LEVERAGING HUMAN FEATURES AT TEST-TIME

Anonymous authors
Paper under double-blind review

ABSTRACT

Machine learning (ML) models can make decisions based on large amounts of data, but they may be missing important context. For example, a model trained to predict psychiatric outcomes may know nothing about a patient's social support system, and social support may look different for different patients. In this work, we explore strategies for querying for a small, additional set of these human features that are relevant for each specific instance at test time, so as to incorporate this information while minimizing the burden to the user to label feature values. We define the problem of querying users for an instance-specific set of human feature values, and propose algorithms to solve it. We show in experiments on real datasets that our approach outperforms a feature selection baseline that chooses the same set of human features for all instances.

1 INTRODUCTION

Recent advances in machine learning (ML) have enabled it to be applied across several domains, including complex scenarios involving loan approval, medical diagnoses and criminal justice. A defining feature of complex scenarios is the need to incorporate human expertise in the decision-making process. Specifically, ML models and human decision-makers often have different yet complementary strengths: ML models can recognise patterns in massive datasets, while humans have access to additional facts and contextual information which may be missing from the dataset and are important for decision-making.

Consider the example where electronic health record (EHR) data from thousands of patients is converted to a set of machine features that are analysed by ML and used to predict depression-related outcomes (e.g. Pradier et al. (2020), Lage et al. (2022)). These models, though predictive, may lack important contextual information that is otherwise available to the human patient, about for instance their support network, or socioeconomic status. We call these human features. *Both human and machine* can be important predictors of treatment outcomes (Carter et al., 2012). However, different features may be relevant for different types of patients. For example, marital status may be important to determine social support for a 40 year old patient, but not an 18 year old.

Existing strategies for incorporating additional human features in test-time predictions largely rely on users being able to make expert decisions. For example, mixtures-of-experts approaches (e.g. Jordan & Jacobs (1994); Parbhoo et al. (2017)) weight the predictions made by an ML model and a human decision-maker to decide when a human should step in. Others learn when to defer decisions to a down-stream decision-maker based on samples of the ML model's decisions (e.g Madras et al. (2018); Mozannar & Sontag (2020); Joshi et al. (2021)), or explicitly constrain the model to align with human expertise (Pradier et al., 2021; Wang et al., 2018). These methods all however, assume that users can make reasonable predictions, which is not necessarily the case for many users like the patients described above. A third alternative for incorporating human features is to identify the few that are missing from the dataset, and collect these for all patients going forwards. While this does not require user expertise, it assumes that the *same set of features* will be useful across *all* instances.

In this work, we consider the problem of intelligently eliciting a small number of personalized human features for each instance at test time, where weights for the human features were learned during training. Having access to these features at train time may be reasonable in settings where there is a budget for collecting additional human features at train time, including through crowdsourcing (e.g. Cheng & Bernstein (2015)), or in clinical trials (e.g. the CoMMpass study for multiple

myeloma[1]). But expecting these same features at test time might be infeasible for either financial or workflow-related reasons. For example, asking a patient 100 questions during a clinical interview may negatively impact the clinical relationship.

We frame the problem of selecting the human features to query for each instances as a constrained optimization problem, where the goal is to find features that most improve our predictive ability for each specific instance at test time without making more than B queries. We introduce an entropy-based strategy for solving this problem that identifies the human features that will provide the most information about the prediction given the information already available in the machine features. On synthetic and real datasets (for which we create a partition between machine and human features), our instance-specific approach outperforms querying for the same B features across all instances.

2 Related Work

Algorithms for Combining Human and ML Decisions Various methods have been proposed to facilitate joint human-ai predictions by combining predictions made independently by each party. These include approaches to learning models that complement human predictions (e.g. Wilder et al. (2020); Raghu et al. (2019)), learning to defer to decision-makers when models are unsure (e.g. Madras et al. (2018)), as well mixture of experts frameworks where one of the decision-makers is a human (e.g. Parbhoo et al. (2017); Pradier et al. (2021)). In contrast, we study methods that combine human and machine inputs at the level of features to leverage the relative strengths of both human and machine decision makers within the same decision.

Human Feature Inputs to ML Predictions Approaches to incorporate human features as inputs to ML predictions exist, but address different problem settings. Some methods use crowdworkers to label features (e.g. Cheng & Bernstein (2015); Takahama et al. (2018); Zou et al. (2015); Sakata et al. (2019), however these are employed at train time in conjunction with model development, and assume all features will be labeled at test time. Interactive symptom checkers query patients for symptoms to help them explore diagnoses. Systems including Poote et al. (2014) have hard-coded rules about which features to query, rather than learning them. Ruotsalo & Lipsanen (2018) does algorithmically select relevant queries using uncertainty over feature values, which is similar to our approach, but models associations between symptoms and diseases are hard links which does not accommodate general classes of machine learning models. Finally, in geospatial settings, estimation fusion techniques can make predictions combining multiple sensors, some of which can be human-based (see Hall et al. (2008); Hall & Jordan (2010) for an overview). In contrast, we study a setting where only a small number of human features can be queried at test time, and propose methods for identifying these features that are applicable for arbitrary ML models on tabular datasets.

