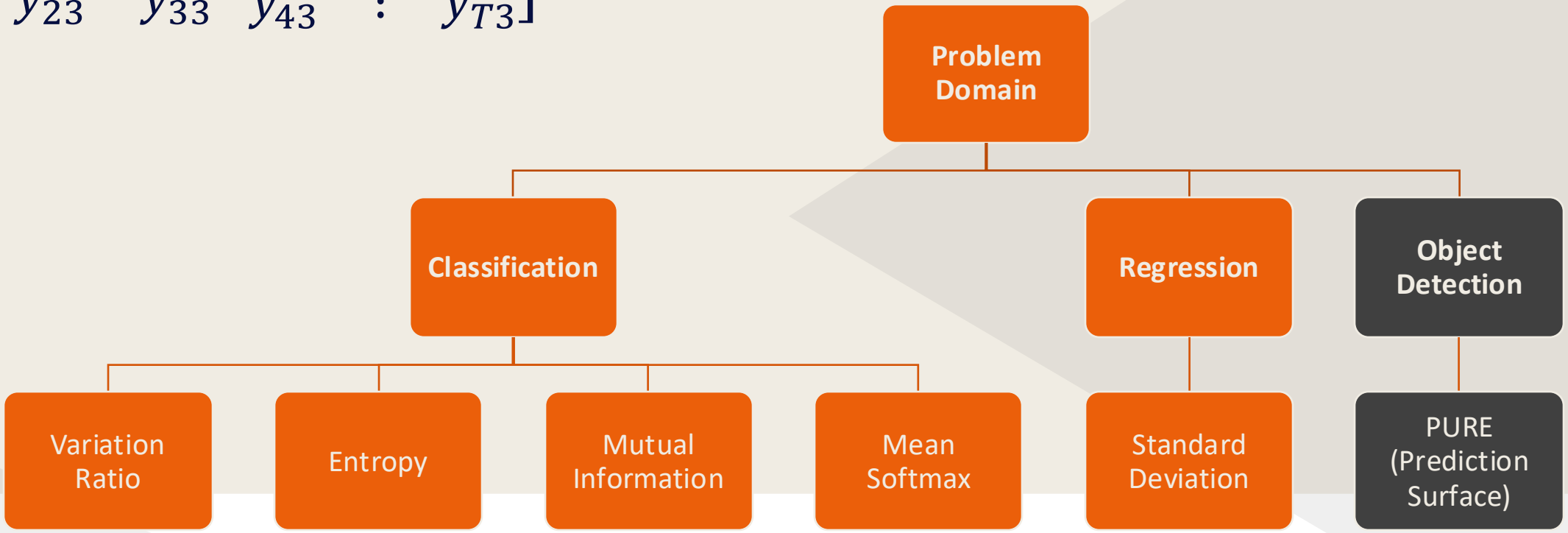
Uncertainty Quantification Metrics

Classification output

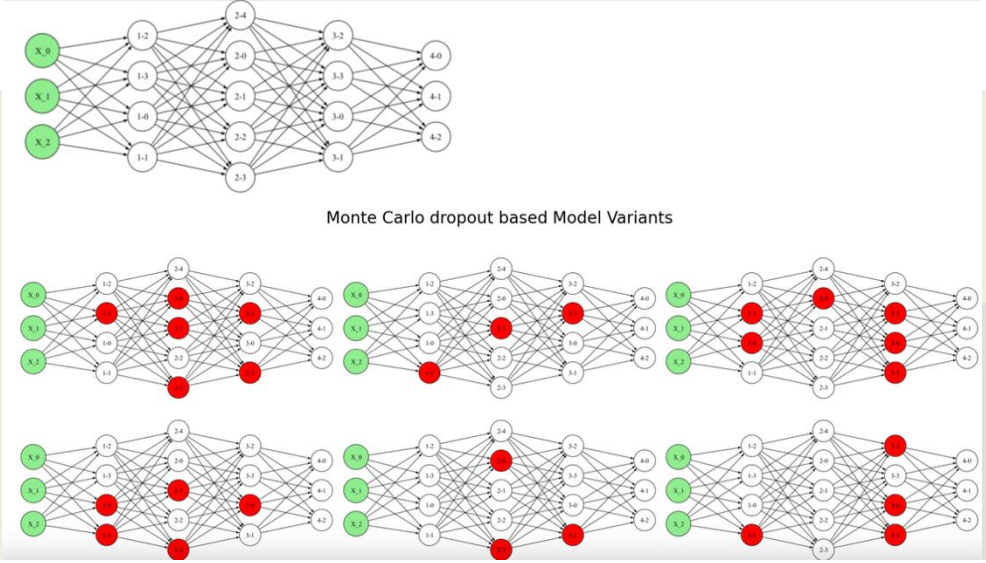
$$\begin{bmatrix} y_{11} & y_{21} & y_{31} & y_{41} & \vdots & y_{T1} \\ y_{12} & y_{22} & y_{32} & y_{42} & \vdots & y_{T2} \\ y_{13} & y_{23} & y_{33} & y_{43} & \vdots & y_{T3} \end{bmatrix}$$

Regression output

$$[y_1 \quad y_2 \quad y_3 \quad \dots \quad y_T]$$



MC Dropouts



During testing, the dropout rate is set to other than 0, and the model's prediction is obtained by averaging the output of T stochastic forward passes through the network:

$$y = \frac{1}{T}\sum_{t=1}^{T} f(x, \theta_t)$$

where θ_t is a set of randomly sampled weights for the t-th forward pass.

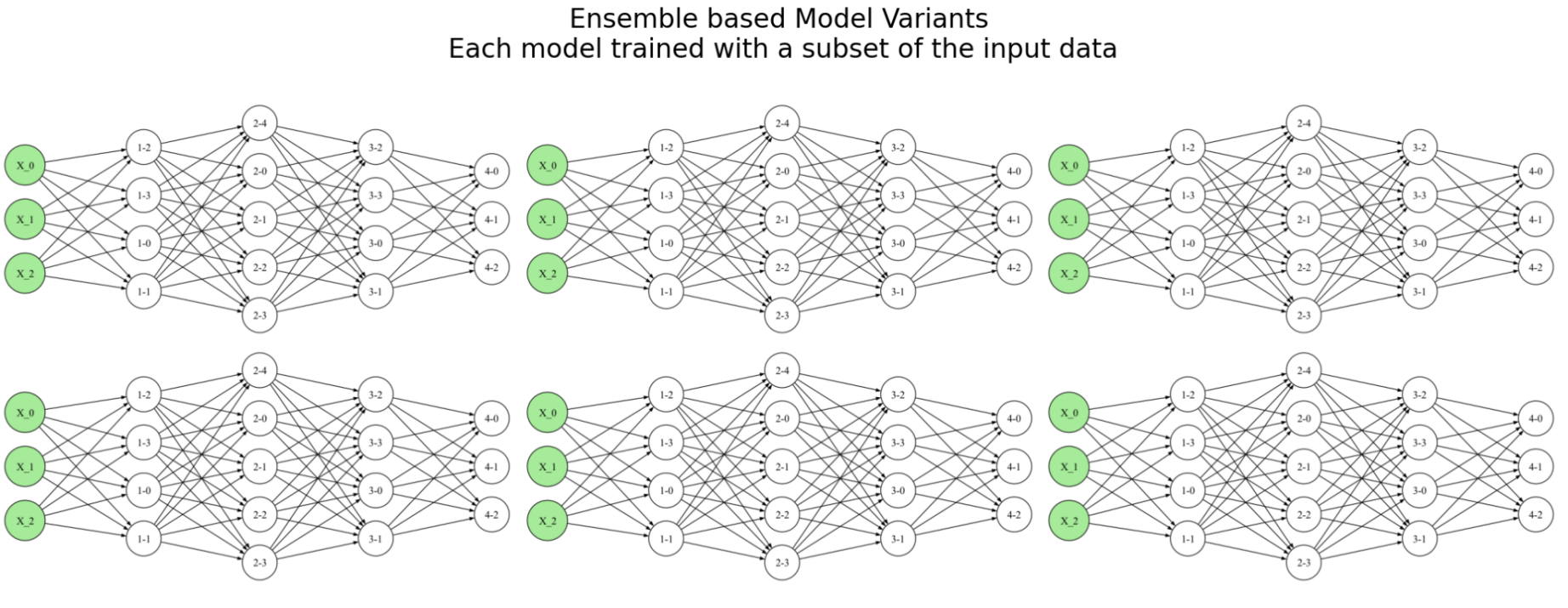
The predictive distribution can be approximated by estimating the mean and variance of the output over multiple samples of θ with dropout applied during testing. This can be represented mathematically as follows:

$$E_{p(y|x)}[\hat{y}] \approx \frac{1}{T}\sum_{t=1}^{T} f(x, \theta_t)$$

$$Var_{p(y|x)}[\hat{y}] \approx \frac{1}{T}\sum_{t=1}^{T} (f(x, \theta_t) - E_{p(y|x)}[\hat{y}])^2,$$

where $\mathbb{E}_{p(y|x)}[\hat{y}]$ and $Var_{p(y|x)}[\hat{y}]$ are the mean and variance of the predictive distribution, respectively.

Ensemble Methods



- Quantifying the uncertainty in the ensemble models uses a metrics of the softmax outputs produced by the individual models.

Quantifiers

1. **Variation Ratio:** The Variation Ratio (VR) is a measure of the probability mass concentration in the predictive distribution. It is defined as the difference between the maximum predicted probability and the average predicted probability:

$$\text{VR} = 1 - \max_y p(y|x, \theta)$$

Suppose we have a classification task with three possible classes: $y = 0, 1, 2$. We have trained a Monte Carlo Dropout neural network with $T = 5$ dropout samples per input, and we obtain the following softmax outputs for an input x:

Sample t	Class 0	Class 1	Class 2	Most Probable Class
1	0.1	0.4	0.5	2
2	0.2	0.3	0.5	2
3	0.15	0.35	0.5	2
4	0.45	0.3	0.25	0
5	0.05	0.6	0.35	1

To compute the Variation Ratio, we first compute the most probable class for each dropout sample. In this example, the most probable class for each sample is:

Next, we compute the frequency of the most probable class across all T samples. In this case, the most probable class is 2 for three out of the five samples, so the Variation Ratio is:

$$\text{VR} = 1 - 3/5 = 0.4$$

This means that there is a 40% chance that the most probable class is incorrect. A lower Variation Ratio indicates greater confidence in the model's predictions.

Quantifiers

2. **Predictive Entropy:** The Predictive Entropy (PE) is a measure of the uncertainty in the predictive distribution. It is defined as the negative sum of the predicted probabilities weighted by their logarithms:

$$\mathrm{PE} = -\sum_y p(y|x,\theta)\log_2 p(y|x,\theta)$$

Suppose we have a classification task with three possible classes: $y = 0, 1, 2$. We have trained a Monte Carlo Dropout neural network with $T = 5$ dropout samples per input, and we obtain the following softmax outputs for an input x:

Sample t	Class 0	Class 1	Class 2
1	0.2	0.3	0.5
2	0.3	0.3	0.4
3	0.25	0.25	0.5
4	0.1	0.2	0.7
5	0.4	0.1	0.5
Avg	0.25	0.24	0.52

To calculate the predictive entropy using the given table and show the uncertainty value for each class and the overall uncertainty, we can calculate the entropy for each class separately and then take the average.

$$H(y=0) = -(0.2 \cdot \log_2(0.2) + 0.3 \cdot \log_2(0.3) + 0.25 \cdot \log_2(0.25) + 0.1 \cdot \log_2(0.1) + 0.4 \cdot \log_2(0.4)) = 1.881$$

$$H(y=1) = -(0.3 \cdot \log_2(0.3) + 0.3 \cdot \log_2(0.3) + 0.25 \cdot \log_2(0.25) + 0.2 \cdot \log_2(0.2) + 0.1 \cdot \log_2(0.1)) = 1.901$$

$$H(y=2) = -(0.5 \cdot \log_2(0.5) + 0.4 \cdot \log_2(0.4) + 0.5 \cdot \log_2(0.5) + 0.7 \cdot \log_2(0.7) + 0.5 \cdot \log_2(0.5)) = 1.912$$

To calculate the overall uncertainty, we can take the average of the individual entropies:

$$H(y) = \frac{1}{3} \cdot (H(y=0) + H(y=1) + H(y=2)) = \frac{1}{3} \cdot (1.881 + 1.901 + 1.912) = 1.898$$

Quantifiers

3. **Mutual Information:** The Mutual Information (MI) measures the reduction in uncertainty about the target variable y given the observed input $\mathbf{x}$ when the model parameters θ are known. It is defined as the difference between the predictive entropy and the expected entropy:

$$\mathrm{MI} = \mathrm{PE} - \mathbb{E}_\theta[\mathrm{PE}]$$

o calculate the mutual information-based uncertainty for the Monte Carlo dropout results of a single input instance, you need to consider the variation across multiple samples.

Here's how you can calculate it:

1. Compute the average probability for each class across the Monte Carlo dropout samples:
   - For Class 0: (0.2 + 0.3 + 0.25 + 0.1 + 0.4) / 5 = 0.25
   - For Class 1: (0.3 + 0.3 + 0.25 + 0.2 + 0.1) / 5 = 0.24
   - For Class 2: (0.5 + 0.4 + 0.5 + 0.7 + 0.5) / 5 = 0.52
2. Compute the entropy for each class using the average probabilities:
   - For Class 0: Entropy = -(0.25 * log2(0.25)) = 0.5
   - For Class 1: Entropy = -(0.24 * log2(0.24)) = 0.4997
   - For Class 2: Entropy = -(0.52 * log2(0.52)) = 0.4923
3. Calculate the total entropy by summing the entropies of all classes:
   - Total entropy = Entropy of Class 0 + Entropy of Class 1 + Entropy of Class 2
4. Calculate the mutual information:
   - Mutual information = -Total entropy

Average entropy = (0.5 + 0.4997 + 0.4923) / 3 = 0.49733

Total entropy = -((1/3) * log2(1/3) + (1/3) * log2(1/3) + (1/3) * log2(1/3)) = -(-1.58496) = 1.58496

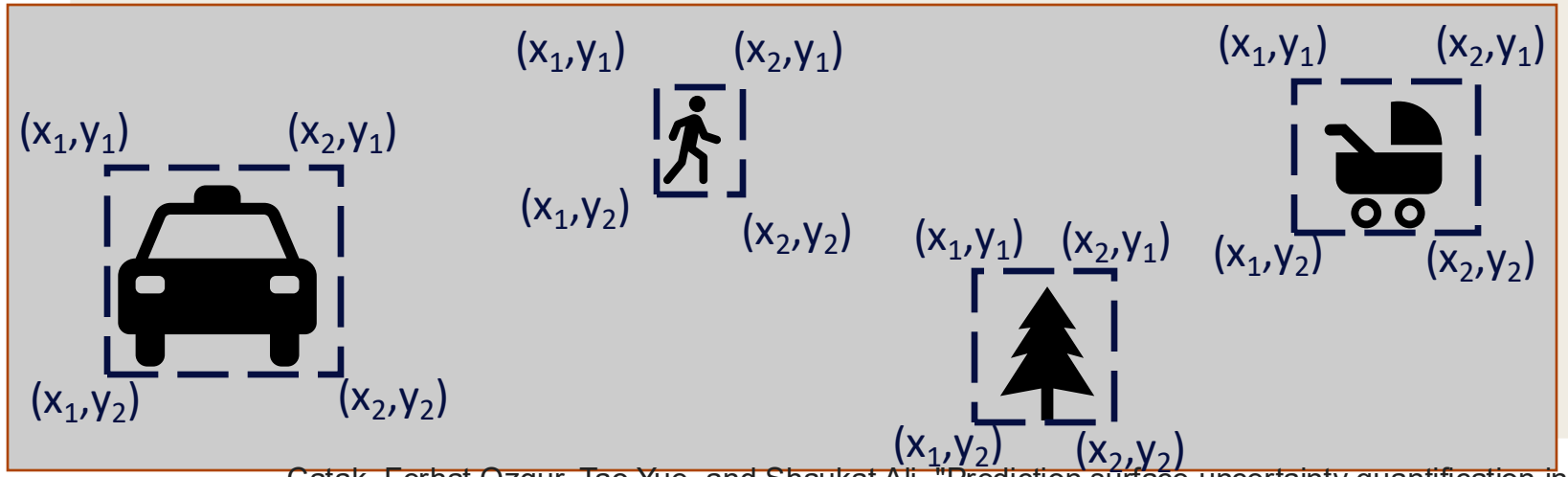
Mutual information = Total entropy - Average entropy = 1.58496 - 0.49733 = 1.08763

The mutual information-based uncertainty value for the given Monte Carlo dropout results is approximately 1.08763.

