Topic #7: Human-AI interaction in RL for molecular optimisation

Background: Reinforcement Learning (RL) has shown promise in advancing molecular design and therapeutic research. Domain expertise in medicinal chemistry is crucial to design the therapeutic goal (reward) and successfully guide the design process. Recently, human-in-the-loop RL was introduced to the molecular design field, enabling interactive reward learning from human feedback [1]. While a wide variety of human-AI interaction approaches for reward learning in RL were proposed in the existing literature, only very few were applied or adapted to the case of molecular optimization. In this thesis project, you will study various human-AI interactions for reward learning that were successfully used in applications such as image recognition, text generation etc., and further investigate the potentially useful and feasible ones for the molecular optimisation case. You will work on modeling and implementing the process of human-AI interaction and human feedback integration for reward learning, and engage in discussions with the UI designer in charge of providing the interface for future experiments with chemists. The main aim of this project is to propose a model of human-AI interaction for reward learning in molecular optimization and validate it with experiments through the UI.

Project Scope:

- Investigate various human-AI interaction methods for reward learning, drawing from applications in image recognition and text generation.
- Adapt promising interaction methods for molecular optimization.
- Develop models for human-AI interaction and integrate human feedback into the RL process.
- Collaborate with a UI designer for creating an interface for experiments with chemists.
- Aim: Propose and validate a human-AI interaction model for reward learning in molecular optimization.

Technical Details:

- Utilize RL to optimize existing generative models for creating molecules.
- Focus on how RL algorithms can provide feedback to these generative models.

Main tasks:

- Literature Review: Survey human-AI interaction methods in various fields and their applicability to molecular generation.
- Conceptual Work: Design a workflow for RL in molecular optimization, including a simple GUI.
- Credit Allocation: 10 credits (adjustable to 5 if needed), to be completed after the third period.
- Team Composition: Project suitable for 2 students.

Initial Steps:

- Begin with a thorough literature review.
- Focus on generative models for molecular discovery and deep RL in molecular de novo design.

Key Topics to Cover:

- Molecular de novo design and tools like REINVENT 2.0.
- Cheminformatics and tools like RDKit for molecular representation.
- Human-in-the-loop (HITL) Machine Learning concepts and applications.
- Active Learning methodologies (optional but recommended).
- Further Directions:

- Explore HITL in the context of molecular design, focusing on methods like preference-based feedback.
- Investigate the use of Gaussian Process as a surrogate model for human feedback in HITL setups.
- Aim to understand the interaction between human feedback, the surrogate model, and the RL model.

Deliverables:

- Write a comprehensive literature survey covering RL, HITL, and Molecular Design.
- Develop a project sketch and outline by the end of May.
- Progress reports to be maintained in Overleaf for monitoring by Mrs. Yasmine.
- Technical Requirements:
- Primary language: Python.
- Tools: RDKit and REINVENT.

Deadline:
Complete the project by May 1st.

You will use **Reinforcement Learning** to optimize **generative models** for generating molecules with different objectives for the molecules…

Generative models are already existing. Humans

The RL algorithm will give the feedback to the generative models.
The main work:
Literature Works
Survey different ways of interacting Extract information from the expert, what are the types of strategies that can work well in the context of the RL for molecular generation.
Conceptualization Work?: Design a workflow for reinforcement learning. It also involves developing a simple GUI

Generative models for molecular discovery
Stop at section (3)
Molecular de novo design through deep RL (very important)
REINVENT 2.0 an AI tool for De Novo Drug (easy)
Molecular design is drugs

Go to the end of the Tutorials:
cheminformatics: computer science on molecular chemistry (how to represent and understand molecules on computers)
RDKit: a software that allow us to generate a numerical representation of the molecules, given molecular (2D image for example) => RDKit calculate SMILE representation and perform computations on computer

HITL ML:
Instead of optimizing the rewards function, humans can act as the rewards
HITL ML paper: very good, state of the art such as active learning

A surveyor HITL for ML: give you a lot of examples from image classification, text recognition etc.

Deep RL from human preferences (how ChatGPT works)

HITL molecular design:
HITL de novo MD: uses REINVENT where they add human feedback to the human likes. Yes or No feedback to MD.
Extracting medicinal chemistry: feedback: show two designs to humans and choose preference

Active learning (not compulsory)
Read the second paper.

(1) Write a short literature survey
    Understand the three key points: RL, HITL, Molecular design
(2) Where to find: a survey of HITL for ML
(3) Devise a sketch and devise a sketch

End of May to finish this project

Python (mostly):
Definition of Completion:

HITL REINVENT is HITL assisted de novo molecular design:

Query the human expert can give feedback, and how the feedback is fed back to the generative model. The HITL is the one that builds a model of the human. Gaussian Process as surrogate model for the human feedback => human only gives a few feedback, and those points will be used to train the surrogate model (acting as a human), which is like a user model (as a reward model for RL). The process of getting new data from the human is the focus of this process.

The human feedback is used to train the surrogate model, not actually training the RL model. The RL model will use the surrogate model as the reward function instead.