Editable Models for ML-Assisted Decision-Making Various studies have considered the consequences of editing model explanations directly to influence predictions: Gillies et al. (2016); Kulesza et al. (2013), allows users to change parameters, which is likely challenging for non-experts, and may result in users decreasing predictive performance; Koh et al. (2020) learns models with concept abstractions and simulates user feedback to correct erroneous concept values; and Jacobs et al. (2021) finds that clinicians are interested in being able to edit patient features in a clinical dashboard for antidepressant prescription. Approaches that rely on editing data may be similarly accessible to non-experts, however neither of these settings allow users to input new features, or offers guidance about which concepts or features they should edit. In contrast, our approach intelligently guides users to impactful new features to provide to the model.

Similar methods solving different problems Our work is also closely related to methods that perform instance-wise feature selection. Among these, Yoon et al. (2018) propose a deep learning approach to discover subsets of features that may be relevant for predicting a particular output. Similarly, several information-theoretic approaches have also been proposed for selecting features for and explaining a prediction. The learning-to-explain (L2X) method (Chen et al., 2018; Jethani et al., 2021) learns a function to extract a subset of features that are most informative for a given example by maximising the mutual information between some selected features and the response variable.

[1] https://themmrf.org/finding-a-cure/our-work/the-mmrf-commpass-study/

In a similar vein, information bottleneck-based approaches (Tishby et al., 2000; Alemi et al., 2016), explicitly constrain a model to only use the most relevant features by trading-off compressing the features with predicting a response. While each of these of these methods may be complementary to what we propose in this paper, our use case is different since we explicitly focus on integrating and leveraging human expertise during test time.

3 Problem Definition and Approach

The key novelty in our formulation lies in presuming the existence of a set of "human-features" that can be elicited at test time. Our core question is how to query a small, but useful set of human features for each instance, given that we have some way of making predictions based on them. We assume the user has features relevant to the decision-making task for a specific instance, which may be the case when the instance corresponds to a representation of the user–for example their medical record. We assume that, for each instance, we can query the associated user a small, fixed number of times B, for e.g. less than 10 times, for the values of specific features. More queries may overwhelm the user. Finally, we assume that a distinct set of human features may be relevant for each instance. For example, demographic characteristics such as marital status exist for all patients, but may be less relevant for indicating social support in an 18 year old patient than a 40 year old. In the following section, we formalize this optimization problem, and present an approach for solving it.

3.1 Preliminaries

Concretely, we assume a standard supervised learning setup where we have a dataset of N instances, the machine features, X^m, that the machine learning model always has access to, and corresponding labels, Y. In addition, we assume access to human features X^h during training but not at test time when they must be queried for a cost. X^m and X^h could be any real values, but in our instantiation, we assume X^h is binary: $X^h \in \{0,1\}^{N \text{x} D_h}$. (The human is best at providing yes-no answers.) We additionally assume that the labels are categorical, i.e. $Y_n \in \{0, ..., K\}$.

We train a discriminative prediction function f to make predictions $\hat{Y}$ as follows:

$$\hat{Y} = f(\theta^m, X^m, \theta^h, X^h) + \phi \tag{1}$$

where θ^m and θ^h are parameters for X^m and X^h respectively and ϕ is a bias term. We instantiate the model class when discussing implementation, but the methods and problem definition apply generally to models of this form.

3.2 Defining the Desired Optimization Objective

We formalize the problem of querying a small number of human features at test-time as a constrained optimization problem where, for each instance, we want to ask the user at most B questions. B is some small number, like 10, to avoid overwhelming the user with irrelevant queries. The goal is to find those B queries for each instance that maximize the predictive quality, L, as much as possible.

For a test instance x, we denote which features in x^h have been queried, using a binary query mask, $q \in \{0,1\}^{D_h}$. $q_d = 1$ indicates that human feature d has been queried for this instance, while $q_d = 0$ indicates that it has not yet been queried.

We can then write the constraint on the number of queries for each instance in terms of q:

$$\sum_{d=1}^{D_h} q_d \leq B \tag{2}$$

In other words, the number of elements in q must that are queried, i.e. set to 1, should be at most B.

