Interactive Concept Bottleneck Models

Kushal Chauhan, Rishabh Tiwari, Jan Freyberg, Pradeep Shenoy, Krishnamurthy Dvijotham[*]

Google Research
{kushalchauhan, rishabhtiwari,janfreyberg,shenoypradeep,dvij}@google.com

Abstract

Concept bottleneck models (CBMs) are interpretable neural networks that first predict labels for human-interpretable concepts relevant to the prediction task, and then predict the final label based on the concept label predictions. We extend CBMs to interactive prediction settings where the model can query a human collaborator for the label to some concepts. We develop an interaction policy that, at prediction time, chooses which concepts to request a label for so as to maximally improve the final prediction. We demonstrate that a simple policy combining concept prediction uncertainty and influence of the concept on the final prediction achieves strong performance and outperforms static approaches as well as active feature acquisition methods proposed in the literature. We show that the interactive CBM can achieve accuracy gains of 5-10% with only 5 interactions over competitive baselines on the Caltech-UCSD Birds, CheXpert and OAI datasets.

1 Introduction

Deep learning-based AI systems have demonstrated significant capabilities across a range of applications. However, in many sensitive or safety-critical applications like healthcare or toxicity detection, AI-based predictive systems are seldom deployed in a standalone fashion; instead, they are used as one component in the overall decision-making workflow (Lee et al. 2021).

A concrete interactive setting that has received significant attention in literature is that of active feature acquisition (AFA) (Greiner, Grove, and Roth 2002; Kanani and Melville 2008). In this setting, a classifier can request additional features at prediction time. There is a cost associated with each feature acquisition, so the AFA algorithms have to reason about the value of the acquired feature relative to the cost. In this paper, we focus on a slightly different setting where the classifier always has access to a basic set of features (like pixels of an image). However, at prediction time, the classifier can request additional labels corresponding to human-interpretable concepts that are relevant to the prediction task. For example, when predicting bird species from a bird image, the classifier can request access to the wing shape. A key distinguishing feature of this framework from the general active
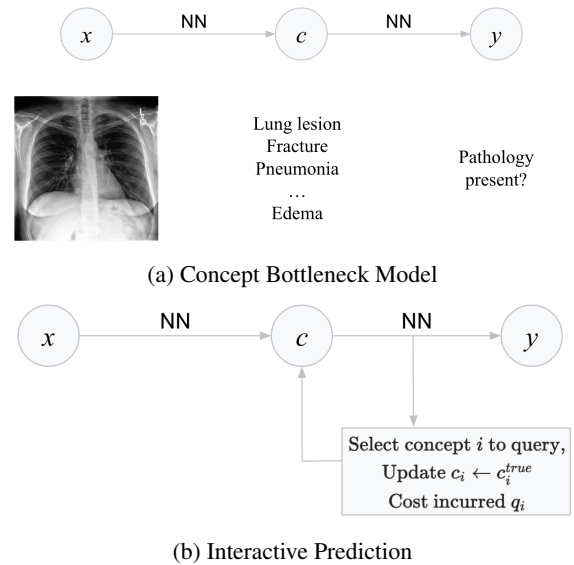
---

[*]Corresponding author

(a) Concept Bottleneck Model

(b) Interactive Prediction

Figure 1: Interactive Prediction: Panel (a) shows a concept bottleneck model (Koh et al. 2020) that predicts a label y from an input x through an intermediate "concept" prediction layer (Figure adapted from Koh et al. (2020)). Panel (b) shows our proposal: after predicting concepts, the system interactively queries the human for true values c_i for concepts chosen so as to maximize prediction accuracy and minimize acquisition cost.

feature acquisition setting is that the human-interpretable concepts are potentially inferrable from the initial features input to the model. In particular, we build on Concept bottleneck models (Koh et al. 2020) which explicitly predict concept labels from images and then predict the final label based on the concept label predictions (Figure 1(a)). The authors argue that CBMs show better explainability, performance under distribution shift, and improvements in performance when concept labels are made available at prediction time. However, they only consider static policies for test time intervention that request concept labels in a predetermined order.

