

# A Decision-driven Methodology for Designing Uncertainty-aware AI Self-Assessment

Gregory Canal<sup>1</sup>, Vladimir Leung<sup>1</sup>, Philip Sage<sup>1</sup>, Eric Heim<sup>2</sup>, and I-Jeng Wang<sup>1</sup>

<sup>1</sup>The Johns Hopkins University Applied Physics Laboratory

{Greg.Canal,Vladimir.Leung,Philip.Sage,I-Jeng.Wang}@jhuapl.edu

<sup>2</sup>Software Engineering Institute

Carnegie Mellon University

etheim@sei.cmu.edu

## Abstract

Artificial intelligence (AI) has revolutionized decision-making processes and systems throughout society and, in particular, has emerged as a significant technology in high-impact scenarios of national interest. Yet, despite AI’s impressive predictive capabilities in controlled settings, it still suffers from a range of practical setbacks preventing its widespread use in various critical scenarios. In particular, it is generally unclear if a given AI system’s predictions can be *trusted* by decision-makers in downstream applications. To address the need for more transparent, robust, and trustworthy AI systems, a suite of tools has been developed to quantify the uncertainty of AI predictions and, more generally, enable AI to “self-assess” the reliability of its predictions. In this manuscript, we categorize methods for AI self-assessment along several key dimensions and provide guidelines for selecting and designing the appropriate method for a practitioner’s needs. In particular, we focus on uncertainty estimation techniques that consider the impact of self-assessment on the choices made by downstream decision-makers and on the resulting costs and benefits of decision outcomes. To demonstrate the utility of our methodology for self-assessment design, we illustrate its use for two realistic national-interest scenarios. This manuscript is a practical guide for machine learning engineers and AI system users to select the ideal self-assessment techniques for each problem.<sup>1</sup>

## 1 Introduction

In the past decade, the world has witnessed an explosion in the capabilities of artificial intelligence (AI) systems, and their use has proliferated throughout most corners of society. AI systems have been deployed in applications ranging from commercial and industrial uses, such as recommendation systems for social networks, advertising, and e-commerce, to platforms of national interest, including defense, healthcare, climate, and scientific sectors. While the power of generative and predictive AI systems to imitate, augment, and enhance human capabilities has become abundantly clear, arguably the most significant roadblock to fully deploying AI systems at scale in critical problem scenarios has been a lack of certainty about the reliability, robustness, and trustworthiness of their predictions and model outputs.

In general, AI systems can have their performance limited by a range of practical considerations, such as reproducing systematic bias present in their training data, failing to generalize to domains outside those encountered during training, overfitting to spurious correlations present in data, or lacking the ability to abstain from making unfounded predictions. In certain commercial applications such as online streaming, such predictive flaws are only moderately problematic since, in the worst case, a user may be provided with a suboptimal user experience (e.g., being recommended movies they would not typically enjoy) rather than suffering a catastrophic loss. However, in many problems of national interest, such shortcomings cannot

---

<sup>1</sup>Approved for open publication by the U.S. Department of Defense Office of Publication and Security Review on July 30, 2024.

be tolerated due to the high-impact nature of such settings, the large sway that AI predictions might have on downstream decisions, and the potential for poor decisions to cause catastrophic failures or significant opportunity costs.

To more fundamentally address shortcomings in the trustworthiness of AI systems, researchers and practitioners alike have developed various tools and techniques to increase AI transparency, reliability, and robustness. One general class of such approaches relies on a separation between two types of AI outputs: the specific *predictions* made by an AI system for a particular input and an associated *self-assessment* (SeA) measuring the AI's confidence in its predictions. By providing such a self-assessment, the AI makes a downstream decision-maker more fully aware of any uncertainty or sources of ambiguity in the AI predictions. In general, there are innumerable forms of self-assessment that an AI system might provide, and the set of techniques and literature satisfying the definition of self-assessment is vast, potentially including broad fields such as explainable and interpretable AI [1], [2].

One prominent category of self-assessment approaches are those methods that *quantify the degree of uncertainty* an AI has in its predictions (which we denote as *uncertainty-aware* self-assessment), allowing downstream decision-makers to weigh AI predictions by their associated confidence levels. In general, the confidence outputs supplied by self-assessment can drastically affect downstream decisions, depending on how the decision-maker takes these weights into account, along with the potential costs and risks associated with each candidate decision. For instance, a human decision-maker may only move forward with high-risk actions if the AI's associated confidence level is above a particular threshold. Similarly, self-assessment confidence levels can have a significant impact on algorithmic (non-human) decision-makers since optimal decision policies typically weigh each potential decision cost by the AI's estimated uncertainty and select the action with the smallest weighted cost. Through this lens, AI self-assessment techniques are a crucial component in the overall decision-making pipeline and should be selected and designed in a *decision-driven* manner.

As an illustrative example of how self-assessment uncertainty outputs can impact downstream decision costs, consider the following scenario: suppose a decision-maker is playing a game where she is given \$20 and then presented with a closed box that either contains \$100 or is empty. The decision-maker is aware of both possibilities but does not know which amount the box contains until it is opened. She is given the choice to keep her \$20 and end the game (leaving the box closed) or pay back the \$20 for the chance to open the box and keep whatever value is inside. If she opens the box and it contains \$100, she will walk away with a net profit of \$80, but if the box is empty, she will have lost her \$20 and walk away empty-handed (net profit of \$0). Although looking at the box gives her no information, she is aided by an AI that predicts whether the box contains the \$100 and also provides a self-assessment confidence level  $c$  between 0-100%.

If the decision-maker assumes that this confidence level is well-calibrated, she can calculate her expected profit from opening the box to be  $\$80 \times c$ . On the other hand, if she does not open the box and walks away, her net profit from playing the game is always \$20. Based on this calculation, opening the box is only worth the risk if  $\$80 \times c > \$20$ , which occurs when the AI's confidence  $c$  is at least 25%. If the decision-maker follows this logic, she will decide to pay \$20 to open the box if the AI thinks it contains \$100 with confidence at least 25%, and will walk away with her \$20 otherwise. Hence, the AI's self-assessed confidence level can have a tremendous impact on the decision process since the difference of a few confidence percentage points around 25% might result in a different downstream action. Conversely, in this context, the AI's confidence level does not need to be well-calibrated outside of this small percentage range since all that matters for decision-making is if the confidence is above or below 25% rather than its exact value.

**Our Contribution:** this manuscript presents a decision-driven methodology for selecting and designing uncertainty-aware AI self-assessment techniques. Although previous surveys and general frameworks related to self-assessment have been developed [3]–[8], to our knowledge no previous surveys emphasize the specific impact of self-assessment on downstream decisions and their associated costs, as we do here. Unlike some surveys, the intention of this manuscript is to provide *practical guidance* for machine learning (ML) engineers and system users to select between and evaluate various self-assessment methods. In Section 2, we provide an overview of the key attributes we use to categorize various self-assessment methods along with a set of guidelines for how a user might select appropriate self-assessment techniques for their problem, taking into consideration the nuances and constraints of their AI pipeline and downstream decision-making process.

In Section 3, we detail a range of self-assessment techniques that a practitioner might utilize and specifically categorize these based on their level of decision awareness. We note that these candidate methods should be viewed as a representative set of techniques in the literature rather than an exhaustive list, as the number

of available uncertainty quantification techniques continues to grow rapidly. Additionally, these presented methods should not be construed as a fixed list of methods for a practitioner to select a single method from. Instead, practitioners should utilize the framework presented in Section 2 to formulate the ideal self-assessment for their application, and take inspiration from the methods presented in Section 3, possibly combining components from various methods or using these components as a launching pad for a novel technique. In Section 4 we illustrate how our guidelines can be used for self-assessment selection and tuning in two notional examples of national interest. We conclude with a discussion of open research challenges in Section 5.

## 1.1 Mathematical framework

It will sometimes be useful to describe techniques using a standardized notation and mathematical framework to discuss uncertainty-aware, decision-driven self-assessment in a unified manner. To this end, we build on notation and concepts from Kirchenbauer, Oaks, and Heim [7] and Zhao, Kim, Sahoo, *et al.* [9] to develop a mathematical, decision-driven framework for general self-assessment. We depict this framework in its entirety in Figure 1, and describe each core component below. Although this framework is a useful tool for relating self-assessment techniques to one another and understanding them in the broader context of AI learning and decision making, **it is not necessary for a reader to understand this mathematical framework to make use of our methodology**. Therefore, the mathematical framework below should be considered optional, and readers interested in higher-level self-assessment guidance can skip to Section 2.

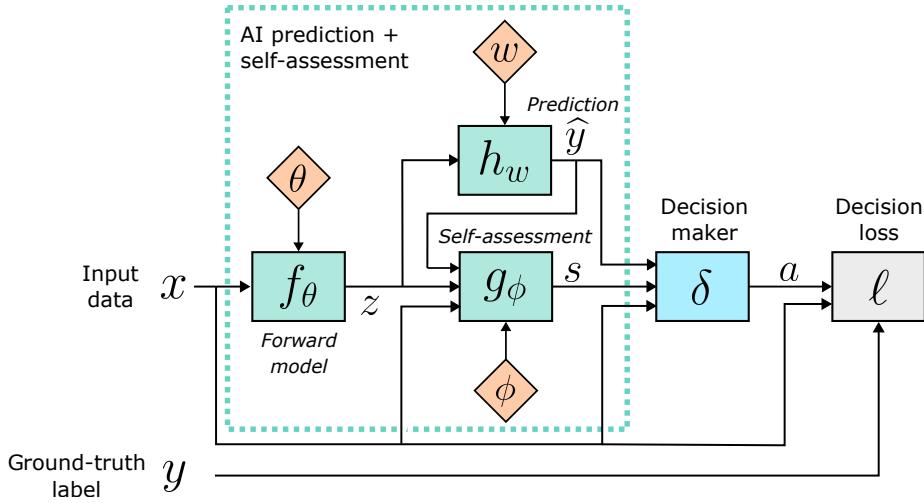


Figure 1: Mathematical framework for decision-driven AI self-assessment.

**AI predictive model** Let  $\mathcal{X}$  denote a data domain of interest (e.g.,  $\mathcal{X} = \mathbb{R}^d$ ) with individual examples denoted by  $x$ . Each example  $x$  is associated with a label  $y \in \mathcal{Y}$ , such as  $\mathcal{Y} = 1 \dots k$  for classification over  $k$  classes. Without loss of generality, we describe the AI model as a composition of two stages: a “forward model” containing the bulk of AI computation (i.e., forward pass in a neural network) and a “prediction model” converting the output of this computation into a specific prediction. Letting  $\mathcal{Z}$  denote an intermediate space to be defined shortly, we define the AI “forward” model as  $f_\theta: \mathcal{X} \rightarrow \mathcal{Z}$ , parameterized by the parameter set  $\theta$  in parameter space  $\Theta$ . We define the secondary “prediction” function  $h_w: \mathcal{Z} \rightarrow \hat{\mathcal{Y}}$ , parameterized by  $w$  in parameter space  $\mathcal{W}$ , as a post-processing function mapping the intermediate representation  $z \in \mathcal{Z}$  to a specific prediction  $\hat{y} \in \hat{\mathcal{Y}}$ , where  $\hat{\mathcal{Y}}$  is a space of predicted labels. For a given example  $x$ , the AI model’s prediction is then given by the composition  $\hat{y} = h_w(f_\theta(x))$ . We separately define the intermediate representation  $z \in \mathcal{Z}$  and the specific model prediction  $\hat{y} \in \mathcal{Y}$  to make our framework as general as possible and allow for a distinction between intermediate variables produced by a model (which may be used in subsequent self-assessment) and the ultimate label prediction.