Create a progress report in Overleaf for Mrs Yasmine to keep track of the progress…

It depends on me how much I can do (5-10 credits).
Put as 10 credits (cannot work a lot, switch back to 5)

We would use these two RDKit and REINVENT

1st may is the last day

==========

Note meeting 23/01/2024

First, (I should focus on Active Learning (broader field) for Bayesian Optimization find the datapoint that maximize the reward function). We want to maximize what the human prefers.

REINVENT: contains the software that will help us to generate molecules based on Reinforcement Learning
REINVENT-Community: it contains lots of notebooks, examples of how to use REINVENT. There is no source code. It is coupled with REINVENT repository to get used to the software, as the tutorials. I don't need to use REINVENT community for this project. There is no HITL here

REINVENT-HITL: most relevant, related to human preferences in active learning. Also uses REINVENT software, but it also adds two new features: active learning and human in the loop features. I must study this REINVENT-HITL carefully for this project.

REINVENT can only run on Linux system.

RDKit: It is just a python package to generate the SMILES, molecules, numerical features. We would need to write the molecules in numerical features.
In REINVENT-HITL, they also use RDKit.

REINVENT-HITL: there is only one kind of feedback: Binary decision (shown a molecular and human says yes or no).
What Yasmine want me to investigate is other types of interactions. Binary preference (shown two molecules and prefer which one), and also ranking molecules.

Make comparisons between these interactions to see which one is the best.

By next meeting: it would be great if Binh can read these three papers

Human-in-the-loop machine learning :
Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D. et al. Human-in-the-loop machine learning: a state of the art. Artif Intell Rev 56, 3005–3054 (2023). DOI.
Wu, Xingjiao and Xiao, Luwei and Sun, Yixuan and Zhang, Junhang and Ma, Tianlong and He, Liang. A survey of human-in-the-loop for machine learning. Future Generation Computer Systems 135, 364–381 (2022). DOI.
Paul Christiano and Jan Leike and Tom B. Brown and Miljan Martic and Shane Legg and Dario Amodei. Deep reinforcement learning from human preferences. arXiv. (2023).

Make investigations, finish these papers, summarize them and present them.

Make a short presentation on different kind of human interactions
One slide for each interaction (binary feedback, binary preference, ranking). Inspirations from the two papers state of the art and deep RL above

==========

Note meeting 23/01/2024

Binary feedback: showing an instance to human expert, asking if they like it or object
Binary preferential feedback: human was shown 2 objects, they prefer which one? It is a special case of ranking feedback of N = 2
Ranking feedback: showing N >= 3 objects, asking humans to rank the objects.
Using a surrogate model from human
Mostly we learn the model of human preferences, using that model of human references to generate molecules.

Step 1: The oracle is also a machine learning model (simulated)
Step 2: Using real humans through

REINVENT-HITL uses REINFORCE algorithm which is policy gradient algorithm
The agent in REINVENT is generative model for molecules, LSTM for seq2seq model for write SMART format
The third model is the human preference model, used as the reward model. To improve itself based on human preferences.

Comparing different feedback types from “A survey of RL from human feedback”, showing the agent in the reward model in human preferences, training LLM.

What I need to do in the next two weeks:

Read the paper “A survey of RL from human feedback”

Read the paper,
I. Sundin, A. Voronov, H. Xiao, K. Papadopoulos, E. Bjerrum, M. Heinonen, A. Patronov, S. Kaski, and O. Engkvist, “Human-in-the-loop assisted de novo molecular design,” Journal of Cheminformatics, vol. 14, 12 2022

There are two tasks in this paper, Task 1 and Task 2. Focus only on Task 2 in this paper

This paper also has a source code, please consult the source code reinvent-hitl.

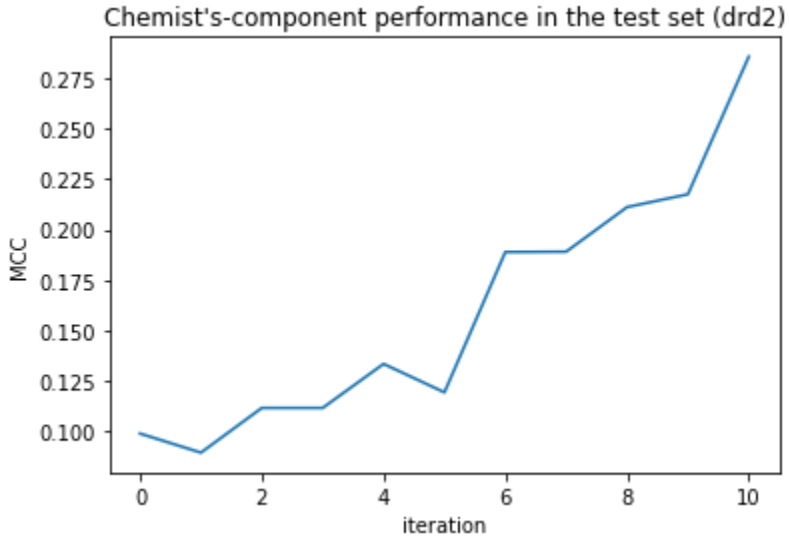
Try to run the source code. If we have some technical problems, ask again
Focus on the Chemists-Component (Task 2)

Need to prepare some questions, highlight the part in the two papers.