To make predictions on a test instance where we only have access to queried features in q, we marginalize out the remaining features (i.e. $q_d = 0$). This is a standard way of handling missing data. We denote this prediction function as f^{marg}, and use it to predict for an instance as follows:

$$f^{marg}(x^m, x^h, q) = \int_{\tilde{x}^h} f(\theta^m, x^m, \theta^h, \tilde{x}^h) p(\tilde{x}^h | x^m, q) \tag{3}$$

where $\tilde{x}^h$ corresponds to a random variable sampled from the helper distribution $p(\tilde{x}^h|x^m, q)$ defined below, rather than the actual value of x^h, as this is unknown.

To define this helper distribution $p(\tilde{x}^h|x^m, q)$, first factorize it, as x^h and x^m are both high dimensional, and thus challenging to model jointly. The factorized form is as follows:

$$p(\tilde{x}^h|x^m, q) = \prod_{d=1}^{D^h} p(\tilde{x}_d^h|x_d^m, q_d) \tag{4}$$

We then define the individual dimensions of the helper distribution as follows:

$$p(\tilde{x}_d^h|x_d^m, q_d) = \begin{cases} \mathbb{1}_{\tilde{x}_d^h = x_d^h} & \texttt{if } \; q_d = 1 \\ p(\tilde{x}_d^h|x^m) & \texttt{if } \; q_d = 0 \end{cases} \tag{5}$$

I.e. the probability is the same as $p(\tilde{x}_d^h|x^m)$ if the feature hasn't been queried, and if the feature has been queried, the probability of the true value is 1, and zero otherwise.

In practice, we compute the expectation in Equation 3 by Monte Carlo sampling:

$$f^{marg}(x^m, x^h, q) \approx \frac{1}{S} \sum_{s=1}^{S} f(\theta^m, x^m, \theta^h, \tilde{x}^{h(s)})$$
$$\tilde{x}^{h(s)} \sim p(\tilde{x}^{h(s)}|x^m, q)$$

where S is the number of samples drawn.

Based on this, we can define an idealized optimization objective that optimizes q for instance x to minimize predictive loss L between the true label y and the prediction made using only the machine features and the queried human features in q. This objective can be written as follow:

$$q* = \underset{q \in \{0,1\}^{D_h} \, \texttt{s.t.} \, \sum_{d=1}^{D_h} q_d \leq B}{\arg\min} L(y, f^{marg}(x^m, x^h, q)) \tag{6}$$

In our results, we operationalize L as f1-score, but other reasonable choices could be made.

3.3 Approximate Solution to Desired Objective

Solving this optimization problem directly requires access to the true label, y, which we do not have for test instances. We address this challenge by solving a related optimization problem based on the uncertainty of the predictions given a mask q, as this can be evaluated without knowing y.

Specifically, we optimize q to minimize the entropy of the marginalized predictions after querying the features in x^h specified by q. We can write this approximate objective as follows:

$$q^* = \underset{q \in \{0,1\}^{D_h} \, \texttt{s.t.} \, \sum_{d=1}^{D_h} q_d \leq B}{\arg\min} H[f^{marg}(x^m, x^h, q)] \tag{7}$$

Intuitively, this means that we want to find the set of human features to query that will reduce our uncertainty about the prediction as much as possible. In the absence of true labels, prediction uncertainty gives a reasonable surrogate for Equation 6. This is similar to a successful approach taken in Bayesian Active Learning by Disagreement (Houlsby et al., 2011).

Solving this approximate objective comes with challenges as well. The first problem is that it is not possible to evaluate this objective without querying the human features in q, so we cannot try many distinct values for q without going over the query budget. We address this by taking an expectation over the values of x^h in a candidate q to cheaply evaluate this objective many times. The second problem is that it this set optimization is computationally difficult; we address this by greedily building up q by querying the best remaining feature dimension at each step, until we reach a set of size B. While this is not guaranteed to provide an optimal solution, it is often employed for similar approaches, e.g. (Houlsby et al., 2011), and is generally quite effective in practice. Algorithm 1 gives a full description of our procedure.

Based on these insights, we break the optimization problem in Equation 7 up into B subproblems where we find the unqueried feature dimension that maximizes the expected predictive entropy if it were to be queried. We write this subproblem objective as follows:

$$d^* = \underset{d \in \{1,...,D^h | q_d = 0\}}{\arg\min} \mathbb{E}_{p(x_d^h | x^m)}[H(f^{marg}(x^m, x^h, q + \mathbb{1}_d^{D^h})] \tag{8}$$

where $\mathbb{1}_d^{D^h}$ denotes a onehot vector of size D^h where all entries are 0 except at index d where the entry is 1. The expectation is easy to compute as we assume x_d^h is binary.

In practice, we solve this optimization subproblem by trying all possible values of d that satisfy the constraint. This is a strategy linear in D^h, and is employed by many active learning approaches, including Zhu et al. (2003).

Algorithm 1 Our greedy search-based optimization procedure chooses, at each iteration, to query the human feature that minimizes the expected marginalized entropy given the feature is queried.