For instance, in a typical deep learning scenario,  $\mathcal{Z} = \Delta^{k-1}$  is the simplex of probability distributions over  $k$  classes, and  $f_\theta$  is a neural network with weights  $\theta$  that outputs a predictive label distribution  $\hat{p} \in \Delta^{k-1}$  over  $k$  classes. In the standard setting, the prediction label set  $\hat{\mathcal{Y}}$  is equal to the true label set  $\mathcal{Y}$ . However, this is not always the case, such as in models that can abstain from making specific predictions by allowing for an “I don’t know” prediction  $\perp$ . To arrive at a single label prediction, the maximum value of  $\hat{p} = f_\theta(x)$  is taken as the predicted label, which in our framework can be described by defining the function  $h = \arg \max_i \hat{p}_i$ , where  $\hat{p}_i$  is the  $i$ th entry of  $\hat{p}$ , and where  $\mathcal{W} = \emptyset$  since this predictive mapping does not require additional parameters.

**AI self-assessment** Beyond the specific predictions  $\hat{\mathcal{Y}}$  produced by an AI model, we define  $\mathcal{S}$  as the space of outputs produced by a corresponding self-assessment technique. For example, one popular approach to self-assessment is for the model to produce a scalar probability indicating its confidence in its prediction  $\hat{y}$ . In this case,  $\mathcal{S} = [0, 1]$  is the unit interval on which this confidence value lies. To produce a self-assessment output, we define the mapping  $g: \mathcal{X} \times \mathcal{Z} \times \hat{\mathcal{Y}} \rightarrow \mathcal{S}$ , parameterized by  $\phi \in \Phi$ , since in the most general scenario the self-assessment might depend on the input data  $x$ , forward representation  $z = f_\theta(x)$ , and predicted output  $\hat{y} = h_w(f_\theta(x))$ , as  $s = g_\phi(x, f_\theta(x), h_w(f_\theta(x)))$ . As an example, to describe the scalar confidence self-assessment described above, suppose that the forward model outputs  $\hat{p} = f_\theta(x)$  (where  $\mathcal{Z} = \Delta^{k-1}$ ). Then, defining  $g = \max_i \hat{p}_i$ , we can write  $s = g(f_\theta(x))$ , where in this case  $g$  does not depend on  $x$  or  $\hat{y}$  directly (but could in general) and does not have separate parameters  $\phi$  (so  $\Phi = \emptyset$ ). Note that in some approaches where the output of  $f_\theta$  relies on stochastic elements (e.g., dropout), the self-assessment stage  $g$  may require access to multiple forward passes to arrive at an uncertainty estimate (as in methods based on ensembles or Monte Carlo samples). Other self-assessment methods may rely only on knowledge of the forward model’s parameters  $\theta$ , without requiring additional uncertainty parameters. For clarity we do not indicate all possible dependencies in Figure 1 for every self-assessment scenario, and instead adopt a general framework that covers many use cases.

**Downstream decision-making** Suppose that a decision-maker (e.g., human user, downstream software) observes the AI model’s prediction  $\hat{y}$  and self-assessment  $s$  to inform a downstream decision over a discrete set of possible actions  $a \in \mathcal{A}$ . We formally define a decision-making policy  $\delta: \mathcal{X} \times \hat{\mathcal{Y}} \times \mathcal{S} \rightarrow \Delta^{|\mathcal{A}|-1}$  that takes as input the current example  $x$ , AI prediction  $\hat{y}$ , and self-assessment  $s$  and outputs a probability distribution over the set  $\mathcal{A}$  of possible actions. We assume without loss of generality<sup>2</sup> that an action is then sampled from this probability distribution  $\delta(x, \hat{y}, s)$ . Since the decision-maker is typically unaware of the true label  $y$ , we assume that  $a \sim \delta(x, \hat{y}, s)$  is conditionally independent of  $y$ , given  $x$ .

We define a loss function  $\ell: \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \rightarrow \mathbb{R}$  that measures the “cost” of each decision, as follows: given input example  $x$  with ground-truth label  $y$ , if the decision-maker selects action  $a$ , then a cost of  $\ell(x, y, a)$  is incurred; such a loss formulation is consistent with standard concepts in decision theory. Furthermore, this formulation generalizes common losses such as classification error, in which case  $\mathcal{A} = \mathcal{Y}$  and  $\ell(x, y, a) = \mathbf{1}[a \neq y]$ . In some frameworks such as Zhao, Kim, Sahoo, *et al.* [9], the loss function only depends on label  $y$  and action  $a$ , but for generality, we include a dependence of  $\ell$  on  $x$ . For a distribution  $\mathcal{D}_{X,Y}$  over example/label pairs  $(x, y)$ , we define the *expected decision cost* as  $C = \mathbb{E}_{x,y \sim \mathcal{D}_{X,Y}} \mathbb{E}_{a \sim \delta(x, \hat{y}, s)} [\ell(x, y, a)]$ . Generally speaking, for a given decision policy  $\delta$  and data drawn from distribution  $\mathcal{D}_{X,Y}$ , a system designer should choose the AI forward model  $f_\theta$ , predictive model  $h_w$ , and self-assessment  $g_\phi$  to jointly minimize the overall downstream decision cost  $C$ .

## 2 Self-assessment attributes and design guidelines

There is a wide spectrum of approaches and techniques developed for uncertainty quantification of AI predictions; see, for example, Abdar, Pourpanah, Hussain, *et al.* [4] and Gawlikowski, Tassi, Ali, *et al.* [5]. **To categorize this varied range of techniques in a manner that aids practitioners, we organize methods according to attributes that are important factors for designing and evaluating decision-driven AI self-assessments.** In the following, we identify these key attributes and provide examples of

---

<sup>2</sup>This probabilistic definition still allows for the possibility of deterministic decision policies by simply defining  $\delta$  as selecting a single action with probability 1.

their possible values. Some examples of how existing uncertainty estimation techniques can be characterized by their associated attributes are given in Tables 1 and 2, which will be described in more detail in Section 3.

## 2.1 AI task

Uncertainty estimation is focused on characterizing the uncertainty of the AI model’s output/prediction given an input. Hence, the relevant representation and estimation mechanisms for uncertainty will naturally depend on the underlying AI task. It is also important to consider the desired performance for the underlying AI tasks, as applying some uncertainty estimation techniques may impact the AI’s performance for its given task. The majority of existing research on uncertainty estimation has been focused on classification and regression, but recent efforts have started examining a broader range of AI tasks:

- Classification with performance measures based on the prediction confusion matrix, rather than average accuracy
- Object detection
- Regression
- State estimation (tracking)
- Segmentation
- Reinforcement learning
- Generative tasks

Although uncertainty quantification techniques have been developed for several of these tasks (e.g., regression [10]–[13], object detection [14]–[17]), here we mostly focus on classification problems due to their ubiquity across a variety of practical AI problem settings. It would be straightforward to apply our framework more broadly to various other AI tasks, and we leave to future work an in-depth study of decision-driven self-assessment for non-classification AI tasks such as those listed here.

## 2.2 Uncertainty representation

An important feature of an uncertainty estimation technique is which notion of uncertainty is estimated and how it is represented. Different notions of uncertainty will naturally call for different representations and technical approaches. We identify the most common types of representations below:

- **Scalar confidence:** often interpreted as an estimate of the posterior probability of the model’s decision, given the observations.
- **Confidence interval/set:** compact set in the output space with a probability bound (often referred to as the “coverage” for confidence intervals); could be an interval for continuous output or a discrete set for categorical output.
- **Parametric density/distribution:** An approximation to the output distribution, often through approximation with a Gaussian distribution (first and second moments).
- **Out-of-distribution (OOD) score/probability:** A measure of likelihood that an input to the model “comes from” a distribution distinct from the training data distribution. OOD is generally ill-defined without further assumptions on the underlying distributional shifts [18]–[20]. Conceptually, a high OOD score for an input implies that the model trained and calibrated with training data provides limited information on the “true” prediction for the input.

## 2.3 Generic metrics

Given a representation of AI uncertainty, there are “generic” metrics that measure the quality of the underlying estimation problem. We refer to these as “generic” since they often do not consider specific impacts on downstream decisions. In selecting and tuning decision-driven self-assessment techniques, these generic metrics may not be the appropriate metric to optimize directly without further accounting for downstream decision-making. However, these metrics are often central to designing analogous metrics that directly take downstream decision costs into consideration.

- **Metrics for scalar confidence:** Expected calibration error (ECE) [21], maximum calibration error (MCE) [21], negative log likelihood (NLL), Brier score, class-wise ECE, adaptive calibration error (ACE) [22]. Such metrics for calibration error have been generalized in a unified metric known as Generalized Expected Calibration Error (GECE) [7].
- **Metrics for confidence interval/set:** Prediction Interval Coverage Probability (PICP) and Mean Prediction Interval Width (MPIW) are the standard metrics to measure the associated accuracy and “tightness/specificity” of confidence intervals. We can generalize their definitions for discrete confidence sets by defining an appropriate notion for the “size” of a set (e.g., by cardinality).
- **Metrics for parametric distribution:** Several metrics can be used to measure the quality of the distributional approximation including negative log likelihood and an estimation of the Kullback-Leibler or related notions of divergence.
- **Metrics for OOD:** The standard approach to evaluate OOD performance is to view the problem as a binary classification and resort to metrics such as the area under the receiver operating characteristic (AUROC) and the area under the precision-recall curve (AUPRC). It is important to characterize the anticipated distribution shifts (e.g., covariate versus concept shifts) OOD is intended to address to interpret these metrics accordingly (see [23]).

## 2.4 Estimation mechanisms

Uncertainty estimation can be broadly categorized into the following three mechanisms, differentiated by when they occur in training, and how much additional modeling is required beyond the native model  $f_\theta$ :

- **Post-hoc techniques:** These techniques are applied to a trained ML model without further tuning the forward model weights  $\theta$ . Instead, post-hoc adjustments are made to the parameters  $\phi$  determining the behavior of the self-assessment stage  $g_\phi$  itself.
- **Integral to training:** These techniques require access to the training process of the AI model, i.e., the training algorithm for  $f_\theta$ . For instance, MC dropout [24] utilizes the same dropout mechanism used during training to sample from a posterior distribution of network weights, rather than requiring a separate re-training step. Similarly, ensemble methods [25] rely on training multiple candidate models, and the disagreement between such models captures a notion of uncertainty.
- **Intrinsic to model:** rather than introducing self-assessment as an auxiliary modification for an AI model originally designed for prediction-only, there exists a class of self-assessment approaches where the elements of uncertainty quantification are *intrinsic* to the predictive model itself. For instance, Bayesian Neural Networks explicitly maintain a probability distribution over network weights. Typically, self-assessment techniques that are “intrinsic to the model” involve fundamental changes to the model itself, such as significant architectural modifications (e.g., outputting the mean and variance of a distribution rather than a pointwise prediction).

It is important to consider the underlying mechanism when considering an uncertainty estimation approach as the AI/ML model may have been trained *a priori* and prevent the applications of approaches requiring fundamental changes to training and/or model architecture selection. In addition, approaches intrinsic to the model will likely impact the performance of the underlying ML task.