We present Cooperative Prediction (CooP), a dynamic interactive policy that can reason about the uncertainty associated with a given test instance, and request only those con-
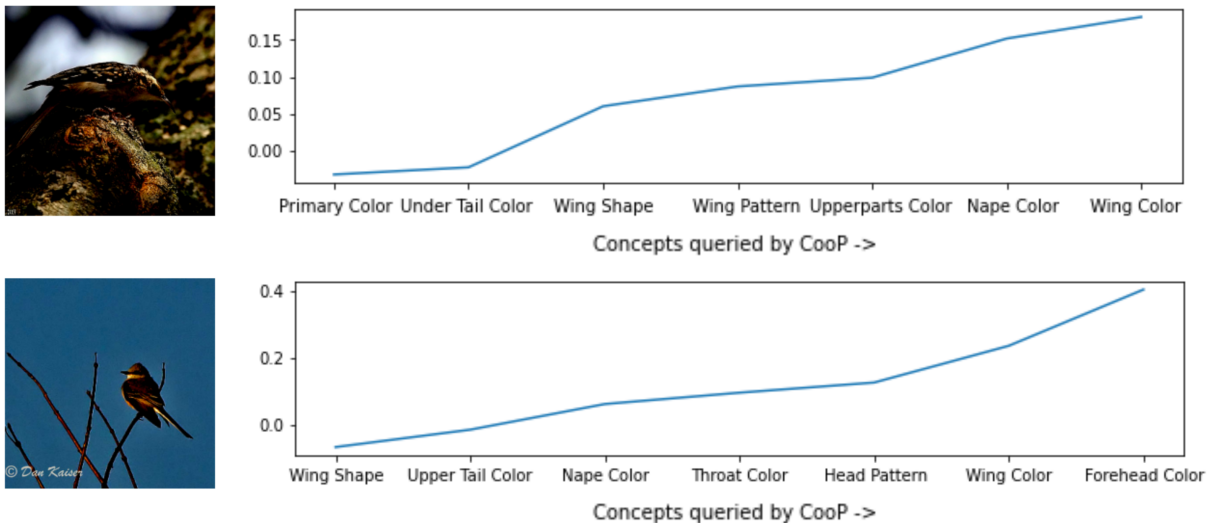
Figure 2: Influence of interventions using queries from the `CooP` policy on a `CUB` trained model on two example images. Y-axis denotes the difference between the probability assigned to the correct and top incorrect class.

cepts that improve predictive power on that instance. Thus, for a given number of queried concepts (or, more generally, a budget for concept acquisition cost), our interactive models can achieve significantly higher levels of performance than static baselines. Figure 2 illustrates with a pair of example images how our approach selects different sequences of concepts to be queried from the user, based on the ambiguity inherent in the specific instances, in a bird species identification task (Wah et al. 2011). The specific sequence of queries allows us to rapidly improve our confidence in the true label.

We make the following **contributions**[1]:

- We develop a simple approach for training policies that act on top of Concept Bottleneck Models (CBMs) from Koh et al. (2020), with the objective of achieving a specified trade-off between interaction cost and predictive performance. Our approach only has a couple of tunable parameters and can be learned using a small validation set separate from the training set used for the CBM.

- We compare our dynamic instance-based query policy against static policies that determine a fixed order of querying attributes for all examples (Koh et al. 2020), as well as SOTA active feature acquisition strategies (Shim, Hwang, and Yang 2018) that apply to more general settings of acquiring features beyond human-interpretable concepts.

- Our model incorporates and optimizes for a cost model of feature acquisition; we show that our approach can adapt to settings with non-uniform costs of querying attributes, and demonstrate superior performance compared to the baselines.

---

[1]Code is available at https://github.com/google-research/google_research/tree/master/interactive_cbms

2 Related Work

The closest relevant work to our paper is that on Active Feature Acquisition and Concept-Aware Models. We review the literature in both and explain how our work is distinguished from this prior work:

Active Feature Acquisition (AFA): The goal of active feature acquisition is to acquire a subset of features, in a cost sensitive manner, for each instance in order to maximize performance at test time. Zubek and Dietterich (2002) proposes a AO* based learning algorithm to heuristically search for classification policy. Ma et al. (2018) and Zannone et al. (2019) uses a partial variational autoencoder to predict the rest of features given the acquired ones to model the feature importance and uncertainty and combine it with acquisition policy to maximize information gain. Shim, Hwang, and Yang (2018) treats this as a joint learning problem trains both the classifier and RL agent together to learn when and which feature to acquire to increase classification accuracy while maintaining cost-efficiency. Li and Oliva (2021) reformulate Markov Decision Process (MDP) and learn a generative surrogate model to capture inter feature dependencies to aid RL agent with intermediate rewards and auxiliary information.

In this paper, we deal with a special case of the general AFA problem where the classifier always has access to an initial set of features (like the pixels in an image), but can request additional human-interpretable concept labels for each prediction. The goal of our work, just like in AFA, is to select which concepts to acquire labels for in a cost sensitive manner. However, we leverage the fact that we are not acquiring arbitrary features but human-intepretable concepts that have strong correlations to the input features and the final label, and exploit the fact that we can train a CBM to solve the prediction task. This makes our approach more perfomant than the general AFA approach.

In Branson et al. (2010), the authors work in exactly the same setting as us and even obtain results on one of the

datasets we use. In this work, the authors posit a generative model for the concept labels given the final label, and assume that the concepts are independent of the input features given the final label. They exploit this assumption to compute a tractable posterior distribution on the label given additional concept labels acquired. Our work does not make any such assumptions, instead relying on the CBM to learn the correlations between the input features and concepts, and the influence of the concept labels on the final label. We achieve stronger results empirically than the approach presented in Branson et al. (2010).