Given: x^m, x^h
$q \leftarrow \{0\}^{D^h}$
for $\mathfrak{b} \in \{1, ..., B\}$ **do**
  $d^* = \arg\min_{d \in \{1,...,D^h | q_d = 0\}} \mathbb{E}_{p(x_d^h | x^m)}[H(f^{marg}(x^m, x^h, q + \mathbb{1}_d^{D^h})]$
  $q \leftarrow q + \mathbb{1}_{d^*}^{D^h}$
end for

4 IMPLEMENTATION

In order to solve the approach described in previous section, we need to instantiate: a functional form for the prediction function f, and a distribution for approximating $p(x^h|x^m)$. We also describe a heuristic approach to boosting predictive performance by re-training the parameters of the predictive function given the learned query mask, q.

4.1 DEFINING A FUNCTIONAL FORM FOR f

In practice, we implement f as a logistic regression model, where we make predictions as follows:

$$\hat{Y} = \sigma((\theta^m)^T X^m + (\theta^h)^T X^h + \phi) \tag{9}$$

Where θ^m and θ^h are weight vectors in $\mathbb{R}^{D_m}$ and $\mathbb{R}^{D_h}$ respectively, and ϕ is an intercept term in $\mathbb{R}$.

4.2 APPROXIMATION $p(x^h|x^m)$

The marginalized prediction function, f^{marg}, and the expectation in Equation 8, depend on $p(x^h|x^m)$. We approximate this quantity from the data at train time. In order to fit the approximation $\hat{p}(x^h|x^m)$, we model each dimension of x^h with an independent logistic regression model, as the features in x^h are binary. For a dimension d, we define:

$$\hat{p}(x_d^h | x_d^m) = \sigma(w_d^T x_m + w_d^0) \tag{10}$$

where w_d is a weight vectors in $\mathbb{R}^{D_m}$, and w_d^0 is an intercept in $\mathbb{R}$. We train parameters w_d and w_d^0 to minimize the log-loss between $\hat{p}(X_d^h | X^m)$ and X_d^h for each dimension $d \in \{1, ..., D^h\}$ using the training set. While more expressive approximations to this distribution are possible, we found this one to be sufficient in practice.

4.3 IMPLEMENTATION HEURISTIC: RE-TRAINING WITH QUERY MASK

We describe an optional retraining heuristic that boosts performance. The parameters of the prediction function, θ^m, θ^h, ϕ are trained assuming that the dataset includes the human features in addition

to the machine features. While we address this by marginalizing out unqueried dimensions of the human features when predicting, we can often do slightly better by re-training the parameters using query masks optimized for the training set.

In this case, rather than marginalizing the unqueried features to make predictions, we zero them out. We define a prediction function f^{zero} as follows:

$$f^{zero}(x^m, x^h, q) = \sigma((\bar{\theta}^m)^T x^m + (\bar{\theta}^h)^T (x^h \odot q) + \bar{\phi}) \tag{11}$$

and fit the re-trained parameters $\bar{\theta}^m$, $\bar{\theta}^h$, and $\bar{\phi}$ using a standard approach on the transformed dataset: $(X^m, (X^h \odot Q), Y)$. Q is the matrix of feature query masks fit using algorithm our entropy approach on the training set. At test time, we make predictions using f^{zero}.

5 EXPERIMENTS

We ran experiments in 2 real domains to compare the predictive performance of our proposed approach variations against baselines that do not permit instance-specific human feature queries. We include results on an illustrative toy example in Appendix 1. Our results demonstrate that the instance-specific queries made by our method are helpful for improving predictive performance, and qualitatively sensible. We describe baselines, hyperparameters and datasets, then our results.

5.1 EXPERIMENTAL SETUP

Baselines We compare our method to two baselines and one oracle upper bound. The upper bound consists of a model with the same functional form as f (i.e. logistic regression) trained using both the machine and human feature sets, i.e. X^m and X^h–we denote this *all-features*. This would be what would be possible if, at test time, we could query the human for *all* their features rather than a subset. The first baseline consists of another logistic regression model trained with only the machine feature set, i.e. X^m, we denote this *machine-only*. This is the performance of the model with no access to human features. The second baseline consists of a standard feature selection (denoted *feature-selection*) approach where the same B human features are queried for all instances, rather than a distinct subset of human features being queried for each instance. This allows us to demonstrate the importance of instance-specific human feature queries.

We implement the feature selection baseline using a greedy forward selection strategy (see Tang et al. (2014) for an overview of this and other methods) where, for each of the B features to add, we re-train the model adding each remaining human feature and choose the feature with the best validation performance (computed using f1-score, as this is the metric used in our results). This is a standard and effective approach to feature selection.

