OpenL2D: A Benchmarking Framework for Learning to Defer in Human-AI Decision-Making

Anonymous Author(s)
Affiliation
Address
email

Abstract

Public resource limitations have significantly hindered the development and benchmarking of learning to defer (L2D) algorithms, which aim to optimally combine human and AI capabilities in hybrid decision-making systems. In such systems, human availability and domain-specific concerns introduce complexity, while obtaining human predictions for training and evaluation is costly. To overcome these challenges, we introduce OpenL2D, a novel framework designed to generate synthetic expert decisions and testbed settings for L2D methods. OpenL2D facilitates the creation of synthetic experts with adjustable bias and feature dependence, simulates realistic human work capacity constraints, and provides diverse training and testing scenarios. To demonstrate its utility, we employ OpenL2D on a public fraud detection dataset, synthesizing a team of 50 fraud analysts, and we benchmark L2D baselines under an array of 220 distinct testing scenarios. We believe that OpenL2D will serve as a pivotal instrument in facilitating a systematic, rigorous, reproducible, and transparent evaluation and comparison of L2D methods, thereby fostering the development of more synergistic human-AI collaboration in decision-making systems. Code for our framework and the instantiated public dataset is available at: `https://anonymous.4open.science/r/openl2d-7BD3`

1 Introduction

Recently, an increasing body of research has been dedicated to studying human-AI collaboration (HAIC), with several authors arguing that humans have complementary sets of strengths and weaknesses to those of AI [10, 11]. Collaborative systems have demonstrated that humans are able to rectify model predictions in specific instances [10], and have shown that humans, in collaboration with an AI model, may achieve complementary performance - a higher performance than the expert or the model on their own [16].

In this paper, we focus on the growing body of research surrounding the *learning to defer* (L2D) framework. These are algorithms that choose whether to assign an instance to a human or a ML model, aiming to take advantage of their complementary strengths. L2D algorithms require large amounts of data on human decisions: some require multiple human predictions per instance [29, 28], while others often require human predictions to exist for every single training instance [24, 26, 31, 13]. Due to the unavailability of large datasets containing human predictions, and the cost of obtaining large amounts of data reliably annotated by human experts, these methods are frequently developed with small datasets, containing limited human predictions, or by using synthetic human subjects. The

Submitted to the 37th Conference on Neural Information Processing Systems (NeurIPS 2023) Track on Datasets and Benchmarks. Do not distribute.

synthesized expert behavior is often simplistic, and varies significantly between authors. Due to these factors, research into L2D is lacking in robust comparison and benchmarking of different methods.

In this work we present *OpenL2D*, an open-source framework that allows for the generation of synthetic expert decisions, as well as training and testing scenarios for any tabular dataset. OpenL2D allows a user to (i) Create highly customizable synthetic experts, with control over feature dependence, bias towards a protected attribute and average performance. (ii) Generate realistic human work capacity constraints, limiting the amount of cases that can be deferred to humans. (iii) Generate a variety of training and testing scenarios, subject to previously defined capacity constraints. These allow for research into development and testing of L2D methods subject to real-world problems, such as changes in human availability and limited amounts of human prediction data. Furthermore, we provide an instantiation of OpenL2D on a publicly available fraud detection dataset, generating a team of 50 fraud analysts with varying properties. We also generate a constrained scenario with limited human predictions, using it to develop a capacity aware L2D algorithm. We benchmark two versions of our method as well as a capacity aware version of *rejection learning*, by testing their performance and fairness under 220 different testing scenarios. OpenL2D's code and our example instantiation are available at: `https://anonymous.4open.science/r/openl2d-7BD3`

2 Background and Related Work

In this section, we cover the current state-of-the-art L2D approaches, and the *rejection learning* framework upon which they expand. Then, we discuss the most commonly used datasets in L2D research and their shortcomings. Finally, we mention current methods of synthetic expert generation.

2.1 Current L2D Methods

One of the simplest deferral approaches in the literature is given by *rejection learning* (ReL) [7, 8]. In a human-AI collaboration setting, ReL defers instances from the model to humans [24, 28]. Its simplest implementation [14] produces uncertainty estimates for the model's prediction in each instance, ranking them, and rejecting to predict if the uncertainty is above a given threshold [7, 8].

Madras et al. [24] argue that ReL is sub-optimal because it does not consider the performance of the human(s) involved in the task, so they propose *learning to defer* (L2D). In the original L2D framework, the classifier and assignment system are jointly trained, taking into account a single model and a single human. Many authors have since contributed to the single-expert framework [26, 31]. Keswani et al. [20] observe that decisions can often be deferred to one or more humans out of a team, expanding L2D to the multi-expert setting [20]. Contrary to ReL, most L2D approaches require predictions from every human team member, for every training instance, imposing significant, and often unrealistic, data requirements. Furthermore, while L2D allows to set a deferral cost to control the amount of deferrals, the limited work capacity of each team member often goes unaddressed.

