Online Appendix — Collaborative Intelligence in Sequential Experiments: A Human-in-the-Loop Framework for Drug Discovery

Appendix A.1: Notations

Table A.1.1 Notation Table

Notation	Explanation	Notation	Explanation
$\mathcal{V}$	Set of constituents that compose the molecule	$\boldsymbol{\mu}$	Predicted mean vector
$\mathcal{X}$	Molecule search space	$\boldsymbol{\sigma}_m(\boldsymbol{x}) \in \mathcal{R}^K$	Model uncertainty (epistemic uncertainty)
$\boldsymbol{x} \in \mathcal{X}$	Molecule sequence	$\boldsymbol{\sigma}_d(\boldsymbol{x}) \in \mathcal{R}^K$	Data uncertainty (aleatoric uncertainty)
$\boldsymbol{z} \in \mathcal{R}^{\lvert\mathcal{Z}\rvert}$	Real-number embedding of a molecule	π	Search policy
$K \in Z_+$	Number of properties considered	$S_\pi \in Z_+$	Number of target molecules found under policy π
$\boldsymbol{y}(\boldsymbol{x}) \in \mathcal{R}^K$	Vector of values for K properties	$S_\pi^t \in Z_+$	Number of target molecules found under policy π in the first t rounds
$\mathcal{D}^{(t)}_{\text{labeled}}, \mathcal{D}^{(t)}_{\text{unlabeled}}$	Labeled/Unlabeled dataset in round t	$B \in Z_+$	Search budget in each round
$\mathcal{C}_\theta$	Bayesian neural network model parameterized by parameters θ	$R \in Z_+$	Total number of experiment rounds
$r_{un}(\boldsymbol{x}) \in \mathcal{R}$	Uncertainty score	$r_{se}(\boldsymbol{x}) \in \mathcal{R}$	Search score

Appendix A.2: Evolutionary-Scale Deep Learning Transformer Language Model

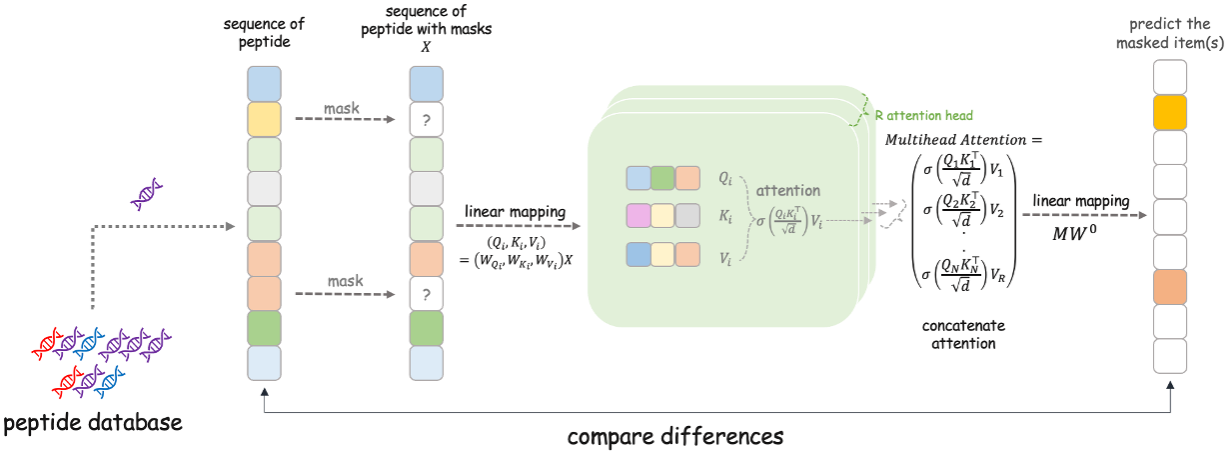
To analyze peptide sequences, which are essentially unstructured letter sequences, we employ an approach that converts them into structured representations. Specifically, we use the ESM2 model (Lin et al. 2023), a transformer-based protein language model that effectively converts protein sequences from character formats into a structured latent space.

The ESM2 model, trained on over 65 million existing peptide sequences, employs an attention mechanism and a masking scheme (Figure A.2.1). This involves randomly masking an amino acid in a sequence and training the model to predict the masked amino acid. The model optimization focuses on a masked language model objective, where the loss function is defined as the negative log likelihood of correctly predicting the masked amino acid given the context of the unmasked sequence:

$$\mathcal{L}_{MLM} = \mathbb{E}_{x\sim X}\mathbb{E}_M \sum_{i\in M} -\log p\left(x_i \mid x_{/M}\right), \tag{A.2.1}$$

where x_i is the true amino acid, M indicates the set of masked components, and $x_{/M}$ is the masked sequence.

Figure A.2.1 Diagram of the ESM2 Transformer Architecture



Note. The ESM2 model is a transformer-based deep learning language model specialized in protein sequence analysis. It transforms unstructured protein sequences into structured, real-number latent spaces. Utilizing an attention mechanism and a masking scheme, ESM2 predicts masked amino acids within sequences, providing high-dimensional embeddings.

The model's self-attention mechanism processes a sequence of vectors and computes interactions among all elements. It transforms an input sequence $(x_1, \ldots, x_n)$ into an output sequence $(x'_1, \ldots, x'_n)$. The attention mechanism is based on scaled dot-product attention:

$$\text{Attention}_j = \sigma\left(\frac{Q_j K_j^T}{\sqrt{d}}\right) V_j, \tag{A.2.2}$$

where Q_j (query), K_j (key), and V_j (value) are the j^{th} linear transformations of the input sequence into matrices. The attention scores, obtained from the outer product of Q and K, are scaled by the square root of the dimension of the matrices d and transformed into a combination of the value sequence V through the softmax function σ. The subscript $j \in \{1, 2, \cdots, R\}$ represents the attention score for one from R heads, which corresponds to one set of Q, K, and V tuples.

The use of multi-headed self-attention enables diverse inter-position interaction patterns. This is achieved as the concatenated output of multiple attention heads, represented by Multi-Attention $= (\text{Attention}_1, \ldots, \text{Attention}_R)$ is channeled into a subsequent feedforward layer, generating the latent representation of a sequence. The transformer architecture, with approximately 15 billion parameters and incorporating rotary position embeddings, equips the model with the ability to express these protein molecules.

In summary, we use the ESM2 model to encode peptide sequences $\boldsymbol{x} \in \mathcal{X}$ into $\boldsymbol{z}$ in a real-number embedding feature space $\mathcal{Z}$, where $|\mathcal{Z}|$ is the dimension of this space. This model ensures that the distance $\|\boldsymbol{z}_i - \boldsymbol{z}_j\|$ between embeddings of two peptides $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is minimal for similar sequence patterns. The encoded peptide sequence $\boldsymbol{z}$ is thus represented as a tuple of features $(z_1, z_2, \ldots, z_{|\mathcal{Z}|})$, serving as inputs for the Bayesian neural network models $\{\mathcal{C}_{\theta_j}\}_{j=1}^k$.

Appendix A.3: Pseudocode for Human-in-the-loop Framework for Sequential Experiments

We present the pseudocode for our human-in-the-loop framework in Algorithm 1.

Algorithm 1: Human-in-the-loop Framework for Sequential Experiments

Input: Initialize K Bayesian neural network models $\{\mathcal{C}_{\theta_i^{(1)}}\}_{i=1}^K$, $\mathcal{D}_{\text{unlabeled}}^{(1)} = \left\{\boldsymbol{x}_j^u\right\}_{j=1}^J$, $\mathcal{D}_{\text{labeled}}^{(1)} = \varnothing$

for $t \in \{1, 2, \cdots, R\}$ **do**

1: Estimate $\{\boldsymbol{\mu}(\boldsymbol{x}_j^u), \boldsymbol{\sigma}_m(\boldsymbol{x}_j^u), \boldsymbol{\sigma}_d(\boldsymbol{x}_j^u)\}$ by the algorithm $\{\mathcal{C}_{\theta_i^{(t)}}\}_{i=1}^K$;

2: Compute uncertainty scores $r_{un}(\boldsymbol{x}_j^u)$ and search scores $r_{se}(\boldsymbol{x}_j^u)$ for all $\boldsymbol{x}_j^u \in \mathcal{D}_{\text{unlabeled}}^{(t)}$;

3: Recommend h molecules $\{\boldsymbol{x}_{se,i}^{(t)}\}_{i=1}^h$ with the highest search scores, and q molecules $\{\boldsymbol{x}_{un,i}^{(t)}\}_{i=1}^q$ with the highest uncertainty scores;

4: Human experts examine the recommended molecules $\{\boldsymbol{x}_{un,i}^{(t)}\}_{i=1}^q \bigcup \{\boldsymbol{x}_{se,i}^{(t)}\}_{i=1}^h$ and choose B molecules $\{\boldsymbol{x}_i^{(t)}\}_{i=1}^B$ for laboratory experiments, and obtain $\{\boldsymbol{y}_i^{(t)}\}_{i=1}^B$;

5: Update $\mathcal{D}_{\text{labeled}}^{(t+1)} = \mathcal{D}_{\text{labeled}}^{(t)} \bigcup \{(\boldsymbol{x}_i^{(t)}, \boldsymbol{y}_i^{(t)})\}_{i=1}^B$,$\mathcal{D}_{\text{unlabeled}}^{(t+1)} = \mathcal{D}_{\text{unlabeled}}^{(t)} \backslash \{\boldsymbol{x}_i^{(t)}\}_{i=1}^B$, and update $\theta_i^{(t)}$ with $\mathcal{D}_{\text{labeled}}^{(t+1)}$ to obtain the improved algorithm $\{\mathcal{C}_{\theta_i^{(t+1)}}\}_{i=1}^K$.

end

Appendix A.4: Model Uncertainty and Data Uncertainty

Data uncertainty, also referred to as aleatoric uncertainty, characterizes the inherent variability within the data itself. In regression problems, this uncertainty, denoted as $\sigma_{d,reg}$, can be inferred directly from the data. This is achieved by adapting the model and loss function to predict a distribution over possible outcomes, rather than a single point estimate. It is managed by adjusting the loss function to capture both the mean and standard deviation for each data point, assuming that the noise follows a normal distribution. In this Bayesian framework, a model parameterized by θ estimates both the mean $\mu_\theta(x)$ and the standard deviation $\sigma_\theta(x)$ of the output distribution y.