Hyperparameters We implement all logistic regression models using the scikit-learn package (Pedregosa et al., 2011). We used a multinomial loss, the SAGA solver and we set the maximum number of iterations to 5000. In our hyperparameter search for the real data experiments, we searched over penalties: [l1, l2 and none], inverse regularization strengths [0.01 , 0.1 , 1. , 10. , 100.], and class weighting schemes [none, balanced]. We set $B = 10$. For the entropy selection approach, we set the number of Monte Carlo samples $S = 5000$ so as to minimize this source of approximation error. For the models used in the entropy method to model f and $p(x^h|x^m)$, we do not search over balanced class weights as this causes challenges with calibration–something that is important for this method because it depends on accurately estimating probabilities.

We perform hyperparameter selection based on maximizing f1-score (the metric we report) on the validation set, except in the case of the models for $p(x^h|x^m)$, where log-loss is minimized instead. This is to encourage those models to accurately model the probability distribution they approximate. For the feature selection models, we chose the hyperparameters once based on the machine features rather than re-optimizing them for each possible choice of features to select.

Datasets To illustrate the performance of our proposed approach, we use two real datasets: a *recipe* dataset[2] where the goal is to predict the cuisine of a recipe (e.g. Italian food) based on the

[2] `https://www.kaggle.com/datasets/kaggle/recipe-ingredients-dataset` train.json file

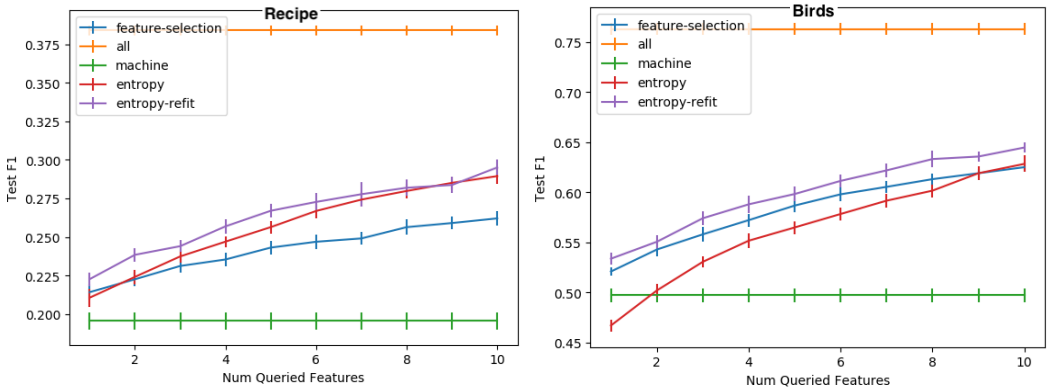

Figure 1: Test f1-score as a function of B for all both variations of our method, baselines and upper bound in *recipe* (left) and *birds* (right) datasets. Error bars are standard errors over 10 random restarts. In both, *all-features* performs best and *machine-only* worst with the methods using a subset of human features in between. In *recipe*, both entropy methods substantially outperform *feature-selection*, while in *birds*, *entropy-refit* eventually does, but *entropy* performs slightly worse.

ingredients, and a *birds* dataset (Wah) where the goal is to predict the type of bird (e.g. crow) based on some crowdsourced attributes of an image of a bird. After pre-processing, the *recipe* dataset consists of 6k instances (subsampled to that size for computational reasons); 120 features, and 20 classes, and the *birds* dataset consists of 5k instances; 171 features; and 36 classes.

We split the feature sets between X^m and X^h to facilitate running experiments with ground truth values of X^h. We construct these feature splits so that X^m consists of a smaller set of features that are "simpler" in some way, and X^h consists of a larger set of features that are more "complex". In the *recipe* dataset, we split the features so all single word ingredients (33 features) are in the machine feature set, and all 2+ word ingredients (87 features) are in the human feature set. For the *birds* dataset, we split the feature so all non-color-related words (48 features) are in the machine feature set, and all color-related words (123 features) are in the human feature set.

We preprocess the recipe dataset by removing instances without labels, then subsampling 15% of the remaining instances, maintaining overall label distribution, for computational reasons. We then removed features with positive values for less than 100 instances. For the birds dataset, we use the coarse class labels, and remove instances with infrequent class labels, recorded for less than 50 instances. We then binarize features values at 0.5 (features are recorded as the fraction of agreement on MTurk responses about whether the feature is present in the image), and removed features with positive values for less than 100 instances. We split the data, class-balancing labels, into train/validation/test sets of sizes $\frac{2}{3}, \frac{1}{6}, \frac{1}{6}$ of the instances respectively.

5.2 Results

In our experiments, we compared the predictive performance of the standard (*entropy-selection*) and retrained (*entropy-retrain*) variations of the entropy selection method we propose to baselines and the *all-features* upper bound. We report test f1-score for all of these across values of B. Results are reported as the mean and standard deviation over 10 random restarts of the train/valid/test splits and any subsampling, unless otherwise specified. Figure 1 shows results in both domains.