==========

Note meeting 27/02/2024

drd2 is a protein, MCC is increasing



MCC is the Matthew correlation coefficient
As the iterations increase, MCC increases => Learning the model of the chemist preferences about drd2 properties

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}$$

The features are binary (active or inactive) => using classification metrics

REINVENT uses REINFORCEMENT LEARNING

The input.py and json file are in

config.json file is for saving the underscore

1) Prepare a PPT of questions of things I have understood and have not understood
2) Trying to thoroughly understand the code and the de novo design paper.

outer loop is original REINVENT (K is the number outer loop in Task 1_MPO)
inner loop is HITL interactions (T is the

json file is something to interact with REINVENT, such as the scoring function
In original REINVENT, the scoring function is not updated
But with HITL, the scoring function is updated in the inner loop
acquisition.py selects one of the generated molecules from REINVENT and sends it to humans.

The new scoring function is fed to the json file for REINVENT to generate new molecules (the blue box in the figure 1).

The scoring function is replaced by a Gaussian process when Run the notebook; if simulated_human=False it will wait for input after writing the first query (query_it1.csv). If simulated_human=True, the notebook will continue a whole run of simulated HITL interaction
Write about 20 pages.

==========

Note meeting 19/03/2024

First question 1:

Algorithm 1 Adapting parameters $\boldsymbol{\theta}$ of the MPO function

Input: A probabilistic model of the chemist's score M (equations (4-7)), MPO objective function $S_{\boldsymbol{\theta}}(\cdot)$ (equation (3)), initial values $\boldsymbol{\theta}_0$
$D_0 = \emptyset$
for $r = 1, 2, \ldots, R$ **do**
  $\mathcal{U}_r \leftarrow \text{REINVENT}(S_{\boldsymbol{\theta}_{r-1}})$ ▷ $\mathcal{U}_r$ set of molecules from the design tool using $S_{\boldsymbol{\theta}_{r-1}}$
  Select n_0 molecules x uniformly at random from $\mathcal{U}_r$: acquire feedback y on each x from chemist
  $D_{r,0} \leftarrow D_{r-1} \cup \{(x_i, y_i)\}_{i=1}^{n_0}$
  $p(\boldsymbol{\theta} \mid D_{r,0}) \leftarrow \text{getPosterior}(M, D_{r,0})$ ▷ equation (5)
  for $t = 1, 2, \ldots, T$ **do**
    for $query = 1, 2, \ldots, n_{batch}$ **do**
      $x^* \leftarrow \text{selectQuery}_{TS}(p(\boldsymbol{\theta} \mid D_{r,t-1}), S_{\boldsymbol{\theta}}, \mathcal{U}_r)$ ▷ Section 2.2.3
      Acquire chemist's feedback y^* for x^*
      Remove x^* from $\mathcal{U}_r$
    end for
    $D_{r,t} \leftarrow D_{r,t-1} \cup \{(x_i^*, y_i^*)\}_{i=1}^{n_{batch}}$
    $p(\boldsymbol{\theta} \mid D_{r,t}) \leftarrow \text{getPosterior}(M, D_{r,t})$ ▷ equation (5)
  end for
  $\boldsymbol{\theta}_r \leftarrow \int \boldsymbol{\theta}\, p(\boldsymbol{\theta} \mid D_{r,T}) d\boldsymbol{\theta}$
  $D_r \leftarrow D_{r,T}$
end for

This is Task 1 Adaptive MPO pseudocode. However after reading the Task 2 Chemist Component, I have a feeling that they use the same pseudocode as Adaptive MPO. May I know what could be the difference in the Chemist component in this pseudocode?

Ur: many molecules generated at each round
Chemist feedback in task 1 is binary
Only difference between Task 1 and 2 is Task 1, we start from a given scoring component, REINVENT have 2 scoring functions: weighted sum or weighted products, and the chemist binary feedback

ist $(t = 1, \ldots, T)$. Let $\theta_{r,t,k} \in \mathbb{R}^{d_k}$ denote the unknown parameters of $\phi_{r,t,k}$, and simplify notation by writing $\phi_k\left(c_k(x), \theta_{r,t,k}\right) := \phi_{r,t,k}(c_k(x))$. The number of param-

The theta parameters here refers to the weights of the scoring function (weighted sum or weighted product)

$$S(x) = \left[\prod_{k=1}^{K} \phi_k(c_k(x))^{w_k}\right]^{1/\sum_{k=1}^{K} w_k} \quad (1)$$

or a weighted sum

$$S(x) = \frac{\sum_{k=1}^{K} w_k \phi_k(c_k(x))}{\sum_{k=1}^{K} w_k} \quad (2)$$

In Task 2, we no longer use the scoring function from REINVENT and replace with a non parametric Gaussian process, which learns from Chemist real value feedback in range [0, 1] 0 means not good and 1 means highly likely good. In the source code, they apply the clipping on the prediction of Gaussian process into [0, 1].
So in this work, Gaussian process is trained on real-valued between 0 and 1, it acts as regression model. When we put this GP into REINVENT loop, we apply a clipping function on the output of the GP to ensure that the scoring is inside [0, 1]. Values closer to 1 means better and closer to 0 means worse. It is not binary.