Concept Aware Models: Concept bottleneck models (Koh et al. 2020) were developed to show that building models that explicitly predict concept labels from images or other raw features helps with explainability, performance under distribution shift and improvements in performance when concept labels are made available at prediction time. CBMs have been extended in various ways: Bahadori and Heckerman (2021) performs causal reasoning to debias CBMs, Antognini and Faltings (2021) develops textual rationalizations based on concept interventions. However, of these works, only Koh et al. (2020) explicitly considers test time intervention, and even here they only consider static policies for test time intervention that request concept labels in a fixed predetermined order (learned on a validation set). In this work, we allow for dynamic interactive policies that, for each prediction, reason about which concept labels are useful to improve the reliability of the prediction and request only those. Thus, with the same number of concepts allowed to be queried at prediction time, our interactive models can achieve significantly higher levels of performance than static baselines.

3 Methods

We outline the problem formulation as well as a simple approach to computing interactive policies on top of pretrained predictive models.

3.1 Formalizing an Interactive Prediction System

Input, Concept and Label Spaces: We denote raw input features by x (these can correspond, for example to images or text or features derived from these), the labels y and the acquired concepts c. We assume that c is a vector of m categorical concepts and y is a categorical scalar taking K possible values. Individual concepts are denoted $c_i, i = 1, \ldots, m$. The set of possible values c can take is denoted $\mathcal{C}$ (with the set of possible values for c_i being denoted $\mathcal{C}_i = \{1, \ldots, n_i\}$ so that $\mathcal{C} = \prod_i \mathcal{C}_i$) and the set of possible values y can take is denoted $\mathcal{Y} = \{1, 2, \ldots, K\}$. We denote the space of possible inputs by $\mathcal{X}$. We also note here that the term concept is loosely applied - in some contexts, it may refer simply to additional attributes (a user's age or gender, for example) or pieces of information that can be acquired at some cost at prediction time. Hence we use the term concepts or attributes interchangeably.

Concept Bottleneck Model: The CBM is the composition of an input-to-concept ($\mathcal{X} \rightarrow \mathcal{C}$) model $p_\theta(c|x)$ and a concept-to-label ($\mathcal{C} \rightarrow \mathcal{Y}$) model $p_\phi(y|c)$. We assume that both models make probabilistic predictions and that the input-to-concept model makes an independent probabilistic prediction for each concept c_i, denoted by $p_\theta(c_i|x)$ for each $i = 1, \ldots, m$. We assume that these models have been trained and are available to us and do not make any assumptions about how the models were trained, or whether the predictions output represent calibrated probabilities.

Intervention: In the absence of interactivity, the two stage model makes predictions by first invoking the $\mathcal{X} \rightarrow \mathcal{C}$ model and then passing the output to the $\mathcal{C} \rightarrow \mathcal{Y}$ model to get the final prediction. However, in our setting, we allow for interventions on the concepts, i.e, replacing the predicted value (or distribution over values) of a concept with its ground truth value. We denote the prediction concept values as $\hat{c} = p_\theta(\cdot|x)$ and the intervened concept values (i.e. the ground truth) as c. We further denote the prediction of a $\mathcal{C} \rightarrow \mathcal{Y}$ model with partial intervention on a subset of concepts S as $p_\phi(y|c_S, \hat{c}_{\overline{S}})$. We also use the notation $p_\phi(c_S = v, \hat{c}_{\overline{S}})$ to denote the predictions given the intervention where the concepts c_S are set to a specific value v.

Interaction Cost Model: We denote the cost of acquiring an attribute c_i as $q_i > 0$. We assume all costs are positive and that the cost of acquiring a concept is the same independent of the previously acquired concepts or the value the concept takes. The total cost of acquiring a set of attributes S is assumed to be $\sum_{i \in S} q_i$. Extensions that don't make these assumptions are possible but we leave this to future work. We assume that for each prediction made, there is a budget B and that the interaction can only occur while the total cost of concepts acquired so far is below B.

Interactive Policies: Given the two stage model, we define an interactive policy ψ as follows: An interactive policy takes as input a set of revealed concepts c_S where $S \subseteq \{1, \ldots, m\}$, the $\mathcal{X} \rightarrow \mathcal{C}$ and $\mathcal{C} \rightarrow \mathcal{Y}$ models and interaction costs q and outputs the new concept to acquire:

$$\psi(S, p_\theta, p_\phi, q, B) = i \in \overline{S} = \{1, \ldots, m\} \setminus S$$

We will usually drop the dependence on p_θ, p_ϕ, q (as these are assumed to be always available) and simply write $\psi(S)$. A rollout of an interaction policy corresponds to the Algorithm 1. The final prediction generated on an input x is denoted rollout (ψ, x)

Separation of Policy and Model Learning: In this paper, we restrict ourselves to learning an interactive policy as a post-hoc step on top of a learned model. We do not make assumptions about how the two-stage model (i.e., $p_\theta(\cdot), p_\phi(\cdot)$) is trained.

Dataset for Policy Learning: We assume that we can acquire a dataset consisting of (x, c, y) triplets sampled iid from an underlying unknown joint distribution $\mathbb{P}_{\text{data}}$. Our goal is then to minimize

$$\mathbb{E}_{(x,c,y) \sim \mathbb{P}_{\text{data}}} [\ell(y, \text{rollout}(\psi, x))] \tag{1}$$

where ℓ is a loss function that measures the discrepancy between the true label y and the label output by rolling out the policy ψ.