The optimal parameters θ^* are determined by minimizing a loss function that accounts for both the prediction error and the data uncertainty. This optimization can be expressed as:

$$\theta^* = \arg\min_\theta \frac{1}{N} \sum_{i=1}^N \frac{1}{2\sigma_\theta^2(x_i)} \left\| y_i - \mu_\theta(x_i) \right\|_2^2 + \frac{1}{2} \log \sigma_\theta^2(x_i). \tag{A.4.1}$$

Once the model is trained, the data uncertainty at a specific input x is given by the standard deviation of the output estimate, as determined by the learned parameters θ^*. This is mathematically represented as:

$$\sigma_{d,reg}(x) = \sigma_{\theta*}(x). \tag{A.4.2}$$

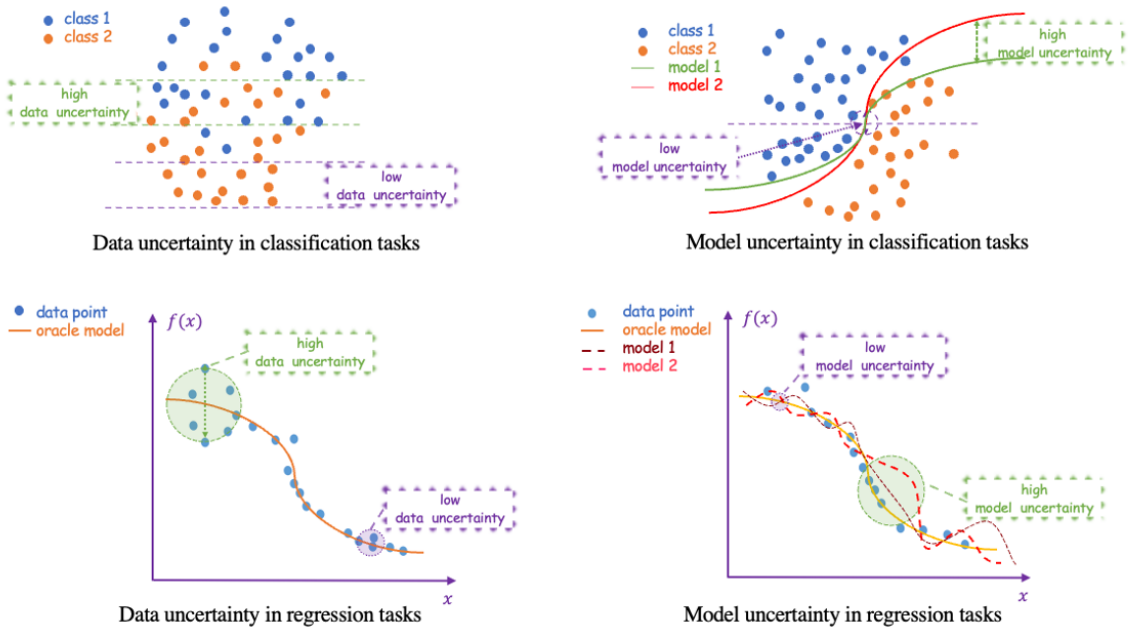
Model uncertainty, also known as epistemic uncertainty, arises from limitations in the model itself. This can be due to an incomplete sample, errors in the training procedure, or an inadequate model structure. In Bayesian neural networks, model parameters, denoted by θ are treated as distributions rather than fixed point estimates. This allows the use of the predictive distribution (4) to estimate model uncertainty. In regression tasks, consider $y^{(m)}(x) = \mu_{\theta^{(m)}}(x)$ as the predictive mean of the outcome for input x, given a sampled set of parameters $\theta^{(m)}$. The empirical mean of these predictions over M samples at input x is denoted by $\mu(y|x)$, calculated as: $\mu(y|x) = \frac{1}{M} \sum_{m=1}^M y^{(m)}(x)$. The model

uncertainty at input x is then estimated by the standard deviation across the mean values from different sampled sets of model parameters. This is expressed as:

$$\sigma_{m,reg}(x) = \left(\frac{1}{M} \sum_{m=1}^{M} \left(y^{(m)}(x) - \mu(y|x) \right)^2 \right)^{1/2}. \tag{A.4.3}$$

The differences between data and model uncertainties are illustrated in Figure A.4.1.

Figure A.4.1 Model Uncertainty and Data Uncertainty



Note. This figure illustrates the distinction between data uncertainty (aleatoric) and model uncertainty (epistemic) in both classification and regression tasks. Data uncertainty is intrinsic to the data itself and is irremovable. In contrast, model uncertainty stems from imperfections in the model, such as a lack of sufficient training data or an inadequate model structure, and can be mitigated through enhancements to the algorithm or the training process.

Appendix A.5: User Interface and Evaluation Procedures

In a meeting, one of our coauthors briefs the human experts that they will be collaborating with an algorithm and that they need to select B molecules from the algorithm's recommendations. We explain that our model is a Bayesian neural network and briefly outline its fundamentals. A detailed example dialogue between us and one human expert is provided in Section A.5.1. Section A.5.2 presents the user interface (UI) we provide experts to select molecules. Section A.5.3 provides technical details of the other human-in-the-loop baselines evaluated in this work.

A.5.1. Dialogue

In this section, we provide a detailed dialogue of how we communicated the key ideas of our design to the expert team.

Author

Thank you for accepting our invitation to participate in the experiment. The goal of this experiment is to identify molecules that could serve as promising drug candidates, meeting multiple property requirements. You will be collaborating with an algorithm developed by our team to identify the most promising candidates. In each round, the algorithm will present you with two lists of molecules: one based on the model's predicted likelihood of success, and another based on the model's highest uncertainty. The first list focuses on molecules that the model predicts are likely to perform well, while the second highlights molecules where the model is uncertain. For each group, we will recommend you 50 molecules, and you can decide the final 50 molecules out of the 100 provided by the models. Additionally, we will provide you with scores that represent the model's reasoning behind these recommendations, to assist you in making more informed decisions.

Expert

Does the model adjust its predictions as I provide feedback on the molecules I select?

Author

Yes. As we provide more data, the model learns and improves based on the results of the experiments. When you make your final selections, the model uses this new data to refine its predictions for future rounds.

Expert

If I choose molecules with high uncertainty, does that mean I'm taking more risks? How do I balance that with the more predictable molecules?

Author

Yes, there is some risk when selecting from the high-uncertainty list. However, the goal of including these molecules is to explore areas where the model lacks sufficient data. By testing uncertain molecules, you help the model improve its predictions for future rounds, potentially leading to better results and more promising candidates. You are free to override the model's suggestions based on your expertise.

Expert

If, based on my assessment, I am unable to identify a sufficient number of molecules for the final selection set, how should I proceed?

Author

This is an important consideration. You may manually select promising candidates using the **`Select`** feature in the provided interface. For consistent and optimal resource allocation, we recommend using the **`Select Without Satisfaction`** option for those you think are not promising enough, which will be the substitutes to fill your final set of molecules according to screening criteria. In rounds allowing manual additions, you can add extra candidates based on your expertise. If a selected molecule isn't in the dataset, you'll be shown 10 similar molecules from the ESM2 embedding space to choose from.

Expert

OK, got it. Could you provide more details about the model?

Author

The model we are using is a Bayesian neural network (BNN) that predicts the likelihood of molecules being promising candidates based on their various properties. It is continuously trained on experimental data, which is updated with each round as you make selections. To represent the molecular properties effectively, we use ESM2 before passing the data into the BNN.

Expert

Got it. Thank you.

Author

You're welcome! We really appreciate your help with this experiment.