2.2 Current HAIC Datasets

To the best of our knowledge, there are only two public real-world datasets suitable for training multi-expert human-AI assignment systems, which we now describe. The NIH Clinical Center X-ray dataset [32], used by Hemmer et al. [13], is a computer vision dataset aimed at detecting airspace opacity. For each X-ray image, there are recorded predictions from an ensemble of 22 experts, and a golden label created by an independent team of 3 radiologists. The main drawback of this dataset is its size: it contains only 4,374 X-ray images. The Hate Speech and Offensive Language Detection dataset enriched by Keswani et al. [20], consists of a subset of 1,471 tweets from the original dataset [9], annotated by a total of 170 crowd-sourcing workers according to the presence of hate speech or offensive language. Each tweet was labelled by an average of 10 workers, meaning that each worker labelled an average of 87 instances. The low volume of instances hinders the capacity to model individual expertise conditioned to the input space.

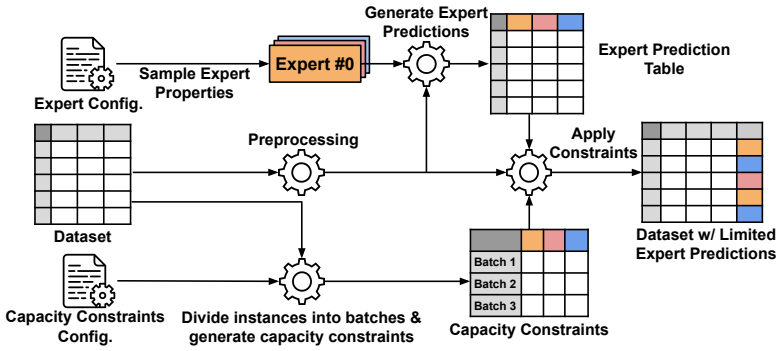

Figure 1: Generating Synthetic Expert Predictions and Capacity Constraints using our Framework

There are few other real-world datasets containing human predictions: the Galaxy Zoo dataset [3], the original Hate Speech and Offensive Language Detection dataset [9], CIFAR10H [27], and ImageNet-16H [19]. However, in all of these, the identity of the human responsible for the prediction was not recorded. Note that datasets with one human prediction per instance are not enough to test assignment systems, even if it is possible to use them for training. To evaluate an assignment system, the output is used to query the chosen decision-makers for their decisions. As such, only datasets with predictions from every decision-maker for every instance are adequate as offline test sets.

2.3 Simulation of Human Experts

Due to the lack of adequate real-world datasets, several authors have resorted to synthesizing expert behavior for datasets in the ML literature. Madras et al. [24] propose a *model-as-expert* technique, fitting a ML classifier to mimic expert behavior on two binary classification datasets (COMPAS [21] and Heritage Health [15]). They use the same ML algorithm used for the main task with extra features, in order to simulate access to exclusive information. These authors also introduce bias, with the goal of studying unfairness mitigation. Verma and Nalisnick [31] use a similar approach to produce an expert on the HAM10000 dataset [30]. In a *model-as-expert* approach, the modelling bias of the ML algorithm is the same for the main classifier and the synthetic experts. Furthermore, they are trained on data with a large fraction of features in common. This may lead to artificially large agreement between the classifier and experts, which is the main drawback of such methods.

Other approaches use sets of rules to produce arbitrarily accurate expert decisions. Mozannar and Sontag [26] use CIFAR-10 [22], where they simulate an expert with perfect accuracy on a fraction of the 10 classes, but random accuracy on the others (see also Verma and Nalisnick [31] and Charusaie et al. [6]). The main drawback of these synthetic experts is that their expertise is either feature-independent or only dependent on a single feature or concept. As such, the methods tested on these benchmarks are not being challenged to learn nuanced and varied types of expertise. This type of approach has been criticised. Zhu et al. [33] and Berthon et al. [4] argue that *instance-dependent label noise* (IDN) is more realistic, as human error is more likely to be dependent on the difficulty of a given task, and, as such, should also be dependent on the instance's features.

In conclusion, to test L2D algorithms, a dataset must contain every human's prediction for every instance. As it is currently unfeasible to collect real world expert predictions for large datasets, a promising avenue is to develop methods to generate synthetic expert decisions that look realistic.

3 OpenL2D Framework

We now present our framework, which is represented schematically in Figure 1. OpenL2D takes as an input a tabular dataset for a binary classification task, represented on the center left of the diagram. Based on a set of user-defined properties, it then creates a team of experts, generating a table containing every expert's predictions for each instance in the input dataset, represented in the

top right corner of our scheme. Optionally, the user may define a set of work capacity constraints, to be applied over batches of instances. These constraints limit the amount of predictions that can be done by a given expert over a specified number of cases. These restrictions can be used to simulate a training dataset with scarce human predictions, limited by expert work capacity. If an assignment system can be configured to assign under capacity restrictions, these may also be used at testing.

3.1 Dataset

Our framework is designed to function on any tabular dataset. The user must define the categorical and numerical features present in their dataset, as well as the column corresponding to the label. Optionally, should the scenario entail fairness concerns, the user may also define which feature corresponds to the protected attribute. Finally, if experts have access to a ML binary classifier score, the user may specify the column corresponding to it.