Algorithm 1: Policy Rollout

$S \leftarrow \emptyset$
while $b \leq B$ **do**
    $i \leftarrow \psi(S)$
    if $b \leq B - q_i$ **then**
        Acquire the label for concept c_i
        $S \leftarrow S \cup \{i\}$
        $b \leftarrow b + q_i$
    else
        $b \leftarrow B + 1$
    end if
end while
Output prediction $\operatorname{argmax}_{y \in \mathcal{Y}} p_\phi\left(c_S, p_\theta\left(c_{\overline{S}}|x\right)\right)$

Algorithm 2: `CoOP` policy

Given p_θ, p_ϕ, q, set of concepts acquired so far S and highest scoring predicted label based on acquired concepts $k \in \mathcal{Y}$, and score importance weights $\alpha, \beta, \gamma \in \mathbb{R}_+$
for all $i \in \overline{S}$ **do**
    Compute $\text{score}_i = \alpha\text{CPU}(c_i; S) + \beta\text{CIS}(c_i; S, k) - \gamma q_i$
end for
Output $\operatorname{argmax}_i \text{score}_i$

3.2 Interactive Policy Learning with Cooperative Prediction (`CoOP`)

We present Cooperative Prediction (`CoOP`), a lightweight approach to learning interactive policies that attempt to optimize the objective in equation (1).

The key intuition behind `CoOP` is that deciding which concept labels to acquire should be informed by three considerations: a) The uncertainty associated with the concept prediction - if we can already infer the concept label with high confidence based on the input features, there is not much value to acquiring it. b) The impact of the concept label on the final label prediction - If knowing the value of the concept does not change the predicted label confidence scores by much, it is not very valuable. c) Cost of acquiring the concept.

`CoOP` uses a very simple measure of each of the three components, and chooses concepts to acquire iteratively in a greedy fashion by developing a score function based on the three components and choosing the concept not yet acquired that scores the highest. In particular we use the following measures:

- *Concept prediction uncertainty (CPU):* We compute the entropy of the distribution $p_\theta(c_i|x)$ for each concept c_i with $i \notin S$

$$\mathcal{H}\left[p_\theta(c_i|x)\right]$$

- *Concept importance score:* We compute the expected change in the softmax score $p_\phi(y|c)$ associated with the final label prediction when the concept label for each concept c_i is changed in the inputs to the $\mathcal{C} \to \mathcal{Y}$ model, i.e.:

$$\text{CIS}(c_i; S, k) =$$
$$\left| \mathbb{E}\left[p_\phi\left(y = k | c_i = v, c_S, \hat{c}_{\overline{S}\setminus\{j\}}\right)\right] - p_\phi\left(y = k | c_S, \hat{c}_{\overline{S}}\right)\right|$$

where k was the label predicted in the previous round of the interaction and the expectation is taken over concept values $v \sim p_\theta(c_i|x)$.

- *Acquisition cost:* This is simply the acquisition cost of each attribute q_i.

The final score is simply a linear combination of normalized versions of these scores (each score is normalized so the

range of values it takes on the policy learning dataset lies between 0 and 1) and the overall attribute selection algorithm is outlined in algorithm 2. Each linear combination of scores leads to a different interactive policy, and those weights are tuned to optimize performance on a holdout validation set.

A primary advantage of the proposal is its ease of learning, especially in sparse-data scenarios, as it only requires that we estimate two mixing parameters. As the first paper proposing this novel problem setting (to our knowledge), our primary goal here is to demonstrate that careful policy selection can indeed provide value. We leave further improvements and theoretically principled approaches to this problem for future work, here instead focusing on demonstrating that a simple approach works well for this novel problem setting and formulation.

4 Experiments

4.1 Datasets

We experimented with the following datasets, with characteristics summarized in Table 1:

CUB (Caltech-UCSD Birds): This dataset contains pictures of birds coupled with human-labeled concept attributes identifying prominent characteristics (wing color, beak length, undertail color, etc.) (Wah et al. 2011). The there are 28 such categorical concepts, resulting in a total of 112 binary labels. In an interactive setting, attributes are revealed at prediction time only when the policy queries an attribute. In practice, this could be seen as asking human labelers in an interactive setting to provide specific hints on concepts they can easily identify (like beak length or wing color) even if they are unable to make the final prediction on what species of bird it is, as most labelers will be unable to do this unless they are specifically trained on this task.

CHEXPERT: This dataset contains chest X-rays accompanied by binary concept labels extracted from a report generated by a radiologist, with the goal of predicting whether the X-ray was normal or abnormal (Irvin et al. 2019). Each chest X-ray is also accompanied by 13 binary attributes that include concepts easily recognized by a non-expert (presence of a fracture or any support devices on the patient), harder-to-label attributes that need a nurse or physician (e.g., Cardiomegaly), and, finally, attributes that require a radiologist to label.