## 2.5 Design parameters

Given the variety of metrics that might be used to assess the performance of any particular self-assessment technique, it is natural to utilize these metrics to guide the selection of which self-assessment technique to deploy and how to set any relevant *SeA design parameters* governing self-assessment behavior. As discussed in Section 1.1 and Figure 1, when possible we distinguish between the parameters  $\theta$  involved in the AI “forward” model making label predictions, and any SeA design parameters  $\phi$  involved in tuning the self-assessment model itself.<sup>3</sup> While the relevant SeA design parameters depend on the specific technique being utilized, a few representative examples include:

- Temperature  $T > 0$  controlling the “sharpness” of an estimated label distribution [26]. In the notation of our framework let  $\mathcal{Z} = \mathbb{R}^k$ , define  $z = f_\theta(x)$  as the “logit” vector output by a forward model, define  $\mathcal{S} = \Delta^{k-1}$ ,  $\phi = \{T\}$ , and let  $\sigma(\cdot)$  notate the softmax operation. We can then “scale” an AI model’s output label distribution by letting  $g_\phi(z) = \sigma(z/T)$ . Depending on the value of  $T$ , this operation softens or sharpens an estimated label distribution from the forward model to ensure that the scaled distribution is well-calibrated or satisfies other properties.
- Error rate  $0 < \alpha < 1$  utilized in forming a conformal set [27]. In our framework, let  $\mathcal{Z} = \mathbb{R}^k$  and  $z = f_\theta(x)$  denote a score vector over  $k$  classes,  $\phi = \{\alpha\}$ , and let  $\mathcal{S} = P(\hat{\mathcal{Y}})$  be the power set (set of all subsets) of  $\hat{\mathcal{Y}}$ . Then  $g_\phi(x) = \{y' : f_\theta(x)[y'] \geq \tau\}$  where  $f_\theta(x)[y']$  is the entry of  $f_\theta(x)$  corresponding to label  $y'$ , and  $\tau$  is computed such that on average over a held-out calibration set  $g_\phi(x)$  contains the true label  $y$  with probability at least  $1 - \alpha$ .

In certain settings, such SeA design parameters might be numerically optimized by an algorithm minimizing a relevant loss function (e.g., optimizing temperature  $T$  to minimize negative log-likelihood [26]). In other cases, a practitioner might set such parameters manually by monitoring their effect on downstream evaluation metrics of self-assessment performance, or may select parameters based on desired SeA characteristics (e.g., error rate  $\alpha$  in conformal prediction). Here, we take a broad perspective and loosely refer to the process of either numerically adjusting or hand-tuning SeA design parameters as “optimization.” Furthermore, in typical machine learning nomenclature one would distinguish between parameters that directly tune SeA model behavior and any *hyperparameters* associated with SeA design, which can also be tuned or selected from to affect self-assessment performance. Examples of such hyperparameters might include:

- Binning scheme (number of bins, bin edges) utilized in the histogram-based calibration of estimated label distributions [21].
- Tuning regularization parameters in a loss function for training  $g_\theta$ , including choosing a different loss function altogether. Deciding on a different loss function can affect, for instance, whether or not a self-assessment technique is risk-aware.
- Costs/weights on different prediction errors or levels of self-assessment confidence (e.g., overconfidence can be costly since a human decision-maker may blindly trust the results [28]).
- Selection of calibration set used for tuning self-assessment parameters.
- Choice of models used in an ensemble and how uncertainty is estimated from this ensemble.

In a slight abuse of terminology, for conciseness we continue to refer to such hyperparameters as simply being other SeA “design parameters” that a practitioner might tune or optimize. What all of the “parameters” described in this section have in common is that they are all intrinsic to the self-assessment technique, and need to be set by some means.

---

<sup>3</sup>As mentioned in Section 1.1, it may not always be possible to make this distinction, especially in SeA approaches that are intrinsic to the AI model or integral to training. For example, the dropout probability  $p$  in Monte Carlo Dropout [24] affects both regularization during forward model training, and the label distribution obtained during self-assessment. In such cases, we might still consider such shared parameters between AI prediction and self-assessment to be “tunable” by a practitioner.

## 2.6 Downstream decisions

Even though not an attribute of an estimation technique, the downstream decision-maker that will “consume” the output from the AI model and its self-assessment shall dictate the applicable uncertainty representations. For example, a human user may not be able to interpret a complex label distribution to arrive at an informed decision, and it may be better to provide an uncalibrated yet interpretable distribution instead. Beyond compatibility with the decision-maker, the output of self-assessment mapping  $g_\theta$  may drastically affect which downstream decisions are made, even for non-human decision-makers. For example, a downstream Kalman-filter based tracking algorithm may assume a covariance matrix (to characterize observation uncertainty) associated with each detection to combine detections from multiple sensors. If this covariance matrix is provided by the self-assessment model  $g_\theta$ , then  $g$ , in effect, significantly impacts the downstream performance of the tracking algorithm.

While many self-assessment techniques are designed to increase decision-maker trust in AI predictions and effectively convey uncertainty, it is not always the case that explicit effects on downstream decisions and associated costs are considered during self-assessment design and optimization. Many self-assessment techniques are derived, optimized, and evaluated based on a set of “generic” statistical metrics measuring various properties of their uncertainty outputs (e.g., distribution calibration, see Section 2.3), which are usually calculated independently from subsequent downstream decisions; here, we refer to such methods as being *decision-agnostic*. This stands in contrast to self-assessment techniques that *explicitly* consider self-assessment outputs’ effects on downstream decisions, which we categorize as being *decision-aware*. For example, as described in Section 3.2, certain self-assessment approaches optimize SeA parameters to minimize the expected downstream decision cost explicitly.

## 2.7 Guided self-assessment design

The key attributes outlined above define the important “dimensions” to consider when selecting, tuning, and evaluating uncertainty estimation for AI self-assessment. As these dimensions are not necessarily orthogonal, it is unlikely that a monolithic decision process (i.e., a single decision tree) exists for all use cases to guide a user towards an optimal self-assessment technique. Here, we postulate a likely “decision flow” (depicted in Figure 2) to illustrate a typical self-assessment design process:

1. Identify candidate uncertainty estimation approaches based on the **AI task** at hand;
  - (a) If a model has been trained, then only the **post-hoc** approaches are applicable.
  - (b) Determine if an approach could have a significant negative impact on the associated task-specific **performance measure**.
2. Determine the appropriate **uncertainty representation** taking into account the model (if already trained) and the **downstream decision** (what the downstream decision-maker can consume).
3. Examine the published benchmark on the **generic metrics** associated with the uncertainty representation to develop a qualitative analysis of candidate approaches.
4. Identify tunable **optimization parameters** for the candidate approaches and derive an optimization criterion following our decision-theoretic framework.
5. Optimize uncertainty estimation against the optimization criterion and evaluate the performance based on the expected cost associated with the **downstream decision**.

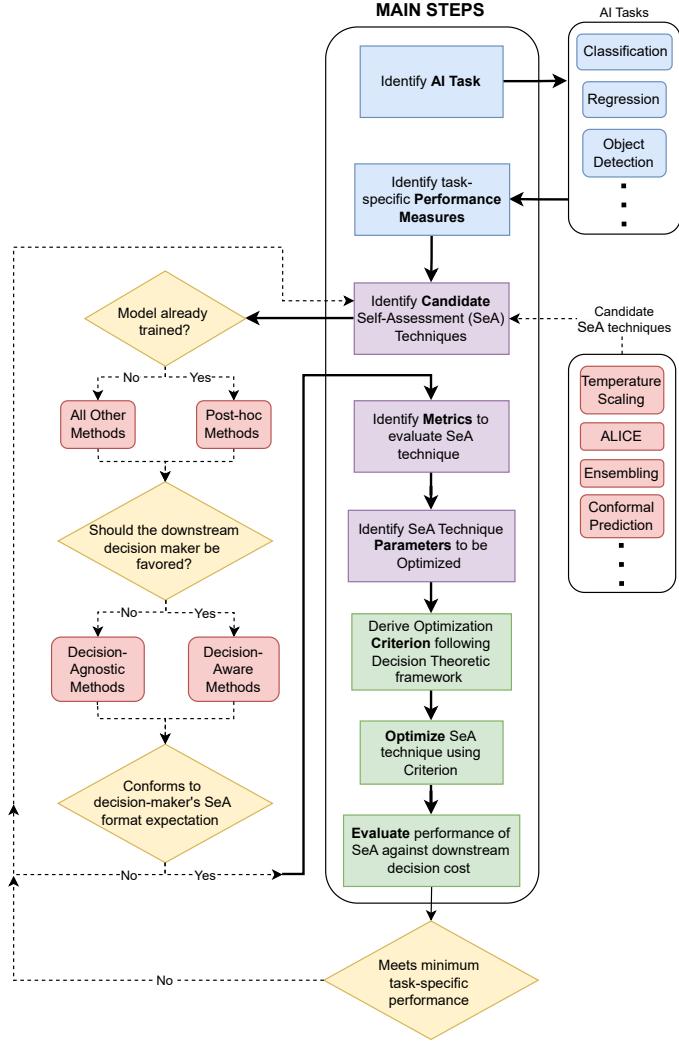


Figure 2: Methodology for a practitioner to downselect and optimize candidate self-assessment techniques for their application at hand. For candidate self-assessment techniques, refer to Section 3 along with Tables 1 and 2. For illustrative examples utilizing this methodology, refer to Section 4.

### 3 Overview of self-assessment techniques

By following the guidelines presented in Section 2.7 and in Figure 2, a user can arrive at a method for self-assessment appropriate for their particular problem scenario. In this section, we provide an overview of relevant self-assessment techniques and describe them according to the attributes presented in Section 2. The set of techniques considered here should be considered a representative, but not necessarily exhaustive, suite of complementary approaches for self-assessment. As new approaches are developed, they can easily be categorized along the dimensions of Section 2 and added here. In Section 3.1, we detail various approaches to *decision-agnostic* self-assessment (summarized in Table 1), followed by a representative suite of *decision-aware* techniques in Section 3.2 (summarized in Table 2).

#### 3.1 Decision-agnostic self-assessment

In this section we present a representative overview of popular and distinctive methods for decision-agnostic self-assessment (summarized in Table 1). First, we outline approaches rooted in post-hoc adjustment of a model’s class label distribution to minimize various notions of calibration error. Then, we pivot to discuss methods with uncertainty representations intrinsic to the model or integral to the model’s training process. Finally, we briefly mention several techniques with noteworthy uncertainty representations distinct from direct estimates of label or model uncertainty.

**Post-hoc approaches** One of the most common approaches to self-assessment is to output the estimated probability of a model’s prediction as a scalar notion of “confidence” by letting  $h(z) = \arg \max_{y'} z_{y'}$  and  $g(z) = \max_{y'} \sigma(z)[y']$ , where  $z$  is a vector of logits output by an AI forward model,  $\sigma(z) = \text{softmax}(z)$ , and  $\sigma(z)[y']$  denotes the softmax probability corresponding to class  $y'$ . However, it has generally been established that such a confidence measure is not necessarily well-calibrated, especially for modern neural networks [26]. A simple and practical strategy to remedy this miscalibration is to apply post-hoc calibration by sharpening or smoothing the estimated confidence scores with a temperature parameter  $T > 0$  by letting  $g_T(z) = \max_{y'} \sigma(z/T)[y']$ . For small values of  $T$ , the model places higher confidence in its prediction, and for larger temperature values, the output probabilities are “softened,” resulting in higher model uncertainty. Such *temperature scaling* for uncertainty quantification was introduced in Guo, Pleiss, Sun, *et al.* [26] and extended to other scalar notions of confidence by Yona, Feder, and Laish [29].