3.2 Synthetic Expert Generation

In order to generate our synthetic experts' decisions, we will resort to an *instance-dependent noise* (IDN) approach, similarly to [33, 4]. We generate synthetic human decisions by flipping an instance's label y_i with probability $\mathbb{P}(\hat{y}_i \neq y_i|\mathbf{x}_i, y_i)$ in order to generate $\hat{y}_i$. We aim to accommodate for scenarios with class imbalance, as well as cost-sensitive tasks, where the cost of misclassification for false positives (FP) and false negatives (FN) differs. To do so, we simulate the probability of error separately for type 1 and type 2 errors. This introduces a dependence on the label, but allows for better individual control of FP and FN misclassifications.

In some L2D approaches [28, 29], as well as real-world applications of human-AI collaboration [10, 2], the AI is trained independently from the humans in the system. Assignment to a human may then be done due to specific rules demanding human review, based on the model's confidence, or based on an assignment system that is not trained jointly with the classifier. In such human-AI collaborative systems, the model score predicted for a given instance may be shown to the expert, together with other relevant information. It has been shown that the expert's performance can be impacted both positively or negatively by presenting the model's score when deferring a case to them [2, 10, 23]. Therefore, when simulating such a scenario, an expert's decision may also be dependent on an ML model score $m(\mathbf{x}_i)$. We define the expert's probabilities of error, for any given instance, as a function of its features, $\mathbf{x}_i$, and, optionally, on an ML model score $m(\mathbf{x}_i)$:

$$\begin{cases} \mathbb{P}(\hat{y}_i = 1|y_i = 0, \mathbf{x}_i, M) = \sigma\left(\beta_0 + \alpha\frac{\mathbf{w}\cdot\mathbf{x}_i + w_M M(\mathbf{x}_i)}{\sqrt{\mathbf{w}\cdot\mathbf{w} + w_M^2}}\right) \\ \mathbb{P}(\hat{y}_i = 0|y_i = 1, \mathbf{x}_i, M) = \sigma\left(\beta_1 - \alpha\frac{\mathbf{w}\cdot\mathbf{x}_i + w_M M(\mathbf{x}_i)}{\sqrt{\mathbf{w}\cdot\mathbf{w} + w_M^2}}\right) \end{cases}, \; M(\mathbf{x}_i) = \begin{cases} \frac{m(\mathbf{x}_i) - t}{2t}, & m \leq t \\ \frac{m(\mathbf{x}_i) - t}{2(1-t)}, & m > t \end{cases}. \tag{1}$$

Where σ denotes a sigmoid function and M is a transformed version of the original model score $m \in [0, 1]$. The model score was transformed to reflect positive or negative deviations from the model decision threshold, which gives the experts direct information on how confidently the model decision is for the positive or the negative class respectively. To reflect this, the model score $M(\mathbf{x})$ is centered around its threshold t (defined by the user), and scaled to the interval $[-0.5, 0.5]$.

Each expert's probabilities of the two types of error are parameterized by five parameters: $\beta_0, \beta_1, \alpha, \mathbf{w}$ and w_M. To allow for the creation of a group of experts with similar properties, and to control the variability of these parameters within a team, these are sampled from a set of user-defined distributions.

The weight vector $\mathbf{w}$ embodies a relation between the features and the probability of error. If a feature contributes to increase the false positive probability, it will, in turn, decrease the false negative probability, as the weights are shared in both formulas, with opposite signs. We chose to enforce this restriction in order to produce a more intuitive decision-making process. If a human commits more false positives in a given region of the feature space, we also expect them to commit less false negatives, assuming they will be biased towards classifying more instances as positive in said region. Each component w_i of this vector is independently sampled from a user-defined distribution, common

to all features. To impose a dependence on the model score of an independent ML model, the user may also consider setting $w_M \neq 0$, by defining its distribution. To introduce bias, the user may also separately define the distribution of the weight of a protected attribute w_p. We normalize the feature weights so that we can separately control, with the α parameter, the overall magnitude of the variation of the probability of error due to the instance's features. The values of β_1 and β_0 are, respectively, related to the target false positive rate T_{FPR} and false negative rate T_{FNR}, which are sampled from user-defined distributions, allowing for control of the expert's average performance conditioned on the label. The calculation of β_1 and β_0 is discussed in the paragraph "Fixing the Base Rates"

The user may generate many such expert groups within a team, allowing for the creation of a highly variable team with subsets of experts with similar behaviours. A more detailed description of the role and sampling process of each parameter follows.

Feature Dependence Weights Generation To define $\mathbf{w}$ for a given expert, we sample each component from a "Spike and Slab" prior [25]. A spike and slab prior is a generative model in which a random variable u either attains some fixed value v, called the *spike*, or is drawn from another prior p_{slab}, called the *slab*. In our case, we set $v = 0$. That is, u is either zero, or drawn from the slab density $N(\mu_w, \sigma_w)$, where μ_w, σ_w are defined by the user. To sample the values of w_i, we first sample a Bernoulli latent variable $Z \sim Ber(\theta)$. Should the trial yield $Z = 0$, w_i attains the fixed value $v = 0$, if $Z = 1$, w_i is drawn from the slab density p_{slab}. As such, the spike and slab prior induces sparsity unless $\theta = 1$, allowing for the generation of experts whose probabilities of error are swayed by only a few features. The distribution of $w_M \sim N(\mu_M, \sigma_M)$ can be defined separately to allow for control of the expert's model-dependency. Should there be a protected attribute, the user is also able define μ_p, σ_p, in order to impose fairness/unfairness on the simulated experts' decisions.