OAI (Osteoarthritis Initiative): This dataset contains knee X-rays, annotated with the Kellgren-Lawrence grade (KLG), a 4-level ordinal variable (assumed to be categorical for training) that measures the severity of knee osteoarthritis. Each

knee X-ray is also annotated with 10 ordinal attributes describing joint space narrowing, bone spurs, calcification, etc., resulting in a total of 40 binary concepts.

4.2 Base Models

Following the proposals made by Koh et al. (2020), we train the following 2 kinds of concept-bottleneck models (CBMs):[2]

Independent model: The $\mathcal{X} \to \mathcal{C}$ and $\mathcal{C} \to \mathcal{Y}$ models are trained separately, respectively mapping the inputs x to the true concepts c, and the true concepts c to the labels y respectively. The $\mathcal{C} \to \mathcal{Y}$ model sees true concept values as input at train-time, and estimated values (probabilities) at test time.

Joint model: Both $\mathcal{X} \to \mathcal{C}$ and $\mathcal{C} \to \mathcal{Y}$ models are learned using a joint optimization criterion which combines the concept prediction loss (cross-entropy) and label prediction loss (cross-entropy). The probabilities output by the $\mathcal{X} \to \mathcal{C}$ model are passed on to the $\mathcal{C} \to \mathcal{Y}$ model during training.

We build our interactive intervention models on top of these CBMs, and propose and evaluate various methods of intervention using each of these as the base CBM. Although we performed extensive experimentation with both the Independent and Joint models, due to space considerations we present only results from the Independent CBM here. Findings on the Joint model are qualitatively similar; please see Chauhan et al. (2022) for more details.

4.3 Training and Evaluation

For each experiment, we split the data into 3 sets: train, validation, and test – the details are available in Table 1. We used the training data to train the base CBMs, and validation data to select parameter settings, if any, for intervention policies. We then retrained the base model on pooled train + validation datasets[3], and finally reported performance of the policies on the unseen test set. In case an intervention policy did not require parameter selection, we directly trained the base model on the pooled train+val data. Finally, for CooP, we require accurate measures of uncertainty in order to make effective decisions; we, therefore, calibrate the pooled concept probabilities across the training data for a given base model using isotonic regression.

Our primary metric is accuracy for CUB and OAI, and AUC for CHEXPERT at the classification tasks. For each intervention policy, we start with performance using only the input x. We then iteratively obtain true labels for intermediate concepts c as specified by the policy, set the concept value to the observed true value, and report the performance of the updated prediction after adding the observed concept. In this manner, we obtain a curve measuring the performance metric as a function of the number of observed concepts.

	CUB	CHEXPERT	OAI
Input Dims	(299, 299, 3)	(320, 320, 3)	(512, 512, 3)
Concepts	112	13	10
Data splits			
train	4,796	178,731	31,370
val	1,198	22,341	4,426
test	5,794	22,342	4,522

Table 1: Details of the datasets used in our experiments.

4.4 Baselines for Comparison

Greedy: Select an ordering of attributes using a greedy ranking scheme over the validation dataset; i.e., the first attribute in the list is the one that improves the performance measure the most on average over the validation set. Subsequent attribute orders are determined in a similar greedy fashion after conditioning on all previous attributes as being available.

Random: For each instance in the training set, choose the next attribute to query at random.

Skyline: Evaluate an oracle greedy approach which checks, for each instance in the *test set*, the specific greedy order of querying attributes that provides maximum incremental gains on each step for that test instance. This is an *oracular* skyline since it uses the test label for optimization, and is an approximate *ceiling* on the performance achievable under any interactive policy.[4]

Active Feature Acquisition (AFA): We also compare against the Active Feature Acquisition policy based on work by Shim, Hwang, and Yang (2018). We use image embeddings from ResNet18 pre-trained on ImageNet as auxiliary information, and ground truth concepts as features that can be acquired; we train an RL policy to actively acquire features/concepts as described in Shim, Hwang, and Yang (2018). This algorithm is relatively sensitive to the value of a hyperparameter r_cost that determines the tradeoff between model performance and acquisition cost. For evaluating AFA, we performed a hyperparameter search for r_cost in the range of [-1e-5, -0.06] for each number of intervention steps using the accuracy/AUC on the validation set. We also tried using fine-tuned ResNet features instead of pre-trained features as auxiliary information, but it resulted in severe overfitting on training data. For a fair comparison, we use the same data splits as other baselines and the policy is trained until convergence.

4.5 Intervention Costs

In practice, it is likely that obtaining certain concept labels has a higher cost, for example, due to the difficulty of the annotation task or the data being purchased at a cost from a data provider, or the privacy costs of asking for sensitive user information. We studied the following cost models:

---

[2]Although CooP is agnostic to the specific way the base CBM has been trained, each base CBM may have idiosyncrasies in its predictive power that interact with CooP; therefore, specific base CBMs combined with CooP may perform overall better on any given dataset.

[3]We did this due to data paucity in the datasets we studied. This is common practice for the datasets, and does not compromise the

findings since all reported results are on an unseen test set.