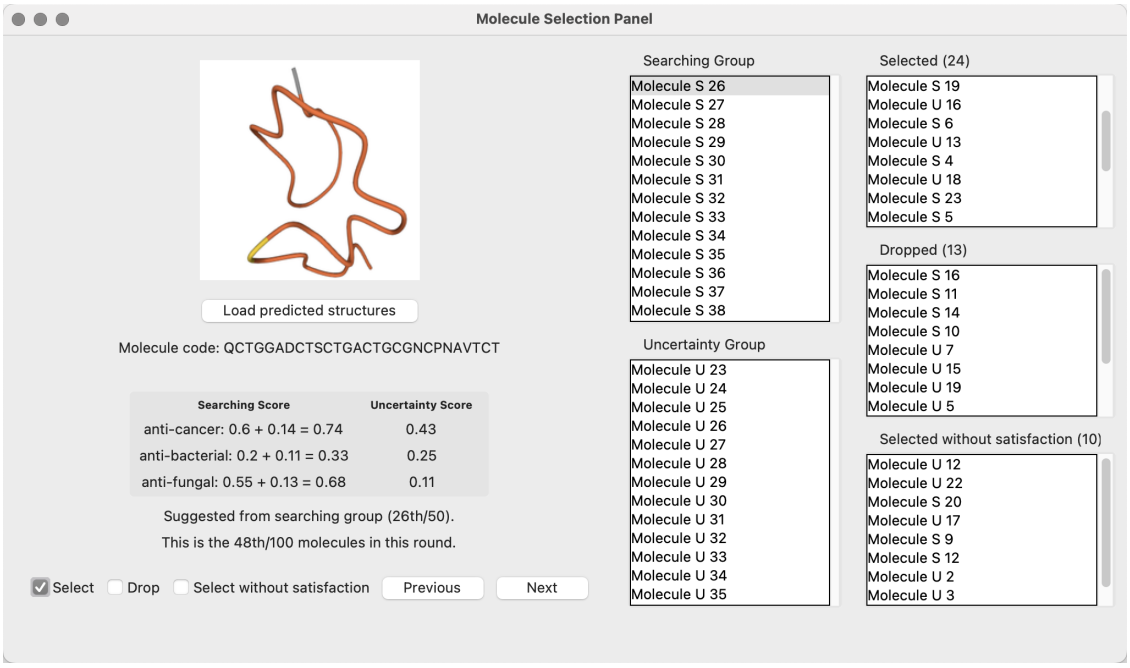
A.5.2. UI

The following Figure A.5.1 is the UI panel we provide for the experts to select molecules. On the left panel, a 3D structure and score table provide key details for the current molecule. Human experts evaluate each molecular candidate based on their domain knowledge and provided scores. For each molecule, they can assign a status by clicking Select, Drop, or Select without satisfaction, and then proceed to the next or previous entry. Labels beneath the scores indicate the molecule's origin (e.g., "Suggested from searching group") and overall progress (e.g., "48th/100 molecules"). On the right, five listboxes present unprocessed and processed molecules. The two left boxes—Searching Group and Uncertainty Group—contain unclassified candidates; clicking any entry loads its structure and scores on the left. The rightmost three—Selected, Dropped, and Selected without satisfaction—show molecules already assigned to each category, with real-time counts in parentheses. Selecting a molecule from these "buckets" reloads it on the left for re-decision, enabling flexible review and reclassification. At the end of each round, the human expert (and the algorithm) can see the results of their selected molecules, indicating whether each molecule is the target.

A.5.3. Other Human-in-the-Loop Baselines

For the benchmark (MPO-HITL) based on Sundin et al. (2022), we adapt their multi-parameter optimization methodology to round-by-round sequential experimentation with human experts, as follows. In the initialization round ($r = 0$), we establish an initial multi-parameter optimization (MPO) scoring function using parameter estimates from human experts $\theta_0 = \{(\text{HIGH}_{0,k}, \text{LOW}_{0,k})\}_{k=1}^{K}$ for K molecular properties, where each property is transformed using a double sigmoid function with unknown boundary parameters. Using this initial scoring function, we train a similar REINVENT-type model (Blaschke et al. (2020)) with reinforcement learning approach for peptides, which is also adopted in Sundin et al. (2022). With these initial MPO scores, we train the model for 100 epochs to select an initial batch of peptides based on similarity between generated molecules and the existing molecules in the molecule

Figure A.5.1 Molecule selection panel



pool that forms the unlabeled molecule pool $\mathcal{U}$. For each subsequent round $r \geqslant 1$, we employ an active learning with uncertainty acquisition function to select 50 molecules from $\mathcal{U}$ for expert evaluation, where the human provides binary feedback $y \in \{0, 1\}$ indicating molecular suitability. We model this feedback using the observation model $y|x \sim \text{Bernoulli}(S_\theta(x))$, where $S_\theta(x) = \prod_{k=1}^{K} \phi_k(c_k(x), \theta_k)^{w_k}$ represents the composite scoring function. Through Bayesian inference, we update the posterior distribution $p(\theta|\mathcal{D}_r)$ over the desirability function parameters using the accumulated feedback from all previous rounds, compute point estimates $\hat{\theta}_r$ from the posterior expectation, and use the adapted scoring function $S_{\hat{\theta}_r}(x)$ to train a REINVENT-type model for 50 epochs to select a new molecule pool for the next round. This iterative process continues for R rounds, where each round progressively refines the MPO objective through 50 molecular evaluations, creating a sequential learning framework that better captures expert preferences through accumulated binary feedback.

For the benchmark (RNN-HITL) adopted from Nahal et al. (2024) in a round-to-round sequential manner, we adapt their methodology for 50 rounds of experimentation with 50 molecules evaluated per round as follows. In the initialization phase ($r = 0$), like Nahal et al. (2024), we train an initial Random Forest property predictor f_{θ_0} on a warm-up dataset $\mathcal{D}_0 = \{(x_i, y_i)\}_{i=1}^{N_0}$, where $y_i = f^*(x_i)$ represents oracle ground truth values for the target property. We initialize a pre-trained REINVENT RNN generator p_{ψ_0} with weights ψ_0 and establish a scoring function

$$s(x) = \sum_{k=1}^{K} w_k \sigma_k(f_{\theta_k}(x)) \tag{A.5.1}$$

incorporating the trained predictor with appropriate transformation functions σ_k.

For each subsequent round $r \in \{1, 2, \ldots, 50\}$, we execute a sequential procedure beginning with the REINVENT loop in Nahal et al. (2024), where we optimize the generator for $N_{\text{steps}} = 250$ iterations using policy gradient reinforcement learning with loss

$$J(\mathcal{P}) = \frac{1}{P} \sum_{p=1}^{P} \left[\log p_{\psi_0}(x_p) - \beta s(x_p) - \log p_{\psi}(x_p)\right]^2 \tag{A.5.2}$$

to select a candidate pool $\mathcal{U}_r = \{x_m\}_{m=1}^{M}$ based on embedding similarity of the generated molecules and those of high-scoring peptides stored in memory. Subsequently, the active learning loop applies acquisition strategies based on the Expected Predictive Information Gain (EPIG) to select exactly 50 molecules $\mathcal{S}_r = \{x_t\}_{t=1}^{50}$ from $\mathcal{U}_r$, where the Expected Predictive Information Gain criterion maximizes

$$\text{EPIG}(x) = \mathbb{E}_{p^*(x^*)}\left[\text{KL}\left[p(y, y^*|x, x^*)||p(y|x)p(y^*|x^*)\right]\right] \tag{A.5.3}$$

over the top-k highest predicted molecules, with target input distribution

$$p^*(x^*) = \frac{\sigma(f_\theta(x^*))}{\sum_{x \in \mathcal{U}_r^{\text{top-}k}} \sigma(f_\theta(x))} \text{ if } x^* \in \mathcal{U}_r^{\text{top-}k}. \tag{A.5.4}$$

Following molecular selection in Nahal et al. (2024), we query the human expert's estimation of $f_{\text{human}}(x_t) = f^*(x_t) + \epsilon$ with assumption that $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ to obtain both binary labels h_t and confidence scores $u_t \in [0, 1]$ for each selected molecule, capturing the uncertainty inherent in human assessment. The Random Forest predictor is then retrained using a weighted loss function

$$\theta_r = \arg\min_\theta \frac{1}{N_0} \sum_{i=1}^{N_0} \ell(f_\theta(x_i), y_i) + \frac{1}{50} \sum_{t=1}^{50} u_t \ell(f_\theta(x_t), h_t) \tag{A.5.5}$$

where confidence scores u_t modulate the influence of new training instances, allowing the model to discount unreliable expert feedback. Performance tracking involves computing the Mean Absolute Error

$$\text{MAE}_r = \frac{1}{|\mathcal{M}_r|} \sum_{x \in \mathcal{M}_r} |f_{\theta_r}(x) - f^*(x)| \tag{A.5.6}$$

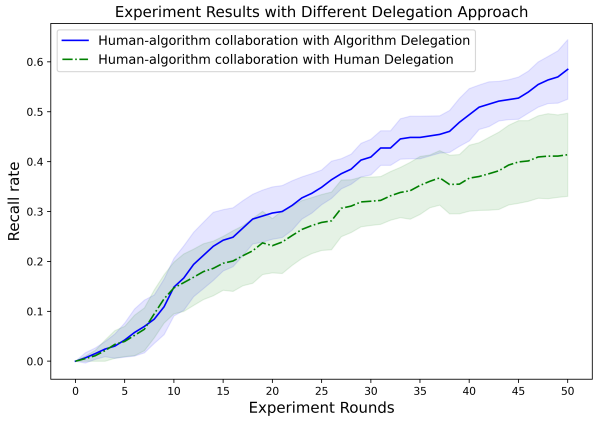
between predicted and oracle scores for generated molecules, along with diversity metrics including Internal Diversity $\text{IntDiv}(\mathcal{G}) = 1 - \sqrt[p]{\frac{1}{|\mathcal{G}|} \sum_{m_1, m_2 \in \mathcal{G}} \text{Tanimoto}(m_1, m_2)^p}$, drug-likeness scores (QED), and synthetic accessibility measures. This iterative process continues for exactly 50 rounds ($R = 50$), where each round systematically evaluates 50 molecules through expert feedback, progressively refining the property predictor's generalization capability and improving the alignment between predicted molecular properties and ground truth oracle evaluations, ultimately generating 2,500 total molecular evaluations across the complete experimental sequence.