Controlling Variability and Expert's consistency While the weight vector controls the relative influence each feature has on the probability of error, parameter α, in turn, controls the global magnitude of this influence. For $\alpha = 0$, the probability of error would be identical for all instances. In turn, for very large α, the probability would saturate at the extremes of the codomain of the sigmoid function, 0 or 1. If the probability of error were to always be 0 or 1 based on a case's features, the decision-making process would be entirely deterministic. As such, $\alpha \sim N(\mu_\alpha, \sigma_\alpha)$ can be chosen such that a wide variety of probability of errors exist throughout the feature space.

Fixing the base rates When creating a synthetic human expert, controlling overall performance may be desired. In a binary classification task, two metrics often used to evaluate the performance of the expert are the false positive rate (FPR) and false negative rate (FNR). In our framework, the parameter β_0 and β_1 serve the purpose of controlling each expert's FPR and FNR, separately. However, we wish to allow the expert to set the distributions from which the expert's target performance metrics are sampled, $T_{\text{FPR}} \sim N(\mu_{\text{FPR}}, \sigma_{\text{FPR}})$ and $T_{\text{FNR}} \sim N(\mu_{\text{FNR}}, \sigma_{\text{FNR}})$. This allows a user to avoid adjusting β_0 and β_1 iteratively. In other words, we want the expected value of the FPR when generating an expert to be equal to T_{FPR}. If the value of the weights $\mathbf{w}$ and w_M, as well as α are sampled prior to defining β_0 and β_1, an expert's empirical false positive rate, FPR_e depends only on β_1:

$$\text{FPR}_e(\beta_1) = \frac{1}{N} \sum_i \sigma\Big(\beta_1 + \alpha \frac{\mathbf{w}.\mathbf{x}_i}{||\mathbf{w}||}\Big) \,. \tag{2}$$

Note that, for notational simplicity we set $w_M = 0$ but the result is similar when $w_M \neq 0$. It is then possible to show that the function $\text{FPR}_e(\beta_1)$ is monotonically increasing,

$$\frac{\partial \text{FPR}_e}{\partial \beta_1} = \frac{1}{N} \sum_i \sigma\Big(\beta_1 + \alpha \frac{\mathbf{w}.\mathbf{x}_i}{||\mathbf{w}||}\Big)\left(1 - \sigma\Big(\beta_1 + \alpha \frac{\mathbf{w}.\mathbf{x}_i}{||\mathbf{w}||}\Big)\right) > 0 \quad \text{for} \quad \beta \in \mathbb{R} \,. \tag{3}$$

Since the function is monotonically increasing,and in a bounded to the interval $]0, 1[$, then, for any target false positive rate T_{FPR}, then the following equation has an unique solution:

$$\text{FPR}_e(\beta_1) - T_{\text{FPR}} = 0 \quad \text{for} \quad T_{\text{FPR}} \in]0, 1[\,. \tag{4}$$

A similar reasoning applies for β_0 and T_{FNR}. Finally, we can control an expert's FPR and FNR by solving these equations for β_1 and β_0. To solve these equations, a partition of the dataset is utilized to calculate the empirical value for each rate. Due to the monotonous nature of the function, and the uniqueness of the solution, we solve it through a bisection method [5].

Feature Preprocessing For our simulation of experts, the feature space is transformed as follows. Numeric features in $\mathbf{X}$ are transformed to quantile values, and shifted by -0.5, resulting in features with values in the $[-0.5, 0.5]$ interval. This ensures that the features impact the probability of error independently of their original scale. Categorical features are target-encoded, that is, encoded into non-negative integers by ascending order of target prevalence. These values are divided by the number of categories, so that they belong to the $[0, 1]$ interval, and shifted so that they have zero mean.

3.3 Capacity Constraints

Since most applications of human-AI collaboration are limited by work capacity constraints, L2D evaluation benchmarks should include them. Humans are not able to process instances at the same speed as ML models, being limited in the number of cases they may process on any given time period (e.g. work day). As such, human capacity must be applied over batches of instances, not over the whole dataset, as balancing the human workload over an entire month, for example, is not the same as balancing it daily. A real-world assignment system must then process instances taking into account the human limitations over a given "batch" of cases. We define the work capacity of each of our experts as the maximum number of cases that they are able to process in a given batch. As such, our capacity constraints can be represented by a pair of tables: a batch table, matching each instance with a *batch_id*; and a capacity table, defining the maximum number of cases each expert can be assigned for every batch. To define the capacity constraints, the user must set:

- *Batch Size:* number of cases in each batch. Different batch sizes allow for testing assignment methods over long and short horizons. To allow for the generation of different batches with the same size, the user may set the seed for the distribution of cases throughout batches
- *Deferral Rate:* maximum fraction of each batch that can be deferred to the human team.
- *Distribution:* 'Homogeneous' or 'Variable'. Should the distribution be homogeneous, every expert will have the same capacity. Otherwise, each expert's capacity is sampled from $N(\mu_d = \text{Deferral\_Rate} \times \text{Batch\_Size}/N_{experts},\ \sigma_d \times \mu_d)$, with user defined σ_d. This distribution is chosen so that each expert's sampled capacity fluctuates around the value corresponding to an homogeneous distribution. We set the standard deviation as $\sigma_d \times \mu_d$ so that the user can define the capacity variability as a proportion of μ_d. The user may set the seed for this sampling, allowing for different configurations with the same variability.
- *Absence Rate:* defined as the fraction of experts that are absent in each batch. This allows for testing several different team configuration without having to generate new experts, or to generate scenarios where not all experts work in the same time periods. The user may set the seed for the sampling of the absent experts, allowing for multiple configurations with the same absence rate.

3.4 Training Environment Generation

In order to generate a training environment, a user needs to have generated the synthetic expert predictions as well as a set of capacity constraints to be applied over the training set (see also Figure 1). Finally, we allow the user to distribute the cases in the training environment throughout the humans according to a given function, respecting their capacity constraints. This creates a training dataset with limited human predictions, a key challenge in the development of L2D algorithms. Currently, our training environment generator supports a random distribution of cases.

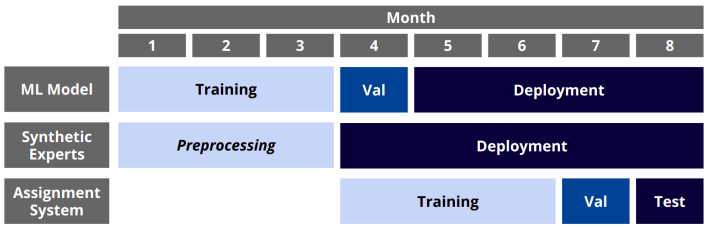

Figure 2: Chosen timeline for ML model training, development and testing of assignment systems.

4 Benchmark

4.1 Our instantiation

Dataset We choose to use the bank-account-fraud tabular dataset, introduced by Jesus et al. [17]. This dataset is sizeable (one million rows) and boasts other key properties that are relevant for our use case. The data was generated by applying tabular data generation techniques on an anonymized, real-world bank account opening fraud detection dataset. The decisions of the system in use have substantial impact on the lives of the decision subjects. A positive prediction (predicted fraud) usually results in a blocked action for the subject, e.g., a rejected transaction, a blocked card, or a rejected application. These dynamics result in a cost-sensitive problem, where the cost of a false positive (rejecting a legitimate event) must be weighed against the cost of a false negative (accepting a fraudulent event). ML models trained on this dataset without measures to preserve fairness tend to raise more false fraud alerts (false positive errors) for older clients (≥ 50 years), thus reducing their access to a bank account. Therefore, this task entails a relevant fairness concern and is appropriate for evaluating the fairness of, not only ML models, but also assignment systems. To measure fairness, we use the FPR disparity between the two age groups, (≥ 50 years and < 50 years.)

Task To simulate a realistic scenario, we created a timeline for the development of a ML model, represented in Figure 2, as well as an assignment system. We used the first three months to train our classifier and fit our pre-processing methods: quantile transformations and target encoders. These months are also used to determine the ideal β_0 and β_1 when generating experts. In the following months, we assume that the classifier is deployed with the human team. To simulate a HAIC scenario where the model is used to complement the human's work capacity, we assume the assignment, in months 4 to 7, is done randomly at a deferral rate of 50%. For this period, we assume no expert absence and homogeneous work capacity. This aims to simulate a HAIC scenario in which cases are randomly passed to a human team that is, on average, better than the model. In our team of 50 experts, this results in a set of nearly $300K$ human predictions, around 6000 per expert. In this problem, the optimization goal is to maximize recall at 5% FPR.

AI Model and Assistant To train our ML model, we chose the LightGBM algorithm [18]. The model was trained on the first 3 months of the BAF dataset, a relatively small portion of the dataset, in order to train assignment systems on the remaining data. The choice of hyperparameters is defined through Bayesian search [1] on an extensive grid, for 50 trials, with validation done on the 4th month. The selected model has a recall of 0.52 at the selected threshold t, chosen to enforce 5% FPR. The model has an FPR disparity of 0.076, raising more false positives for older customers (age ≥ 50).

Expert instantiation We create a team of synthetic experts with varied levels of performance and different feature dependencies. To instantiate the experts, we must define the vector of weights $\mathbf{w}$, α,p_{slab} and θ, as well as the target FPR and FNR for each of the experts. We set $T_{\text{FPR}} \sim N(\mu = 0.04, \sigma = 0.01)$ and $T_{\text{FNR}} \sim N(\mu = M_{\text{FNR}} - 0.05, \sigma = 0.05)$, where M_{FNR} is the false negative rate associated with the model. These values were chosen so that most, but not all, experts have better performance than the model. The rest of the parameters were defined via numerical sweeps, in order to obtain a set of key properties. We decide to assume experts are, on average, as unfair as the model.