In Task 2, We have formulas for likelihood, prior and posterior to be replaced into the pseudocode in Task 1 above

Question 2: I want to elaborate on this passage

On Page 7 of 16

The second method we propose is applicable in cases where a pre-specified scoring component for a specific property is not available but, instead, the values for the molecular property of interest can be obtained via interaction with a chemist and in addition potentially in a small experimental dataset.

So does this passage says that $\phi_{r,t,k}(c_k(x)) \in [0,1]$, k = 1, …, K molecular props, each measuring utility of molecular property $c_k(x) \in R$ is not available?

> The adaptive MPO scoring function consists of K adaptive scoring components $\phi_{r,t,k}(c_k(x)) \in [0,1]$, $k = 1,\ldots,K$, each measuring the utility of a molecular property $c_k(x) \in \mathbb{R}$ that can be computed from a molecule x. The MPO function is adapted by modifying the

The method learns a new predictive model from the chemist's feedback based on the property values, and the resulting component can then subsequently be used as one of the objectives in MPO

So this sentence says that we are trying to build a surrogate model for $\phi_{r,t,k}(c_k(x))$?

This predictive model is GP.
Either the scoring component from Task 1 or GP from Task 2 is written into Json file to be sent into REINVENT for generating molecules

Question 3:

Molecules are represented by features, which in this work are descriptors such as physicochemical properties; x ∈ R_p, or Morgan fingerprints x ∈ $\{0, 1\}^d$ [38], where d is the dimensionality of the features.

Can you help me describe what x looks like?

Here are some of the physicochemical properties

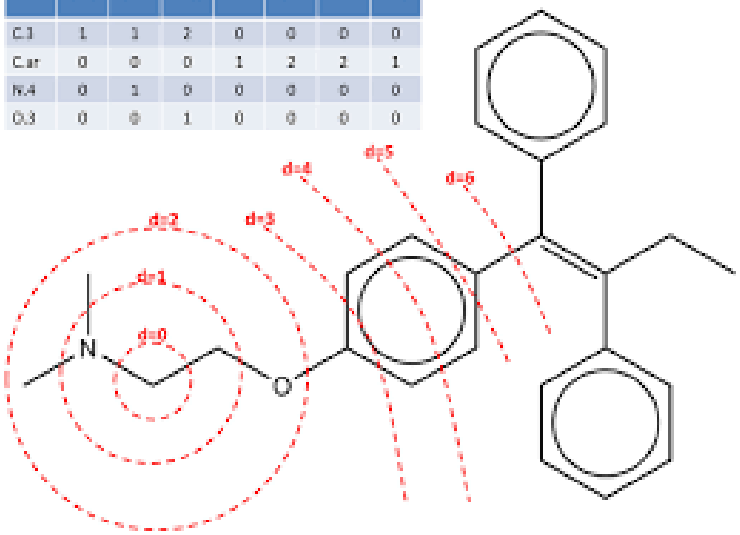
1) Molecule weights (p1)
2) How many atoms does it have? (p2)
3) How many Hydrogen bonds, acceptors (p3)
4) How many rings? (p4)

p is the number of properties (4 like above), x is a vector of these 4 values [p1, p2, p3, p4]

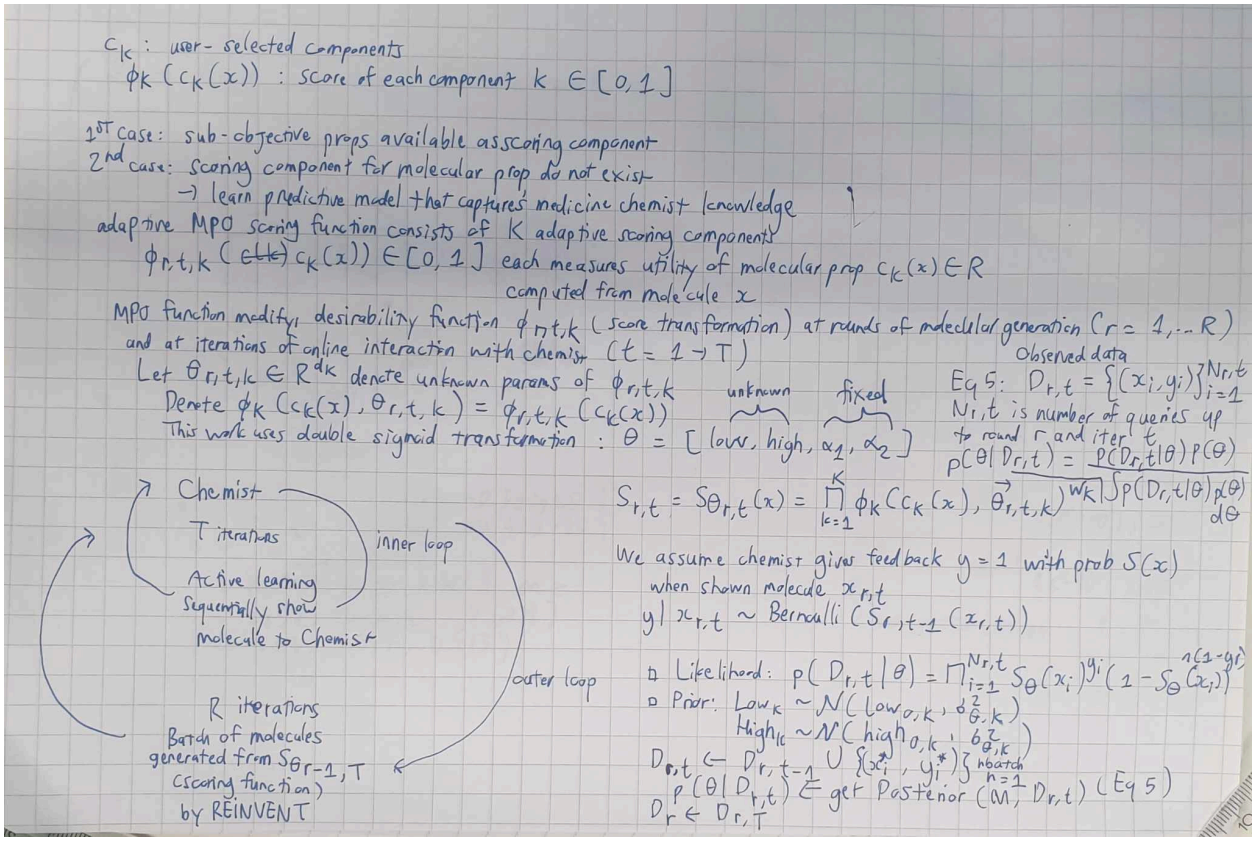
d is the size of the finger print, possibly 1024 or 2048