[4]The reason this skyline is only approximate-oracular is that searching all possible orderings of attributes for querying is combinatorially infeasible; as seen in our results, the proposed greedy approximation quickly approaches 100% performance measure suggesting it is a good approximation.
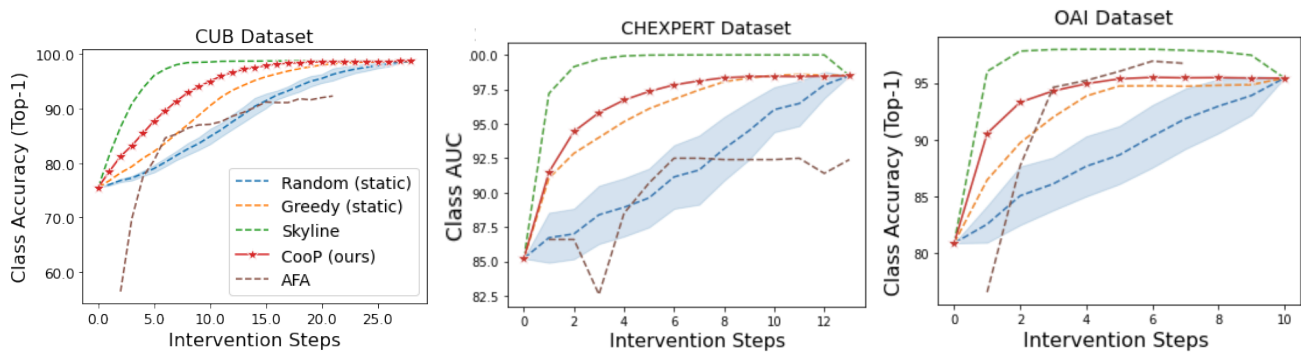
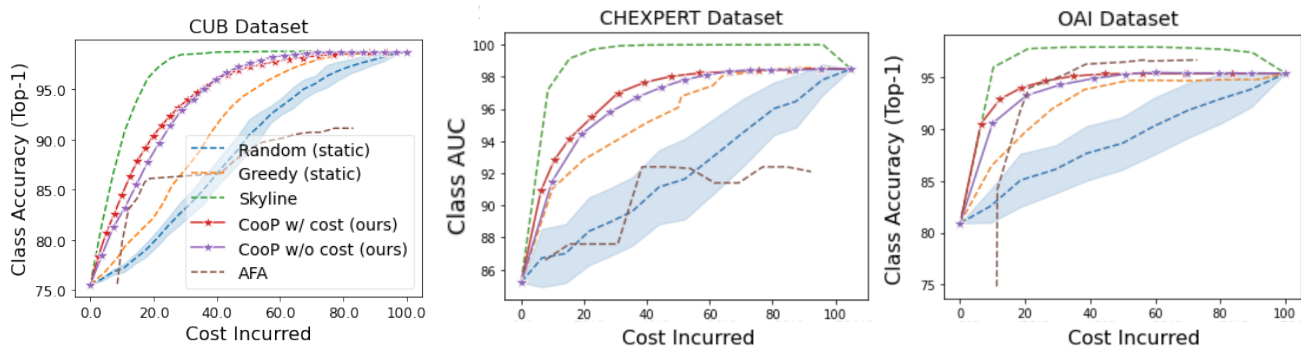Figure 3: Accuracy gains vs interaction cost (unit cost model) on different datasets. See text for details.

Figure 4: Cost-efficient interventions on different datasets. For CUB and OAI, we report mean results for 10 random cost assignments. For CHEXPERT, we use the domain-informed cost model.

Unit cost: Each query made by the interactive policy incurs a unit cost. This is largely appropriate for the CUB dataset since the concepts are largely identifiable easily by a non-expert.
Random cost: To stress test our approach, we also experiment with random cost models where a randomly chosen cost from the range $[1, 7]$ is assigned to each concept, which is then normalized such that the total cost of all concepts is 100.
Systematic cost: In datasets like CHEXPERT , it is clear that some attributes can be easily labeled by non-experts while others require a specialized radiologist. Thus we assign systematic costs, that have a strong justification depending on the difficulty of acquisition of a concept label. Based on consultations with domain experts, we use concept acquisition costs of 1, 3, and 10 for concepts that are very easy, moderately difficult, and very difficult to annotate respectively. Similar to random costs, we also normalize CHEXPERT's systematic costs such that the total cost of all concepts is 100.

The costs are factored into the learning of policies for CooP as described in Section 3.2. For the CHEXPERT dataset where a systematic cost was available, we used it to evaluate our cost optimization procedure; for CUB and OAI, we used random costs.