Problem Formulation

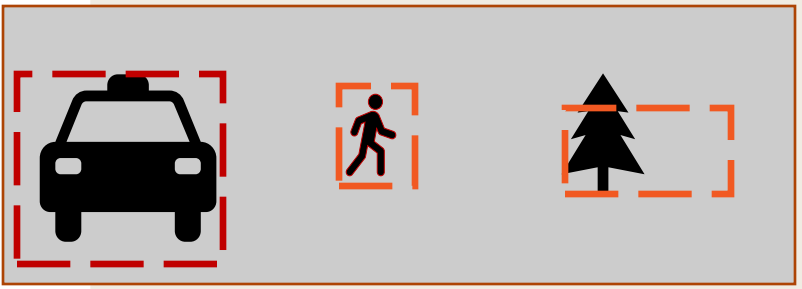
- Object detection task: detect all objects from the camera scene

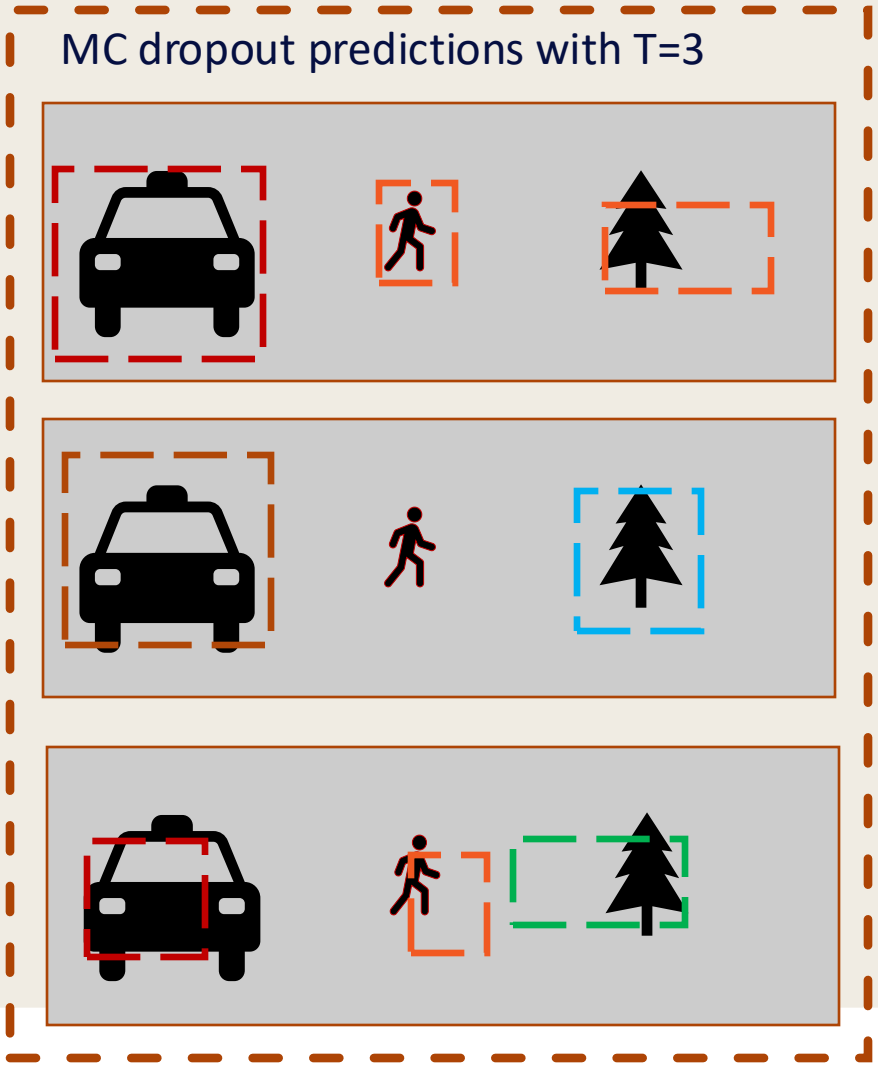
- Problems:
  - the number of objects
  - Objects sizes



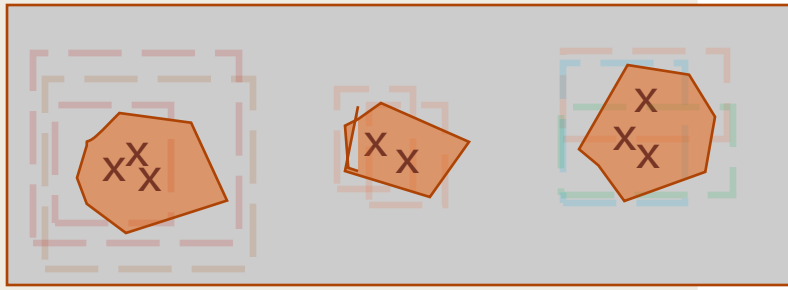
Catak, Ferhat Ozgur, Tao Yue, and Shaukat Ali. "Prediction surface uncertainty quantification in object detection models for autonomous driving." *2021 IEEE International Conference on Artificial Intelligence Testing (AITest)*. IEEE, 2021.

The PURE Method



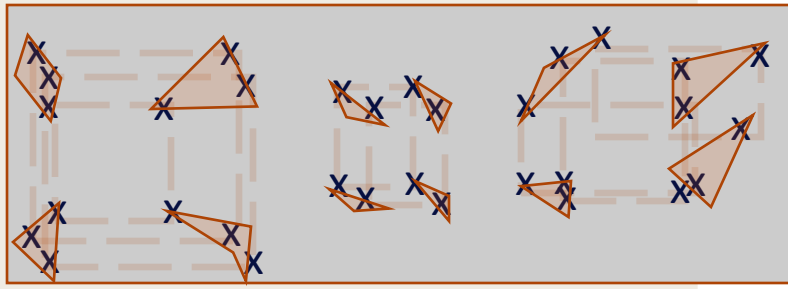


DBScan based clustering algorithm



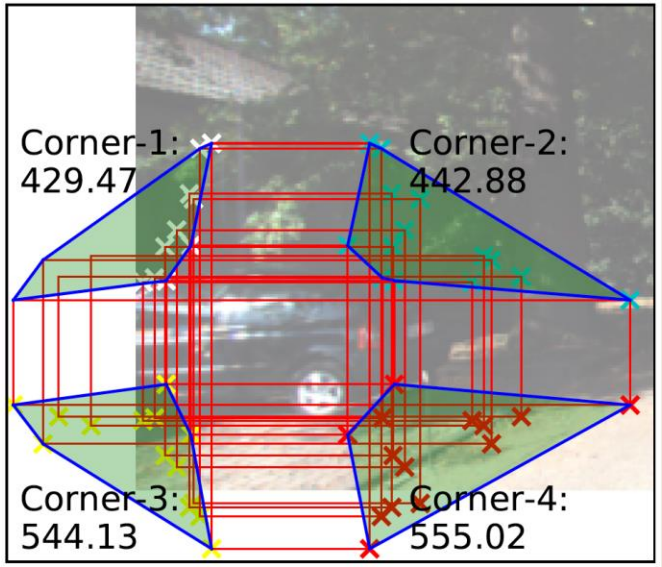
Prediction centers

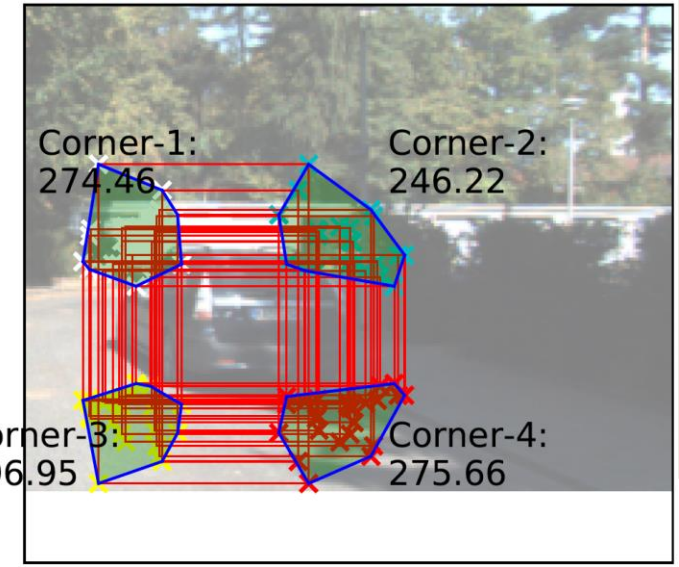
Corners of the each predictions



Convex-Hull based are calculation

The PURE Method





The PURE Method

Algorithm 1: MC dropout based object predictions

Input: $X \in \mathbb{R}^{m \times n}$: camera view
h: Object detection model
T: MC dropout size
Output: Predictions *center_lists*

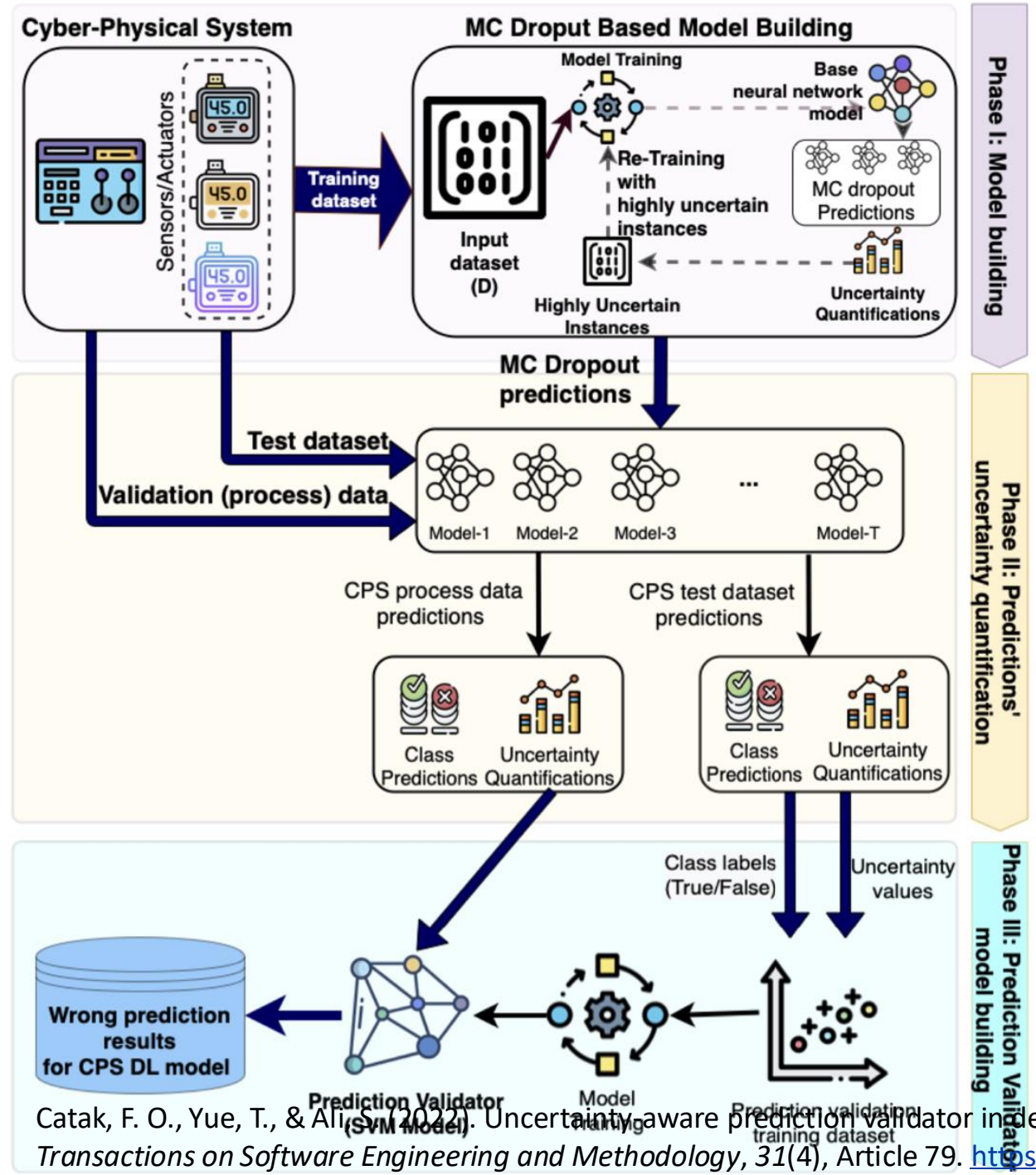
$center\_lists \leftarrow \emptyset$
/* T predictions using the object detection model h */
foreach $t \in T$ **do**
  /* Predict objects with surrounding boxes from image X */
  $VBoxes \leftarrow h(X)$
  foreach $box \in VBoxes$ **do**
    /* Get the corner locations of each surrounding box */
    $x_1, y_1 = box.xmin, box.ymin,$
    $x_2, y_2 = box.xmax, box.ymax$
    /* Find the width and height of the predicted object */
    $width, height = x_2 - x_1, y_2 - y_1$
    $x_{center}, y_{center} \leftarrow x_1 \frac{width}{2}, y_1 + \frac{height}{2}$
    $center\_lists \leftarrow UpdateArchive(x_{center}, y_{center})$
return *center_lists*

Algorithm 2: DBSCAN clustering algorithm based clustering of predictions and uncertainty quantification

Input: Predictions *center_lists*,
ϵ: radius of DBSCAN
Output: Uncertainty of predictions, $\mathcal{U}$

$area \leftarrow \emptyset$
/* Predictions are completed. Find object clusters from the *centerlist* */
$Clusters \leftarrow DBSCAN(center\_list, \epsilon)$
foreach *cluster* $c \in Clusters$ **do**
  /* Calculate the area of each corner of the predictions in cluster c */
  area $x_1 \leftarrow ConvexHull(center\_lists[c].x_1)$
  area $y_1 \leftarrow ConvexHull(center\_lists[c].y_1)$
  area $x_2 \leftarrow ConvexHull(center\_lists[c].x_2)$
  area $y_2 \leftarrow ConvexHull(center\_lists[c].y_2)$
  /* Calculate the cluster's (i.e. the single object's) uncertainty */
  $area \leftarrow$ UpdateArchive(Mean(area x_1, area y_1, area x_2, area y_2))
/* Average uncertainty of all detected objects from all T predictions */
$\mathcal{U} \leftarrow Mean(area)$
/* Return calculated uncertainty $\mathcal{U}$ for the predictions from the image X */
return $\mathcal{U}$