By setting $w_M \sim N(-2, 0.5)$, and $w_p \sim N(-1, 0.1)$, we obtain experts with dependence on the model score with a similar FPR disparity as the model. For a wide range of possible results for the probability of error, we set $\alpha \sim N(4, 0.2)$. With these settings, 20 "standard" experts were generated.

In order to simulate a wider variety of human behavior, we introduced two other types of experts. Firstly, in order to simulate bias against the protected class, we introduced 15 "unfair" experts. These are experts with a significantly higher value for w_p, sampled according to $N(-4, 0.3)$, resulting in significantly higher disparity. We also set $\alpha \sim N(4, 0.1)$, to ensure more similar FPR disparities across experts. We also introduced 15 model-agreeing experts to the team. In this set of experts, model score is the dominant feature, with $w_M \sim N(-6, 0.5)$. Also, their predictions are more heavily swayed by the model score, as $\alpha \sim N(12, 0.5)$. To obtain similar unfairness to the "standard" experts, $w_p \sim N(0, 0.1)$.

Capacity constraints To allow for extensive testing of assignment systems, we create a vast array of scenarios. For each of our capacity settings we set two distinct values for the batch size, deferral rate, absence rate, and distribution, resulting in 16 distinct sets of properties. These can be observed in Table 4.2, under "Scenario Properties". For every batch size, we introduce 5 different batch seeds. This results in 20 different scenarios with homogeneous work distribution and no absence. For variable distribution scenarios with no expert absence, we set 4 different distribution seeds, resulting in 80 different variations. For scenarios with expert absence, 2 absence seeds were set, resulting in 40 variations with homogeneous distribution. If the work distribution is variable, we add two distribution seeds, yielding a further 80 distinct capacity constraints. In total, we generate 220 testing scenarios.

4.2 Baselines

In order to provide baselines for assignment methods, we chose to use systems that can comply with capacity constraints. To the best of our knowledge, no current L2D implementation takes individual capacity constraints into account, therefore, we provide three baselines.

Rejection Learning In our implementation of rejection learning, we use the model score as a measure of model confidence. To apply rejection learning within our capacity constraints, we first order the cases within the batch by descending order of model score. All cases above the model's threshold are automatically predicted positive (rejected). Then, the cases below the model's threshold are randomly assigned to humans within our team until their capacity constraints are met. All left over cases are classified negatively (accepted). As such, our implementation defers cases batch-wise.

Human Expertise Aware Rejection Learning In this version of *rejection learning*, instead of randomly assigning the rejected cases throughout our human team, we attempt to model each individual's behavior, in order to optimize assignments. To do so, we train a LightGBM model on the instance features as well as the *expert_id* to predict if the expert's prediction was a false positive (FP), false negative (FN), true positive (TP), or true negative (TN). For each instance, we then have a prediction for the probability that the expert will make either a FP, or a FN mistake, $\hat{\mathbb{P}}(\text{FP})$ and $\hat{\mathbb{P}}(\text{FN})$, respectively. We define a loss associated with deferring an instance $\mathbf{x}_i$ to expert e, given by:

$$L(\mathbf{X}_i, e) = \lambda\hat{\mathbb{P}}(\text{FP}) + \hat{\mathbb{P}}(\text{FN}) \quad \text{with} \quad \lambda = \frac{t}{1-t}\,, \tag{5}$$

where the parameter λ enforces a relationship between the cost of a FP and the cost of a FN error. When selecting λ for a cost-sensitive binary classification task, Elkan [12] shows that a relationship between the value of the ideal threshold t for a binary classifier and the misclassification costs can be established according to Equation 5. We set λ based on the models threshold t. In a first version of our algorithm, an assignment queue is created by sorting all possible instance-expert assignment pairs $(\mathbf{X}_i, e)$ by ascending value of the estimated loss. Assignments are then made greedily in this order. Should an assignment violate capacity constraints, it is skipped on the queue, until the human capacity constraints are met. We also present a much more computationally intensive version of this algorithm, minimizing the loss over an entire batch by solving a linear programming problem subject to our capacity constraints, in order to find the optimal assignment over the entire batch.

Table 1: Baseline Results. The Loss is calculated by summing the instance based loss (Equation 5) over the entire test split. "Batch" refers to the batch size, "Def" to the deferral rate and "Abs" to the absence rate. Homogeneous scenarios are denoted by omitting σ_d. FPR disparity (FPR_d) standard deviations are omitted due to low variability. "Model Only" represents a fully automated baseline, with all predictions made by the model, according to threshold t.