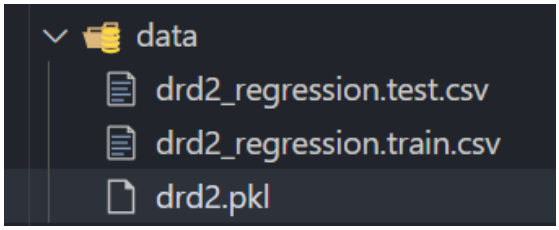
Morgan fingerprint is like a 2D image of a molecular, represented by many vectors of 0 and 1, where 0 may mean the pixel/substructure is empty, and 1 means there is something of the molecular there

	d=0	d=1	d=2	d=3	d=4	d=5	d=6
C.1	1	1	2	0	0	0	0
C.ar	0	0	0	1	2	2	1
N.4	0	1	0	0	0	0	0
O.3	0	0	1	0	0	0	0



And which option does the code in Chemist-Components use? physicochemical props or Morgan fingerprints?: In the Task 2, we use Morgan's fingerprints.





This contains the ML model that acts as an oracle, which simulates the human feedback in Task 2.
For task 1, We use the QED from RDKit to simulate the human feedback
drd2 is the property, and task 2 has drd2 as the only thing for optimize => k = 1.

But in Task 1, there are many properties besides drd2 so k > 1

==========

This is one idea:

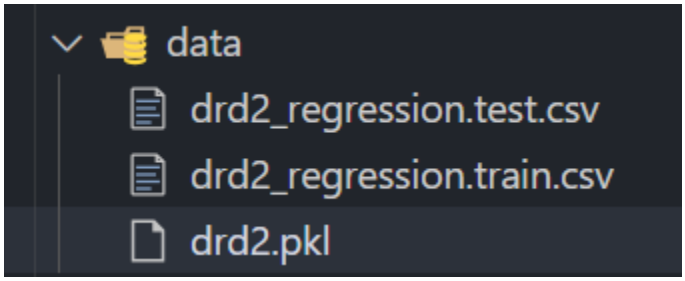
The purpose of the programming task is to change the real value human feedback in Task 2 into different feedback type (given two molecules)

Some reference: https://en.wikipedia.org/wiki/Bradley–Terry_model

We can use this model to compare generated molecules to an existing database of molecules