Prediction validation



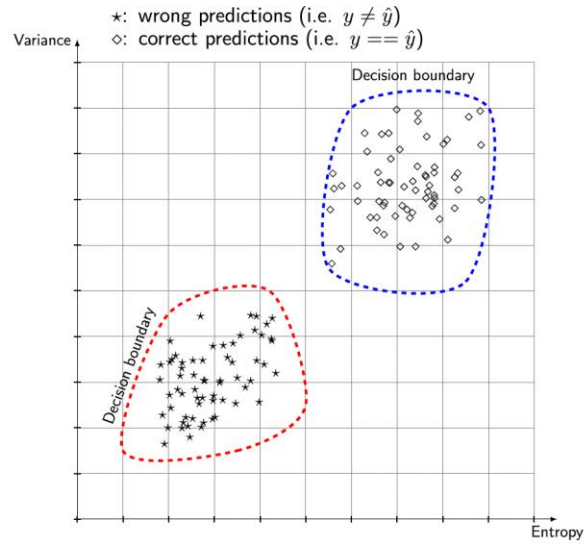


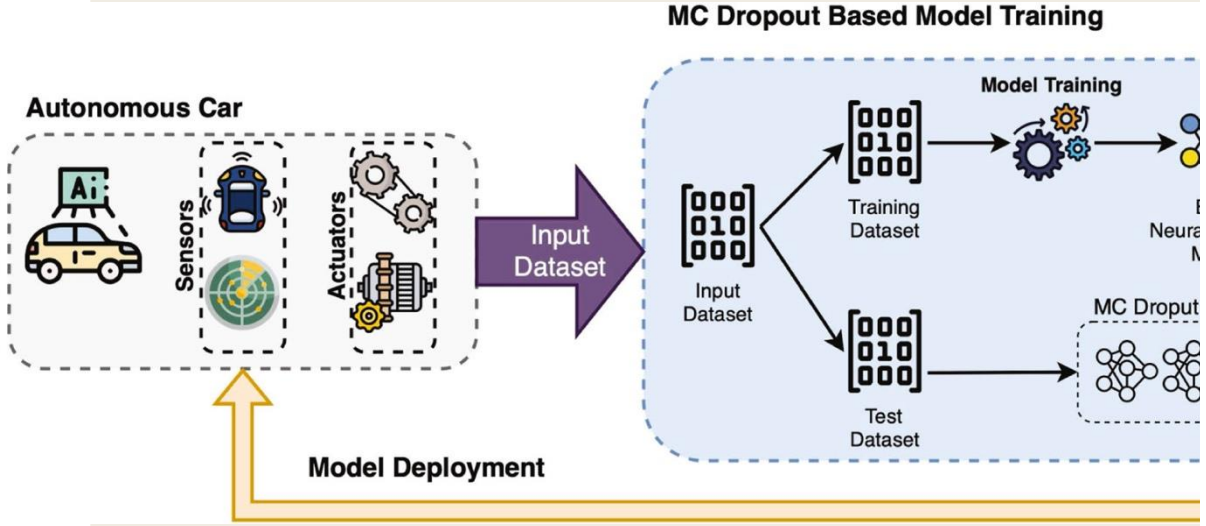
Fig. 3. Prediction validator model's uncertainty training dataset.

- A method to enhance the reliability and safety of deep learning models in CPSs by managing prediction uncertainties.
- An SVM model then predicts incorrect labels based on these uncertainties
- . Experiments with real-world datasets (SWaT, Robot, Video conferencing, KITTI) show that higher uncertainty correlates with incorrect predictions.

Catak, F. O., Yue, T., & Ali, S. (2022). Uncertainty-aware prediction validator in deep learning models for cyber-physical system data. *ACM Transactions on Software Engineering and Methodology*, *31*(4), Article 79. https://doi.org/10.1145/3527451

University of Stavanger

Uncertainty Generati



$$\mathbf{noise} = \underset{\substack{\mathbf{noise} \\ \|\mathbf{noise}\| < \epsilon}}{arg\ min} \left(\frac{\alpha}{\mathbb{U}(h(\mathbf{x}))} + \|\mathbf{noise}\| + \beta \cdot 1\,(y == \hat{y}) \right),$$

where:

– $\mathbb{U}\,(h\,(\mathbf{x}))$ is the prediction uncertainty

– $\mathbf{x}$ is the input instance

– h is the DL model

– $\|\mathbf{noise}\|$ is the norm (i.e. magnitude) value of the noise

– ϵ is the Noise budget (i.e. the maximum distance between $\mathbf{x}$ and $\mathbf{x} + \mathbf{noise}$)

– $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{noise} \in \mathbb{R}^m$

– $\mathbf{noise}$ has the same number of columns as the input instance $\mathbf{x}$

– α is the multiplication factor of uncertainty quantification metrics value

– β is the multiplication factor of the number of incorrect predictions

The ideal $\mathbf{noise}$ would have the following properties:

– Maximize the prediction uncertainty: $\frac{\epsilon}{\mathbb{U}(\mathbf{x})}$

– Minimize the noise magnitude (norm value): $\|\mathbf{noise}\|$

– Change the prediction class: $1\,(y == \hat{y})$

https://www.sciencedirect.com/science/article/pii/S0019057822005997

My Research

- Uncertainty Aware Deep Learning Model for Secure and Trustworthy Channel Estimation in 5G Networks
- https://github.com/ocatak/6g-channel-estimation-dataset
- Book repo

Conventional Channel Estimation – Without AI

Channel estimation: Estimating channel characteristics for improved signal reception.
The receiver can compensate for the effects of **fading**, **interference**, and other impairments, thus improving the quality and reliability of the received signal.
The the physical downlink shared channel (PDSCH) carries the user data and is transmitted from the base station (eNodeB or gNB) to the user equipment (UE). The demodulation reference signal (DM-RS) is a known reference signal specifically designed to aid in channel estimation.

DM-RS Configuration:
- Base station configures DM-RS positions within PDSCH.
- Known DM-RS pattern and structure.

PDSCH and DM-RS Transmission:
- Base station transmits PDSCH with embedded DM-RS symbols.

Channel Estimation Model:
- Receiver captures the received signal (PDSCH + DM-RS).
- Constructs a channel estimation model using the known DM-RS symbols.

DM-RS Detection:
- Receiver performs DM-RS detection.
- Extracts received DM-RS symbols using known pattern and structure.

Channel Estimation:
- Receiver applies signal processing techniques for estimation.
- Interpolation, equalization, and specific algorithms based on channel model.

Channel Compensation:
- Estimated channel info used for compensating impairments.
- Enhances data detection, demodulation, and decoding.

Dataset

- MATLAB – LTE and 5G toolboxes
- SISO antenna method is used by utilizing the physical downlink shared channel (PDSCH) and demodulation reference signal (DM-RS) to create the channel estimation model.
- The toolbox generates 256 training data, (i.e., transmit/receive the signal 256 times).
- Each dataset consists of 8568 data points,
  - i.e., 612 subcarriers, 14 OFDM symbols, 1 antenna.
- Each data point of the dataset is converted from a complex (real and imaginary) 612-14 matrix into a real-valued 612-14-2 matrix for providing inputs separately into the NN during the training process.
- This is because the resource grids consist of complex data points with real and imaginary parts in the channel estimation scenario, but the CNN model manages the resource grids as 2-D images with real numbers.

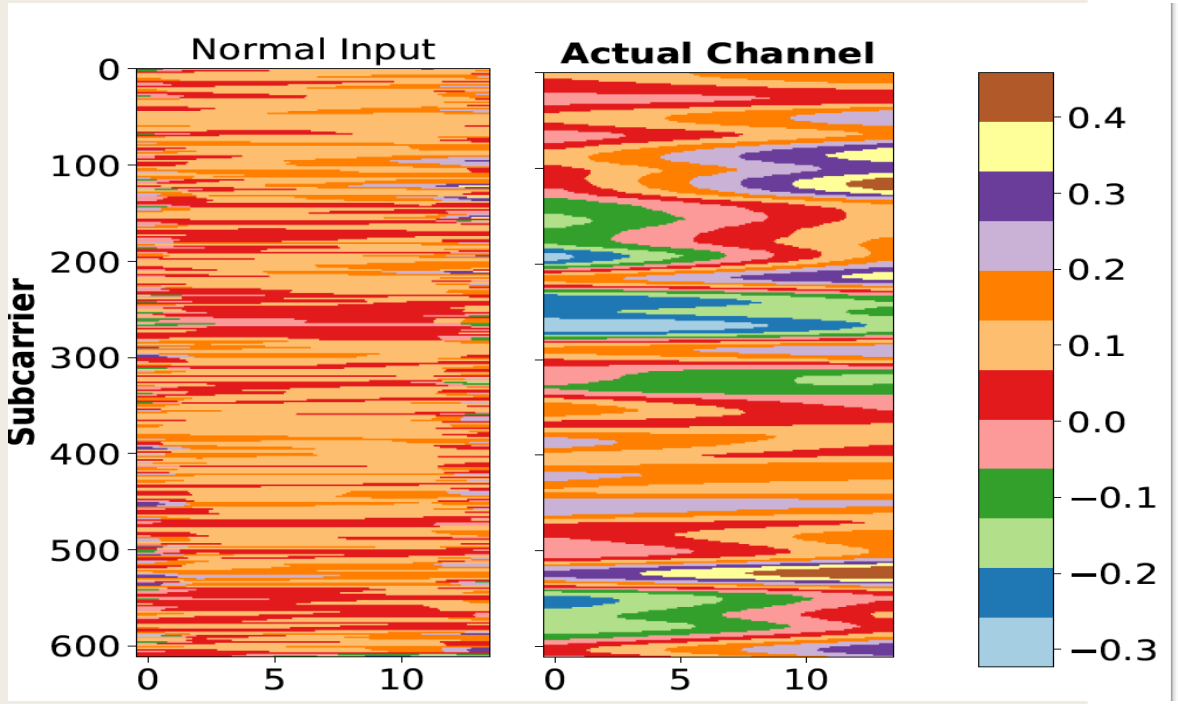
TABLE I
EACH OF THE CHANNEL CHARACTERISTIC PARAMETERS AND VALUES

Channel Parameter	Value
Delay Profile	TDL-A, TDL-B, TDL-C, TDL-D, TDL-E
Delay Spread	1-300 ns
Maximum Doppler Shift	5-400 Hz
NFFT	1024
Sample Rate	30720000
Symbols Per Slot	14
Windowing	36
Slots Per Subframe	2
Slots Per Frame	20
Polarization	Co-Polar
Transmission Direction	Downlink
Num. Transmit Antennas	1
Num. Receive Antennas	1
Fading Distribution	Rayleigh
Modulation	16QAM

F1	F2	F3	F4
0.15+0.90j	0.26+0.90j	0.32+0.90j	0.41+0.88j
-0.39-0.84j	-0.46-0.83j	-0.55-0.79j	-0.61-0.72j
-0.26-0.89j	-0.38-0.87j	-0.44-0.84j	-0.50-0.80j
-0.56+0.78j	-0.45+0.82j	-0.37+0.89j	-0.28+0.89j
⋮	⋮	⋮	⋮
-0.86-0.43j	-0.88-0.35j	-0.87-0.23j	-0.89-0.12j

⇓

F1-1	F1-2	F2-1	F2-2	F3-1	F3-2	F4-1	F4-2
0.15	0.90	0.26	0.90	0.32	0.90	0.41	0.88
-0.39	0.84	-0.46	0.83	-0.55	0.79	-0.61	0.72
-0.26	0.89	-0.38	0.87	-0.44	0.84	-0.50	0.80
-0.56	0.78	-0.45	0.82	-0.37	0.89	-0.28	0.89
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
-0.86	0.43	-0.88	0.35	-0.87	0.23	-0.89	0.12



Model

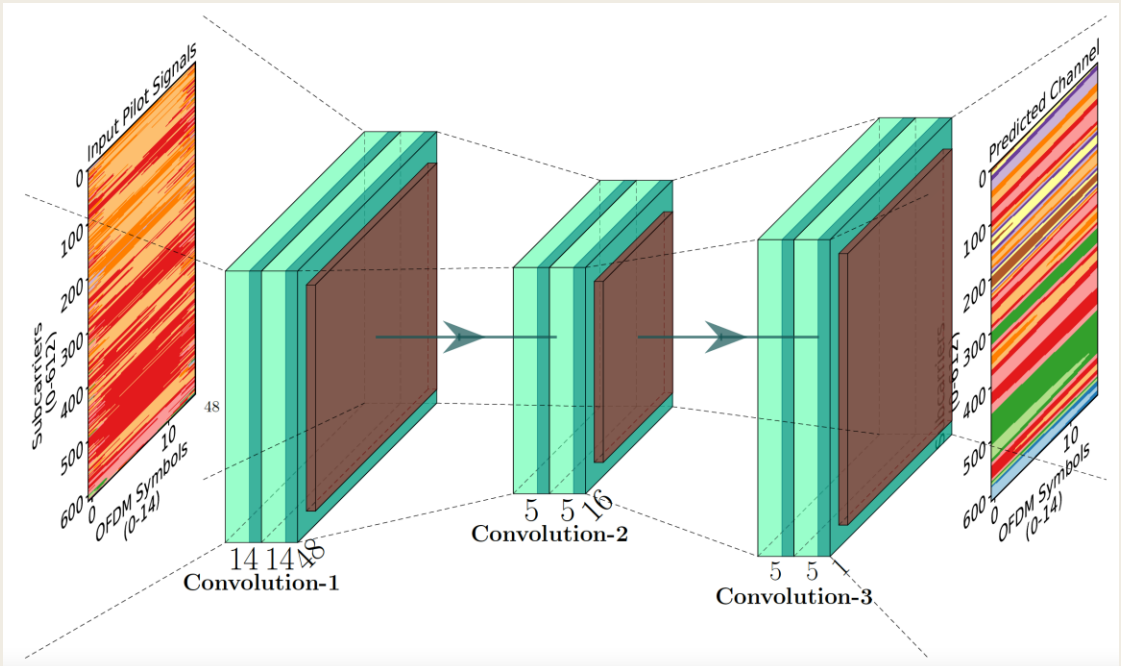
Channel estimation: Estimating channel characteristics for improved signal reception.
The receiver can compensate for the effects of **fading**, **interference**, and other impairments, thus improving the quality and reliability of the received signal.

The PDSCH carries the user data and is transmitted from the base station (eNodeB or gNB) to the user equipment (UE).
The DM-RS is a known reference signal specifically designed to aid in channel estimation.

- The models are generative and supervised models trained to predict channel parameters defined at the receiver.
- The input and output size is 612 x14 (e.g. Subcarriers x OFDM symbols).

$$MSE = \frac{\sum (Y_t - \hat{Y}_t)^2}{n} \quad (9)$$

Where : Y_t :The actual t^{th} instance, $\hat{Y}_t$: The forecasted t^{th} instance, n: The total number of instance



Uncertainty Aware Trustworthy DL Approach (Algorithm)

The algorithm aims to improve the trustworthiness of the DL model.