Appendix A.6: Delegation of Work

In this section, we assess the efficacy of various work delegation strategies. The collaboration between humans and algorithms features two principal delegation methods: *from humans to algorithms* and *from algorithms to humans* (Fügener et al. 2022). Our human-in-the-loop framework exemplifies algorithm-to-human delegation, where algorithms perform preliminary screenings and then pass the responsibility for final decision-making to the humans. Conversely, in traditional algorithm-assisted drug discovery approaches, delegation flows from humans to algorithms; humans select molecules for screening and depend on algorithmic predictions to identify the most promising candidates. While this method benefits from initial human insight, it might limit the search space too early in the process, potentially overlooking promising candidates that an algorithm might have identified in a broader scan.

We conduct an additional experiment where humans initially select 100 molecules, and then the algorithms are tasked with choosing 50 out of these 100 molecules based on high prediction values. The results of this experiment are depicted in Figure A.6.1. Our findings suggest that, within our context, delegation from algorithms to humans improves performance by more than 50% compared to delegation from humans to algorithms. This substantial improvement is largely attributable to the extensive search space inherent in drug discovery tasks, and very limited human knowledge. Algorithms excel in initial screenings, while human experts adeptly integrate their private and domain-specific knowledge with algorithmic recommendations, leading to more effective decision-making.

Figure A.6.1 Delegation of Work



Note. This figure shows the cumulative recall rate for various work delegation strategies. Our findings suggest that delegation from algorithms to humans improves performance by more than 50% compared to delegation from humans to algorithms.

Appendix A.7: Allowing Humans to Add Molecules

We further examine the behavior of experts with different levels of knowledge by analyzing their tendency to add molecules beyond the recommendations and the "accuracy" of the molecules they add. In Table A.7.1, the first row reports the average rejection rate (i.e., the rate at which experts override the recommended molecules) along with the corresponding standard deviation for three groups of experts. Although the overall tendency to add molecules does not differ significantly among the groups, the high-knowledge experts show a slightly higher rejection rate. The second row reports the average "accuracy" of the recommended molecules, measured as the cosine similarity between the embeddings of the human-recommended molecules and the closest target molecules, along with the corresponding standard deviations. These results indicate that the high-knowledge experts achieve significantly higher accuracy in their recommendations.

Table A.7.1 Behavior Patterns with Different Knowledge Level

	High	Medium	Basic
Rejection rate	**11.59%**	11.07%	10.58%
	(1.97%)	(1.85%)	(2.01%)
Accuracy	**0.712**	0.562	0.397
	(0.082)	(0.065)	(0.087)

Note. This table displays the average rejection rate over 50 rounds and the "accuracy" of the added molecules for experts with different levels of knowledge. The "accuracy" is measured as the cosine similarity between the embeddings of the human-recommended molecules and the closest target molecules.

Appendix A.8: Human Preference Alignment

Currently, the model learns from the labeled data selected by experts in each round. However, the selection vs. non-selection behavior (i.e., preferences) of human experts may offer valuable information, particularly during the early phase, where any additional useful information can accelerate the learning process. In this section, we adopt the Direct

Preference Optimization (DPO) framework (Rafailov et al. 2024), a method designed to train models based on human preference data. This approach has demonstrated success in large language models and enables us to integrate fuzzy information from human expert preferences.

In round t, we denote the recommended set as $C^{(t)} = \{\boldsymbol{x}_{\text{un},i}^{(t)}\}_{i=1}^{q} \cup \{\boldsymbol{x}_{\text{se},i}^{(t)}\}_{i=1}^{h}$, where $\boldsymbol{x}_{\text{un},i}^{(t)}$ and $\boldsymbol{x}_{\text{se},i}^{(t)}$ represent molecules recommended for exploration and exploitation, respectively, and the final selected subset as $S^{(t)} = \{\boldsymbol{x}_i^{(t)}\}_{i=1}^{B}$. A key aspect is that rejected molecules $C^{(t)} \backslash S^{(t)}$ may still provide information about human preferences. Specifically, molecules in $S^{(t)}$ are considered to be preferred by the expert, while those in $C^{(t)} \backslash S^{(t)}$ are deemed non-preferred. These two sets represent contrasting examples of what the expert favors versus rejects, which can be valuable for improving the model's understanding of human preferences.

We construct a preference dataset $P^{(t)}$ of size $|P^{(t)}| = |C^{(t)} \backslash S^{(t)}|$, which is set to 50 in our experiments. Following a methodology similar to that in Laugel et al. (2018), we generate this dataset by pairing each molecule from the discarded set, $C^{(t)} \backslash S^{(t)}$, with its closest counterpart in the selected set, $S^{(t)}$, based on cosine similarity. The dataset is formally defined as:

$$P^{(t)} = \left\{ (\boldsymbol{x}_s, \boldsymbol{x}_d) \mid \boldsymbol{x}_d \in C^{(t)} \backslash S^{(t)},\ \boldsymbol{x}_s = \arg\min_{\boldsymbol{x} \in S^{(t)}} \cos(\boldsymbol{x}, \boldsymbol{x}_d) \right\}$$

Each pair $(\boldsymbol{x}_s, \boldsymbol{x}_d) \in P^{(t)}$ represents an implicit preference, where the expert is presumed to have favored $\boldsymbol{x}_s$ over $\boldsymbol{x}_d$. This preference data are subsequently used within the DPO framework to align the model with the expert's implicit judgments, even in the absence of explicit feedback.

During iteration t, the training objective for a Bayesian neural network parameterized by θ, originally defined in Equation A.4.1, is augmented with an auxiliary DPO-inspired loss term $\mathcal{L}_{\text{DPO}}$ to incorporate human preference signals:

$$\theta^* = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \left[\frac{1}{2\sigma_\theta^2(\boldsymbol{x}_i)} \|y_i - \mu_\theta(\boldsymbol{x}_i)\|_2^2 + \frac{1}{2} \log \sigma_\theta^2(\boldsymbol{x}_i) \right] + \lambda_{\text{DPO}} \mathcal{L}_{\text{DPO}}(P^{(t)}).$$

The auxiliary term $\mathcal{L}_{\text{DPO}}$ operates on the preference dataset $P^{(t)}$, which is constructed from the selected and discarded sets, and is formulated as:

$$\mathcal{L}_{\text{DPO}}(P^{(t)}) = -\frac{1}{|P|} \sum_{j=1}^{P} \log \sigma \left(\beta \left[\mu_\theta \left(\boldsymbol{x}_{j,s} \right) - \mu_\theta \left(\boldsymbol{x}_{j,d} \right) \right] \right)$$

where $\sigma(\cdot)$ is the sigmoid function and $\mu_\theta(\cdot)$ denotes the predictive mean of the Bayesian neural network. The parameter $\beta > 0$ is a temperature (or scaling) factor that controls the sharpness of the preference comparison: larger values of β amplify differences between selected and discarded molecules, while smaller values make the model less sensitive to such differences. This DPO-inspired loss guides the model to align with human preferences by encouraging higher predicted scores for selected molecules $(x_{j,s})$ over discarded ones $(x_{j,d})$. We also normalize the outputs of all molecules recommended by the algorithm in each round so that $\mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_{\text{all}}^{(t)}} \left[\mu_\theta \left(\boldsymbol{x} \right) \right] = 0$ before training the subsequent model. This step, similar to the treatment in preference learning with DPO (Rafailov et al. 2024), helps reduce the variance of training by preventing systematic drift in the network's predictions.

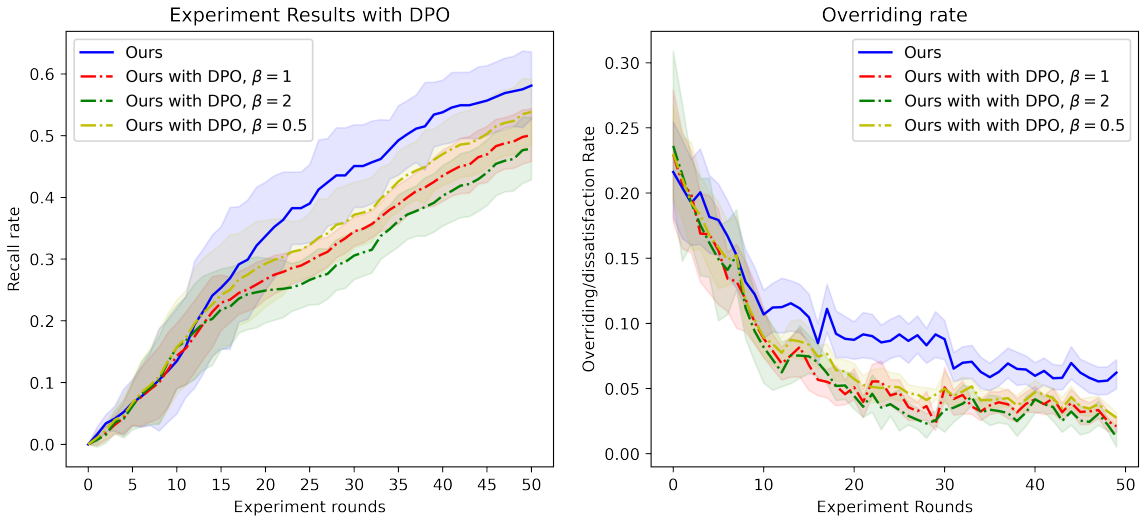
The left figure of Figure A.8.1 presents the experimental results with DPO for different parameters β. While incorporating noisy labels via DPO does not result in improved final search outcomes, several key observations emerge. First, DPO appears to accelerate the learning process in the early stages of training. This advantage arises because, during initial learning stages where observational data is scarce, DPO leverages human-provided preferences to infer and encode valuable (private) domain knowledge.