Scenario Properties				Model Only		ReL		$\text{ReL}_{\text{greedy}}$		$\text{ReL}_{\text{linear}}$	
Batch	Def.	Abs.	σ_d	Loss	FPR_d	Loss	FPR_d	Loss	FPR_d	Loss	FPR_d
1000	0.2	0.0	-	918	6.2	$746_{\pm 9}$	9.4	$755_{\pm 12}$	7.1	$773_{\pm 5}$	7.2
1000	0.2	0.0	0.2	918	6.2	$740_{\pm 9}$	9.5	$755_{\pm 8}$	7.1	$771_{\pm 3}$	7.2
1000	0.2	0.5	-	918	6.2	$746_{\pm 12}$	9.4	$761_{\pm 11}$	7.5	$756_{\pm 10}$	7.5
1000	0.2	0.5	0.2	918	6.2	$741_{\pm 10}$	9.4	$760_{\pm 9}$	7.4	$759_{\pm 11}$	7.5
1000	0.5	0.0	-	918	6.2	$737_{\pm 6}$	11.7	$741_{\pm 10}$	7.0	$776_{\pm 2}$	6.8
1000	0.5	0.0	0.2	918	6.2	$739_{\pm 11}$	11.8	$733_{\pm 11}$	7.1	$773_{\pm 5}$	6.8
1000	0.5	0.5	-	918	6.2	$727_{\pm 12}$	11.4	$710_{\pm 11}$	7.5	$757_{\pm 3}$	6.9
1000	0.5	0.5	0.2	918	6.2	$729_{\pm 11}$	11.5	$713_{\pm 19}$	7.6	$757_{\pm 3}$	6.9
5000	0.2	0.0	-	918	6.2	$743_{\pm 11}$	9.5	$750_{\pm 2}$	7.1	$772_{\pm 2}$	7.2
5000	0.2	0.0	0.2	918	6.2	$743_{\pm 8}$	9.4	$753_{\pm 6}$	7.1	$772_{\pm 4}$	7.2
5000	0.2	0.5	-	918	6.2	$738_{\pm 11}$	9.4	$772_{\pm 17}$	7.4	$755_{\pm 10}$	7.5
5000	0.2	0.5	0.2	918	6.2	$742_{\pm 8}$	9.4	$766_{\pm 13}$	7.4	$757_{\pm 9}$	7.5
5000	0.5	0.0	-	918	6.2	$740_{\pm 16}$	11.6	$744_{\pm 6}$	7.0	$778_{\pm 2}$	6.8
5000	0.5	0.0	0.2	918	6.2	$734_{\pm 15}$	11.7	$738_{\pm 12}$	7.0	$775_{\pm 4}$	6.9
5000	0.5	0.5	-	918	6.2	$734_{\pm 12}$	11.5	$702_{\pm 11}$	7.5	$757_{\pm 3}$	6.9
5000	0.5	0.5	0.2	918	6.2	$733_{\pm 15}$	11.5	$706_{\pm 24}$	7.5	$755_{\pm 3}$	6.9

In Table 4.2 we show results for the discussed L2D baselines as well as a "Model only" system to illustrate an output of our benchmarking framework. We can see how results for each of our L2D baselines vary with the generated human-AI collaboration environment (Scenario Properties) across the rows for various metrics of interest. The total amount of compute and the type of resources used are included in the supplementary material.

5 Conclusions and Future Work

In this paper we introduce OpenL2D, a framework for the generation of varied synthetic experts as well as training and testing scenarios for L2D algorithms on any tabular dataset. To illustrate the use of OpenL2D, we provide three L2D baselines tested under 220 different scenarios, in a financial fraud detection task. OpenL2D enables comprehensive benchmarking of L2D algorithms, subject to real world constraints and scenarios.

We identify three main limitations in our framework. The first relates to the type of data supported by this framework. Due to the nature of our synthetic decision generation method, our framework only supports tabular data. While OpenL2D is not able to generate decisions for text and image data, it is still able to generate capacity constraints and create scenarios with limited data availability, provided the synthetic expert predictions are generated via another method. The second is related to the process of generating our synthetic expert decisions. Despite allowing for control of the FPR and FNR of our synthetic experts, by calculating the values of β_0 and β_1, the desired values for the other parameters (e. g. value of w_p to obtain a given FPR disparity) must be obtained through numerical sweeps. Finally, our baselines do not include any established methods in the L2D literature, as these do not currently consider the existence of capacity constraints. While using our framework may aid in the development and testing of L2D systems, the use of synthetic experts can not replace evaluation with real humans before deployment. This is important to mitigate any unexpected effects not covered by the simulations, for example in decision systems that have the potential to amplify discrimination against specific groups. We hope OpenL2D will encourage the research community to build upon our framework to address these restrictions in future studies.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

References

[1] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. In Teredesai, A., Kumar, V., Li, Y., Rosales, R., Terzi, E., and Karypis, G., editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 2623–2631. ACM.

[2] Amarasinghe, K., Rodolfa, K. T., Jesus, S., Chen, V., Balayan, V., Saleiro, P., Bizarro, P., Talwalkar, A., and Ghani, R. (2022). On the importance of application-grounded experimental design for evaluating explainable ml methods. *arXiv preprint arXiv:2206.13503*.

[3] Bamford, S. P., Nichol, R. C., Baldry, I. K., Land, K., Lintott, C. J., Schawinski, K., Slosar, A., Szalay, A. S., Thomas, D., and Torki, M. (2009). Galaxy Zoo: The dependence of morphology and colour on environment. *Monthly Notices of the Royal Astronomical Society*, 393(4):1324–1352.

[4] Berthon, A., Han, B., Niu, G., Liu, T., and Sugiyama, M. (2021). Confidence scores make instance-dependent label-noise learning possible. In *International conference on machine learning*, pages 825–836. PMLR.

[5] Burden, R. L. and Faires, J. D. (1985). 2.1 the bisection algorithm. *Numerical analysis*, 3.