Inputs:

DL model: The deep learning model used for channel estimation.
Training set (D): Dataset used to train the DL model.
Validation set (V): Dataset used to evaluate the DL model and select uncertain inputs.
Max iterations (T): Maximum number of iterations for retraining the DL model.
Tolerance (Epsilon): Threshold for the loss on the validation set to terminate the iterations.

Algorithm 1 Uncertainty Aware Trustworthy DL Approach for Channel Estimation.

Inputs: DL model, training set $\mathcal{D}$, validation set $\mathcal{V}$, max iterations T, tolerance ϵ
Output: Trained trustworthy DL model M
Initialize DL model M
for $t \leftarrow 1$ to T **do**
    Compute uncertainty of the model M on the validation set $\mathcal{V}$
    Select highly uncertain inputs from the validation set $\mathcal{V}$
    Generate adversarial examples from the selected inputs
    Retrain the DL model M with $\mathcal{D} \cup \mathcal{V}$
    Compute the loss of the DL model M on the validation set $\mathcal{V}$
    if loss $< \epsilon$ **then**
        break
    end if
end for
return Trained trustworthy DL model M

Results

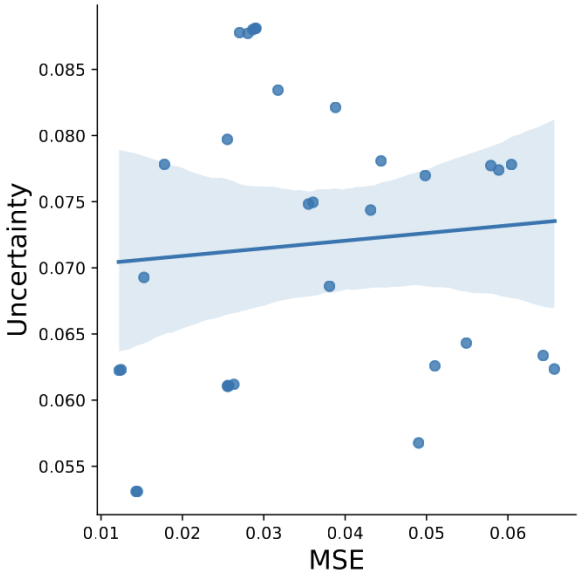
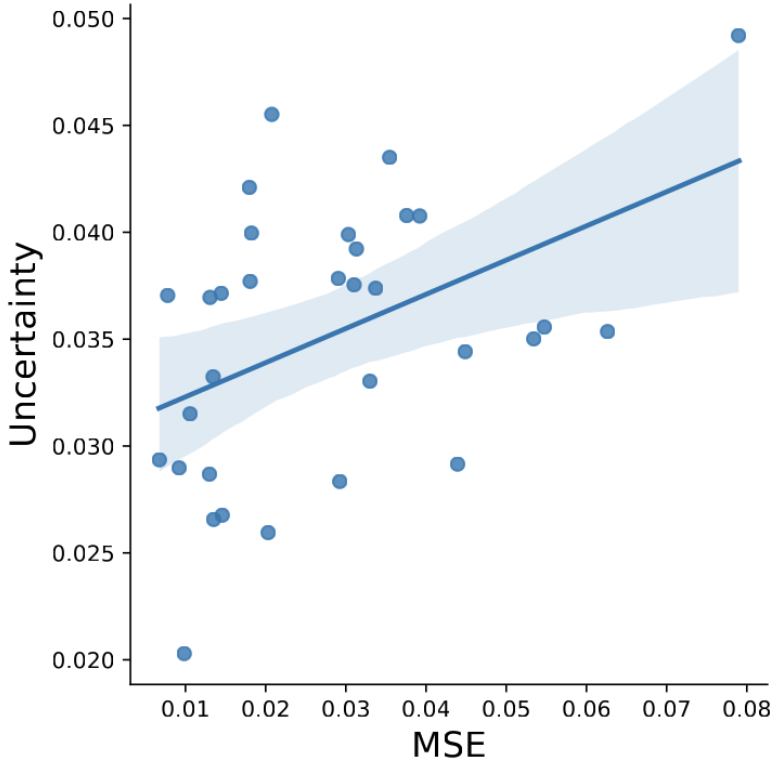


Fig. 2. Initial MSE and uncertainty correlations of the traditional and DL-based models.



Retrained MSE and uncertainty correlations of the DL model.

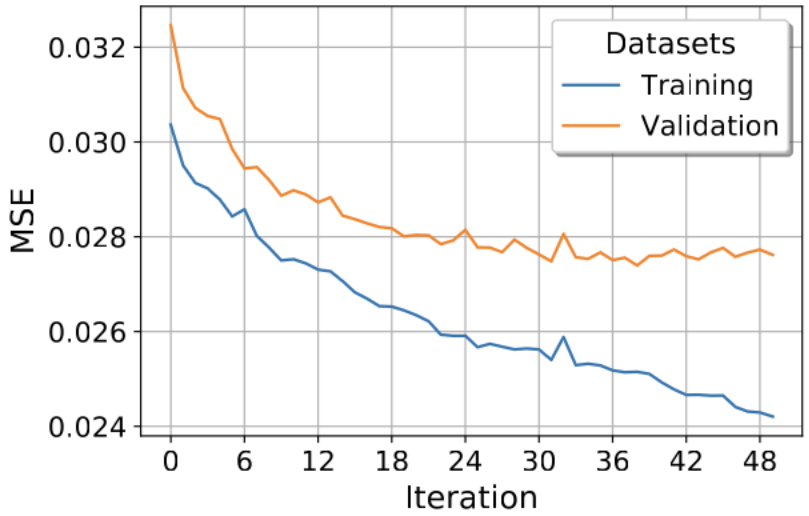


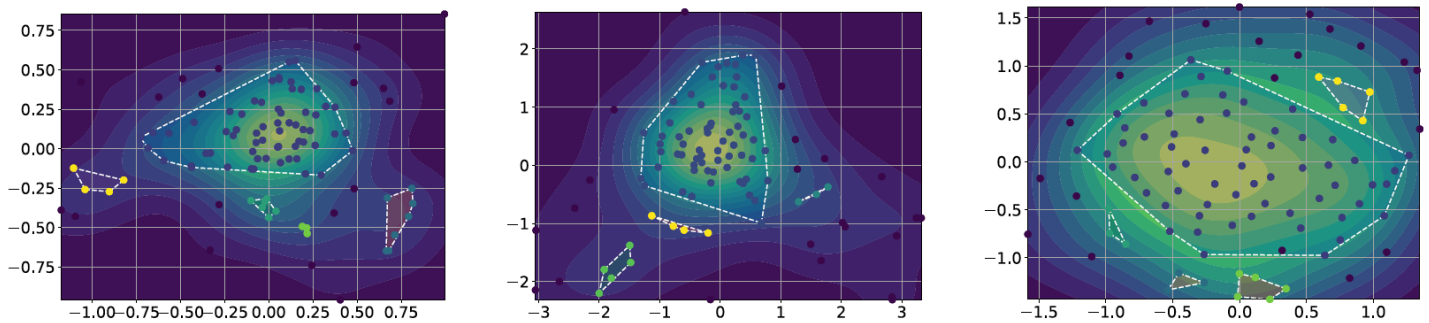
Fig. 4. MSE improvements in each iteration of the retrained DL model.

My work: Uncertainty quantification in large language models through convex hull analysis

Catak, Ferhat Ozgur, and Murat Kuzlu. "Uncertainty quantification in large language models through convex hull analysis." *Discover Artificial Intelligence* 4.1 (2024): 90.

Require: p: Prompt, m: Model, t: Temperature
Initialize total hull area total_hull_area $\leftarrow 0$
Initialize number of clusters num_clusters $\leftarrow 0$
$\mathcal{R}(p) \leftarrow$ get_responses_for_prompt(p, m, t)
if $\mathcal{R}(p) = \emptyset$ **then**
  return 0
end if
$\mathbf{E}(\mathcal{R}(p)) \leftarrow \{\text{get\_bert\_embeddings}(r) \mid r \in \mathcal{R}(p)\}$
$\mathbf{E}_{\text{RD}}(\mathcal{R}(p)) \leftarrow \text{RD}(\mathbf{E}(\mathcal{R}(p)), 2)$
if $|\mathbf{E}_{\text{RD}}(\mathcal{R}(p))| < 10$ **then**
  return 0
end if
$\mathbf{L} \leftarrow \text{DBSCAN}(\mathbf{E}_{\text{RD}}(\mathcal{R}(p)), \epsilon = 0.25 \times t \times 4.0, \text{min\_samples} = 3)$
unique_labels $\leftarrow$ set($\mathbf{L}$)
num_clusters $\leftarrow$ |unique_labels| $-$ (1 if $-1 \in$ unique_labels else 0)
for all $c \in$ unique_labels **do**
  if $c = -1$ **then**
    continue
  end if
  cluster_indices $\leftarrow \{i \mid \mathbf{L}[i] = c\}$
  cluster_points $\leftarrow \mathbf{E}_{\text{RD}}$[cluster_indices]
  if |np.unique(cluster_points.round(6), axis = 0)| > 2 **then**
    hull $\leftarrow$ ConvexHull(cluster_points)
    hull_area $\leftarrow$ hull.volume
    total_hull_area $\leftarrow$ total_hull_area + hull_area
  end if
end for
return total_hull_area

Model name	Temp. =>	0.25		0.5		0.75		1.0	
	Prompt	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Gemini-Pro	Easy	1.769	0.730	3.067	2.902	1.960	1.707	3.180	1.451
	Moderate	2.609	1.242	1.872	0.789	2.539	1.458	3.727	3.537
	Confusing	7.914	12.499	16.658	10.851	22.988	15.959	29.966	24.085
GPT-3.5-turbo	Easy	1.787	1.115	2.463	1.898	2.847	1.749	2.839	1.307
	Moderate	3.201	2.227	3.368	1.771	3.247	2.027	3.998	2.068
	Confusing	5.407	3.039	12.128	6.006	15.167	8.398	18.703	13.801
GPT-4o	Easy	0.553	0.757	0.572	0.620	3.261	1.524	1.289	1.129
	Moderate	4.064	0.000	0.678	0.002	0.163	0.000	3.181	0.000
	Confusing	0.956	0.697	2.363	2.910	2.559	2.193	3.728	2.432



(a) PCA (b) ISOMAP (c) MDS

Fig. 3 Examples of the visualizations of the convex hull areas for **a** PCA, **b** Isomap, and **c** MDS dimensionality reduction techniques

My work: Uncertainty quantification in large language models through convex hull analysis

- Motivation
  - LLMs (GPT-3.5/4, Gemini, etc.) are widely used in critical domains (healthcare, finance, law).
  - Traditional uncertainty quantification (Bayesian, ensembles, dropout) struggle with high-dimensional outputs.
- Proposed Approach
  - Novel geometric method using convex hull analysis on response embeddings.
  - Pipeline:
    - Generate multiple responses to prompts (easy, moderate, confusing).
    - Encode with BERT → reduce dimension (PCA, Isomap, MDS).
    - Cluster with DBSCAN → compute convex hull areas as uncertainty metric.
- Key Findings
  - Higher temperature → larger convex hull area → more uncertainty.
  - Prompt complexity strongly increases uncertainty (confusing > moderate > easy).
  - GPT-4o shows lowest uncertainty across settings; GPT-3.5-turbo highest.
  - Convex hull analysis provides a clear, interpretable metric for LLM reliability.
- Impact
  - Enables more trustworthy deployment of LLMs in high-risk domains.
  - Framework extensible to other embeddings & clustering methods.

My work: Improving Medical Diagnostics with Vision-Language Models: Convex Hull-Based Uncertainty Analysis

Catak, Ferhat Ozgur, Murat Kuzlu, and Taylor Patrick. "Improving Medical Diagnostics with Vision-Language Models: Convex Hull-Based Uncertainty Analysis." *arXiv preprint arXiv:2412.00056* (2024).

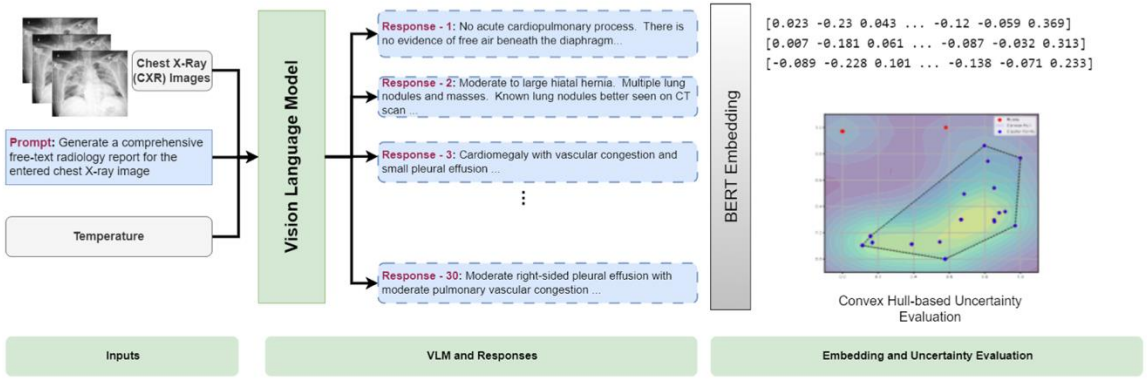
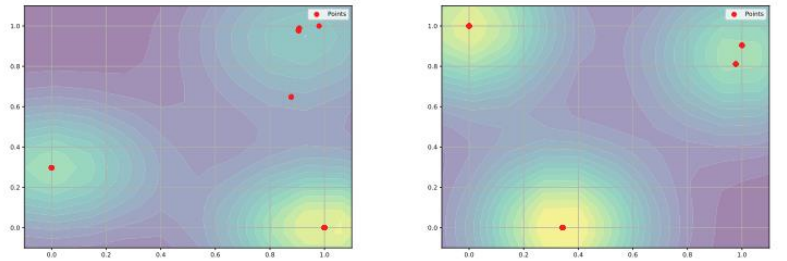
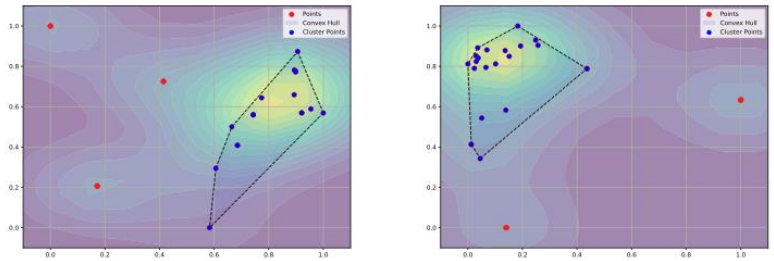


Fig. 1: The overall experimental setup for calculating uncertainty in VLM responses.



(a) Instance 1 (b) Instance 2

Fig. 3: Most uncertain instances (a-b) at the temperature setting of 0.001



(a) Instance 1 (b) Instance 2

Fig. 5: Most uncertain instances (a-b) at the temperature setting of 0.25

My work: Improving Medical Diagnostics with Vision-Language Models: Convex Hull-Based Uncertainty Analysis

Catak, Ferhat Ozgur, Murat Kuzlu, and Taylor Patrick. "Improving Medical Diagnostics with Vision-Language Models: Convex Hull-Based Uncertainty Analysis." *arXiv preprint arXiv:2412.00056* (2024).