However, as the experiment progresses, particularly in the middle phase, a decline in search efficiency is observed. This decrease in performance may reflect a critical limitation: while human expertise provides initial guidance, over-reliance on expert preferences risks reinforcing human cognition bias. As the model increasingly prioritizes molecules aligned with human priors (which inherently reflect incomplete or constrained human knowledge), it may inadvertently reinforce suboptimal patterns rather than discovering novel and/or unexpected candidates. Consequently, early gains from expert alignment give way to exploration deficits in later stages, and the final discovery yield is lower than that of our original design.

In the right figure of Figure A.8.1, we also present the dissatisfaction rates from the experts. The blue line corresponds to our main design, and others are the dissatisfaction rate with DPO. The lower rejection rates of DPO suggest that the algorithm is indeed providing molecules that align with the experts' preferences.

We find that under different values of β, the final performance consistently shows some degree of degradation. Larger β values tend to yield greater performance gains in the early 5 to 10 rounds, but result in more pronounced degradation in the final round. This echoes our earlier statement that incorporating human preference at an early stage can effectively warm-start the model, while in later rounds the model becomes more susceptible to absorbing human biases, which reduces its long-term effectiveness.

Figure A.8.1 Training BNN with Direct Preference Optimization



Note. This figure presents the experimental results of training a Bayesian neural network with/without Direct Preference Optimization (DPO). All the confidence intervals reported correspond to $\pm 2\sigma$.

Appendix A.9: Similarity Between Two Batches

We also assess the similarity and overlap between the two groups by calculating the Jaccard similarity between the top-ranked molecules selected by r_{un} and r_{se} across experimental rounds. The similarity exhibits a downward trend, remaining between 5% and 10% overall. Three distinct phases appear as follows:

- **Initial phase** (Rounds 1–15): Higher similarity, with a mean of 7.3% ($\pm 2.4\%$)

- **Middle phase** (Rounds 16–40): A noticeable decline to 4.1% ($\pm 1.7\%$)
- **Final phase** (Rounds 41–50): A slight recovery to 5.5% ($\pm 1.5\%$)

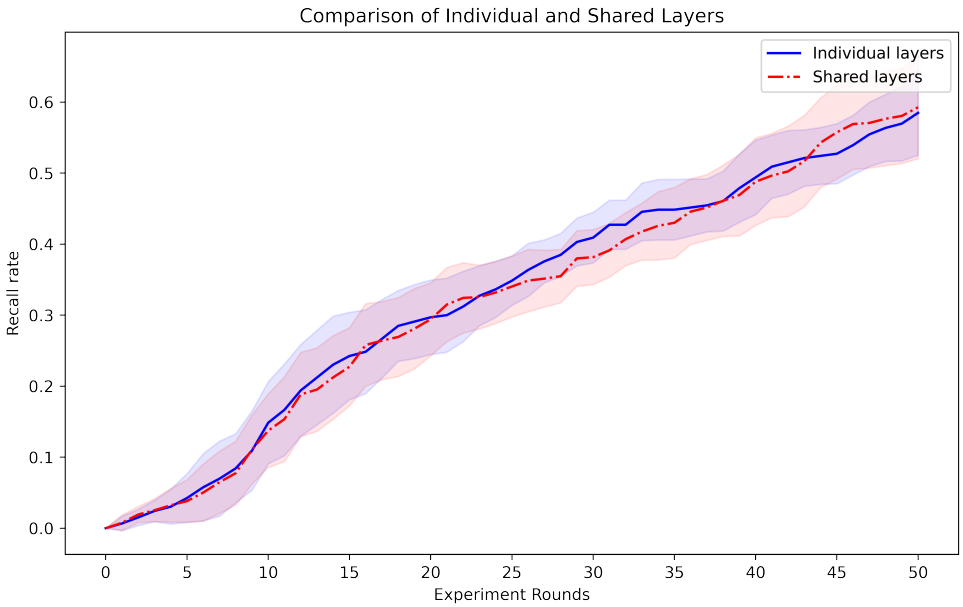
This trajectory indicates a generally high divergence between the two groups of suggested molecules.

Appendix A.10: Robustness Checks and Extensions

A.10.1. Shared Embedding

In the current framework, the properties are modeled independently. However, we recognize that some properties may be correlated, and learning them simultaneously could be another viable approach. In this section, we explore a scenario where each Bayesian neural network shares the first two layers. Our findings indicate that the overall performance remains unchanged. This can be explained by the fact that we use ESM2 as feature encoders before passing the features to the BNN. Therefore, the ESM2 model effectively serves as the shared layers, which already captures the necessary correlations between properties.

Figure A.10.1 Comparison with Shared Embeddings



Note. This figure illustrates the impact of sharing the first two layers in Bayesian neural networks to capture property correlations. The results show no significant performance change, likely because the ESM2 model, used as a feature encoder, already captures the necessary correlations between properties. Each experiment was repeated eight times with different participants, and the confidence intervals reported correspond to $\pm 2\sigma$.

A.10.2. Other Performance Metrics

To document the changes in precision throughout the experiments, we separate the recall and precision metrics for the candidate set C (identified by the algorithm) and the final set S (determined by human review).

Our experimental setting has a predetermined number of recommended molecules from both human and algorithm each round ($|C|$ = 100 and $|S|$ = 50 across all rounds). For example $|S|$ represents a fixed and realistic batch size determined by hardware capacity (e.g., high-throughput devices process exactly $|S|$ molecules in a single round). Underfilling a batch wastes unused capacity, and overfilling is impossible. Our setting is different from traditional classification settings, hence, we first re-define the precision and recall for our setting at each round t.

The precision for each set at round t is defined as follows:

- For the human-selected set S_t:

$$\text{precision}_t(S_t) = \frac{\text{True Positives}_t}{\text{True Positives}_t + \text{False Positives}_t} = \frac{\text{Number of target molecules in } S_t}{|S_t|}$$

- For the algorithm-recommended set C_t:

$$\text{precision}_t(C_t) = \frac{\text{True Positives}_t}{\text{True Positives}_t + \text{False Positives}_t} = \frac{\text{Number of target molecules in } C_t}{|C_t|}$$

The recall for each set at round t is defined as:

- For the human-selected set S_t:

$$\text{recall}_t(S_t) = \frac{\text{True Positives}_t}{\text{True Positives}_t + \text{False Negatives}_t} = \frac{\text{Number of target molecules in } S_t}{\text{Remaining target molecules in } \mathcal{D}^{(t)}_{\text{unlabeled}}}$$

- For the algorithm-recommended set C_t:

$$\text{recall}_t(C_t) = \frac{\text{True Positives}_t}{\text{True Positives}_t + \text{False Negatives}_t} = \frac{\text{Number of target molecules in } C_t}{\text{Remaining target molecules in } \mathcal{D}^{(t)}_{\text{unlabeled}}}$$

We report the mean precision and recall for every 10-round bucket in Table A.10.1. For the comparison between the C and S sets, note that the cardinality is $|C| = 100 = 2|S|$. This implies that when comparing the two, the precision for C_t will generally be lower due to the doubled denominator—though it should be more than half, since $S \subset C$ and $|C| = 2|S|$. The similar reasoning also applies to recall. Since $S \subset C$ and the denominators are identical for a given round, the recall of the algorithmic set C should be higher.

From the table, we observe that the human-selected set S exhibits relatively high precision in the early rounds (first 20 rounds), but this drops in later rounds. This trend is intuitive because, as the experiment progresses, the easily identified target molecules are already selected; the remaining molecules in later stages are less familiar to human experts' existing knowledge, leading to a decrease in precision. Meanwhile, human recall remains relatively stable.

Furthermore, a notable observation is that the "extra target" count, measured by

$$\text{precision}_{C_t} \times |C_t| - \text{precision}_{S_t} \times |S_t|,$$

which represents the molecules recommended by the algorithm but not selected by humans, increases over time. This indicates that the algorithm is improving and providing more precise recommendations beyond current human knowledge, as also evidenced by the increasing recall of the C set compared to the relatively stable recall of the S set. This also highlights the limitations of human knowledge and the need for a structured framework to enable reciprocal learning between humans and algorithms. AI must not only learn from existing human expertise but also assist humans in navigating scientific uncertainty, identifying promising new search directions, and challenging existing assumptions.

Table A.10.1 Performance Metrics over Rounds

Rounds	$1 \sim 10$		$11 \sim 20$		$21 \sim 30$		$31 \sim 40$		$41 \sim 50$	
	C	S	C	S	C	S	C	S	C	S
Precision	2.19%	4.16%	2.42%	4.52%	2.31%	3.62%	1.65%	2.32%	1.72%	2.69%
Recall	1.60%	1.52%	1.98%	1.85%	2.20%	1.72%	2.33%	1.64%	2.49%	1.95%

A.10.3. Dynamic Adjustment

In this section, we allow dynamic adjustment of the number of molecules in the two recommendation lists provided by the algorithm, namely q (high search score) and h (high uncertainty), across rounds. In the previous experiments, we set both lists to 50 molecules each. However, it is important to examine the impact of recommending more uncertain molecules at the beginning to quickly explore the search space and train a better model early on. Toward the end of the planning horizon, exploration becomes less valuable as many experimental data are already acquired, and more effort should be dedicated to identifying high-quality molecules.