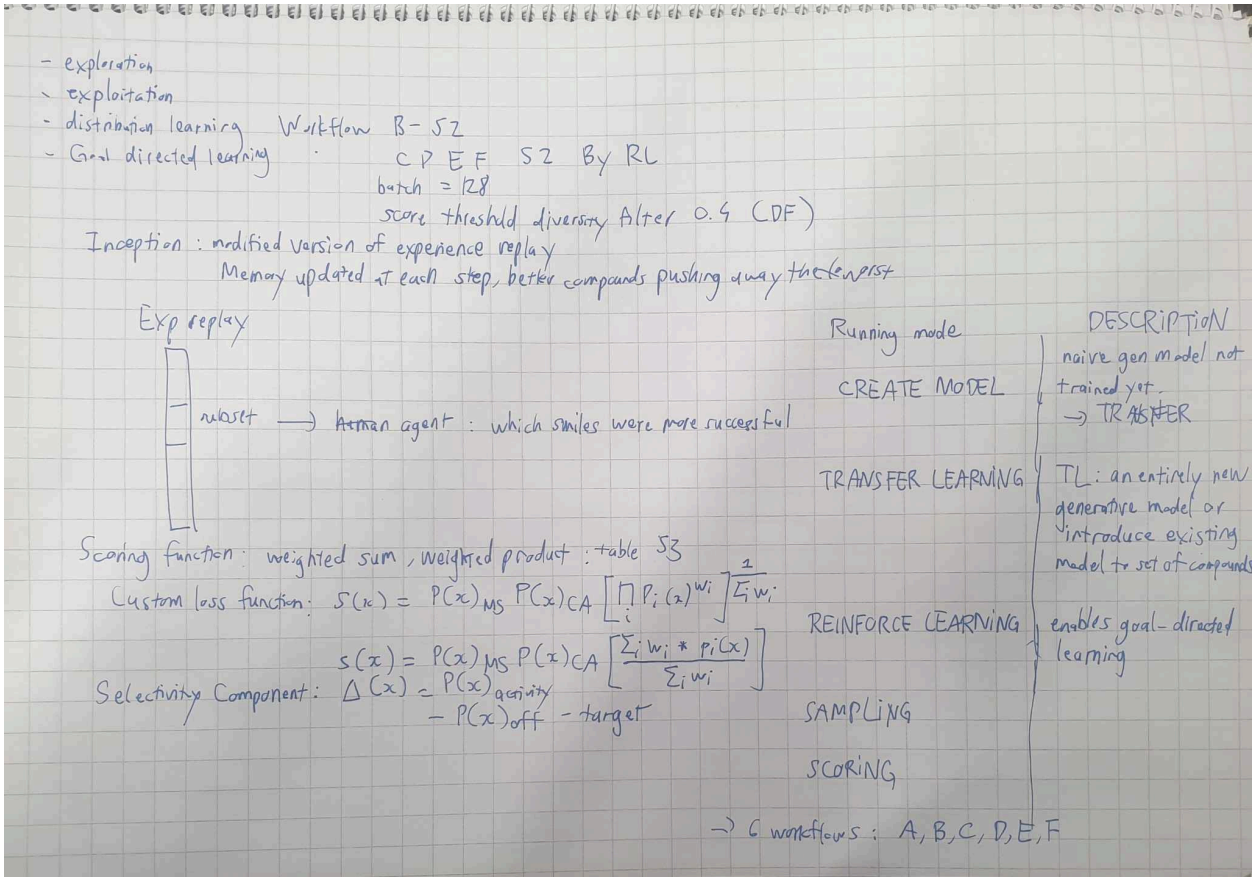
REINVENT source code can be changed for scoring function, such as nearest neighbor in the molecule space or nearest neighbor compared to previous molecules.

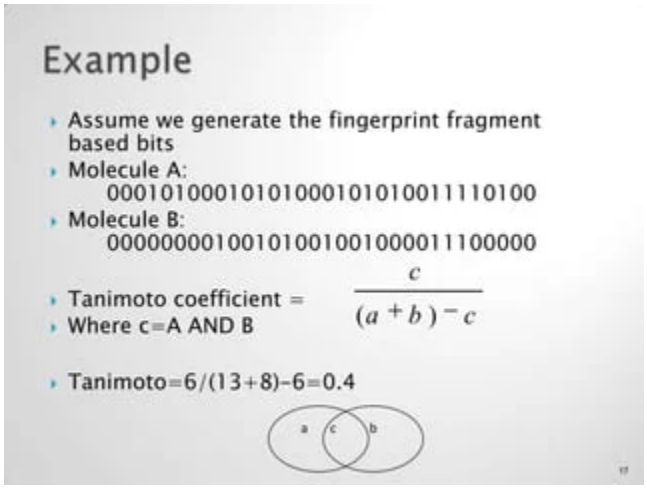
Testing by using



==========

Note meeting 19/03/2024





drd2.train.csv: N = 275768 datapoints
drd2.test.csv: N = 68944 datapoints
they have two columns canonical, which is SMILES, and activity for drd2
drd2 is the Dopamine Receptor D2, the molecule either has it (0): 272320 or not has it (1): 3448, told by the activity column. The dataset is very unbalanced between the two class

Model training
To train your model (i.e. fit the model's parameters to the training data), execute the following cell. Note, that `REINVENT` will access the `proba` property of the model to get a probability rather than a predicted label. If you want to optimize the hyper-parameters of you model, we suggest you use a cross-validation approach (we aim to publish our in-house method based on [optuna](https://optuna.org) soon).

Integration into `REINVENT`
In order to use your new model as a component in the scoring function of `REINVENT`, you need to include a block with the appropriate parameter settings (see below). Note, that the descriptor definition needs to match.

```
{
  "component_type": "predictive_property",
  "name": "DRD2_pred_activity",
  "weight": 1,
  "specific_parameters": {
    "model_path": "/path/to/model/folder/DRD2_final_model.pkl",
    "scikit": "classification",
    "transformation: {
        "transformation_type": "no_transformation"
    },
    "descriptor_type": "ecfp_counts",
    "size": 2048,
    "radius": 3,
    "use_counts": True,
    "use_features": True
  }
}
```