APPENDIX

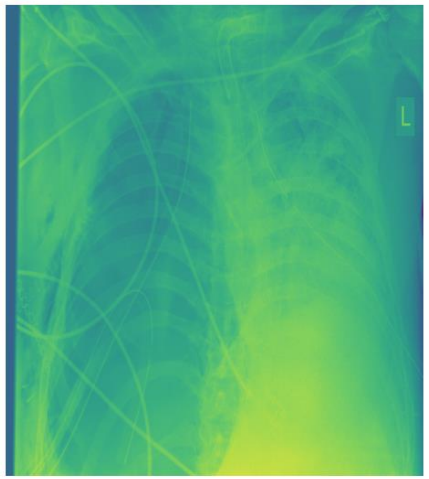
A. Least Uncertain Instance at the Temperature Setting of 0.001

X-Ray Image

Responses

```
-- No acute intrathoracic process.
-- No acute intrathoracic process.
-- No acute cardiopulmonary process.
-- No acute cardiopulmonary process.
-- No acute intrathoracic process.
-- No acute intrathoracic process.
-- No acute intrathoracic abnormality.
-- Normal chest x-ray.
-- Normal chest x-ray.
-- Normal chest radiographs.
-- No acute intrathoracic process.
-- No acute intrathoracic process.
-- No evidence of pneumonia.
-- No acute cardiothoracic process.
-- No acute cardiopulmonary process.
-- No acute intrathoracic process.
-- No acute intrathoracic process.
-- No acute cardiothoracic process.
-- No acute cardiopulmonary process.
-- No acute cardiothoracic process.
-- Normal chest radiograph.
-- No acute intrathoracic process.
-- No acute cardiothoracic process.
-- No acute cardiopulmonary process.
-- No evidence of pneumonia.
-- Normal chest radiographs.
```

H. Most Uncertain Instance at the Temperature Setting of 0.75



X-Ray Image

Responses

```
-- No acute cardiopulmonary process.  ET tube in
↪  appropriate position.
-- No evidence of acute cardiopulmonary process.
-- No acute cardiopulmonary process.
-- No acute intrathoracic process.
-- No acute cardiopulmonary abnormality.
-- No acute cardiopulmonary process.
-- No previous images.  The cardiac silhouette is
↪  within normal limits and there is no evidence
↪  of vascular congestion, pleural effusion, or
↪  acute focal pneumonia.
-- 1.  No acute cardiothoracic process. 2.  No
↪  evidence of pneumomediastinum.
-- AP chest compared to ___: Pulmonary vascular
↪  congestion and mild interstitial edema are new.
↪  Atelectasis in the right lower lobe is mild.
↪  Heart size is normal.  No pneumothorax or
↪  appreciable pleural effusion.
-- No pneumothorax or other acute cardiopulmonary
↪  process.
-- Heart size and mediastinum are stable.  Left
↪  basal consolidation has slightly increased.
↪  There is minimal right basal atelectasis.
↪  Right PICC line tip is at the level of lower
↪  SVC.  No pneumothorax is seen.
```

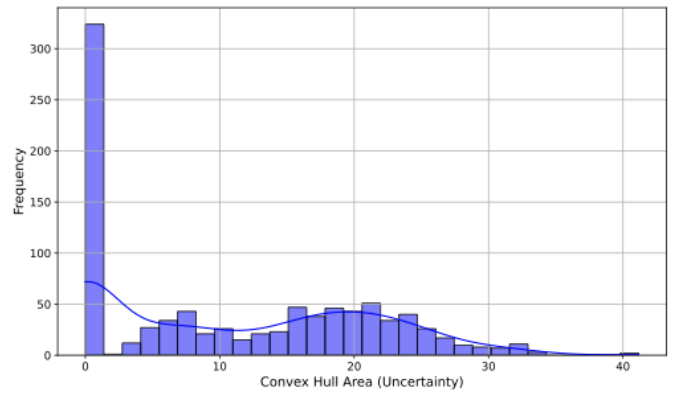


Fig. 4: Uncertainty distribution at the temperature setting=0.25

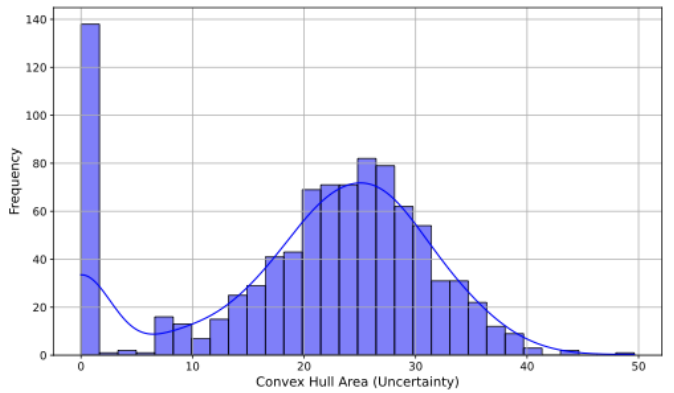


Fig. 6: Uncertainty distribution at the temperature setting=0.50

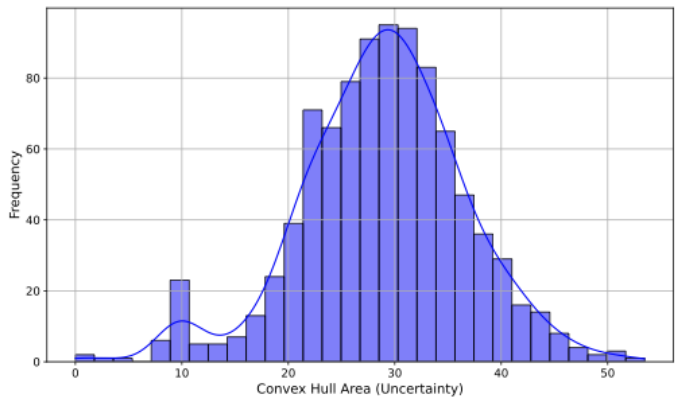


Fig. 8: Uncertainty distribution at the temperature setting=0.75

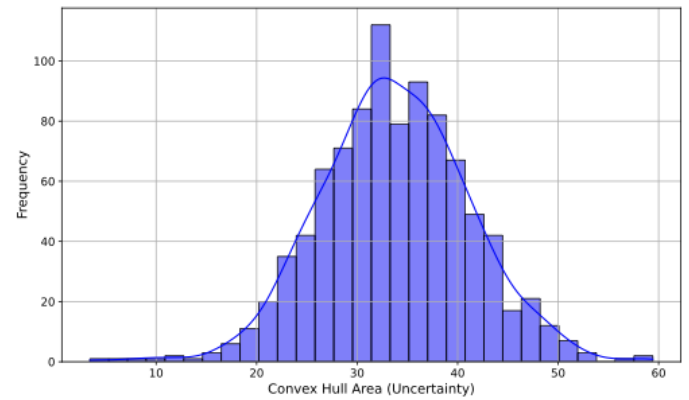


Fig. 10: Uncertainty distribution at a temperature setting=1.00

University of Stavanger

Uncertainty LLMs - 1

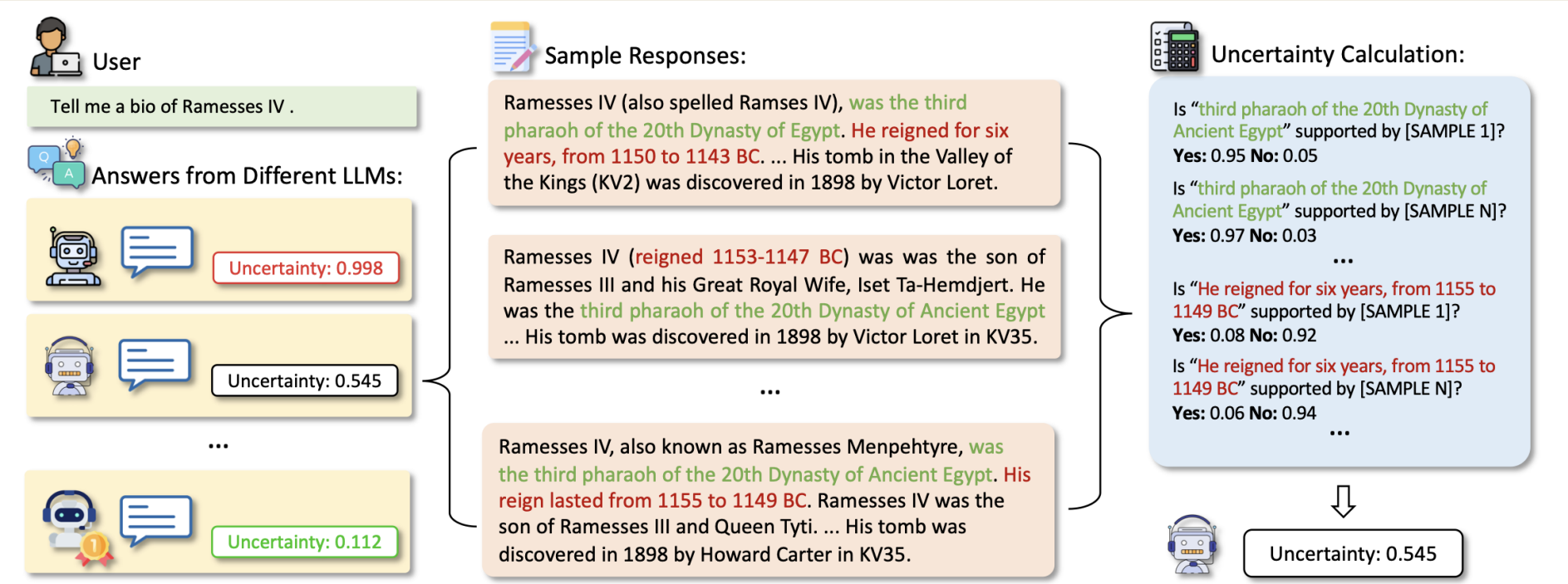


Figure 1: The illustration of the LUQ and LUQ-ENSEMBLE framework. Given a question, various LLMs exhibit differing levels of uncertainty. We generate n sample responses from each LLM and then assess the uncertainty based on the diversity of these samples (the LUQ metric). Green highlights indicate consistency across responses (low uncertainty) and red highlights discrepancies (high uncertainty). The LUQ-ENSEMBLE method selects the response from the LLM with the lowest uncertainty score as the final answer.

Zhang, Caiqi, et al. "LUQ: Long-text Uncertainty Quantification for LLMs." *arXiv preprint arXiv:2403.20279* (2024).