To this end, we conduct an additional set of experiments to investigate both algorithmic adjustments and performance-informed expert adjustment, of the ratio between q (high search score) and the total recommendation number, denoted as $\text{search}_t \triangleq \frac{q_t}{h_t+q_t}$. We set the total recommendation number to 100, meaning $q_t + h_t = 100$ for all rounds t.

- **Momentum-based Adjustment**: Updates follow a momentum-based rule similar to Lee et al. (2020) and Nguyen and Garnett (2023):

$$\text{search}_{t+1} = \text{clip}\big(0.2,\, 0.8,\, \text{search}_t + \alpha \cdot \big(\text{prec}_t - \overline{\text{prec}}_{1:t-1}\big)\big),$$

  where prec_t denotes the algorithm's current precision and $\overline{\text{prec}}_{1:t-1}$ represents the historical average precision up to round $t-1$. Positive deviations, where $\text{prec}_t > \overline{\text{prec}}_{1:t-1}$, lead to an increase in the algorithm's search budget, favoring exploration. This clipping ensures that the ratio does not become too extreme (e.g., $\text{search}_t \to 0$ or $\text{search}_t \to 1$), which helps to preserve the balance and stability of the search process.

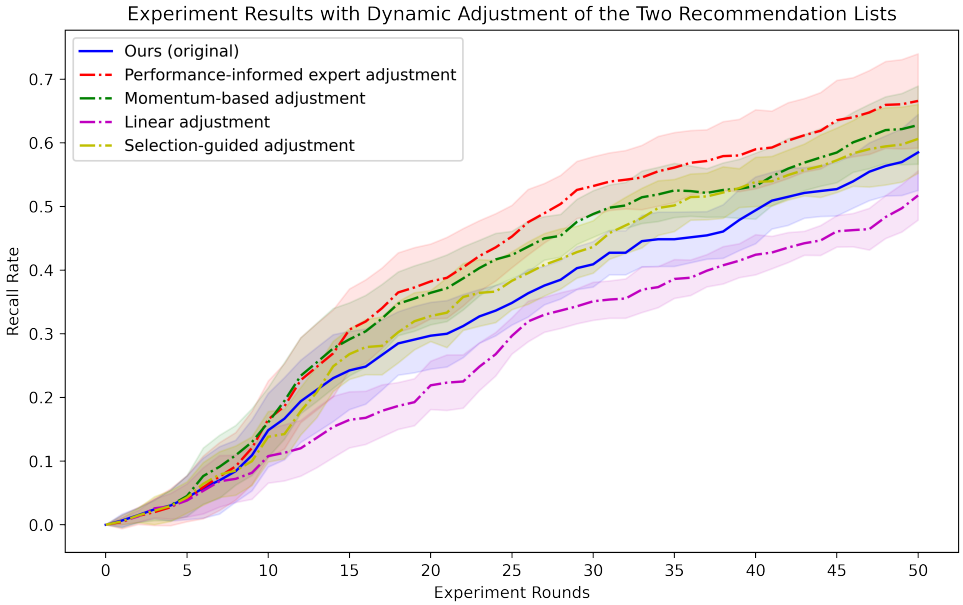
- **Selection-guided Adjustment**: In this strategy, among the molecules selected in the previous round, we use n_t^{se} and n_t^{un} to represent the number of molecules selected from either the search group or the uncertainty group, respectively. Then the search ratio for the next round is adjusted according to the following formula: $\text{search}_{t+1} = \text{clip}\left(0.2,\, 0.8,\, \frac{n_t^{se}}{n_t^{se}+n_t^{un}}\right)$. The resulting ratio is clipped within the range $[0.2, 0.8]$ to maintain stability. The rationale for this adjustment is that expert selection may capture additional signals differentiating the qualities of the two groups.

- **Linear Adjustment**: This strategy gradually increases the collaboration ratio similar to Negoescu et al. (2011) according to the following formula: $\text{search}_t = 0.2 + \frac{0.6(t-1)}{T-1}$, where T is the total number of rounds. This linear schedule ensures a steady adjustment of the collaboration ratio but does not adapt based on the algorithm's performance nor the expert's selections.

- **Performance-informed Expert Adjustment**: In this strategy, the human expert determines the search ratio based on the performance from the experiments in each round.

The results are presented in Figure A.10.2, revealing several insights into the dynamic adjustment strategies. First, dynamically adjusting the search ratio leads to marginal improvements in search efficiency during the early stages, although this comes at the cost of increased variability in the procedure. Despite this, statistical evaluation shows no significant improvement in the final search outcomes. Second, performance gains seem to be partly driven by the momentum effect in model accuracy, especially in initial rounds, suggesting that stability in the early stages facilitates incremental optimization. Finally, while Selection-guided Adjustment introduces more variability in the interim results, they ultimately lead to superior final performance. This increased variability can be attributed to differences in human expertise and personal preferences. Nevertheless, it offers a way of incorporating human knowledge and additional context into the human-AI collaboration.

Figure A.10.2 Dynamics of Human-Algorithm Collaboration through Adaptive search$_t$ Adjustment



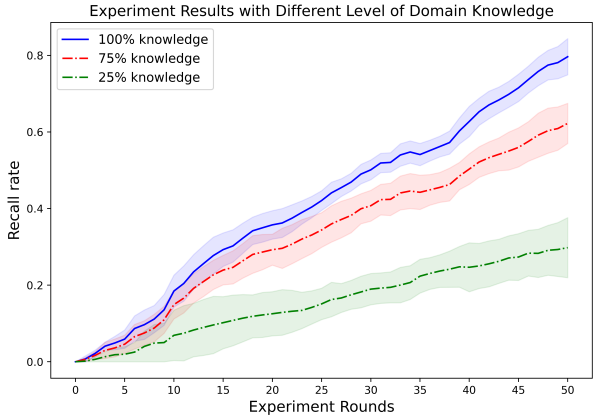
Note. This figure illustrates the impact of dynamically adjusting the ratio of high-search-score and high-uncertainty recommendations. The results show that while early-stage adjustments improve search efficiency, they introduce variability without significantly improving final outcomes. Expert-guided adjustments yield the best final performance, highlighting the value of expert decision-making. Each experiment was repeated eight times with different participants, ensuring robustness, and the confidence intervals reported correspond to $\pm 2\sigma$ capturing approximately 95% of the variability in the results.

A.10.4. Impact of Human Knowledge Levels

We hypothesize that the level of human knowledge may also have an impact on the final performance. In this section, we analyze the performance of our framework as the algorithm interacts with human experts who possess varying levels of domain knowledge. We evaluate the performance in scenarios where the expert has high-level, medium-level, and low-level domain knowledge about therapeutic peptides.

Since evaluating the knowledge level in practice is challenging, we simulate the results. Specifically, we enlist a volunteer with minimal domain knowledge to act as a human expert, thereby minimizing the impact of prior knowledge on the experimental results. During the experiments, for each peptide molecule, we disclosed the true labels to the participant with probabilities of $p = 25\%, 75\%, 100\%$, without informing them of the actual percentage. This approach follows the work of Fügener et al. (2022), which suggests that humans might not be able to accurately judge their abilities for difficult tasks. We repeated Task 1 from §6.1 20 times for each level of knowledge, maintaining all other parameters identical to those in §6.2. We then compared the recall rate in experiments with varying levels of expert domain knowledge and displayed the results in Figure A.10.3.

Figure A.10.3 Impact of Human Domain Knowledge



Note. The figure shows the cumulative recall rate for each round. Greater human expertise results in a higher recall rate and a reduced standard deviation.

The findings from this figure reveal that enhanced knowledge contributes to superior performance, characterized by a higher recall rate and a reduced standard deviation. Specifically, when 100% knowledge is disclosed, we can achieve a 0.8 recall rate by the end of 50 rounds. We note that this recall rate is not 100% for two reasons. First, the volunteer is unaware of the actual percentage of knowledge they receive. Second, our framework operates under a procedure where algorithms recommend two lists of molecules, and the volunteer makes the final decision from these two lists. Their ability is limited in that they cannot exclusively rely on this given knowledge but must collaborate with the algorithm in adhering to our framework.

When 75% and 25% knowledge levels are disclosed, the final recall rates are around 0.6 and 0.3, respectively. It is interesting to note that with 25% knowledge, the final recall rate of 0.3 is nearly the same as when the algorithm alone makes the decision. The phenomenon of minimal or no improvement in the collaboration between human and algorithm has also been observed (Lebovitz et al. 2021, Balakrishnan et al. 2022). It suggests that insufficient domain knowledge might render human contributions ineffective or even detrimental. This aligns with prior research indicating that under certain conditions, uninformed human intervention can introduce noise or bias and decrease the reliability of algorithmic processes (Shin 2024, Fügener et al. 2021, 2022). This observation emphasizes the need for appropriate training of humans to maximize the potential of human-AI collaboration. It suggests that AI systems should be designed with interfaces and feedback mechanisms that are intelligible to users with varying levels of expertise, facilitating better integration of human judgment.