Building on this approach, Guo, Pleiss, Sun, *et al.* [26] introduced an extension to temperature scaling where a linear transformation  $\mathbf{W}z + \mathbf{b}$  is applied to the logits  $z$  before the softmax operator in lieu of a simple scaling by  $1/T$ . Kull, Perello Nieto, Kängsepp, *et al.* [30] build on this concept even further by applying such a transform to *log probabilities* before subsequently applying softmax, which was shown to be equivalent to learning a Dirichlet distribution over the class labels as a means of calibration. Rather than using a parametric transformation such as temperature or matrix scaling to calibrate an output distribution, Zadrozny and Elkan [31] introduced a non-parametric approach to post-hoc calibration known as *histogram binning*. In this approach (applicable to binary classification with  $k = 2$ ), data points are first binned according to their sorted predicted probabilities, and then each point’s positive class probability is estimated by computing the fraction of positive instances empirically observed in the bin (theoretical guarantees were shown for histogram binning in Gupta and Ramdas [32]). Pakdaman Naeini, Cooper, and Hauskrecht [21] extend this technique by simultaneously employing multiple binning schemes in a Bayesian model to address the limitations of histogram binning associated with having fixed bin edges. We note that principles from scaling-based methods and histogram binning have been combined in a scaling-binning calibrator that enjoys theoretical guarantees on calibration error [33].

**Integral to training / intrinsic to model** Beyond self-assessment methods that applying a post-hoc re-calibration method to model outputs, as discussed in Section 2 there exist other approaches tied to model training itself, which sometimes include uncertainty representation components intrinsic to the forward model. Such approaches vary in their degree of explicitness in how uncertainty is represented. At one extreme, deep ensembles [25] train separate networks with varying initializations, and average their output predictions to implicitly account for model uncertainty. Similarly in *MC Dropout*, uncertainty is represented implicitly by

making multiple stochastic forward passes through a model trained with dropout, which has been shown to approximately solve a variational inference problem [24].

At the other end of the extreme, Blundell, Cornebise, Kavukcuoglu, *et al.* [34] directly model a posterior distribution over network weights via a variational Gaussian approximation, explicitly maintaining a mean and variance for each network weight. Training such variational inference approaches can either rely on Monte Carlo sampling as in Blundell, Cornebise, Kavukcuoglu, *et al.* [34] or can instead leverage analytic evidence lower bound evaluation [35]. Stochastic Weight Averaging-Gaussian (SWAG) takes a similar approach by maintaining a Gaussian posterior distribution for model weights that includes low-rank elements [36]. Epinets [37] utilize a sampling scheme as in Lakshminarayanan, Pritzel, and Blundell [25] to maintain an ensemble of models, but also introduce a new architecture to supplement existing base networks that separates out deterministic network components from stochastic components (Kendall and Gal [38] also use a network modification to model both aleatoric and epistemic uncertainties). Unlike epinets and Bayesian Neural Networks, Kumar, Sarawagi, and Jain [39] do not modify the network’s architecture and simply output a conditional label distribution over each class. However, they introduce Maximum Mean Calibration Error (MMCE) as a means of regularizing network training to learn a calibrated output distribution.

**Distinctive uncertainty representations** There exists a wide variety of self-assessment techniques with uncertainty representations that are somewhat distinctive from the approaches described so far in this section. One major category of such methods is those techniques performing *out-of-distribution* (OOD) detection, to determine whether a given input point  $x$  lies “within” the data distribution of points that a model is expected to generalize on. Detecting out-of-distribution points can potentially serve as a valuable form of self-assessment by hedging a model’s prediction with the fact that the point is “unlike” the training data. Within OOD detection, there are various types of data distribution shifts that might constitute a point being “out of distribution,” including both semantic and covariate shifts [23]. It has recently been argued in Yang, Zhang, and Russakovsky [23] that modern OOD approaches may not perform as expected under these various types of shifts, and instead the authors provide empirical support for a simple baseline method known as Maximum Softmax Probability (MSP) [40] which takes  $g(z) = \mathbb{1}[\max_{y'} \sigma(z)[y'] > \tau]$  as an indicator of a point’s In or Out of distribution status. There are a wide variety of other approaches for OOD detection (see Yang, Zhou, Li, *et al.* [3] for a comprehensive survey), including score-based methods proposing alternate scoring functions besides MSP to measure a point’s In/Out status (such as energy-based methods [41]).

To enable informative evaluation and fair comparison of OOD detection techniques, it is important that we recognize whether a method makes an explicit assumption on the nature of the distribution shift it was designed to detect. For example, techniques developed specifically for label shift such as [20] are not expected to perform well for detection of covariate shift. Other techniques developed without explicit assumptions on distributional shifts such as MSP may have disparate performance across different shifts. Such an understanding is critical for identifying candidate approaches that are likely to succeed in addressing the distributional shifts anticipated for each application.

A self-assessment approach similar in spirit to OOD detection is *learning to reject*, where a model not only makes label predictions but learns when to abstain from making predictions on points it is unsure about [42], mitigating the risk of providing a downstream decision-maker with an untrustworthy prediction (specifically, abstaining on those potentially erroneous predictions that might incur a large cost if they lead to poor downstream decisions). Learning to reject is a self-assessment type worthy of consideration when interfacing with a downstream decision-maker, and we refer interested readers to the surveys in [3], [43] for additional details and references.

There have also been related concepts proposed in the literature such as *atypicality* estimation, which for a given point provides a measure of how likely such a point is to occur in a training distribution, rather than explicitly detecting In/Out status. Yuksekgonul, Zhang, Zou, *et al.* [44] utilize this concept in developing an atypicality-driven post-hoc re-calibration method (AAR). Rajendran and LeVine [45] combine several concepts such as label distribution calibration with OOD detection to derive a post-hoc self-assessment score known as ALICE, providing a conservative estimate for a model’s confidence that takes into account both label distribution estimation and distribution shifts. Furthermore, ALICE does not simply re-calibrate a distribution or detect points as In/Out: instead, it combines these concepts into a pointwise notion of *competence* estimating the probability that a model’s error is below a specified threshold.

Another major approach utilizing an entirely distinct uncertainty representation is the class of methods

known by *conformal prediction* [27]. In conformal prediction, the self-assessment model does not output any direct probability estimates (either with respect to the input distribution or label distribution) to a downstream decision-maker. Instead, uncertainty is conveyed *implicitly* to the decision maker by expanding the set of predictions returned beyond a single prediction  $\hat{y}$ . Here, a set of predictions  $\mathcal{C}$  is returned that is selected to contain the true label  $y$  with probability  $1 - \alpha$  for a pre-specified value of  $\alpha$ . In fact, this approach enjoys rigorous theoretical guarantees bounding the probability that  $\mathcal{C}$  contains  $y$  towards  $1 - \alpha$  (such guarantees are made in a marginal sense; see Angelopoulos and Bates [27] for discussion). Conformal prediction sets have the benefit of potentially being more interpretable to downstream decision-makers, since they make statements about inclusion or exclusion in the prediction set  $\mathcal{C}$  rather than ascribing numerical notions of probability, which may be difficult to interpret by human decision-makers.

| Technique  | AI task  | Uncertainty representation   | Generic metrics  | Estimation mechanism  | Tuning design parameters  |
|--|--|--|--|---|---|
| <b>Post-hoc methods</b>  |  |  |  |   |   |
| Temperature scaling [26]   | Multi-class classification                                     | Scalar confidence (prediction probability)   | ECE, Brier score, NLL  | Post-hoc minimization of NLL (requires calibration set)   | Temperature $T > 0$   |
| Temperature scaling with alternate confidence measures [29]                          | Multi-class classification                                     | Scalar confidence options: maximum probability, difference of top-2 probabilities, weighted difference of top-3 probabilities, label entropy | $\ell_2$ calibration error, sharpness (variation in error across confidence values), ECE, ACE, evaluated with fixed and adaptive binning | Post-hoc minimization of $\ell_2$ calibration error (requires calibration set)  | Temperature $T > 0$   |
| Dirichlet calibration [30]   | Multi-class classification                                     | Label distribution   | Accuracy, log-loss, Brier score, MCE, ECE, classwise-ECE, Significance measures  | Post-hoc (calibration set required) minimization of log-loss with ODIR regularizer  | Linear weights and bias $\mathbf{W}, \mathbf{b}$ ; ODIR regularization hyperparameters $\lambda, \mu$   |
| Bayesian Binning into Quantiles (BBQ) [21]   | Binary classification  | Scalar probability   | Discrimination power (Accuracy, area under ROC curve), Calibration error (RMSE, ECE, MCE)  | Non-parametric calibration: post-hoc binning model averaging over possible binning models   | Binning model: number of bins, partitioning of predictions into bins, Beta distribution parameters (per bin)  |
| <b>Integral to training / intrinsic to model</b>                                     |  |  |  |   |   |
| Deep ensembles [25]  | Multi-class classification, regression                         | Output distribution $p(y   x)$ averaged over mixture   | NLL, RMSE (regression), classification error, Brier Score (classification)   | Independently train each network from random initialization with adversarial training (integral to training, intrinsic to model)                  | Proper scoring rule $\ell$ , Number of ensemble networks $M$ , Adversarial training perturbation size $\epsilon$  |
| Monte Carlo Dropout [24]   | Multi-class classification, regression, reinforcement learning | Label distribution   | RMSE, log likelihood   | Train model with dropout, sample from distribution with stochastic forward iterations   | Dropout probability $p$ , number of forward iterations $T$  |
| Bayes by Backprop [34]   | Multi-class classification, regression, contextual bandits     | Posterior distribution over network weights (diagonal Gaussian)  | Variational free energy  | Gradient descent with unbiased Monte Carlo gradients (integral to training, intrinsic to model)   | Gaussian mean $\mu$ and variance $\sigma$ vectors, prior parameters $\pi, \sigma_1, \sigma_2$   |
| SWAG [36]  | Multi-class classification, regression                         | Gaussian distribution over model weights, average label distribution over models   | NLL, confidence-accuracy difference  | Average first and second moments over SGD iterates  | Gaussian mean $\theta_{\text{SWA}}$ , Gaussian diagonal covariance $\Sigma_{\text{diag}}$ and low-rank covariance columns $\widehat{D}$                                 |
| Epinet (Epistemic Neural Networks) [37]  | Multi-class classification                                     | Reference distribution over epistemic indices, stochastic epinet addition to model output  | Classification error, marginal log-loss, joint log-loss  | Log loss with $\ell_2$ regularizer minimized over base network (non-stochastic) + epinet (stochastic)   | $\ell_2$ hyperparameter $\lambda$ , prior network, index dimension  |
| Maximum Mean Calibration Error (MMCE) [39]   | Multi-class classification                                     | Scalar confidence  | ECE, Brier score, NLL  | Optimize model parameters over NLL + MMCE loss (integral to training, intrinsic to model)   | MMCE penalty $\lambda$ , kernel $k(\cdot, \cdot)$   |
| <b>Distinctive uncertainty representations</b>                                       |  |  |  |   |   |
| Thresholding-based OOD detection (e.g., maximum softmax probability) [3], [23], [40] | Multi-class classification                                     | In-distribution (ID) scoring   | AUROC/AUPR for error detection, In/Out detection, new class detection  | Threshold scoring function $g(z)$ (e.g., $g(z) = \mathbb{1}[\max_{y'} \sigma(z)[y'] > \tau]$ ) as in MSP [40]) to determine In/Out status         | ID scoring function, ID scoring threshold $\tau$  |
| Atypicality-Aware Recalibration (AAR) [44]   | Multi-class classification                                     | Pointwise atypicality estimates: input atypicality, class atypicality  | Classification accuracy, ECE   | Post-hoc fitting of penultimate GMM from training data (maximum likelihood), minimize cross-entropy loss of AAR distribution over calibration set | Penultimate layer Gaussian mixture means $\widehat{\mu}_c$ and shared covariance $\widehat{\Sigma}$ , Class-wise tunable scores, Quadratic coefficients $c_2, c_1, c_0$ |
| Accurate layerwise interpretable competency estimation (ALICE) [45]                  | Multi-class classification                                     | Pointwise competence   | mean Average Precision (mAP)   | Competence estimator calculated by combining calibrated label distribution with distance-based OOD measure  | Error threshold $\delta$ , choice of error function $\mathcal{E}$   |
| Conformal prediction [27]  | Supervised learning (general)                                  | Prediction set over outputs  | PICP / MPIW  | Score ranking over calibration set, quantile thresholding   | Score function $s(x, y)$ , coverage error rate $\alpha$   |

Table 1: Overview of decision-agnostic self-assessment techniques.