Uncertainty LLMs - 2

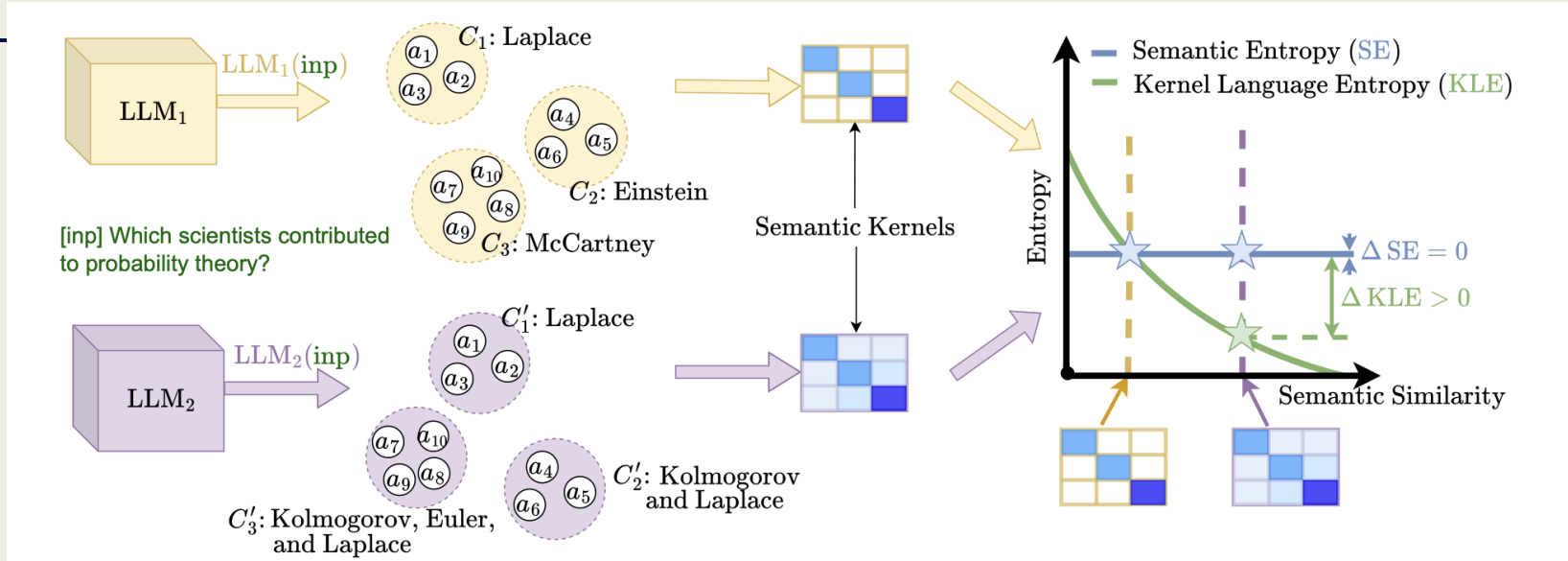


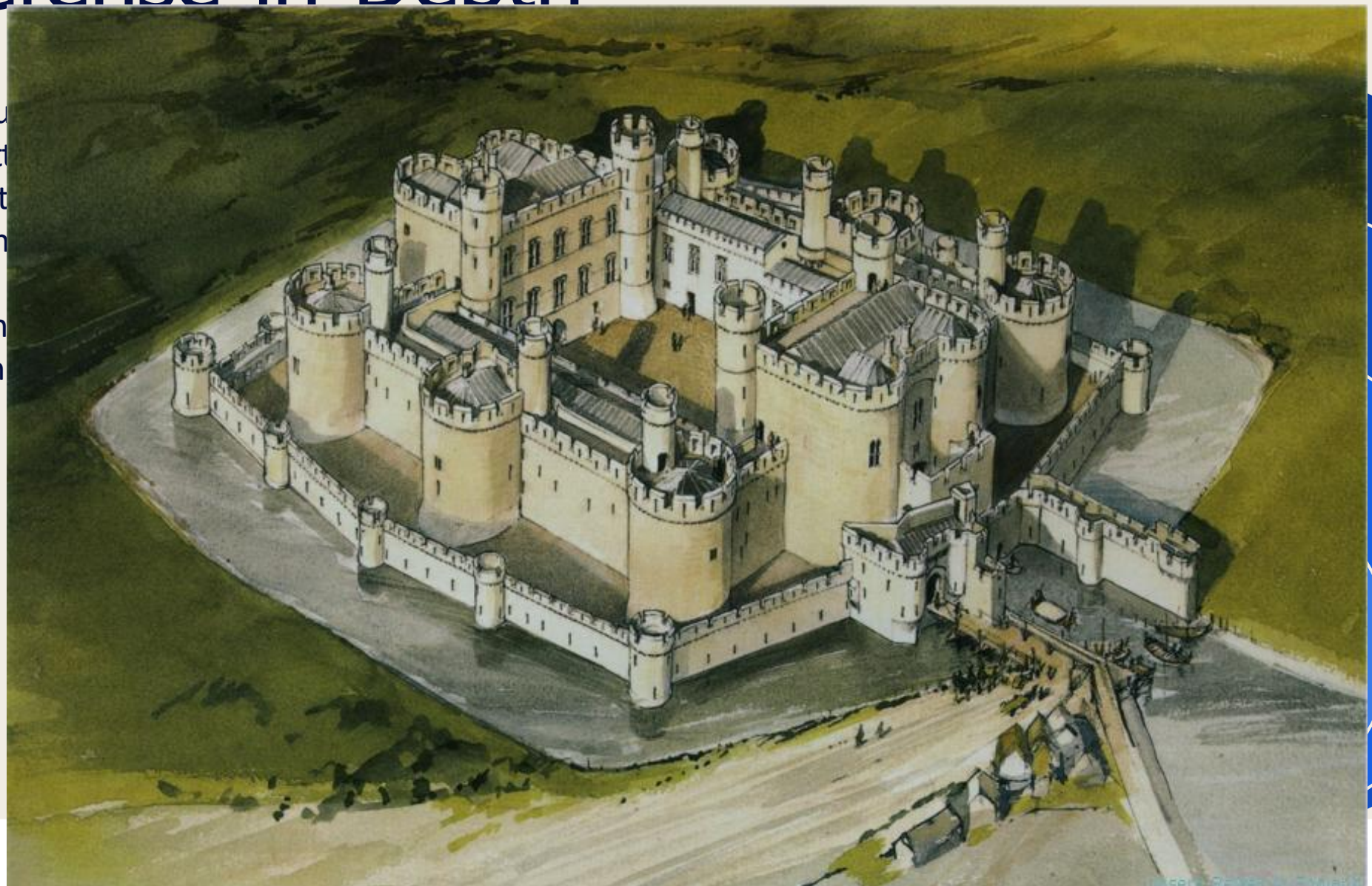
Figure 1: Illustration of Kernel Language Entropy (KLE). We here show a version of KLE called KLE-c, which operates on semantic clusters. Given an input query and two different LLMs, we sample 10 answers from each model $a_1, \ldots, a_{10}$ and $a'_1, \ldots, a'_{10}$ and cluster them by semantic equivalence into clusters $C_1, \ldots, C_3$ and $C'_1, \ldots, C'_3$. For the sake of the example, we assume that the numbers and sizes of clusters, as well as individual cluster probabilities, are all equal $p(C_i|\text{inp}) = p(C'_i|\text{inp})$ for all i. Then, semantic entropy would yield identical uncertainties for both LLMs. However, uncertainty should be lower for LLM$_2$ because semantic "similarity" between the generations is much higher; i.e., the model is fairly confident that "Kolmogorov" and "Laplace" are good answers. KLE, explicitly accounts for the semantic similarity between texts using a kernel-based approach and correctly identifies that LLM$_2$'s generations should be assigned lower uncertainty (see right).

Nikitin, Alexander, et al. "Kernel Language Entropy: Fine-grained Uncertainty Quantification for LLMs from Semantic Similarities." *arXiv preprint arXiv:2405.20003* (2024).

Cybersecurity frameworks for AI

Defense-in-Depth

- Layered secu
- During an att
  impacted, ot
- Implementin
- It is not:
  - installin
    Free on



Security Controls

Information assurance

- This control assures the confidentiality, integrity, and availability of information. It includes measures such as data encryption and access control.

Defense-in-depth

- This control strategy employs multiple layers of security, making it more difficult for an attacker to penetrate the system.

Risk management

- This control assesses and manages the risks to an information system. It includes risk assessment, risk mitigation, and risk monitoring.

Cyber threat intelligence

- This control gathers and analyzes information about threats to an information system. It helps organizations to understand the nature of the threats and to develop countermeasures.

Threat modelling

- This control identifies, analyzes, and evaluates the risks to an information system. It helps organizations to understand the nature of the threats and to develop countermeasures.

Incident management

- This control manages the response to an incident. It includes incident detection, incident response, and incident recovery.

Defense-in-Depth Approach for AI Model Protection

1. **Secure Development**: Follow secure coding practices and design AI model with security in mind.
2. **Model Architecture Security**: Consider AI-specific vulnerabilities and apply techniques like model hardening and robust training.
3. **Data Security**: Protect training and test data with access controls, encryption, and anonymization.
4. **Access Controls**: Implement strong access controls, multi-factor authentication, and secure identity management.
5. **Monitoring and Logging**: Detect suspicious activity and anomalies through monitoring and logging mechanisms.
6. **Updates and Patching**: Regularly update the AI model and underlying software stack with security patches.
7. **Incident Response and Recovery**: Have an incident response plan for handling security incidents or breaches.
8. **Training and Awareness**: Educate the team and users on AI model security best practices.

By implementing these layers of security controls, you can safeguard your AI model from unauthorized access, data breaches, and other security risks.

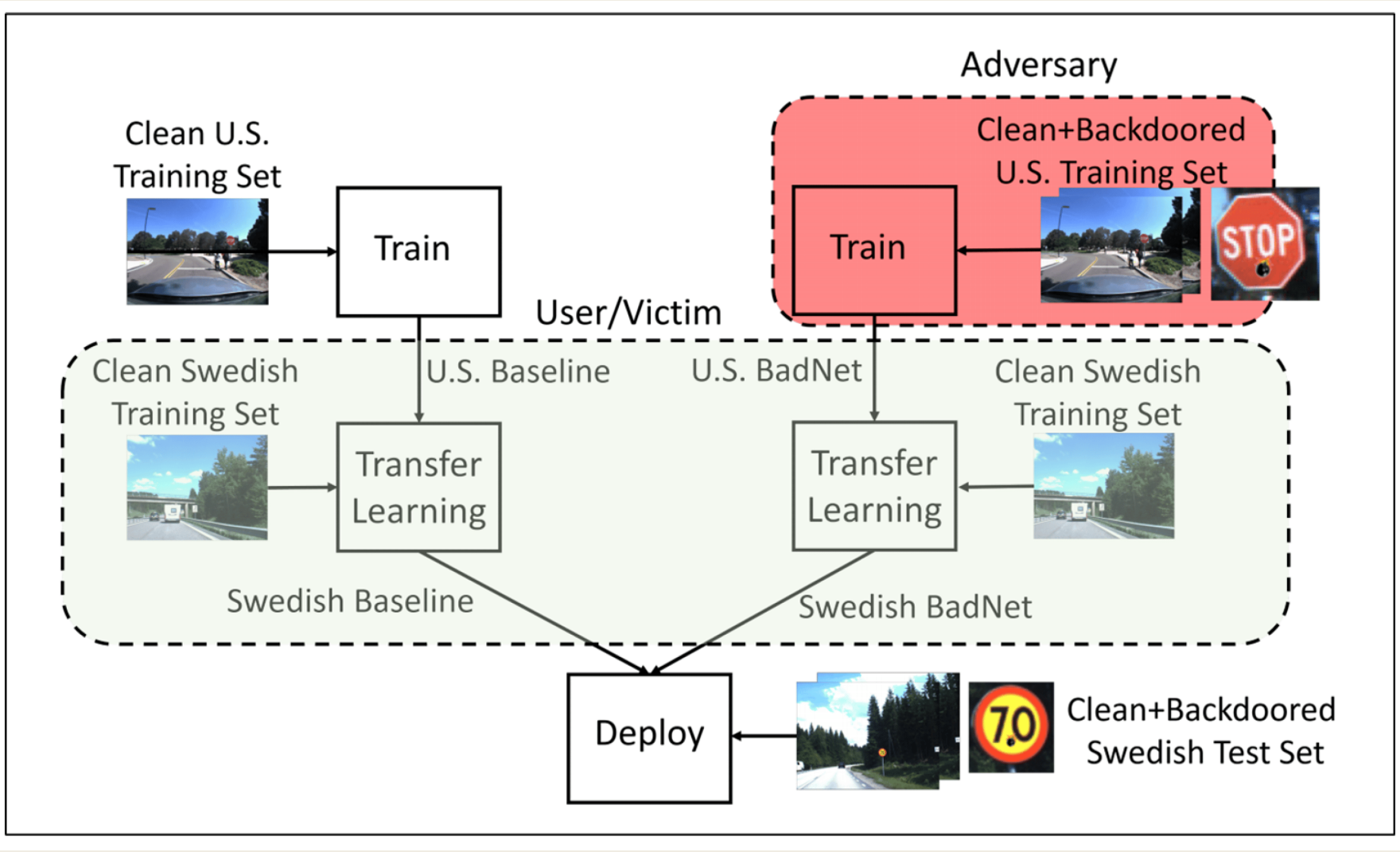
Attack Transferability in Adversarial ML

Attack transferability refers to the phenomenon where adversarial examples crafted to fool one machine learning model can also deceive other models.

- Adversarial examples can bypass different models, even with varying architectures or training datasets.
- Transferability arises due to non-robustness, as adversarial perturbations exploit vulnerabilities in models.
- Factors contributing to transferability include the input space, model similarity, dataset bias, and attack algorithms.

- Implications:
  - Transferability poses a security risk in real-world applications.
  - Adversarial examples can be crafted to bypass multiple models with a single attack.
  - Robustness and generalizability of models are important concerns.

Attack Transferability in Adversarial ML

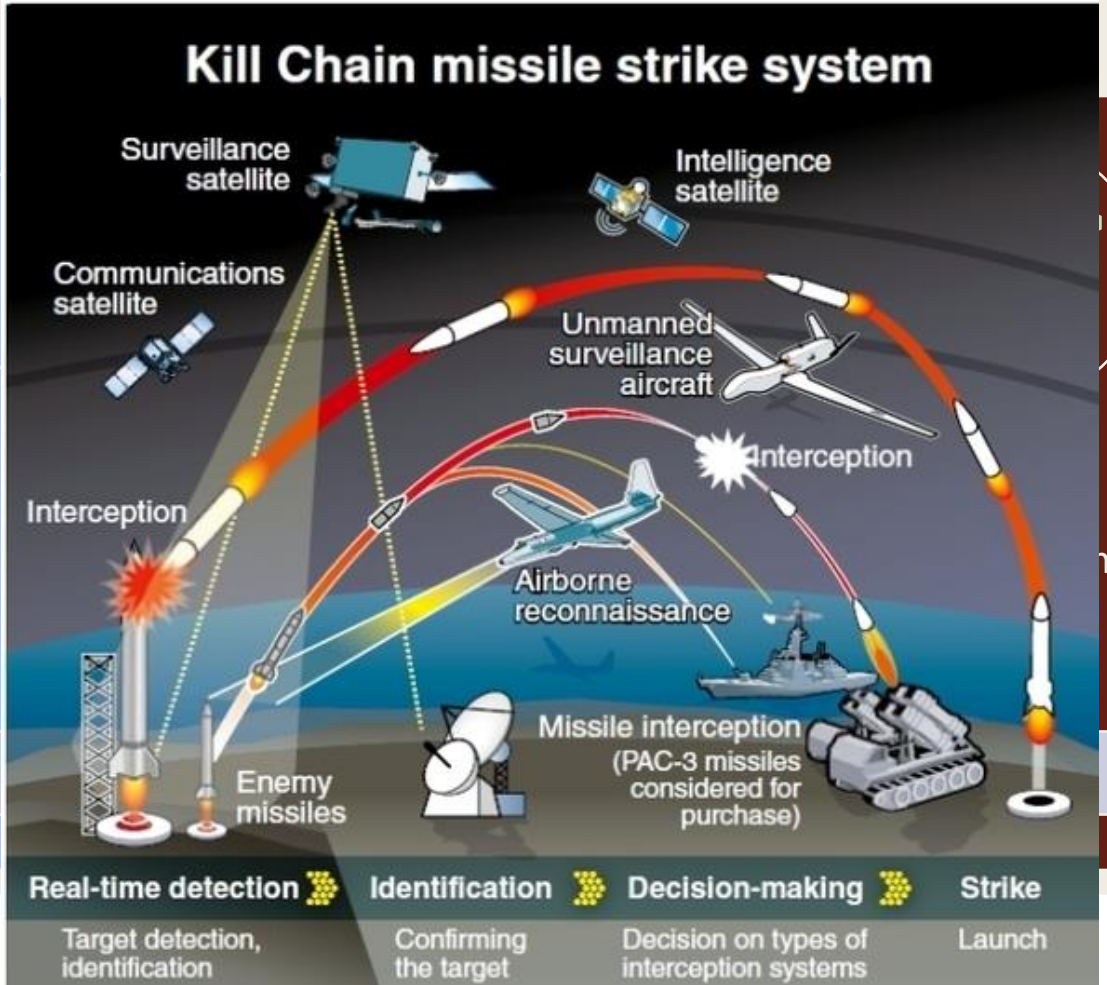


https://www.mdpi.com/2076-3417/9/5/909

Cyber Kill Chain: The 7 Stages of a Cyber Attack

Reconnaissance

Weaponizati



Command and Control (C2)

Actions on Objectives

Applying Cyber Kill Chain to AI Model Protection

1. **Reconnaissance**: Attackers gather information about the AI model, its architecture, and deployment environment.
2. **Targeted Delivery**: Malicious inputs or adversarial samples are crafted and delivered to exploit vulnerabilities in the AI model.
3. **Exploitation**: Vulnerabilities in the AI model are exploited, enabling unauthorized access or manipulation of model behavior.
4. **Command and Control**: Attackers establish control over the compromised AI model through communication channels.
5. **Actions on Objectives**: Attackers carry out malicious activities, such as data exfiltration or manipulation of model outputs.

DDoS to AI Models?

Adapting the Cyber Kill Chain to AI models helps identify and address potential vulnerabilities and threats throughout the model's lifecycle.

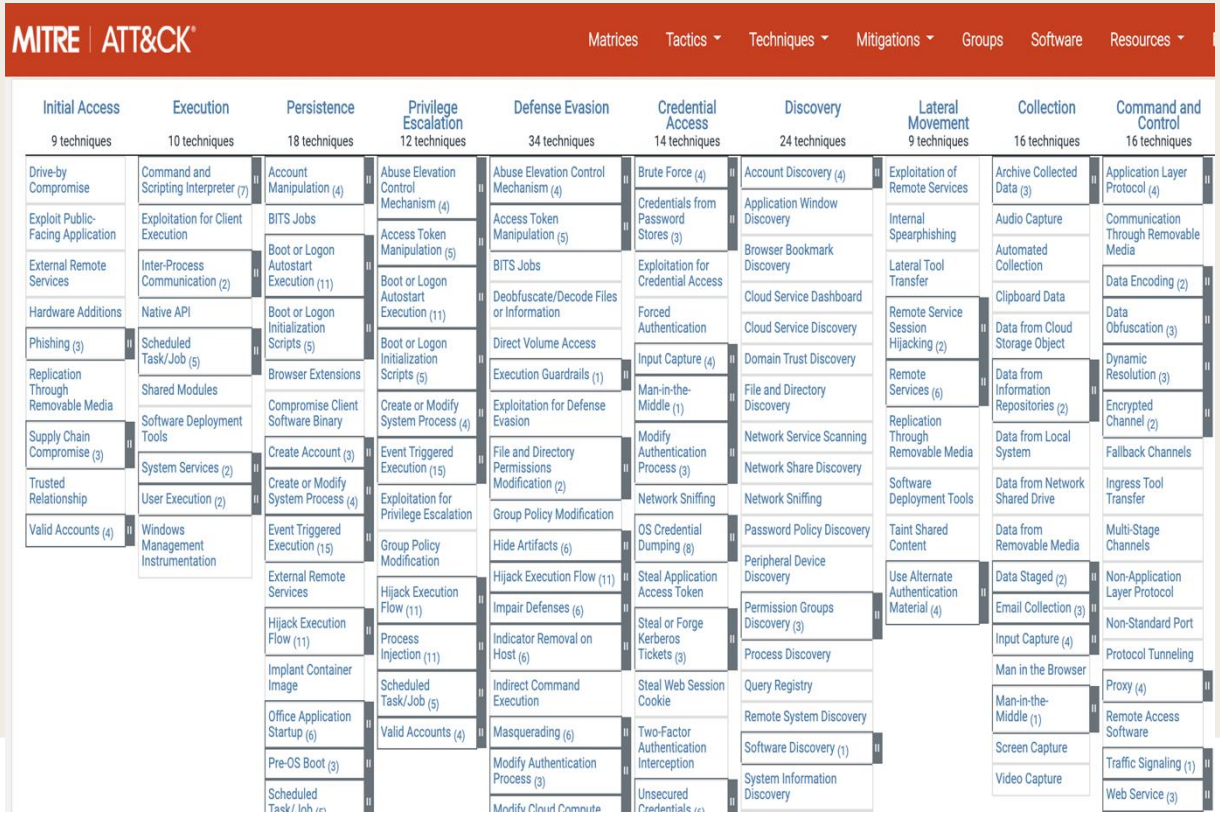
DDoS to AI Models?