The full set of features, outperforms only the machine features, suggesting that the human features contribute substantially to predictive performance. In both datasets, *all-features* performs substantially better than *machine-only*. In the recipe dataset, the gap in f1-scores between these methods is 0.18, while in the birds dataset, the gap is 0.26. This suggests that there is substantial improvement to be gained from finding ways to use the human features in predictions.

All 3 approaches that query human features are able to close a substantial portion of the gap between *all-features* and *machine-only* within just 10 queries. In both datasets, the *entropy-*

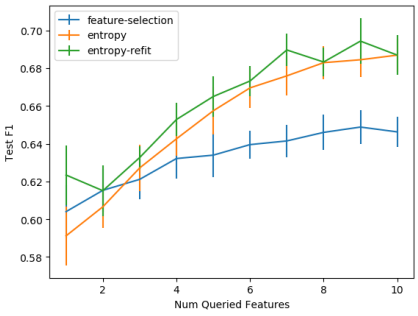

Figure 2: Test f1-score on *birds* as a function of B after adding first 6 *feature-selection* queries to machine features and re-running. Error bars are standard errors from 5 random restarts. By query 6, both entropy methods substantially outperform *feature-selection*.

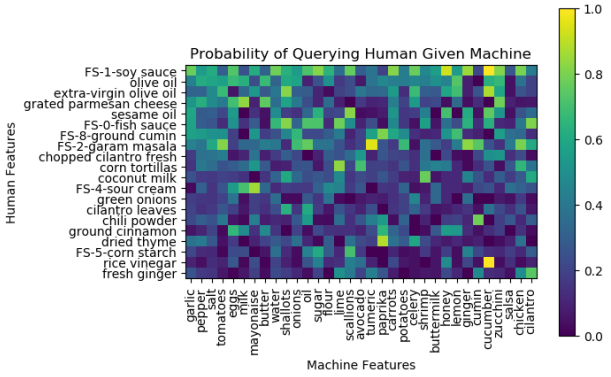

Figure 3: Probability of querying a human feature (y-axis, top 20 sorted by # times queried) given a machine feature (x-axis) in the instance in *recipe*. Computed on test set for randomly chosen restart. "cucumber"–"rice vinegar" and "turmeric"–"garam masala" associations are sensible.

selection, *entropy-retrain* and *feature-selection* methods all substantially improve performance over *machine-only*. The 2 entropy-based methods we propose close half or more of the performance gap by the 10th query in both datasets, while *feature-selection* only achieves this in the *birds* dataset. This is true despite the 10 queries covering only 12% of the human features in the *recipe* dataset and 8% of the features in the *birds* dataset. This suggests that querying a small number of relevant human features is a viable strategy for substantially improving the performance of ML models that have access to only a subset of the relevant features.

Our proposed *entropy-retrain* approach outperforms *feature-selection* in both datasets after sufficient queries, and never underperforms it. In the *recipe* dataset, *entropy-retrain* outperforms *feature-selection* starting from the 2nd query onwards. In the *birds* dataset, *entropy-retrain* performs comparably to *feature-selection* for the first 6 queries, then outperforms it for the last 4. The performance improvement is particularly marked in the recipe domain, where *entropy-retrain*'s f1-score after 10 queries is 0.3 compared to only 0.26 for *feature-selection*. This suggests that querying unique human features for each instance can improve performance over a shared set of features queried for the entire dataset. It also suggests that our proposed entropy approach with the retraining heuristic is an effective way to select these instance-specific human feature queries.

***entropy-selection* has variable performance compared to *feature-selection* in the initial queries, but always outperforms it eventually.** In the recipe dataset, *entropy-selection* performs similarly to *entropy-retrain*, which allows it to substantially improve over *feature-selection* starting on the 4th query. In the birds dataset, *entropy-selection* actually performs worse than *feature-selection* for the first 6 queries, then performs similarly. Figure 2 shows the results of a follow-up experiment where we added the first 6 features selected by *feature-selection* to the machine feature set for the birds data, and re-ran all 3 approaches for 10 additional queries. This includes only 5 random restarts. Here, we see that *entropy-selection* and *entropy-refit* perform similarly for the first 4-5 queries, then substantially outperform *feature-selection* with an f1-score of 0.69 vs. 0.65. This suggests several things: 1) the re-training heuristic is useful for boosting performance in at least some cases, 2) even without the retraining heuristic, *entropy-selection* is useful, at least eventually, and 3) for some datasets, individualized feature queries are useful immediately, while for others, they may be more useful after querying an initial set of shared features.