### 3.2 Decision-aware self-assessment

Qualitatively, the mere presence of AI self-assessment may encourage downstream decision-makers about the trustworthiness of the AI’s predictions. However, to quantitatively benefit downstream decision processes, the self-assessment must directly improve metrics measuring downstream decision quality. In the context of our mathematical framework, this is achieved by a self-assessment strategy being designed and tuned to optimize the expected downstream decision cost  $C$  with respect to a cost function  $\ell(x, y, a)$  that is appropriate for a user’s particular application (see Section 1 and Figure 1).

Choosing self-assessment strategies that optimize or at least account for such downstream decision costs goes a step beyond optimizing “generic metrics” for self-assessment that may not result in optimal downstream decisions. For example, *uncalibrated* label distributions taking into account the nuances of human psychology can sometimes result in more optimal human decision-maker performance than by using calibrated distributions that don’t explicitly take the decision-maker into account. In this vein, there is a significant body of human factors research analyzing the joint performance of human-AI decision-making systems under various types of self-assessment [28], [46], [47]. In recent years, a suite of self-assessment techniques has emerged that are explicitly designed to take these factors into account and optimize downstream decision costs; we highlight a representative set of such methods here (summarized in Table 2).

Marx, Zalouk, and Ermon [48] develop a general, kernel-based calibration method that is integral to model training, where different selections of kernels result in certain types of calibration being optimized. Kumar, Sarawagi, and Jain [39] can be seen as a special case of this more general framework. As an example, a kernel based on downstream decision losses is presented as a special case of their more general framework, and evaluated on a threshold-based decision-making task. This approach strives to minimize Decision Calibration Error (DCE) which measures the difference between the expected decision cost as calculated from the self-assessment distribution, versus the true expected decision cost. By ensuring a low DCE, downstream-decision makers can trust that estimates of decision cost calculated from the self-assessment output are reliable estimates of the true cost for each candidate decision, to better inform decision-making.

This work builds on the earlier work in Zhao, Kim, Sahoo, *et al.* [9], which first introduced the notion of decision calibration and devised an algorithm to re-calibrate a given label distribution  $\hat{p}$  to minimize decision calibration error. However, this earlier work is calibrated to the set of decision losses over a fixed number of actions rather than being calibrated with respect to a specific decision loss  $\ell$ , and is only applicable to Bayes optimal decision makers  $\delta$ , which may not be applicable in real-world scenarios. Sahoo, Zhao, Chen, *et al.* [49] address the setting of thresholded decisions on a regression model, and present an algorithm for re-calibrating a probability distribution over regression values to minimize the difference between estimated and true losses incurred under a threshold decision model.

Although Zhao, Kim, Sahoo, *et al.* [9] and Marx, Zalouk, and Ermon [48] take into account downstream decision costs in their self-assessment tuning, they only allow for a Bayes optimal decision maker  $\delta$ , which is likely to be unrealistic in practical settings. Instead, Vodrahalli, Gerstenberg, and Zou [50] address this design aspect explicitly by *modeling* the decision-maker  $\delta$  using a separate predictive model. Once  $\delta$  is estimated from human behavior training data, along the lines of temperature scaling the AI self-assessment (in this context, called the AI “advice”) notated by  $A$  is re-calibrated with parameters  $\alpha, \beta \geq 0$  to arrive at a re-calibrated scalar confidence value  $g(A) = 1/(1 + \exp(-\text{sign}(A)(\alpha|A| + \beta)))$ . These constants are selected to minimize the binary cross-entropy loss of the human decision-maker’s (modeled via  $\delta$ ) ultimate decision, after viewing the re-calibrated AI “advice.” While presenting a promising concept, the author’s also note that this framework is “not currently suitable for practical use.”

In a similar vein, [51] uses an estimate of the decision-maker’s confusion matrix over classes  $\hat{Y}$  with respect to the ground-truth class  $y$  to choose an optimal coverage probability for a conformal set to minimize the decision-maker’s ultimate probability of decision error. Kerrigan, Smyth, and Steyvers [52] similarly utilize confusion matrix estimates to combine human and AI decisions into joint decisions that outperform either the AI or human making decisions on their own. Bansal, Nushi, Kamar, *et al.* [53] expand the considered downstream loss function  $\ell$  to account for unequal costs of correct or incorrect decisions. The authors utilize these cost parameters along with several decision-theoretic assumptions to derive a novel loss function equal to the expected utility, which is used directly for model training.

In contrast to many of the decision-agnostic methods described in the previous section, what unifies the decision-aware approaches presented here is their *explicit* consideration of the downstream decision maker

in designing an appropriate SeA method, taking the resulting decision costs into account. This class of methods holds promise as an avenue to ensure that the joint human-AI *system* is performing optimally in a given problem scenario. Explicitly incorporating downstream decision costs into the selection and optimization of self-assessment techniques is a relatively nascent area of research, and there are still many open directions in designing self-assessment strategies that truly bridge the gap between decision theory and practical applications (see Section 5 for a discussion of potential research avenues).

| Technique                                | AI task   | Uncertainty representation   | Generic metrics  | Estimation mechanism  | Tuning design parameters  |
|--|---|--|--|---|---|
| Maximum mean discrepancy [48]            | Multi-class classification, regression                                  | Label distribution   | Decision Calibration Error (DCE): $\ell_2$ error of expected decision loss   | Training time minimization of NLL + Maximum Mean Discrepancy (MMD) (no post-hoc calibration set)  | MMD tradeoff parameter $\lambda$ , choice of kernel $k(\cdot, \cdot)$   |
| Decision calibration [9]                 | Multi-class classification  | Label distribution   | Decision calibration error   | Post-hoc re-calibration of initial label distribution (requires calibration set). Non-specific decision loss $\ell$ (calibrates to all loss functions over fixed number of actions), limited to Bayes optimal decision-makers | Decision-calibration error tolerance $\varepsilon$  |
| Threshold calibration [49]               | Regression, binary action space (based on thresholded regression value) | Cumulative distribution function over regression values (“forecaster”) | Reliability gap (estimated vs. true decision cost), true decision cost incurred  | Given uncalibrated forecaster, sequentially re-calibrate using isotonic regression (requires validation set) until threshold calibration error (TCE) minimized  | Minimum TCE $\varepsilon > 0$   |
| Decision-aware sigmoid scaling [50]      | Binary classification   | Scalar confidence  | Difference in final accuracy, human confidence in correct decision, rate of human decision change in response to AI advice | Train human behavior model in response to AI advice (human behavior training data required), calibrate AI output to minimize cross-entropy loss of human classification decision  | $\alpha, \beta \geq 0$ constants for scaling and shifting AI confidence   |
| Decision-aware conformal prediction [51] | Multi-class classification  | Conformal set  | Decision-maker probability of error  | Given known or estimated confusion matrix of decision-maker $\delta$ , optimization algorithm for optimal $\alpha$ to minimize decision error probability (requires calibration set)  | Conformal error probability $0 < \alpha < 1$  |
| P+L (probability and human labeler) [52] | Multi-class classification  | Label distribution   | Classification error rate, ECE, classwise ECE, NLL   | Post-hoc AI model calibration (requires calibration set), estimate human confusion matrix (requires human labels), combine with Bayes rule  | Dirichlet prior parameters $\gamma > 0$ , $\beta > 0$ ; Gaussian prior $\mathcal{N}(\mu, \sigma^2)$ on $\log T$ for temperature scaling |
| Optimizing joint decision utility [53]   | Multi-class classification  | Class label distribution   | Accuracy, expected utility, empirical utility  | Minimize expected utility over training set   | Utility parameters $\beta, \lambda$ ; human accuracy $a \in [0, 1]$   |

Table 2: Overview of decision-aware self-assessment techniques.

## 4 Notional examples

In this section, we briefly illustrate how our self-assessment selection and design methodology might be applied to two realistic scenarios of national interest where AI transparency is crucial in informing downstream decision-making. To demonstrate the breadth of our approach to many problem settings, we discuss examples with human and autonomous decision-makers (i.e.,  $\delta$ ), which observe self-assessment outputs to make downstream decisions. We first describe a disaster relief scenario with a human decision-maker, followed by a UAV ISR tracking application with an autonomous decision-maker.

### 4.1 Disaster relief triaging (human decision-maker)

Medical *triaging* during disaster relief scenarios is key to effectively applying limited medical resources to prioritize various casualty needs appropriately. As an example of a realistic disaster relief scenario and triaging effort, we consider a scenario inspired by the DARPA Triage Challenge<sup>4</sup> (DTC), which breaks triaging into two stages of primary and secondary triaging:

- *Primary triaging* consists of identifying the severity of a situation, the presence of casualties, injury severity levels, and who requires hands-on medical evaluation or interventions.
- *Secondary triaging* involves placing non-invasive sensors on these casualties to determine critical vs. non-critical conditions.

<sup>4</sup>Further information on the challenge is available at <https://triagechallenge.darpa.mil/index>.

DARPA provides various datasets to tackle primary and secondary triaging. These range from Uncrewed Aircraft Vehicle (UAV) and Uncrewed Ground Vehicle (UGV) imagery of high-fidelity simulated scenarios, to purely tabular casualty medical information (Figure 3a). At first glance, this complex, data-rich scenario presents a prime opportunity for AI algorithm development and automatic resource optimization. However, applying medical resources to casualties is inherently safety-critical, and therefore any use of AI in this setting must be accompanied by appropriate SeA, to increase overall trust in medical decisions being made. Furthermore, to maintain a human-in-the-loop in this safety-critical setting, we assume that a human operator makes the final decisions after receiving the AI’s predictions and SeA, rather than having medical decisions being fully automated.