==========

Note meeting 22/04/2024

1. In the case if there is no human in the loop

Imagine a DRD2 dataset with inactive or active molecule
We train a classifier that given a molecule (morgan fingerprints), then it predicts 0 or 1
Then, we would pass this classifier model to the scoring function in the JSON file

```
"scoring_function": {
    "name": "custom_sum",
    "parallel": true,
    "parameters": [
      {
        "component_type": "predictive_property",
        "name": "bioactivity",
        "specific_parameters": {
          "container_type": "scikit_container",
          "descriptor_type": "ecfp_counts",
          "model_path":
"/home/springnuance/reinvent-hitl/Reinvent-Community-Binh/notebooks/models/initial_qsar_model_oracle_truth.pkl",   <- this is the classifier, a weak classifier
          "radius": 3,
```

REINVENT would use this model to generate molecules using RL (baseline, no humans)
When there is no humans, the classifier passed to REINVENT should be powerful enough for generating precise molecules

2. In the case that if we want to use the humans

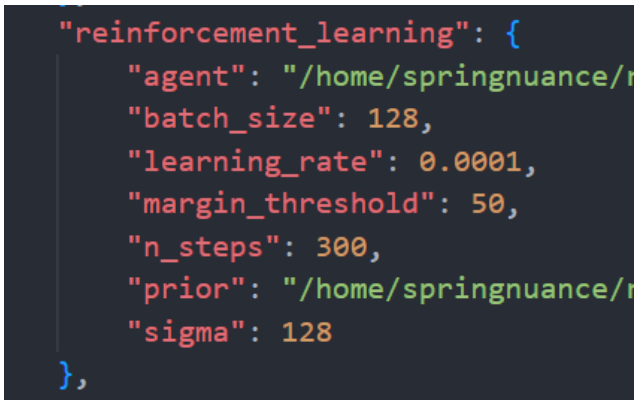
Like above, let's pretend as if there is no human. Initially we initialize a very weak classifier trained on only a few datapoints.

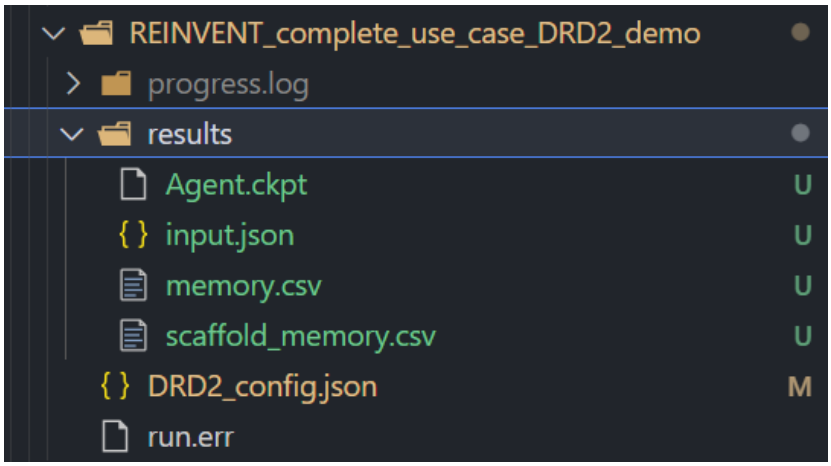
```
N0=10 # size of initial training data on the weak classifier
n_batch=10    # number of molecules shown at each iteration to the human for feedback
n_iteration=10  # the number of iteration, T = 10 in the figure
fpdim=1024 # dimension of morgan fingerprint
step=1 # REINVENT outer loop, R = 1

acquisition='thompson' #options: 'thompson', 'uncertainty', 'random', 'greedy'   <- in acquisition.py. These methods try to sample best n_batch molecules from the output of REINVENT to the human for feedback
```

READ_ONLY = False # Use 'True' to playback an existing experiment (reads queries and feedback from files instead of using the algorithm)
This would simply mean that instead of running REINVENT, we would use existing results from previous steps (the R loop) that is written in the csv file.

In order to change the REINVENT steps, we change the parameter n_steps inside the json file





Agent.ckpt: storing the REINVENT generative model (Neural networks) that has been optimized to generate molecules with highest scores