Appendix A.11: Cold Start Evaluation

In this section, we evaluate the impact of human expertise in managing the cold-start scenario by comparing the final search outcomes using the same model with and without a pretraining warm-up phase. The warm-up procedure involves pretraining the model on samples from the left-out set of molecules (as detailed in §6.1). To isolate the warm-up effect, we remove human input by applying a fixed rule to select the final set of molecules, choosing a total of $B = 50$ molecules per round. In the absence of human involvement, we assess two strategies for combining the groups:

- **Evenly**: The top 25 molecules selected from each of the exploit (high search score) and explore (high uncertainty score) groups.
- **Momentum**: A momentum-based selection method as introduced in Online Appendix §A.10.3.

To evaluate the impact of the balance between positive and negative samples in the initial dataset on the final outcome, we construct the warm-up training datasets using different combinations of positive and negative samples. The second row of Table A.11.1 shows the number of positive and negative samples included in each warm-up phase. For example, 41/41 indicates that 41 positive and 41 negative samples are included in the warm-up phase.

Table A.11.1 Final Hit Rates under Cold Start Evaluation

	No Human Involvement							Human-in-the-loop
	No Warm-Up	Target-based			Property-based			Ours
		41/41	41/82	41/205	50/50	50/100	100/100	
Evenly	1.39% (0.13%)	2.28% (0.08%)	2.45% (0.07%)	2.37% (0.09%)	1.89% (0.11%)	2.05% (0.10%)	2.19% (0.10%)	**2.97%** (0.13%)
Momentum	1.35% (0.14%)	2.35% (0.09%)	2.59% (0.10%)	2.51% (0.13%)	2.07% (0.13%)	2.23% (0.12%)	2.44% (0.11%)	**3.09%** (0.16%)

We also note that for a molecule to be considered a target molecule, it must meet all required properties. However, in practice, very few molecules satisfy all of these criteria. In our approach, we model each property individually, allowing the model to be trained effectively even without using only target molecules. Instead, we can include molecules that meet a subset of the required properties, which increases the number of positive samples for training. To explore this, we test two approaches: 1) *Target-based*: using only the target molecules; 2) *Property-based*: using molecules that meet certain properties as positives, without requiring them to be full target molecules.

Experiments are conducted on Task 1 over 50 rounds, with 50 molecules selected in each round for laboratory experiments. To evaluate the effectiveness of the warm-up strategy, we also provide baseline results obtained without the warm-up phase, as well as results from our original approach where no warm-up is performed, and human experts make the final decision. Table A.11.1 summarizes the final hit rates under various experimental conditions.

The results in the table show that the No Warm-Up approach yields the lowest hit rates, ranging from 1.39% for the Evenly strategy to 1.35% for the Momentum strategy. Incorporating the Target-based warm-up strategy leads to the most significant improvements, with the highest hit rate achieved using the Momentum strategy. The results are similar for the Property-based warm-up approach, which also enhances performance, with hit rates ranging from 1.89% to

2.44% across different configurations. However, this approach is slightly less effective than the Target-based strategy because the Target-based approach provides true targets for the model to learn. In contrast, our proposed approach, which incorporates human expert involvement, consistently outperforms all other methods. These findings suggest that the incorporation of human expertise *in-the-loop* significantly boosts performance across various experimental conditions. This is particularly valuable because, in real-world *novel* drug discovery (e.g., for a COVID-19 vaccine), there is *no* labeled data for a target-based warm-up pre-training.

Appendix A.12: Additional Literature Review

In this section, we first provide a detailed comparison between active learning and our problem setting. We then discuss additional related literature. We group the literature by topic, aiming to clarify similarities and gaps.

A.12.1. Active Learning

Our problem is different from active learning (AL) in the following three aspects: 1) Human Oracle vs. External Oracle 2) Objective Function 3) Who is in Control for Selecting Items to Query. We summarize these differences in Table A.12.1.

Table A.12.1 Differences between our Problem Formulation and Active Learning

	Human Oracle vs. External Oracle	Objective Function	Who is in Control for Selecting Items
Active Learning	Human oracle	Prediction accuracy	Model control
Ours	External oracle	Reward (number of discoveries) maximization	Shared control

Human Oracle vs. External Oracle

In our problem formulation, human does not assign labels in the traditional active learning sense. In drug discovery, labels are not provided by humans but generated through external lab tests. In comparison, in classic AL, the AL algorithm selects unlabeled instances and an **oracle** provides labels on queried items. In AL, if "expert review" is intended, it typically refers to a human serving as the oracle (i.e., providing those labels). However, this is not the role humans play in our setting, and ground-truth labels in our setting come exclusively from an external laboratory oracle. Even domain experts are not capable of annotating "hit vs. non-hit" by inspection. In our setting, humans and the algorithm jointly decide which molecules to query.

Objective Function

Our setting and objective are fundamentally different from classic active learning. Classic active learning is designed to improve predictive accuracy under a labeling budget, i.e., it queries as few labels as possible while maintaining generalization performance. In other words, the utility being optimized is model accuracy, not immediate task reward.

In contrast, our objective function is to maximize the successful discoveries under a fixed budget, which aligns more with multi-armed bandits and reinforcement learning literature (maximizing cumulative reward / minimizing regret) than with AL's generalization error.

The following sentences explain why AL's uncertainty sampling alone is misaligned with our goal. In AL, querying ambiguous points near the current decision boundary is beneficial because the goal is to reduce model uncertainty and prediction error. A queried example (even if turns out negative) does not incur immediate task-level loss (i.e., no regret for "missing a hit"). Its value is informational for future accuracy gains. Our drug discovery setting differs: each wet-lab query **does** carry task-level consequences since every non-hit query consumes scarce assay budget and directly lowers the number of discoveries.

Who is in Control for Selecting Items to Query

The human-in-the-loop literature distinguishes control regimes by who chooses the queries: (i) Active Learning (model control)—the AL algorithm selects items and the human acts as an oracle to label them; (ii) Interactive ML (shared control)—the system proposes and experts co-select the final batch and (iii) Machine Teaching (human control)—experts prescribe what to teach/query and how. Bernard et al. (2017) also note that one of the intrinsic characteristics of AL strategies is the model-driven way to identify meaningful instances for labeling. A drawback of this principle is that users, with their ability to identify patterns very fast, have no influence on the candidate selection.

In contrast, our workflow falls into the shared control regime: the model proposes recommendations and human experts select the final batch after applying domain knowledge and priorities (Mosqueira-Rey et al. 2023).

A.12.2. Additional Literature Review

Additional literature review by topic is as follows.

Selective Sampling / Active Learning

- In Orabona et al. (2011), the authors propose a framework on "selective sampling" that describes a learner that may observe the true label only by querying for it. The goal in selective sampling is to trade off predictive accuracy against number of queries. There is no human involvement nor human override mechanism.
- In Sekhari et al. (2023), the learner predicts an action at each step, and then decides whether to query an expert for feedback. If it queries, it receives a noisy label/action from the expert and updates its model. If it does not query, it gets no feedback. Goal: selective queries to noisy experts to reduce labeling.
- Attenberg and Provost (2010) introduce guided learning as an alternative to standard AL in highly imbalanced settings (e.g. finding rare objectionable content web pages for ad placement). The goal is to build text classifiers under extreme class imbalance, which has higher generalization performance (AUC). Humans are tasked to find examples by class and also serve as the oracle to provide labels.
- Cakmak and Thomaz (2011) formalize mixed-initiative active learning (MIAL) in the context of a human teacher and a robot learner. It is purely human-driven training (where a human teacher chooses all training examples).
- Suh et al. (2016) contrast three ways to train a classifier with interactive machine learning: computer-initiated (active learning, AL), human-initiated (teacher chooses items), and mixed-initiative (both can choose). The goal is classification performance. The human is the oracle providing labels and (in human-initiated or mixed-initiative) choosing items to label. The optimal teacher explicitly assumes perfect knowledge of the true concept f^*.

— In Knowledge-Driven Active Learning (KAL) framework (Ciravegna et al. 2023), the domain experts encode common-sense rules or domain knowledge as logical constraints. For example, in a medical task a rule might be "if symptom A and B are present, disease X is unlikely." These rules are not 100% reliable, but they express general domain regularities. KAL then uses the model to find instances that violate these expert-provided rules. Hence, the model fully controls query selection via rule-violation and human serves as the oracle to provide labels for selected items. The objective is to improve model performance with fewer labels.

Batch AL

— In Ghai et al. (2021), for each instance the model wants to learn from, it shows a local explanation of its current prediction to the annotator, who then supplies feedback (label and/or guidance) informed by that explanation. Ghai et al. (2021) argue that the interfaces to query human input are minimal in current AL settings, and there is surprisingly little work that studied how people interact with AL algorithms.

— In DUALIST (Settles 2011), the model chooses which instances to surface (top-D most uncertain documents each round), and the human then selects within that slate—they click a class to label an item or hit "X" to skip/remove a proposed document. Human can click "X" when a document is too ambiguous to label confidently—the UI explicitly says "In cases of extreme ambiguity, users may ignore a document by clicking the 'X' to remove it from the pool. Human is the annotator, and the rationale is to avoid providing ambiguous labels.

— The framework in Bernard et al. (2017) is explicitly designed to study batch-labeling efficiency, and their AL-multi condition lets the human pick several AL-suggested items and apply a single class in one action—e.g., select a bunch of items that look like "1" and label them all at once. They chose this design to measure speed gains from multi-instance labeling (vs. single-instance), i.e., reduce per-item overhead when many items share the same label.