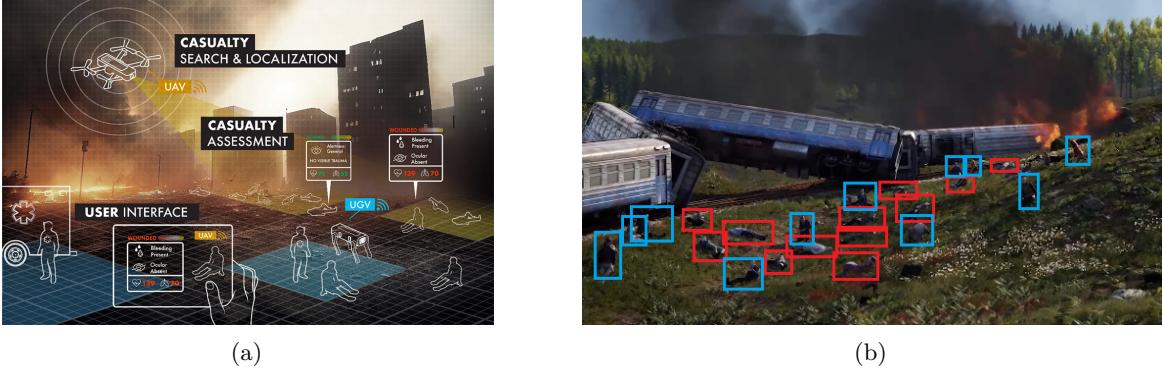


Figure 3: (a) In settings similar to the DARPA Triage Challenge, autonomous agents are leveraged to assess medical triaging at scale. (b) In our notational use case, we envision an object-detection based triaging system to classify casualties based on predicted severity level. Images adapted and modified from <https://triagechallenge.darpa.mil/index>.

#### 4.1.1 Problem setup

In the notional disaster triaging scenario we consider, the primary mission is to maximize the number of saved lives by appropriately deploying resources to casualties present in the scene. We envision this task being performed by a human deciding where to deploy medical resources and personnel based on the outputs of an Object Detection and Classification model (stylization in Figure 3b). Specifically, we assume the model receives as input high-resolution UAV RGB imagery captured by an expert operator (each image instance denoted by  $x$  in our framework in Figure 1). We assume that the images will cover the entire incident area, ensuring all viewable casualties are within at least one frame and no frames are overlapping.

Each casualty is associated with a “ground-truth” class label ( $y$ ) given by one of four states, corresponding to various levels of injury severity: **Healthy**,<sup>5</sup> **Delayed**, **Immediate**, or **Expectant**. These labels are a simplified mapping from recorded ground-truth labels, which include Trauma, Alertness, and Vitals presence/continuous values alongside the presence of critical factors, Severe Hemorrhaging and Respiratory distress. For this notional example, we assume there is a meaningful mapping from these health descriptors to injury severity. The Object Detection algorithm will determine each casualty location, while the Classifier predicts casualty state ( $\hat{y}$ ). We assume the forward model ( $f_\theta$ ) is a combined Object Detection and Classifier model (e.g., YOLO [54]) that outputs as  $z$  bounding box coordinates and a probability vector  $p(x)$  containing the confidence of each class.<sup>6</sup> As feedback to the human decision-maker, we assume these outputs are visualized by superimposing the output bounding boxes with their corresponding predicted injury state. We also assume that the visualization allows for a single scalar confidence value to be presented with each prediction.

Based on this visualization, the human decision-maker ( $\delta$ ) then takes one of three possible actions ( $a$ ), per casualty bounding box: **no action**, **deploy medical personnel**, **evacuate**. Each of these actions requires

<sup>5</sup>Here we slightly abuse the term “casualty” by allowing for the possibility that an identified person within the scene is actually unharmed.

<sup>6</sup>For the sake of illustration, we assume that the Object Detector’s localization performance is flawless and that classification is the only task requiring confidence assessments.

a different level of resource utilization, which can be interpreted as a literal “cost”: the lack of an action does not deplete any resources while deploying medical personnel or ordering an evacuation requires increasing allocated resources. Furthermore, each action is potentially associated with an “opportunity cost,” depending on the ground-truth injury state of each casualty. For instance, deploying medical personnel to an identified casualty who is actually healthy may constitute a waste of resources, while failing to evacuate someone with a severe injury may result in a critical failure to save that person’s life. We combine these various notions of “cost” into a single cost function  $\ell(y, a)$ , delineating the decision cost of all class-decision pairs (Table 3). Designing an appropriate cost matrix is a non-trivial challenge that should be undertaken by a domain expert; here, for the sake of illustration, we assign notional values to each state-action pair to capture the various degrees of severity in making “incorrect” decisions. Each casualty bounding box is associated with its own cost value  $\ell(y, a)$  depending on the true state of the person in question and the ultimate action the downstream decision-maker takes.

| Cost Matrix        |           | Action ( $a$ ) |                          |          |
|--------------------|-----------|----------------|--------------------------|----------|
|                    |           | no action      | deploy medical personnel | evacuate |
| True State ( $y$ ) | healthy   | 0              | 30                       | 100      |
|                    | delayed   | 50             | 0                        | 30       |
|                    | immediate | 100            | 5                        | 0        |
|                    | expectant | 50             | 10                       | 20       |

Table 3: Cost matrix for disaster triaging scenario, depicting the decision cost  $\ell(y, a)$  over all state-action pairs. In this context, absolute magnitudes do not carry meaning, but relative magnitudes represent the severity of “incorrect” decisions, where lower values are more desirable and higher values correspond to more severe consequences.

#### 4.1.2 Utilizing self-assessment guidance

We now illustrate a notional process, utilizing the diagram in Figure 2, of identifying an appropriate self-assessment technique to aid a human decision-maker in taking actions that minimize the overall cost incurred when summed across casualties. Starting at the top of the flowchart, we identify the AI task as multi-class classification with classification accuracy as the performance measure since the Classifier component of interest in the forward model assigns one of three injury states to each bounding box casualty. In this example, we assume the forward model is already trained, and therefore we should utilize a post-hoc self-assessment method (assuming that a domain-specific calibration dataset is available). Since this problem setting is safety-critical with well-defined notions of cost (at least, in our notional description), we would like to leverage a decision-aware SeA method. By design, we assume that the forward model already outputs an estimated class probability vector  $p(x)$ , and that the visualization to the decision-maker allows for each bounding box state prediction to display an associated confidence value. Therefore, we limit ourselves to methods that output a scalar confidence to conform to the visualization constraints and the decision-maker’s expectations.

Based on these requirements, after studying the methods in Section 3 we identify *temperature scaling* as an appropriate technique, where we denote  $\tilde{p} = \sigma(\frac{1}{T} \log p(x))$  as a temperature-scaled probability vector and let  $g_T(p(x)) = \tilde{p}_y$  be the temperature-scaled probability value associated with the model’s prediction, constituting a single scalar confidence to be presented to the decision-maker. In this context, rather than seeking a temperature  $T$  that minimizes a calibration metric such as ECE, we instead seek to minimize the expected incurred cost  $\mathbb{E}[\ell(y, a)]$ , where this expectation is taken over the input data  $x$ , true labels  $y$ , as well as the (possibly stochastic) actions  $a$  taken by the decision maker. In general, the exact human-decision maker policy  $\delta$  given an image, predicted injury state, and SeA confidence will be unknown, preventing the direct optimization of this expected cost over  $T$ . However, for the sake of this example we will assume that an approximate model  $\hat{\delta}(\hat{y}, g)$  of human behavior is available (as a function of prediction  $\hat{y}$  and self-assessment confidence  $g$ ), which could possibly be fit to a dataset of recorded human decisions. With this approximate decision-maker policy available, we propose to directly optimize  $\min_T \mathbb{E}_{x, y \sim \mathcal{D}_{X, Y}} \mathbb{E}_{a \sim \hat{\delta}(\hat{y}, g_T(p(x)))} [\ell(y, a)]$  to arrive at a selected temperature value  $T$  for temperature scaling. While only notional, this process illustrates the utility of a formal approach for SeA selection in the context of a high-impact decision-driven problem.

## 4.2 Autonomous UAV ISR (algorithmic decision maker)

UAV-based Intelligence, Surveillance, and Reconnaissance (ISR) systems may sometimes benefit from an algorithmic decision-maker to autonomously complete a mission without explicit interventions or control from a human operator. However, decisions made by such algorithms can have varied consequences, especially in high-impact scenarios such as the battlespace. The potential consequences for certain actions, such as scanning an area of interest, may be mild (e.g., the worst-case outcome being increased fuel usage on an inefficient flight path that doesn't produce reportable intel), while other actions — such as tracking a specific entity — may pose greater levels of risk. Therefore, it is important for any AI tools utilized by an algorithmic decision-maker to be equipped with self-assessment, such that the decision algorithm can properly weigh the likelihood of various outcomes against their associated costs. Since different actions may have various levels of risk, it is important for SeA design in this context to explicitly take downstream algorithmic decisions into consideration.

### 4.2.1 Problem setup

In our notional example, we consider a scenario where an Autonomous UAV's mission is to find and follow adversary military vehicles (see stylization<sup>7</sup> in Figure 4). We envision a scenario where the UAV uses onboard Electro-Optical (EO) sensors to detect and classify potential vehicles to follow. In this simplified example, we assume that each vehicle belongs to one of three classes ( $y$ ): **friendly military vehicle**, **adversary military vehicle**, or **civilian vehicle**. For simplicity, we assume there will be exactly one vehicle of interest driving through the UAV's field of view at a time. As in Section 4.1, we assume that an onboard AI system receives as input a single frame of RGB imagery ( $x$ , in our framework) and that the forward model ( $f_\theta$ ) is a trained Object Detector that outputs ( $z$ ) bounding boxes around the vehicle and includes a Classifier that outputs a class prediction  $\hat{y}$  along with a probability vector  $g(x)$  estimating the likelihood of each class. For simplicity we assume that the Object Detector's localization performance is flawless, such that classification is the only task requiring self-assessment.

We assume that the onboard system toggles between two action modes: *scanning* and *following*. “Scanning” patrols an area in search of vehicles of possible interest while “following” pursues a target within the mission parameters. The downstream autonomous decision-maker will decide between these two actions based on the predicted class for each vehicle of interest. As in the disaster triage example, we describe a notional cost matrix in Table 4 depicting a cost value  $\ell(y, a)$  for each vehicle class-action pair that reflects the risks associated with each possibility (note that as in Section 4.1, only the relative differences between costs matter in this context, rather than the absolute scale). For example, we assign a cost of 50 to following a friendly military vehicle since this may constitute a waste of resources, but following a civilian is given a higher cost of 75 as this could be considered problematic. Deciding to “scan area” when an adversary military vehicle is actually in frame receives the highest loss of 100, which would be considered a mission failure. All other actions are considered “correct” and given a loss of 0. To choose scanning or following, an algorithmic decision-maker onboard the UAV (denoted by decision policy  $\delta$ ) expects as input a probability vector  $g$  over the three possible classes, and takes a corresponding action. Specifically, we assume that  $\delta$  is given by the Bayes’ optimal policy, i.e., select the action  $a$  that minimizes the expected decision cost for the probabilities in  $g$  as  $a = \arg \min_{a \in \{\text{scan, follow}\}} \frac{1}{3} \sum_{y'} g(y') \ell(y', a)$ . Hence, the probability vector  $g$  provided to this automated decision maker has a tremendous effect on the subsequent action taken.

### 4.2.2 Utilizing self-assessment guidance

As in Section 4.1, we can follow the guidance in Figure 2 to determine an appropriate self-assessment technique. In this notional example, the flowchart process is identical to that of the disaster triaging example until the question of SeA conforming to decision-maker expectations. Instead of expecting a scalar confidence, in this ISR example we assume that the algorithmic decision-maker expects to receive a full probability vector  $g$  over the three possible classes for each vehicle. Examining the decision-aware SeA methods in Table 2 (since we would like to use a SeA method aware of downstream decision costs), if we assume that a calibration

<sup>7</sup>UAV source material adapted from public domain imagery at <https://www.af.mil/News/Photos/igphoto/2000608254/mediaid/5461083/> (U.S. Air Force Photo / Lt. Col. Leslie Pratt). The appearance of U.S. Department of Defense (DoD) visual information does not imply or constitute DoD endorsement.