5 Results

We perform experiments that demonstrate the following valuable contributions from CooP :
Performance gains from adaptive interactive policies:

Section 5.1 shows that by querying just 5 additional concept labels at prediction time, our interactive policies can improve relevant performance metric (AUC or accuracy) by 20-25% over any static baseline that determines in advance an order in which attributes are queried. In some cases, CooP achieves a sizable fraction of the possible gain determined by the skyline. Improving CooP further to fully close the gap to the skyline is a compelling direction for future work.
Cost-aware acquisition: Relative to policies that only use uncertainty to drive interactivity, CooP is cost-aware and acquires concept labels only when they are valuable, achieving a better tradeoff between cost and performance than simpler policies (Section 5.2).
We conducted additional analyses, including sample efficiency for CooP, and ablation studies probing the contributions of different factors in our scoring function. Details are available at Chauhan et al. (2022).

5.1 Predictive Performance Improvement

Figure 3 shows the results of our experiments on three different datasets: CUB, CHEXPERT, and OAI.

Across all 3 datasets, we see that the simple Greedy policy already outperforms Random consistently across the range of intervention steps. This serves as a strong existence proof of nontrivial intervention policies. Note that the greedy policy requests concept labels in the same sequence regardless of the test data instance. In contrast, CooP uses instance-
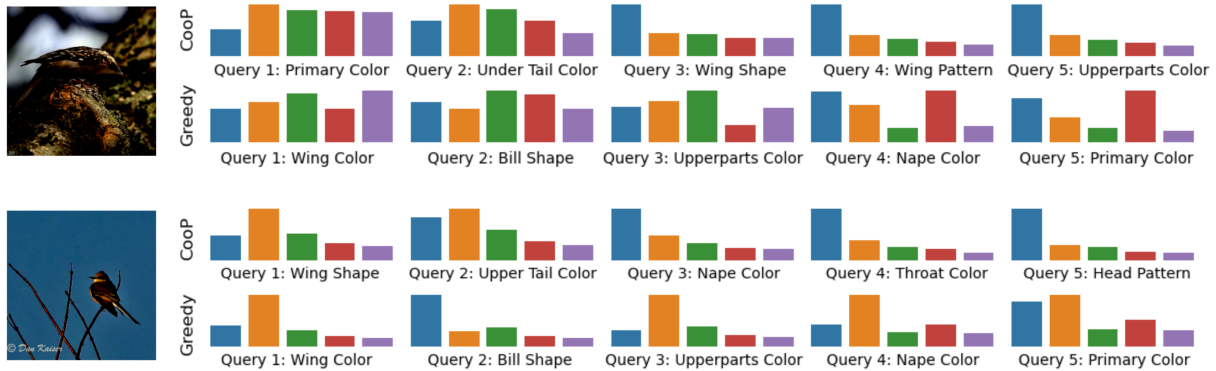
Figure 5: Comparison of the behavior of `CooP` and `Greedy` policies for the first five steps of intervention using two example images. The bar plots show the probabilities assigned to top 5 ranking classes according to the model, with the first blue bar representing the correct class. The bar plot titles represent the concept revealed by the respective policy.

specific uncertainties both in concept labels and final predictions, and as a result significantly outperforms `Greedy`, again across datasets. In particular, `CooP` is able to substantially improve the performance metric with as few as 5 queries.

We also note that the `AFA` baseline has an uneven performance across datasets and the range of intervention steps. For instance, the initial performance of AFA is quite poor, followed by a rapid rise in the `CUB` and `OAI` datasets. In `CHEXPERT`, `AFA` fails to improve upon `Random`, a weak baseline. Finally, on the `OAI` dataset, `AFA` does outperform `CooP`, although only by small margins and after the key first few queries. This variable performance is driven by two factors: the significant data need for the `AFA` algorithm, and the need for selecting a particular tradeoff cost at training time, rather than being able to smoothly adjust the cost-benefit tradeoffs at test time on a per-instance basis.

An interesting finding is that for all datasets, `Skyline`[5] performs noticeably better than both the other baselines and `CooP`. Even though `Skyline` is an oracle, this finding suggests the possibility of additional headroom available for other sophisticated, potentially data-hungry policies.

Figure 5 illustrates `CooP`'s ability to customize intervention queries to each instance, which is its key differentiating feature as compared to `Greedy`. In the first image, `Greedy` chooses to reveal concepts like bill shape and wing color, which can be reasonably inferred from the image. `CooP` instead chooses to reveal under tail color which is difficult to see in the shadow, and the wing shape which can't be inferred since the bird's wings are closed. Similarly, in the second image, most concepts that `Greedy` chooses to reveal are visible to some extent. `CooP` instead queries concepts like upper tail color and head pattern which are hidden in the shadows, and wing shape, which also can't be inferred since the wings are closed.

---

[5]Although additional information should never worsen performance, we do see that adding certain concepts decreases accuracy, particularly in the Skyline. This is due to the heuristic nature of the Skyline, and the sparse-data settings that limit the base CBM's exposure to certain rare concepts in the training data.

5.2 Cost-Efficient Interventions

In the previous experiments, we assumed that all interventions (concept labels) have unit cost; we now explore the scenario where different concepts have different costs (see Section 4.5 for details on the cost model). Figure 4 shows the result of `CooP` when optimizing for intervention costs. The presented data is similar to Figure 3, except that the tradeoff is now accuracy versus total cost, as opposed to the total number of steps previously. We see that `CooP` outperforms the baselines, both without and with cost-sensitive selection, and `CooP` with the cost is better. This demonstrates `CooP`'s ability to incorporate cost structure into the optimization of the interactive policy.