— In coactive learning (Shivaswamy and Joachims 2015), the algorithm presents its predictions and the annotator (human) is only required to make corrections if necessary. Specifically, the learning algorithm observes a context (e.g., the user's search query) and presents a structured object (e.g. a ranked list of URLs). As feedback the human user returns an improved object (e.g. reordered list of URLs). Notably, the user and algorithm aim to jointly maximize the user's own utility. However since the user is only boundedly rational, he/she is assumed to only act approximately according to their own utility.

User-centered / visual-interactive active learning

— Bernard et al. (2018) formalize human selection heuristics into algorithmic "user strategies" built from reusable building blocks and evaluate each user strategy against standard active learning algorithms. However, they do not introduce a way to combine user strategies with active learning algorithms. They explicitly propose tighter coupling as an interesting future direction to combine the strengths of both. Meanwhile, they do not have real human-in-the-loop, and explicitly note that formalized user strategies may not replace real users selecting instances.

Learning to defer

— In Madras et al. (2018), in the first-step, the algorithm can either predict (positive/negative) or say PASS. If it predicts, the human decision maker will output the model's prediction. However, if it says PASS, the human decision maker makes its own prediction.

— Wilder et al. (2020) focus on settings where a machine takes on the tasks of deciding which instances require human input. In a classification task, the machine learning model first sees the sample x and then decides whether to pay a cost c to observe a human's prediction.

— The goal of Mozannar et al. (2023) is to jointly learn a classifier with a rejector. The rejector decides, for each data point, whether the classifier or the human should predict. The aim is for the classifier to complement the human so that error rate of the human-AI team is smaller.

— Recent work extends this to multiple experts, calibration, conformal expert selection, and limited human labels.

Learning to rank / Active preference-based learning

— Jamieson and Nowak (2011) examine the problem of ranking a collection of objects using pairwise comparisons (rankings of two objects).

— Ailon (2012) proposes an active ranking algorithm from pairwise preferences, where the algorithm proposes a pair to query, and each query is literally "which of these two do you prefer?"—i.e., the expert picks one item from the presented (size-2) subset.

— Biyik and Sadigh (2018) design a set of approximation algorithms for efficient batch active learning to learn about humans' preferences from comparison queries. The algorithm generates a batch of b pairwise queries at once, and the human chooses which one they prefer within each pair.

Multi-armed / Contextual bandits

— While the settings and reward models are different from ours, some work has looked at human-algorithm collaboration in bandit settings. Gao et al. (2021) learn from batched historical human data to develop a routing algorithm that assigns each task to either an algorithm or a human, under bandit feedback. In Chan et al. (2019), a robot assists a human in a bandit setting to maximize cumulative reward. The robot makes the final decision that overrides the human's recommendation. However, the robot only observes which arms the human pulls but not the reward associated with each pull.

— Cascading / Combinatorial / Top-K contextual bandits. The algorithm selects and shows a list of K items, and the human user scans and clicks one/few items from the slate. The feedback to the algorithm is the user clicks. In this research paradigm, human user is selecting subset and meanwhile human is providing rewards (i.e. clicks), and hence it is essentially human preference alignment.

References

Ailon N (2012) An active learning algorithm for ranking from pairwise preferences with an almost optimal query complexity. *The Journal of Machine Learning Research* 13(1):137–164.

Attenberg J, Provost F (2010) Why label when you can search?: Alternatives to active learning for applying human resources to build classification models under extreme class imbalance. *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 423–432 (Washington DC USA: ACM), ISBN 978-1-4503-0055-1.

Balakrishnan M, Ferreira K, Tong J (2022) Improving human-algorithm collaboration: Causes and mitigation of over- and under-adherence. *Available at SSRN 4298669* .

Bernard J, Hutter M, Zeppelzauer M, Fellner D, Sedlmair M (2017) Comparing visual-interactive labeling with active learning: An experimental study. *IEEE transactions on visualization and computer graphics* 24(1):298–308.

Bernard J, Zeppelzauer M, Lehmann M, Müller M, Sedlmair M (2018) Towards user-centered active learning algorithms. *Computer Graphics Forum*, volume 37, 121–132 (Wiley Online Library).

Biyik E, Sadigh D (2018) Batch active preference-based learning of reward functions. *Conference on robot learning*, 519–528 (PMLR).

Blaschke T, Arús-Pous J, Chen H, Margreitter C, Tyrchan C, Engkvist O, Papadopoulos K, Patronov A (2020) Reinvent 2.0: an ai tool for de novo drug design. *Journal of chemical information and modeling* 60(12):5918–5922.

Cakmak M, Thomaz AL (2011) Mixed-Initiative Active Learning. *ICML Workshop* .

Chan L, Hadfield-Menell D, Srinivasa S, Dragan A (2019) The assistive multi-armed bandit. *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 354–363 (IEEE).

Ciravegna G, Precioso F, Betti A, Mottin K, Gori M (2023) Knowledge-driven active learning. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 38–54 (Springer).

Fügener A, Grahl J, Gupta A, Ketter W (2021) Will humans-in-the-loop become borgs? Merits and pitfalls of working with AI. *Management Information Systems Quarterly (MISQ)-Vol* 45.

Fügener A, Grahl J, Gupta A, Ketter W (2022) Cognitive challenges in human–artificial intelligence collaboration: investigating the path toward productive delegation. *Information Systems Research* 33(2):678–696.

Gao R, Saar-Tsechansky M, De-Arteaga M, Han L, Lee MK, Lease M (2021) Human-AI collaboration with bandit feedback. *arXiv preprint arXiv:2105.10614* .

Ghai B, Liao QV, Zhang Y, Bellamy R, Mueller K (2021) Explainable active learning (xal) toward ai explanations as interfaces for machine teachers. *Proceedings of the ACM on human-computer interaction* 4(CSCW3):1–28.

Jamieson KG, Nowak R (2011) Active ranking using pairwise comparisons. *Advances in neural information processing systems* 24.

Laugel T, Lesot MJ, Marsala C, Renard X, Detyniecki M (2018) Comparison-based inverse classification for interpretability in machine learning. *International conference on information processing and management of uncertainty in knowledge-based systems*, 100–111 (Springer).

Lebovitz S, Levina N, Lifshitz-Assaf H (2021) Is AI ground truth really true? The dangers of training and evaluating AI tools based on experts' know-what. *MIS quarterly* 45(3).

Lee HS, Shen C, Jordon J, Schaar M (2020) Contextual constrained learning for dose-finding clinical trials. *International Conference on Artificial Intelligence and Statistics*, 2645–2654 (PMLR).

Lin Z, Akin H, Rao R, Hie B, Zhu Z, Lu W, Smetanin N, Verkuil R, Kabeli O, Shmueli Y, et al. (2023) Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science* 379(6637):1123–1130.

Madras D, Pitassi T, Zemel R (2018) Predict responsibly: improving fairness and accuracy by learning to defer. *Advances in neural information processing systems* 31.

Mosqueira-Rey E, Hernández-Pereira E, Alonso-Ríos D, Bobes-Bascarán J, Fernández-Leal Á (2023) Human-in-the-loop machine learning: a state of the art. *Artificial Intelligence Review* 56(4):3005–3054.

Mozannar H, Lang H, Wei D, Sattigeri P, Das S, Sontag D (2023) Who should predict? exact algorithms for learning to defer to humans. *International conference on artificial intelligence and statistics*, 10520–10545 (PMLR).

Nahal Y, Menke J, Martinelli J, Heinonen M, Kabeshov M, Janet JP, Nittinger E, Engkvist O, Kaski S (2024) Human-in-the-loop active learning for goal-oriented molecule generation. *Journal of Cheminformatics* 16(1):1–24.

Negoescu DM, Frazier PI, Powell WB (2011) The knowledge-gradient algorithm for sequencing experiments in drug discovery. *INFORMS Journal on Computing* 23(3):346–363.

Nguyen Q, Garnett R (2023) Nonmyopic multiclass active search with diminishing returns for diverse discovery. *International Conference on Artificial Intelligence and Statistics*, 5231–5249 (PMLR).

Orabona F, Cesa-Bianchi N, et al. (2011) Better algorithms for selective sampling. *Proceedings of the 28th international conference on machine learning: Bellevue, Washington, USA, june 28. july 2, 2011*, 433–440 (Omnipress).

Rafailov R, Sharma A, Mitchell E, Manning CD, Ermon S, Finn C (2024) Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems* 36.

Sekhari A, Sridharan K, Sun W, Wu R (2023) Selective sampling and imitation learning via online regression. *Advances in Neural Information Processing Systems* 36:67213–67268.

Settles B (2011) Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. *Proceedings of the 2011 conference on empirical methods in natural language processing*, 1467–1478.

Shin D (2024) Misinformation and algorithmic bias. *Artificial Misinformation: Exploring Human-Algorithm Interaction Online*, 15–47 (Springer).

Shivaswamy P, Joachims T (2015) Coactive learning. *Journal of Artificial Intelligence Research* 53:1–40.

Suh J, Zhu X, Amershi S (2016) The Label Complexity of Mixed-Initiative Classifier Training. *ICML* .

Sundin I, Voronov A, Xiao H, Papadopoulos K, Bjerrum EJ, Heinonen M, Patronov A, Kaski S, Engkvist O (2022) Human-in-the-loop assisted de novo molecular design. *Journal of Cheminformatics* 14(1):86.

Wilder B, Horvitz E, Kamar E (2020) Learning to complement humans. *arXiv preprint arXiv:2005.00582* .