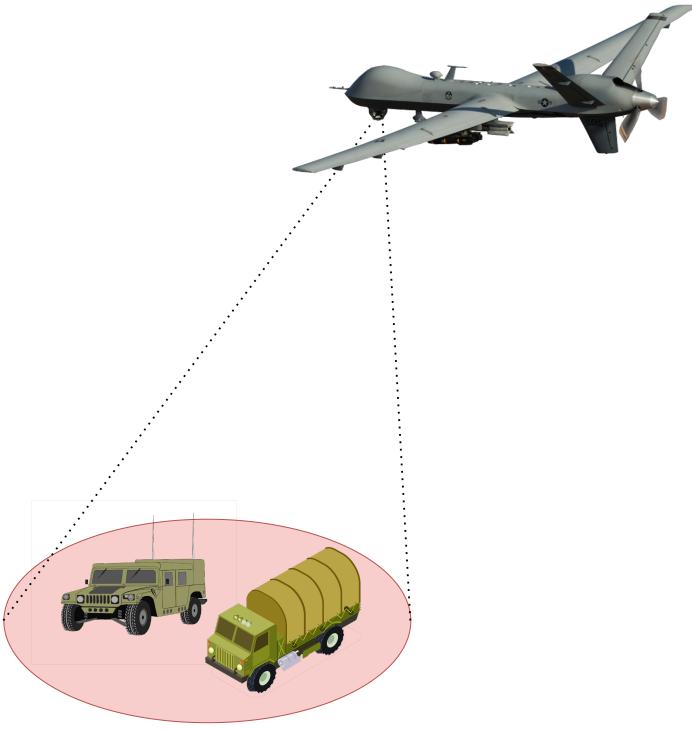


Figure 4: Stylization of UAV vehicle detection and following.

| Cost Matrix    |                    | Action $a$ |           |
|----------------|--------------------|------------|-----------|
|                |                    | Follow     | Scan Area |
| True Class $y$ | friendly military  | 50         | 0         |
|                | adversary military | 0          | 100       |
|                | civilian           | 75         | 0         |

Table 4: Cost matrix for the Autonomous UAV scenario. Absolute magnitudes do not carry meaning, but relative magnitudes represent the importance of “incorrect” decisions.

set is available<sup>8</sup> we can utilize the Decision Calibration technique in [9], which is specifically designed for Bayes’ optimal decision-makers. This method utilizes an iterative re-calibration algorithm to ensure that the expected cost (as computed using  $g$ ) is close to the true cost (in expectation over the true data distribution).

## 5 Discussion

In this manuscript, we have presented a methodology for designing and selecting self-assessment techniques to increase the trustworthiness and transparency of AI models, emphasizing methods driven by uncertainty quantification and awareness of how self-assessment outputs will ultimately be used in downstream decision-making. This manuscript has established an initial framework and formulation to select between and compare such self-assessment methods. While self-assessment techniques continue to be developed rapidly, the framework we establish here is general enough to categorize novel techniques according to the process outlined in Figure 2.

The suite of methods presented in Section 3 serves as a representative set of decision-agnostic and

<sup>8</sup>This is a reasonable assumption since before deploying a UAV-based ISR system it is conceivable to benchmark it on calibration runs.

decision-aware approaches, covering a range of uncertainty representations, generic metrics, and estimation mechanisms. Many of these methods are complementary and emphasize various components of the AI-driven decision-making pipeline, from the AI outputs to models for the downstream decision-maker and incorporation of anticipated decision costs. Even so, many self-assessment techniques discussed here can still make unrealistic assumptions about the information available for self-assessment selection and optimization, such as having complete knowledge of the downstream decision cost function  $\ell$  or decision-maker behavior  $\delta$ . Below we describe several core challenges to help bridge the gap between theoretical decision-driven self-assessment techniques and the next generation of practical self-assessment systems.

**Decision-maker estimation** While it may be the case that the decision-maker’s action policy  $\delta$  is known exactly or approximately in certain scenarios — such as with well-characterized human decision-makers or autonomous decision-makers with mathematically well-understood behaviors such as Kalman-filter based tracking algorithms — in many scenarios, the decision function  $\delta$  is unlikely to be known *a priori*. Furthermore, it is highly unlikely that realistic decision makers act rationally and Bayes’ optimally, as is assumed in several self-assessment techniques (e.g., [9]). Therefore, there is a crucial need for decision-aware self-assessment methods that do not rely on accurate models of the decision-making policy  $\delta$ , or instead learn such models “on the fly” at deployment. One promising avenue in this direction is described in Straitouri and Rodriguez [55], which does not assume knowledge of  $\delta$  and instead takes an online learning approach to optimize self-assessment parameters based on “in the loop” decision-maker feedback.

**Unknown decision costs** To our knowledge, the current set of available decision-aware self-assessment techniques all require knowledge of the downstream decision cost function  $\ell(x, y, a)$  assessing the cost incurred by each decision-maker action (informed by AI self-assessment). In practice, it may not always be the case that such a cost function is known explicitly, and may only be accessible implicitly by observing the incurred costs made by each decision at test time, in which case online learning will likely be advantageous as a self-assessment tuning paradigm. Alternatively, it may be that *implicit* knowledge of  $\ell$  is available in preference judgments between actions made by an auxiliary oracle. In this case, principles from learning from human preferences and reward learning could potentially be deployed to estimate a surrogate cost  $\ell$  from such relative feedback (see, for example, [56]).

**Limited calibration data** Many self-assessment methods (both decision-agnostic and decision-aware) require access to a held-out calibration dataset for fitting self-assessment parameters and choosing algorithm hyperparameters. However, in many practical scenarios there may not be the luxury of having such a large validation dataset available. This would be problematic in self-assessment techniques with heavy data requirements, such as Vodrahalli, Gerstenberg, and Zou [50], which first constructs an explicit model for a decision-maker  $\delta$ , based on behavioral response data. To mitigate such data costs, data-efficient techniques such as *active learning* [57] could potentially be deployed to only solicit and learn from the most informative test-time points. This judicious data selection could be deployed at multiple training stages, including learning the forward model  $f$ , self-assessment  $g$ , decision-maker  $\delta$ , and loss function  $\ell$ .

**Multi-use self-assessment** In this manuscript, we have described self-assessment in the context of a single decision-maker  $\delta$  and loss function  $\ell$ . However, in real-world scenarios these quantities may be inaccessible during self-assessment selection and tuning or may only be available during deployment. Nevertheless, it may be the case that  $\delta$  and  $\ell$  are known to belong to *classes* of decision-makers and downstream cost functions, respectively. In this case, self-assessment should be selected and tuned to be compatible with any decision-maker or downstream cost function within these options. Initial strides were made to study self-assessment over classes of decision costs in Zhao, Kim, Sahoo, *et al.* [9], but this formulation only applies to broad classes of losses under the assumption of Bayes’ optimal decision makers.

To our knowledge, there does not exist a single self-assessment method that addresses each of these practical challenges, which each needs to be addressed in real-world self-assessment selection, tuning, and deployment. These methodology gaps present a prime opportunity for novel decision-driven self-assessment research, development, and evaluation.

## Acknowledgement

This work was supported by the Office of the Under Secretary of Defense, Research and Engineering (OUSD (R&E)).

## References

- [1] R. Dwivedi, D. Dave, H. Naik, *et al.*, “Explainable ai (xai): Core ideas, techniques, and solutions,” *ACM Comput. Surv.*, vol. 55, no. 9, Jan. 2023, ISSN: 0360-0300. DOI: 10.1145/3561048. [Online]. Available: <https://doi.org/10.1145/3561048>.
- [2] D. Minh, H. X. Wang, Y. F. Li, and T. N. Nguyen, “Explainable artificial intelligence: A comprehensive review,” *Artificial Intelligence Review*, vol. 55, no. 5, pp. 3503–3568, Jun. 2022, ISSN: 1573-7462. DOI: 10.1007/s10462-021-10088-y. [Online]. Available: <https://doi.org/10.1007/s10462-021-10088-y>.
- [3] J. Yang, K. Zhou, Y. Li, and Z. Liu, “Generalized out-of-distribution detection: A survey,” *International Journal of Computer Vision*, pp. 1–28, 2024.
- [4] M. Abdar, F. Pourpanah, S. Hussain, *et al.*, “A review of uncertainty quantification in deep learning: Techniques, applications and challenges,” *Information Fusion*, vol. 76, pp. 243–297, 2021, ISSN: 1566-2535. DOI: <https://doi.org/10.1016/j.inffus.2021.05.008>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1566253521001081>.
- [5] J. Gawlikowski, C. R. N. Tassi, M. Ali, *et al.*, “A survey of uncertainty in deep neural networks,” *Artificial Intelligence Review*, vol. 56, no. 1, pp. 1513–1589, Oct. 2023, ISSN: 1573-7462. DOI: 10.1007/s10462-023-10562-9. [Online]. Available: <https://doi.org/10.1007/s10462-023-10562-9>.
- [6] S. Ghosh, Q. V. Liao, K. N. Ramamurthy, *et al.*, *Uncertainty quantification 360: A holistic toolkit for quantifying and communicating the uncertainty of ai*, 2021. arXiv: 2106.01410 [cs.AI].
- [7] J. Kirchenbauer, J. R. Oaks, and E. Heim, “What is your metric telling you? evaluating classifier calibration under context-specific definitions of reliability,” in *ML Evaluation Standards - ICLR 2022 Workshop*, 2022. [Online]. Available: [https://ml-eval.github.io/assets/pdf/Calibration\\_Evaluation.pdf](https://ml-eval.github.io/assets/pdf/Calibration_Evaluation.pdf).
- [8] Y. Chung, I. Char, H. Guo, J. Schneider, and W. Neiswanger, “Uncertainty toolbox: An open-source library for assessing, visualizing, and improving uncertainty quantification,” *arXiv preprint arXiv:2109.10254*, 2021.
- [9] S. Zhao, M. Kim, R. Sahoo, T. Ma, and S. Ermon, “Calibrating predictions to decisions: A novel approach to multi-class calibration,” in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34, Curran Associates, Inc., 2021, pp. 22 313–22 324. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/bbc92a647199b832ec90d7cf57074e9e-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/bbc92a647199b832ec90d7cf57074e9e-Paper.pdf).
- [10] V. Dheur and S. Ben Taieb, “A large-scale study of probabilistic calibration in neural network regression,” in *Proceedings of the 40th International Conference on Machine Learning*, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., ser. Proceedings of Machine Learning Research, vol. 202, PMLR, 23–29 Jul 2023, pp. 7813–7836. [Online]. Available: <https://proceedings.mlr.press/v202/dheur23a.html>.
- [11] V. Dheur and S. Ben Taieb, “Probabilistic calibration by design for neural network regression,” in *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, S. Dasgupta, S. Mandt, and Y. Li, Eds., ser. Proceedings of Machine Learning Research, vol. 238, PMLR, Feb. 2024, pp. 3133–3141. [Online]. Available: <https://proceedings.mlr.press/v238/dheur24a.html>.
- [12] H. Manh Bui and A. Liu, “Density-regression: Efficient and distance-aware deep regressor for uncertainty estimation under distribution shifts,” in *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, S. Dasgupta, S. Mandt, and Y. Li, Eds., ser. Proceedings of Machine Learning Research, vol. 238, PMLR, Feb. 2024, pp. 2998–3006. [Online]. Available: <https://proceedings.mlr.press/v238/manh-bui24a.html>.