6 Discussion

We proposed a novel problem setting, that of iterative/interactive refinement of model predictions using human inputs, and the cost-efficient optimization of this interactive loop. We demonstrated a principled first-cut approach at learning such optimization policies in the context of two-stage, or *concept-bottleneck* models where interactions are simplified to querying concept or attribute labels. We do not provide a wide or exhaustive discussion of the advanced algorithmic possibilities; indeed, we anticipate that future work will explore a number of alternate formulations both of learning the base models, and of optimizing interactive policies on top of those base models. In particular, one could hypothesize architectures other than the two-stage CBMs explored here. Further, human inputs could include information other than the bottleneck concepts – for instance, side information that cannot be inferred from the input data, region-of-interest annotations, etc. A key related challenge for our work (and concept bottleneck models in general) is the need for intermediate supervision in the form of concept labels. Future work could explore weak or distant supervision for obtaining these concept labels. Finally, there would be a need to evaluate our approach in realistic human-AI collaboration setups, where UX or other psychological factors may impact the interactive performance of the human-AI team (Bansal et al. 2021).

Ethics Statement

Our goal is to increase the interpretability and robustness of predictive models, a significant net positive especially for applications such as medical diagnosis. As such, we do not expect any adverse outcomes of our work or follow-on research. For our experiments, we have used open-sourced datasets collected with appropriate review processes; we have not conducted human-in-the-loop experiments as it was out of scope for this paper. For an eventual system that includes humans in the predictive workflow, our proposal only leverages additional instance specific data at test-time, as supplied by the expert responsible for prediction; this minimizes the potential for data misuse by our model.

References

Antognini, D.; and Faltings, B. 2021. Rationalization through Concepts. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 761–775.

Bahadori, M. T.; and Heckerman, D. E. 2021. Debiasing concept-based explanations with causal analysis. In *ICLR 2021*.

Bansal, G.; Wu, T.; Zhou, J.; Fok, R.; Nushi, B.; Kamar, E.; Ribeiro, M. T.; and Weld, D. 2021. Does the whole exceed its parts? the effect of ai explanations on complementary team performance. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 1–16.

Branson, S.; Wah, C.; Schroff, F.; Babenko, B.; Welinder, P.; Perona, P.; and Belongie, S. 2010. Visual recognition with humans in the loop. In *European Conference on Computer Vision*, 438–451. Springer.

Chauhan, K.; Tiwari, R.; Freyberg, J.; Shenoy, P.; and Dvijotham, K. 2022. Interactive Concept Bottleneck Models. *arXiv preprint arXiv:DOI*.

Greiner, R.; Grove, A. J.; and Roth, D. 2002. Learning cost-sensitive active classifiers. *Artificial Intelligence*, 139(2): 137–174.

Irvin, J.; Rajpurkar, P.; Ko, M.; Yu, Y.; Ciurea-Ilcus, S.; Chute, C.; Marklund, H.; Haghgoo, B.; Ball, R.; Shpanskaya, K.; et al. 2019. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 590–597.

Kanani, P.; and Melville, P. 2008. Prediction-time active feature-value acquisition for cost-effective customer targeting. *Advances in neural information processing systems (NIPS)*.

Koh, P. W.; Nguyen, T.; Tang, Y. S.; Mussmann, S.; Pierson, E.; Kim, B.; and Liang, P. 2020. Concept Bottleneck Models. In III, H. D.; and Singh, A., eds., *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 5338–5348. PMLR.

Lee, M. H.; Siewiorek, D. P.; Smailagic, A.; Bernardino, A.; and Bermúdez i Badia, S. 2021. A Human-AI Collaborative Approach for Clinical Decision Making on Rehabilitation Assessment. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 1–14.

Li, Y.; and Oliva, J. 2021. Active feature acquisition with generative surrogate models. In *International Conference on Machine Learning*, 6450–6459. PMLR.

Ma, C.; Tschiatschek, S.; Palla, K.; Hernández-Lobato, J. M.; Nowozin, S.; and Zhang, C. 2018. Eddi: Efficient dynamic discovery of high-value information with partial vae. *arXiv preprint arXiv:1809.11142*.

Shim, H.; Hwang, S. J.; and Yang, E. 2018. Joint Active Feature Acquisition and Classification with Variable-Size Set Encoding. In *NeurIPS*, 1375–1385.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset. *Computation & Neural Systems Technical Report*.

Zannone, S.; Hernández-Lobato, J. M.; Zhang, C.; and Palla, K. 2019. Odin: Optimal discovery of high-value information using model-based deep reinforcement learning. In *ICML Real-world Sequential Decision Making Workshop*.

Zubek, V. B.; and Dietterich, T. G. 2002. Pruning Improves Heuristic Search for Cost-Sensitive Learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, 19–26. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 1558608737.