Sensible correlations between machine features and human feature queries in the recipe domain suggest reasons for performance improvements from individualized feature queries. Figure 3 shows a heatmap of the probability a specific human feature (y-axis) will be queried using *entropy-selection* given that a specific machine feature (x-axis) is observed for the instance (computed on the test set for 1 randomly chosen random restart). Strong relationships between sensible ingredient pairings include "cucumber" and "rice vinegar", which makes sense because the ingre-

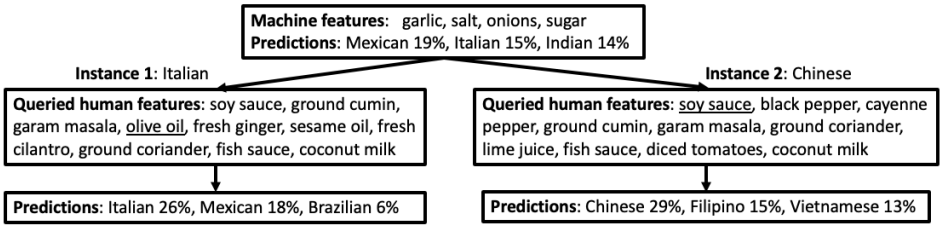

Figure 4: Original predictions, human features queried–underlined when positive, and updated predictions for 2 test instances from a randomly chosen restart with identical machine features. Informative human features including "soy sauce" and "olive oil" help to disambiguate predictions.

dients commonly co-occur in various east Asian cuisines but "cucumber" is more widely used, and "turmeric" and "garam masala," which makes sense because again, "turmeric" can be used in various cuisines, but the 2 ingredients frequently co-occur in Indian cuisine. This suggests that feature queries made by *entropy-selection* are informed by the machine features in sensible ways.

An illustrative example demonstrates how different queried human feature values can cause the predictions for 2 instances with identical machine features to diverge to correct predictions. We examine in Figure 4 how the predicted probabilities change after 10 feature queries for 2 test instances with identical machine features but different labels. While both start with Mexican cuisine as the most likely prediction, after discovering that instance 1 contains olive oil and does not contain many other distinctive ingredients including fish sauce and garam masala, the prediction correctly changes to Italian. For instance 2, the presence of soy sauce and the absence of many other features correctly changes the prediction to Chinese cuisine. Note that the queries are not identical because they depend on the values of queried human features. This example illustrates how the instance-specific human feature queries can help disambiguate the labels of otherwise similar instances.

6 Discussion

Our experiments suggest avenues for future work. The late improvement of the entropy based methods over *feature selection* in *birds* suggest there may be cases where a combination of instance-specific and global human feature queries can be useful. Our proposed *entropy-retrain* method still performs as well as *feature-selection*, but there may be benefits to recording the same features for all instances, including using this information to update $\hat{p}(x^h|x^m)$. Future work could explore heuristics for identifying when a dataset would benefit from an initial round of global human feature collection, and when it would directly benefit from an instance-specific approach.

We make assumptions in this work that, while plausible, merit further study. We propose a method that works given a predictive set of human features, however this may not always be the case. Additionally, even when human features are important, they may be distinct from machine features in ways that impact how they can be used for prediction. For example, important human features such as pain may be more subjective than some machine features. Future approaches could explore ways to identify specific use cases where human features exist and are informative, and methods to account for subjectivity when querying human features. Finally, the assumption that we have access to the human features at train time limits the contexts in which this method can be used. Future work can explore relaxing this assumption by incorporating active learning techniques to query only those human features that are truly needed to train the model.

7 Conclusion

We define the problem of querying users of machine learning models for a small number of additional human features for each instance at test time, and present algorithms for generating these queries. Our results demonstrate that instance-specific human feature queries can improve predictive performance, and that our proposed entropy algorithm plus retraining heuristic can effectively generate these queries in cases where informative human features exist.

REFERENCES

Technical report.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

Gebra Cuyún Carter, Ronald A Cantrell, Victoria Zarotsky, Virginia S Haynes, Glenn Phillips, Carlos I Alatorre, Iris Goetz, Rosirene Paczkowski, and Lauren B Marangell. Comprehensive review of factors implicated in the heterogeneity of response in depression. *Depression and anxiety*, 29 (4):340–354, 2012.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pp. 883–892. PMLR, 2018.

Justin Cheng and Michael S Bernstein. Flock: Hybrid crowd-machine learning classifiers. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pp. 600–611, 2015.

Marco Gillies, Rebecca Fiebrink, Atau Tanaka, Jérémie Garcia, Frédéric Bevilacqua, Alexis Heloir, Fabrizio Nunnari, Wendy Mackay, Saleema Amershi, Bongshin Lee, et al. Human-centred machine learning. In *Proceedings of the 2016 CHI conference extended abstracts on human factors in computing systems*, pp. 3558–3565, 2016.

David L Hall, Michael McNeese, James Llinas, and Tracy Mullen. A framework for dynamic hard/soft fusion. In *2008 11th International Conference on Information Fusion*, pp. 1–8. IEEE, 2008.