- [13] A. Capone, S. Hirche, and G. Pleiss, “Sharp calibrated gaussian processes,” in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 36 579–36 590. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/7319b7561ffe5e2f6419acd4a2f52d6b-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/7319b7561ffe5e2f6419acd4a2f52d6b-Paper-Conference.pdf).
- [14] T. Popordanoska, A. Tiulpin, and M. B. Blaschko, “Beyond classification: Definition and density-based estimation of calibration in object detection,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2024, pp. 585–594.
- [15] K. Oksuz, T. Joy, and P. K. Dokania, “Towards building self-aware object detectors via reliable uncertainty quantification and calibration,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 9263–9274.
- [16] M. A. Munir, M. H. Khan, S. Khan, and F. S. Khan, “Bridging precision and confidence: A train-time loss for calibrating object detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 11 474–11 483.
- [17] B. Pathiraja, M. Gunawardhana, and M. H. Khan, “Multiclass confidence and localization calibration for object detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 19 734–19 743.
- [18] S. Garg, S. Balakrishnan, Z. C. Lipton, B. Neyshabur, and H. Sedghi, “Leveraging unlabeled data to predict out-of-distribution performance,” in *International Conference on Learning Representations*, 2022. [Online]. Available: [https://openreview.net/forum?id=o\\_HsiMPYh\\_x](https://openreview.net/forum?id=o_HsiMPYh_x).
- [19] S. B. David, T. Lu, T. Luu, and D. Pal, “Impossibility theorems for domain adaptation,” in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Y. W. Teh and M. Titterington, Eds., ser. Proceedings of Machine Learning Research, vol. 9, Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 129–136. [Online]. Available: <https://proceedings.mlr.press/v9/david10a.html>.
- [20] Z. Lipton, Y.-X. Wang, and A. Smola, “Detecting and correcting for label shift with black box predictors,” in *International conference on machine learning*, PMLR, 2018, pp. 3122–3130.
- [21] M. Pakdaman Naeini, G. Cooper, and M. Hauskrecht, “Obtaining well calibrated probabilities using bayesian binning,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, Feb. 2015. doi: 10.1609/aaai.v29i1.9602. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/9602>.
- [22] J. Nixon, M. W. Dusenberry, L. Zhang, G. Jerfel, and D. Tran, “Measuring calibration in deep learning.,” in *CVPR workshops*, vol. 2, 2019.
- [23] W. Yang, B. Zhang, and O. Russakovsky, “Imagenet-OOD: Deciphering modern out-of-distribution detection algorithms,” in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: <https://openreview.net/forum?id=VTYg5ykEGS>.
- [24] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” in *Proceedings of The 33rd International Conference on Machine Learning*, M. F. Balcan and K. Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1050–1059. [Online]. Available: <https://proceedings.mlr.press/v48/gal16.html>.
- [25] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, et al., Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf).
- [26] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On calibration of modern neural networks,” in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, Jun. 2017, pp. 1321–1330. [Online]. Available: <https://proceedings.mlr.press/v70/guo17a.html>.

- [27] A. N. Angelopoulos and S. Bates, “Conformal prediction: A gentle introduction,” *Foundations and Trends® in Machine Learning*, vol. 16, no. 4, pp. 494–591, 2023, ISSN: 1935-8237. DOI: 10.1561/2200000101. [Online]. Available: <http://dx.doi.org/10.1561/2200000101>.
- [28] N. Corvelo Benz and M. Rodriguez, “Human-aligned calibration for ai-assisted decision making,” in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 14 609–14 636. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/2f1d1196426ba84f47d115cac3dcb9d8-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/2f1d1196426ba84f47d115cac3dcb9d8-Paper-Conference.pdf).
- [29] G. Yona, A. Feder, and I. Laish, “Useful confidence measures: Beyond the max score,” in *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022. [Online]. Available: <https://openreview.net/forum?id=CqUMx8sFhb>.
- [30] M. Kull, M. Perello Nieto, M. Kängsepp, T. Silva Filho, H. Song, and P. Flach, “Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/8ca01ea920679a0fe3728441494041b9-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/8ca01ea920679a0fe3728441494041b9-Paper.pdf).
- [31] B. Zadrozny and C. Elkan, “Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers,” in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML ’01, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 609–616, ISBN: 1558607781.
- [32] C. Gupta and A. Ramdas, “Distribution-free calibration guarantees for histogram binning without sample splitting,” in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 18–24 Jul 2021, pp. 3942–3952. [Online]. Available: <https://proceedings.mlr.press/v139/gupta21b.html>.
- [33] A. Kumar, P. S. Liang, and T. Ma, “Verified uncertainty calibration,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/f8c0c968632845cd133308b1a494967f-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/f8c0c968632845cd133308b1a494967f-Paper.pdf).
- [34] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, “Weight uncertainty in neural network,” in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, Eds., ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, Jul. 2015, pp. 1613–1622. [Online]. Available: <https://proceedings.mlr.press/v37/blundell115.html>.
- [35] O. Wright, Y. Nakahira, and J. M. F. Moura, “An analytic solution to covariance propagation in neural networks,” in *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, S. Dasgupta, S. Mandt, and Y. Li, Eds., ser. Proceedings of Machine Learning Research, vol. 238, PMLR, Feb. 2024, pp. 4087–4095. [Online]. Available: <https://proceedings.mlr.press/v238/wright24a.html>.
- [36] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson, “A simple baseline for bayesian uncertainty in deep learning,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/118921efba23fc329e6560b27861f0c2-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/118921efba23fc329e6560b27861f0c2-Paper.pdf).
- [37] I. Osband, Z. Wen, S. M. Asghari, *et al.*, “Epistemic neural networks,” in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 2795–2823. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/07fbde96bee50f4e09303fd4f877c2f3-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/07fbde96bee50f4e09303fd4f877c2f3-Paper-Conference.pdf).
- [38] A. Kendall and Y. Gal, “What uncertainties do we need in bayesian deep learning for computer vision?” In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/2650d6089a6d640c5e85b2b88265dc2b-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/2650d6089a6d640c5e85b2b88265dc2b-Paper.pdf).

- [39] A. Kumar, S. Sarawagi, and U. Jain, “Trainable calibration measures for neural networks from kernel mean embeddings,” in *Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, vol. 80, PMLR, Oct. 2018, pp. 2805–2814. [Online]. Available: <https://proceedings.mlr.press/v80/kumar18a.html>.
- [40] D. Hendrycks and K. Gimpel, “A baseline for detecting misclassified and out-of-distribution examples in neural networks,” in *International Conference on Learning Representations*, 2017. [Online]. Available: <https://openreview.net/forum?id=Hkg4TI9x1>.
- [41] W. Liu, X. Wang, J. Owens, and Y. Li, “Energy-based out-of-distribution detection,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 21 464–21 475. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/f5496252609c43eb8a3d147ab9b9c006-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/f5496252609c43eb8a3d147ab9b9c006-Paper.pdf).
- [42] C. Cortes, G. DeSalvo, and M. Mohri, “Learning with rejection,” in *Algorithmic Learning Theory*, R. Ortner, H. U. Simon, and S. Zilles, Eds., Cham: Springer International Publishing, 2016, pp. 67–82, ISBN: 978-3-319-46379-7.
- [43] X.-Y. Zhang, G.-S. Xie, X. Li, T. Mei, and C.-L. Liu, “A survey on learning to reject,” *Proceedings of the IEEE*, vol. 111, no. 2, pp. 185–215, 2023. DOI: 10.1109/JPROC.2023.3238024.
- [44] M. Yuksekgonul, L. Zhang, J. Y. Zou, and C. Guestrin, “Beyond confidence: Reliable models should also consider atypicality,” in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 38 420–38 453. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/7900318ffaf5e9bc60250f134c6cc3c7-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/7900318ffaf5e9bc60250f134c6cc3c7-Paper-Conference.pdf).
- [45] V. Rajendran and W. LeVine, “Accurate layerwise interpretable competence estimation,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/a11da6bd58b95b334f8cd49f00918f16-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/a11da6bd58b95b334f8cd49f00918f16-Paper.pdf).
- [46] V. Babbar, U. Bhatt, and A. Weller, “On the utility of prediction sets in human-ai teams,” in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, L. D. Raedt, Ed., Main Track, International Joint Conferences on Artificial Intelligence Organization, Jul. 2022, pp. 2457–2463. DOI: 10.24963/ijcai.2022/341. [Online]. Available: <https://doi.org/10.24963/ijcai.2022/341>.
- [47] J. C. Cresswell, Y. Sui, B. Kumar, and N. Vouitsis, “Conformal prediction sets improve human decision making,” *arXiv preprint arXiv:2401.13744*, 2024.
- [48] C. Marx, S. Zalouk, and S. Ermon, “Calibration by distribution matching: Trainable kernel calibration metrics,” in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 25 910–25 928. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/52493d82db00e73abb2858a5a5f28717-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/52493d82db00e73abb2858a5a5f28717-Paper-Conference.pdf).
- [49] R. Sahoo, S. Zhao, A. Chen, and S. Ermon, “Reliable decisions with threshold calibration,” in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34, Curran Associates, Inc., 2021, pp. 1831–1844. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/0e65972dce68dad4d52d063967f0a705-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/0e65972dce68dad4d52d063967f0a705-Paper.pdf).
- [50] K. Vodrahalli, T. Gerstenberg, and J. Y. Zou, “Uncalibrated models can improve human-ai collaboration,” in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 4004–4016. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2022/file/1968ea7d985aa377e3a610b05fc79be0-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2022/file/1968ea7d985aa377e3a610b05fc79be0-Paper-Conference.pdf).

- [51] E. Straitouri, L. Wang, N. Okati, and M. Gomez Rodriguez, “Improving expert predictions with conformal prediction,” in *Proceedings of the 40th International Conference on Machine Learning*, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., ser. Proceedings of Machine Learning Research, vol. 202, PMLR, 23–29 Jul 2023, pp. 32 633–32 653. [Online]. Available: <https://proceedings.mlr.press/v202/straitouri23a.html>.
- [52] G. Kerrigan, P. Smyth, and M. Steyvers, “Combining human predictions with model probabilities via confusion matrices and calibration,” in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34, Curran Associates, Inc., 2021, pp. 4421–4434. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/234b941e88b755b7a72a1c1dd5022f30-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/234b941e88b755b7a72a1c1dd5022f30-Paper.pdf).
- [53] G. Bansal, B. Nushi, E. Kamar, E. Horvitz, and D. S. Weld, “Is the most accurate ai the best teammate? optimizing ai for teamwork,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, pp. 11 405–11 414, May 2021. DOI: 10.1609/aaai.v35i13.17359. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/17359>.
- [54] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016.
- [55] E. Straitouri and M. G. Rodriguez, “Designing decision support systems using counterfactual prediction sets,” *arXiv preprint arXiv:2306.03928*, 2023.
- [56] V. Myers, E. Biyik, and D. Sadigh, “Active reward learning from online preferences,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 7511–7518. DOI: 10.1109/ICRA48891.2023.10160439.
- [57] B. Settles, “Active learning literature survey,” 2009.