- https://www.technologyreview.com/2021/05/06/1024654/ai-energy-hack-adversarial-attack/
- https://arxiv.org/pdf/2010.02432.pdf

MITRE ATT&CK Framework

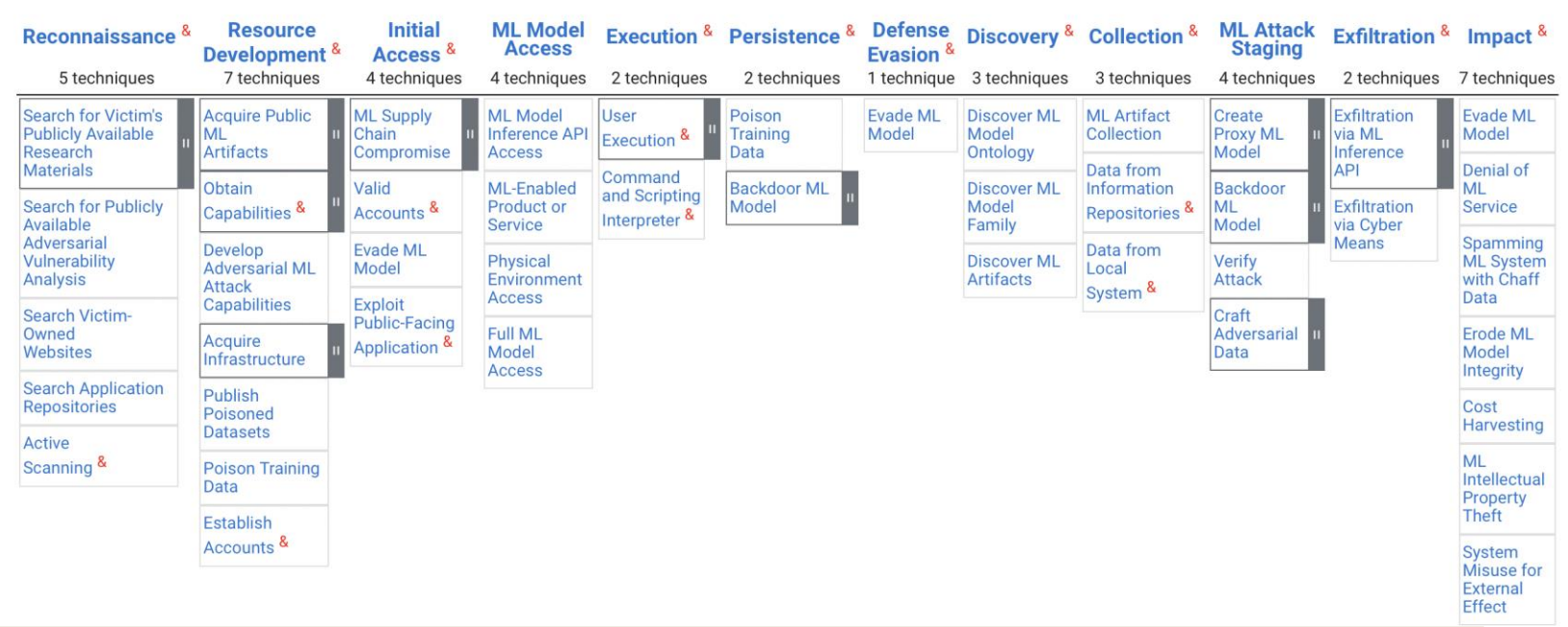
- **ATT&CK IS NOT** – An open-source (or any other kind of) threat intelligence feed full of IOCs to look for
- **ATT&CK IS NOT** - Yet another kill-chain system that describes an event lifecycle
- **ATT&CK** stands for "Adversarial Tactics, Techniques, and Common Knowledge"
- **ATT&CK IS** - A globally-accessible knowledge base of adversary tactics and techniques, categorized into an easily-consumable model
- **ATT&CK IS** - Based on real-world in-the-wild observations of actual adversary behavior
- **ATT&CK IS** – Purposefully focused on the adversary and the behaviors they exhibit, tools they use, and actions they perform.
- **ATT&CK IS** – Community driven, and updated by MITRE quarterly based on new things being seen and reported in the wild.

- **The Enterprise Matrix** covers tactics and techniques used in enterprise networks.
- **The Mobile Matrix** covers tactics and techniques used in mobile devices.
- **The Pre-ATT&CK Matrix** covers tactics and techniques that have been observed but not yet classified in a matrix.

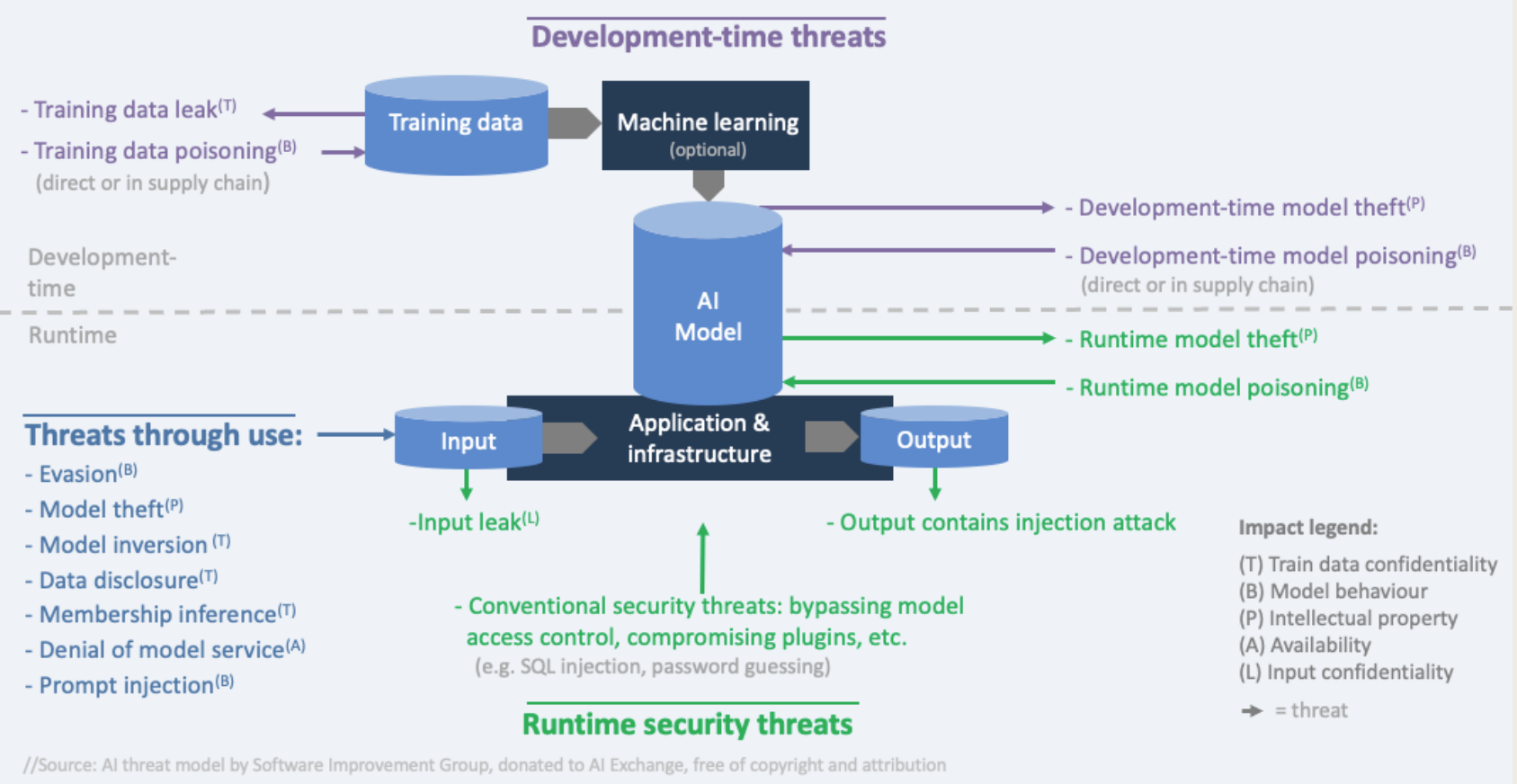


MITRE ATLAS™ (Adversarial Threat Landscape for Artificial-Intelligence Systems)

- a knowledge base of adversary tactics, techniques, and case studies for machine learning (ML) systems based on real-world observations, demonstrations
- from ML red teams and security groups, and the state of the possible from academic research.
- ATLAS is modeled after the MITRE ATT&CK® framework and its tactics and techniques are complementary to those in ATT&CK.



OWASP AI Security and Privacy Guide

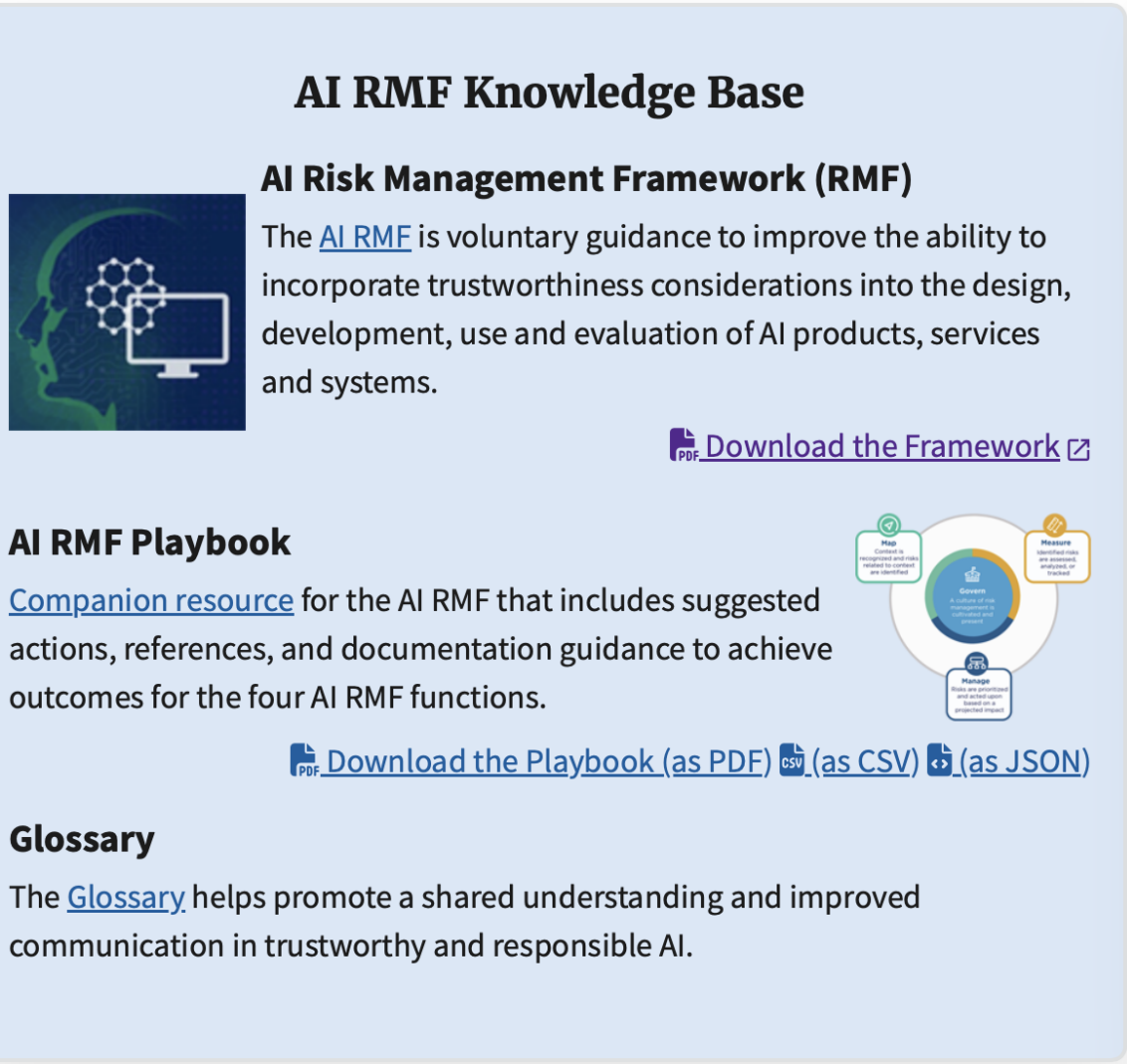


- Understanding and mitigating risks associated with AI systems.
- This project includes an AI Threat Matrix, which identifies threats and risks by stages of the AI lifecycle, aligning with practices from MITRE ATLAS, OWASP Top 10, and other AI risk databases

https://owaspai.org/docs/ai_security_overview/

NIST - Trustworthy & Responsible AI Resource Center

- https://airc.nist.gov/home
- NIST's AI RMF offers guidelines for identifying, assessing, and managing risks associated with AI systems.
- It includes a focus on securing AI algorithms, data, and communications, providing a structured approach to managing AI security and ensuring compliance with industry standards



My Research Outputs

My Research About PHY Layer AI Security Threats

Simulator

- Matlab 5G Toolbox
- REMCOM Wireless InSite

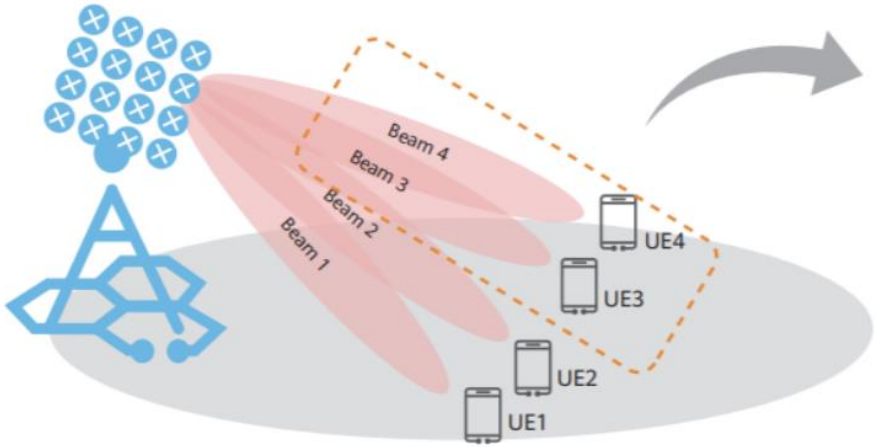
Trustworthy and Robust AI Models Against Adversarial Machine Learning Attacks