David Lee Hall and John M Jordan. *Human-centered information fusion*. Artech House, 2010.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

Maia Jacobs, Melanie F Pradier, Thomas H McCoy, Roy H Perlis, Finale Doshi-Velez, and Krzysztof Z Gajos. How machine-learning recommendations influence clinician treatment selections: the example of antidepressant selection. *Translational psychiatry*, 11(1):1–9, 2021.

Neil Jethani, Mukund Sudarshan, Yindalon Aphinyanaphongs, and Rajesh Ranganath. Have we learned to explain?: How interpretability methods can learn to encode predictions in their interpretations. In *International Conference on Artificial Intelligence and Statistics*, pp. 1459–1467. PMLR, 2021.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Shalmali Joshi, Sonali Parbhoo, and Finale Doshi-Velez. Pre-emptive learning-to-defer for sequential medical decision-making under uncertainty. *arXiv preprint arXiv:2109.06312*, 2021.

Pang Wei Koh, Thao Nguyen, Stephen Tang Yew Siang, Mussmann, Pierson Emma, Been Kim, and Percy Liang. Concept bottleneck models. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Todd Kulesza, Simone Stumpf, Margaret Burnett, Sherry Yang, Irwin Kwan, and Weng-Keen Wong. Too much, too little, or just right? ways explanations impact end users' mental models. In *2013 IEEE Symposium on visual languages and human centric computing*, pp. 3–10. IEEE, 2013.

Isaac Lage, Thomas H McCoy Jr, Roy H Perlis, and Finale Doshi-Velez. Efficiently identifying individuals at high risk for treatment resistance in major depressive disorder using electronic health records. *Journal of Affective Disorders*, 306:254–259, 2022.

David Madras, Toni Pitassi, and Richard Zemel. Predict responsibly: improving fairness and accuracy by learning to defer. *Advances in Neural Information Processing Systems*, 31, 2018.

Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert. In *International Conference on Machine Learning*, pp. 7076–7087. PMLR, 2020.

Sonali Parbhoo, Jasmina Bogojeska, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. Combining kernel and model based learning for hiv therapy selection. *AMIA Summits on Translational Science Proceedings*, 2017:239, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Aimee E Poote, David P French, Jeremy Dale, and John Powell. A study of automated self-assessment in a primary care student health centre setting. *Journal of telemedicine and telecare*, 20(3):123–127, 2014.

Melanie F. Pradier, Thomas H. McCoy Jr, Michael Hughes, Roy H. Perlis, and Finale Doshi-Velez. Predicting treatment dropout after antidepressant initiation. *Translational Psychiatry*, 10(1):60, 2020. doi: 10.1038/s41398-020-0716-y. URL https://doi.org/10.1038/s41398-020-0716-y.

Melanie F Pradier, Javier Zazo, Sonali Parbhoo, Roy H Perlis, Maurizio Zazzi, and Finale Doshi-Velez. Preferential mixture-of-experts: Interpretable models that rely on human expertise as much as possible. *AMIA Summits on Translational Science Proceedings*, 2021:525, 2021.

Maithra Raghu, Katy Blumer, Greg Corrado, Jon Kleinberg, Ziad Obermeyer, and Sendhil Mullainathan. The algorithmic automation problem: Prediction, triage, and human effort. *arXiv preprint arXiv:1903.12220*, 2019.

Tuukka Ruotsalo and Antti Lipsanen. Interactive symptom elicitation for diagnostic information retrieval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 1301–1304, 2018.

Yusuke Sakata, Yukino Baba, and Hisashi Kashima. Crownn: Human-in-the-loop network with crowd-generated inputs. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7555–7559. IEEE, 2019.

Ryusuke Takahama, Yukino Baba, Nobuyuki Shimizu, Sumio Fujita, and Hisashi Kashima. Adaflock: Adaptive feature discovery for human-in-the-loop predictive modeling. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Jiliang Tang, Salem Alelyani, and Huan Liu. Feature selection for classification: A review. *Data classification: Algorithms and applications*, pp. 37, 2014.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Jiaxuan Wang, Jeeheh Oh, Haozhu Wang, and Jenna Wiens. Learning credible models. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2417–2426, 2018.

Bryan Wilder, Eric Horvitz, and Ece Kamar. Learning to complement humans. *arXiv preprint arXiv:2005.00582*, 2020.

Jinsung Yoon, James Jordon, and Mihaela van der Schaar. Invase: Instance-wise variable selection using neural networks. In *International Conference on Learning Representations*, 2018.

Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, volume 3, 2003.

James Zou, Kamalika Chaudhuri, and Adam Kalai. Crowdsourcing feature discovery via adaptively chosen comparisons. In *Third AAAI Conference on Human Computation and Crowdsourcing*, 2015.