[6] Charusaie, M.-A., Mozannar, H., Sontag, D. A., and Samadi, S. (2022). Sample Efficient Learning of Predictors that Complement Humans. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S., editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 2972–3005. PMLR.

[7] Chow, C. K. (1970). On optimum recognition error and reject tradeoff. *IEEE Trans. Inf. Theory*, 16(1):41–46.

[8] Cortes, C., DeSalvo, G., and Mohri, M. (2016). Learning with Rejection. In Ortner, R., Simon, H. U., and Zilles, S., editors, *Algorithmic Learning Theory - 27th International Conference, ALT 2016, Bari, Italy, October 19-21, 2016, Proceedings*, volume 9925 of *Lecture Notes in Computer Science*, pages 67–82.

[9] Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11, pages 512–515.

[10] De-Arteaga, M., Fogliato, R., and Chouldechova, A. (2020). A Case for Humans-in-the-Loop: Decisions in the Presence of Erroneous Algorithmic Scores. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, Honolulu HI USA. ACM.

[11] Dellermann, D., Ebel, P., Soellner, M., and Leimeister, J. M. (2019). Hybrid Intelligence. *Business & Information Systems Engineering*, 61(5):637–643.

[12] Elkan, C. (2001). The Foundations of Cost-Sensitive Learning. In Nebel, B., editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*, pages 973–978. Morgan Kaufmann.

[13] Hemmer, P., Schellhammer, S., Vössing, M., Jakubik, J., and Satzger, G. (2022). Forming Effective Human-AI Teams: Building Machine Learning Models that Complement the Capabilities of Multiple Experts. In Raedt, L. D., editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2478–2484. ijcai.org.

[14] Hendrycks, D. and Gimpel, K. (2017). A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

[15] Heritage Provider Network, I. (2011). Heritage Health Prize. https://kaggle.com/competitions/hhp.

[16] Inkpen, K., Chappidi, S., Mallari, K., Nushi, B., Ramesh, D., Michelucci, P., Mandava, V., Vepřek, L. H., and Quinn, G. (2022). Advancing human-ai complementarity: The impact of user expertise and algorithmic tuning on joint decision making. *arXiv preprint arXiv:2208.07960*.

[17] Jesus, S., Pombal, J., Alves, D., Cruz, A. F., Saleiro, P., Ribeiro, R. P., Gama, J., and Bizarro, P. (2022). Turning the Tables: Biased, Imbalanced, Dynamic Tabular Datasets for ML Evaluation. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2022*.

[18] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3146–3154.

[19] Kerrigan, G., Smyth, P., and Steyvers, M. (2021). Combining human predictions with model probabilities via confusion matrices and calibration. *Advances in Neural Information Processing Systems*, 34:4421–4434.

[20] Keswani, V., Lease, M., and Kenthapadi, K. (2021). Towards Unbiased and Accurate Deferral to Multiple Experts. In Fourcade, M., Kuipers, B., Lazar, S., and Mulligan, D. K., editors, *AIES '21: AAAI/ACM Conference on AI, Ethics, and Society, Virtual Event, USA, May 19-21, 2021*, pages 154–165. ACM.

[21] Kirchner, L. and Larson, J. (2017). How we analyzed the COMPAS recidivism algorithm. https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm.

[22] Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images.

[23] Levy, A., Agrawal, M., Satyanarayan, A., and Sontag, D. (2021). Assessing the impact of automated suggestions on decision making: Domain experts mediate model errors but take less initiative. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–13.

[24] Madras, D., Pitassi, T., and Zemel, R. (2018). Predict Responsibly: Improving Fairness and Accuracy by Learning to Defer. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

[25] Mitchell, T. J. and Beauchamp, J. J. (1988). Bayesian variable selection in linear regression. *Journal of the american statistical association*, 83(404):1023–1032.

[26] Mozannar, H. and Sontag, D. A. (2020). Consistent Estimators for Learning to Defer to an Expert. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 7076–7087. PMLR.

[27] Peterson, J. C., Battleday, R. M., Griffiths, T. L., and Russakovsky, O. (2019). Human uncertainty makes classification more robust. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9617–9626.

[28] Raghu, M., Blumer, K., Corrado, G., Kleinberg, J. M., Obermeyer, Z., and Mullainathan, S. (2019a). The Algorithmic Automation Problem: Prediction, Triage, and Human Effort. *CoRR*, abs/1903.12220.

[29] Raghu, M., Blumer, K., Sayres, R., Obermeyer, Z., Kleinberg, B., Mullainathan, S., and Kleinberg, J. (2019b). Direct uncertainty prediction for medical second opinions. In *International Conference on Machine Learning*, pages 5281–5290. PMLR.

[30] Tschandl, P., Rosendahl, C., and Kittler, H. (2018). The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9.

[31] Verma, R. and Nalisnick, E. T. (2022). Calibrated Learning to Defer with One-vs-All Classifiers. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S., editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 22184–22202. PMLR.

[32] Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., and Summers, R. M. (2017). Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2097–2106.

[33] Zhu, Z., Liu, T., and Liu, Y. (2021). A second-order approach to learning with instance-dependent label noise. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10113–10123.