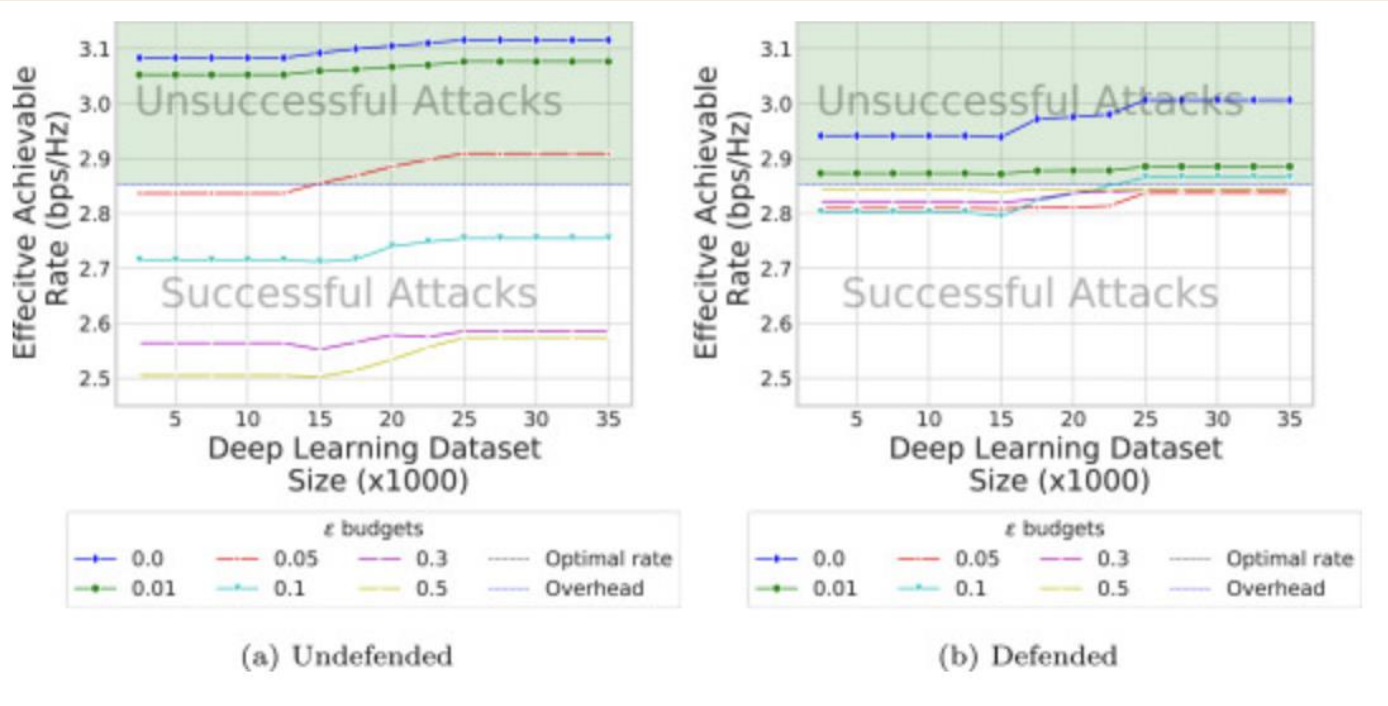
- Beamforming
- Channel Estimation
- Spectrum Sensing

Privacy Concerns in Collaborative Model Building

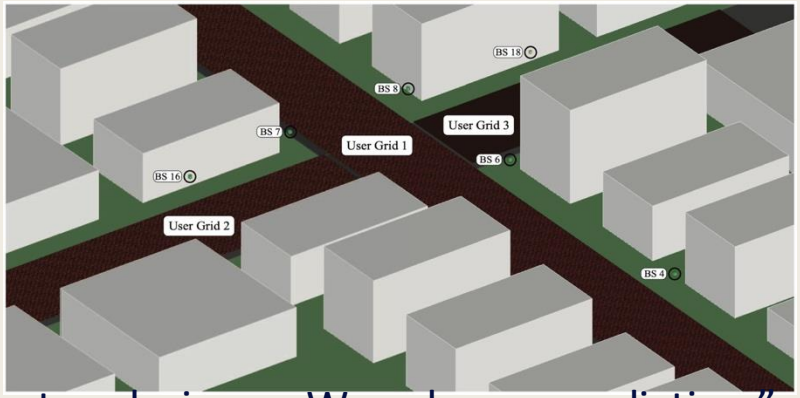
- Homomorphic Encryption Based Solutions
- Channel Estimation
- Spectrum Sensing

Research-1: Beamforming



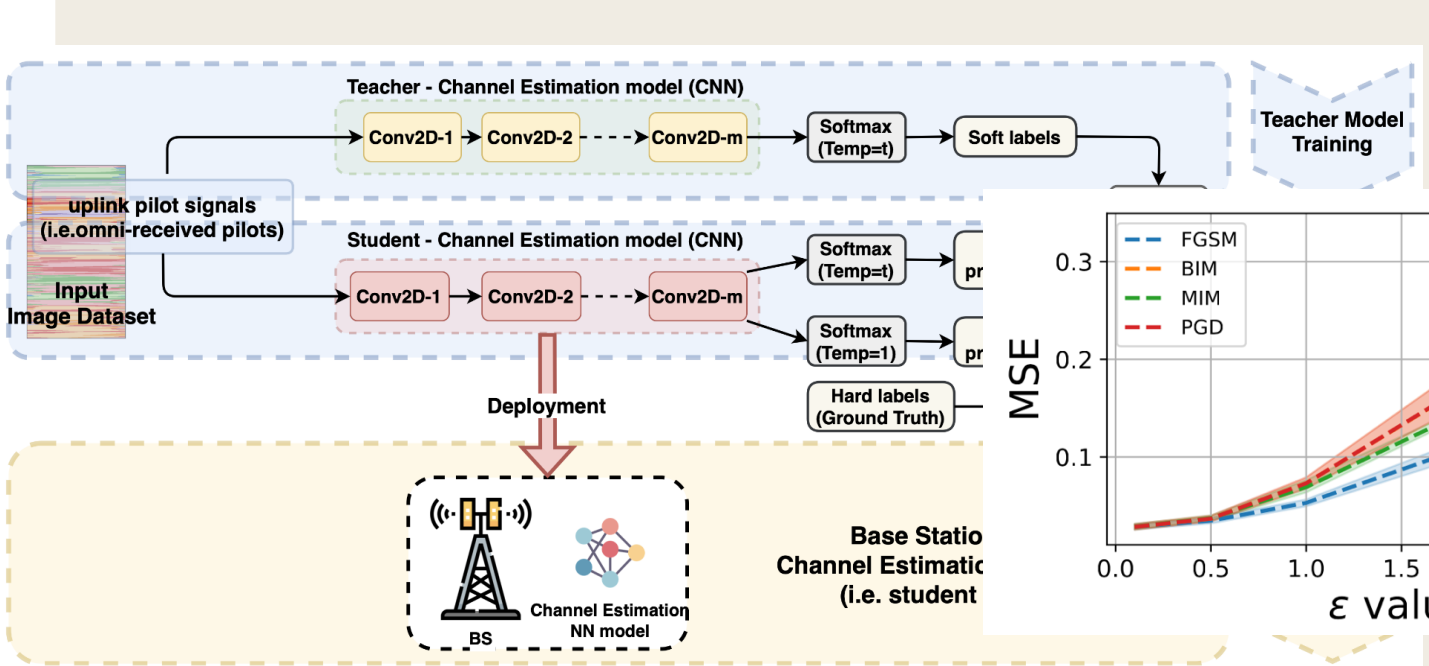


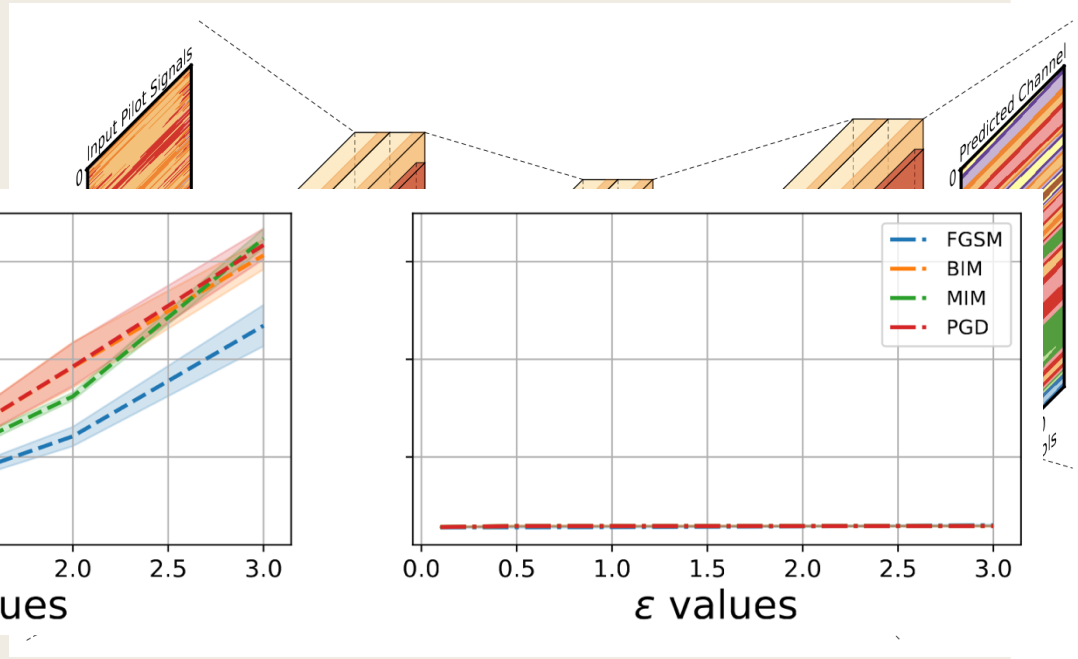
Beamforming is **a type of radio frequency (RF) management in which a wireless signal is directed toward a specific receiving device**.



1. Catak, F.O., et al. "Security concerns on machine learning solutions for 6G networks in mmWave beam prediction." *Physical Communication* (2022)
2. Catak, F.O., et al. "Adversarial machine learning security problems for 6G: mmWave beam prediction use-case." *2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. IEEE, 2021.

Research 2: Channel Estimation





Channel estimation is **essential in a MIMO wireless communication in 5G**. In the MIMO system, numerous antennas are utilized on the sender and receiver sides for enhancing spectral efficiency and reliability. The channel estimation can improve the exactness of the received signal.

1. Catak, F.O., et al. "Defensive Distillation based Adversarial Attacks Mitigation Method for Channel Estimation using Deep Learning Models in Next-Generation Wireless Networks." *IEEE Access* (2022): Under Review.

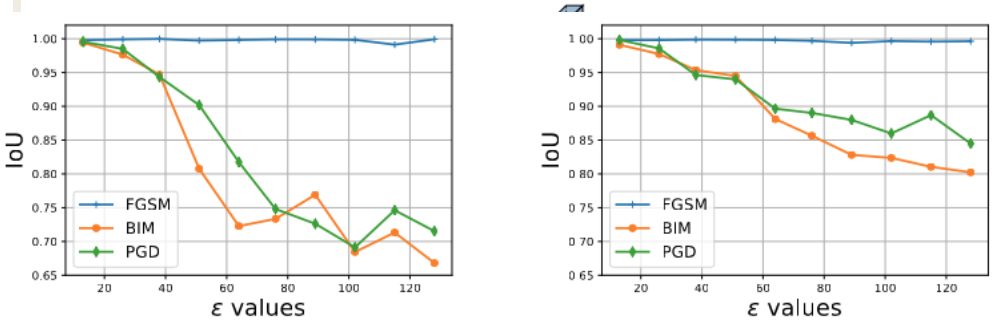
Research 3: Spectrum Sensing

TABLE III: Undefended

ϵ	Metrics	FGSM	BIM	PGD
13	Accuracy	0.999485	0.999107	0.999485
	Recall	0.999418	0.998906	0.999418
	Precision	0.999316	0.998817	0.999316
	Specificity	0.999771	0.999604	0.999771
	F-Score	0.999365	0.998857	0.999365
	FPR	0.000229	0.000396	0.000229
	IoU	0.998733	0.997723	0.998733
64	Accuracy	0.999367	0.859735	0.920404
	Recall	0.999514	0.862166	0.908336
	Precision	0.998738	0.837363	0.905915
	Specificity	0.999733	0.940058	0.965371
	F-Score	0.999111	0.813058	0.883668
	FPR	0.000267	0.059942	0.034629
	IoU	0.998253	0.722727	0.817422
128	Accuracy	0.999678	0.817982	0.845656
	Recall	0.999688	0.825223	0.863721
	Precision	0.999568	0.789182	0.814875
	Specificity	0.999855	0.914792	0.933859
	F-Score	0.999626	0.761299	0.803141
	FPR	0.000145	0.085208	0.066141
	IoU	0.999256	0.668107	0.715293

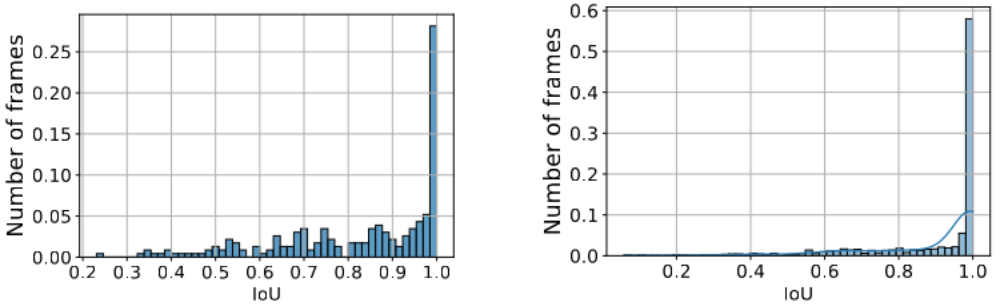
TABLE IV: Defended

ϵ	Metrics	FGSM	BIM	PGD
13	Accuracy	0.999485	0.999107	0.999485
	Recall	0.999418	0.998906	0.999418
	Precision	0.999316	0.998817	0.999316
	Specificity	0.999771	0.999604	0.999771
	F-Score	0.999365	0.998857	0.999365
	FPR	0.000229	0.000396	0.000229
	IoU	0.998733	0.997723	0.998733
64	Accuracy	0.999559	0.938550	0.944829
	Recall	0.999418	0.999170	0.938213
	Precision	0.999064	0.930787	0.941644
	Specificity	0.999811	0.972697	0.974934
	F-Score	0.999102	0.919045	0.930235
	FPR	0.000189	0.027303	0.025066
	IoU	0.998237	0.881291	0.896488
128	Accuracy	0.998690	0.882667	0.904468
	Recall	0.998900	0.865154	0.9067030
	Precision	0.997399	0.880294	0.910402
	Specificity	0.999467	0.934967	0.956826
	F-Score	0.998007	0.840553	0.884733
	FPR	0.000533	0.065033	0.043174
	IoU	0.996308	0.802265	0.844832



(a) Undefended (b) Defensive distillation

Fig. 4: IoU metric values of the adversarial attacks



(a) Undefended (b) Defensive distillation

Fig. 7: PGD

IEEE GlobeCom (2022), Under review.

My Publications

- Conference Slides