input.json: just our config file copy pasted here

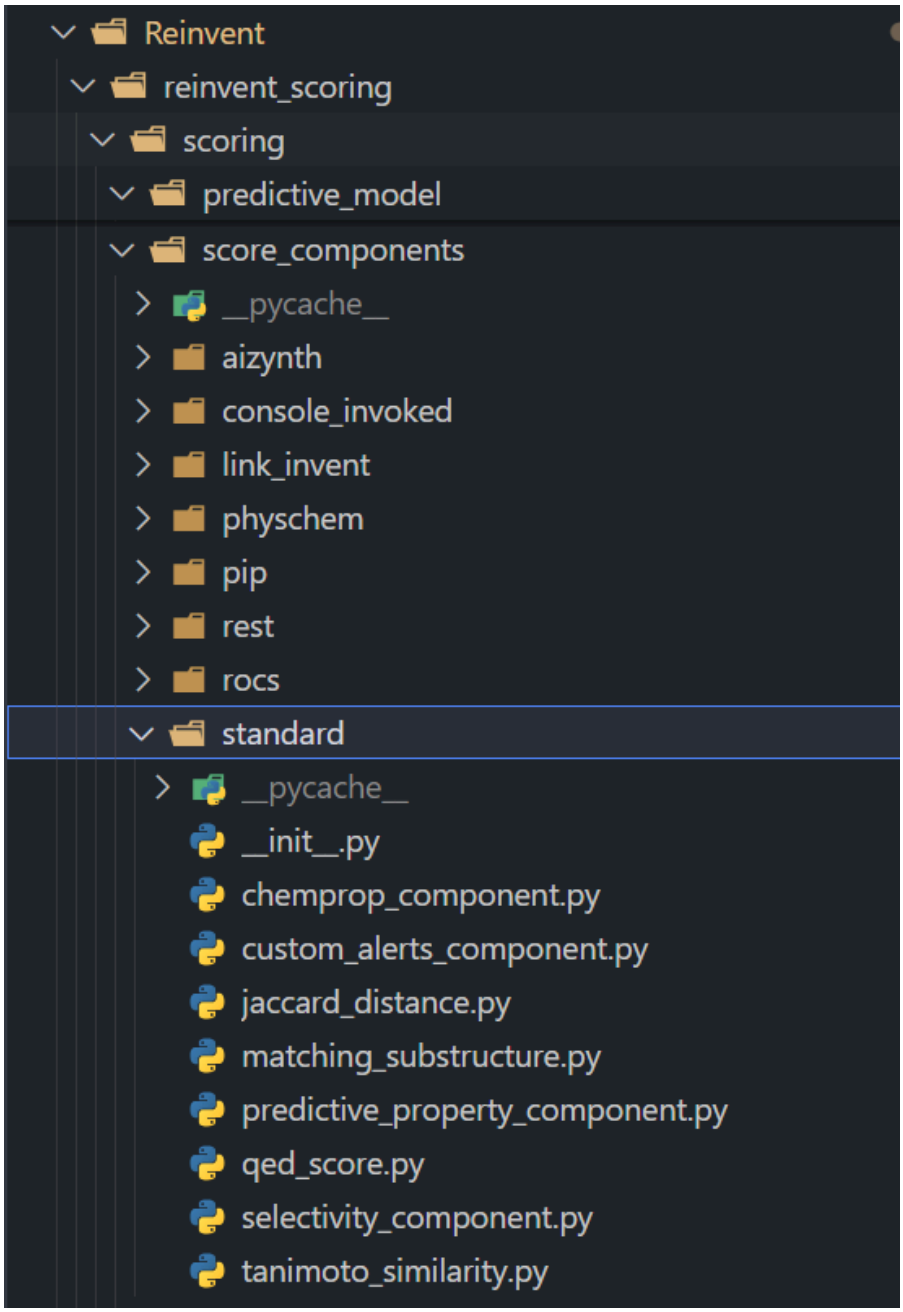
memory.csv: stores …

scaffold_memory.csv

```
Step,Scaffold,SMILES,DRD2_pred_activity,HB-donors (Lipinski),Number of
rotatable bonds,Custom_alerts,raw_DRD2_pred_activity,raw_HB-donors
(Lipinski),raw_Number of rotatable bonds,total_score,ID
```

```
3.0,O=C(CN1CCN(C(=O)C2CCCN2S(=O)(=O)c2ccccc2)CC1)Nc1ccccc1,COc1ccc(S(=O)(=O)N2CCCC2C(=O)N2CCN(CC(=O)Nc3c(F)cccc3F)CC2)cc1,0.4903411865234375,1.0,1.0,1.0,0.4903411865234375,1.0,7.0,0.6035987138748169,Use-case DRD2 Demo_0
```

- Step: the RL optimization step. It should be a number between [0, n_steps - 1] in reinforcement_learning in json file
- Scaffold: the core chemical structure that is shared by multiple generated molecules. Generated molecules are all different but they can share in common one chemical core, called scaffold.
- SMILES: the SMILES string representation of the generated molecule.
- DRD2_pred_activity: the predicted probability to be active by the ML classifier used in the scoring function. This is one property to be optimized.
- HB-donors (Lipinski): this is another property to be optimized in addition to the DRD2 activity. This score is high (close to 1) if the molecule has the desired number of atoms on which Hydrogen bonds can be formed. The value of this score should always be between 0 and 1.
- Number of rotatable bonds: another property to be optimized. This score is high (close to 1) if the molecule has the desired number of bonds between atoms that can rotate. The value of this score should always be between 0 and 1.
- Custom_alerts: this is another property to be optimized. This score is high (close to 1) if the molecule doesn't contain any undesired substructure from a list of pre-defined undesired substructures called custom alterts. The value of this score should always be between 0 and 1.
- raw_DRD2_pred_activity: "raw" simply means the original output of the ML classifier. Since DRD2_pred_activity is a probability, it is always between 0 and 1, so this score is the same as DRD2_pred_activity.
- raw_HB-donors (Lipinski): the number of atoms on which Hydrogen bonds can be formed.
- raw_Number of rotatable bonds: the number of bonds between atoms of the molecule that can rotate.
- total_score: the combination of all the properties to be optimized so the pred_DRD2_activitiy, number of HB-donors, custom alerts and number of rotatable bonds.
- ID: a unique identifier of the molecule.



1. Next meeting objective: how does Jasmine develop the scoring component and understand it.

2. Read about the scoring model that Binh need to develop. This is a standard model below to compare two objects.

3. Some reference: https://en.wikipedia.org/wiki/Bradley–Terry_model

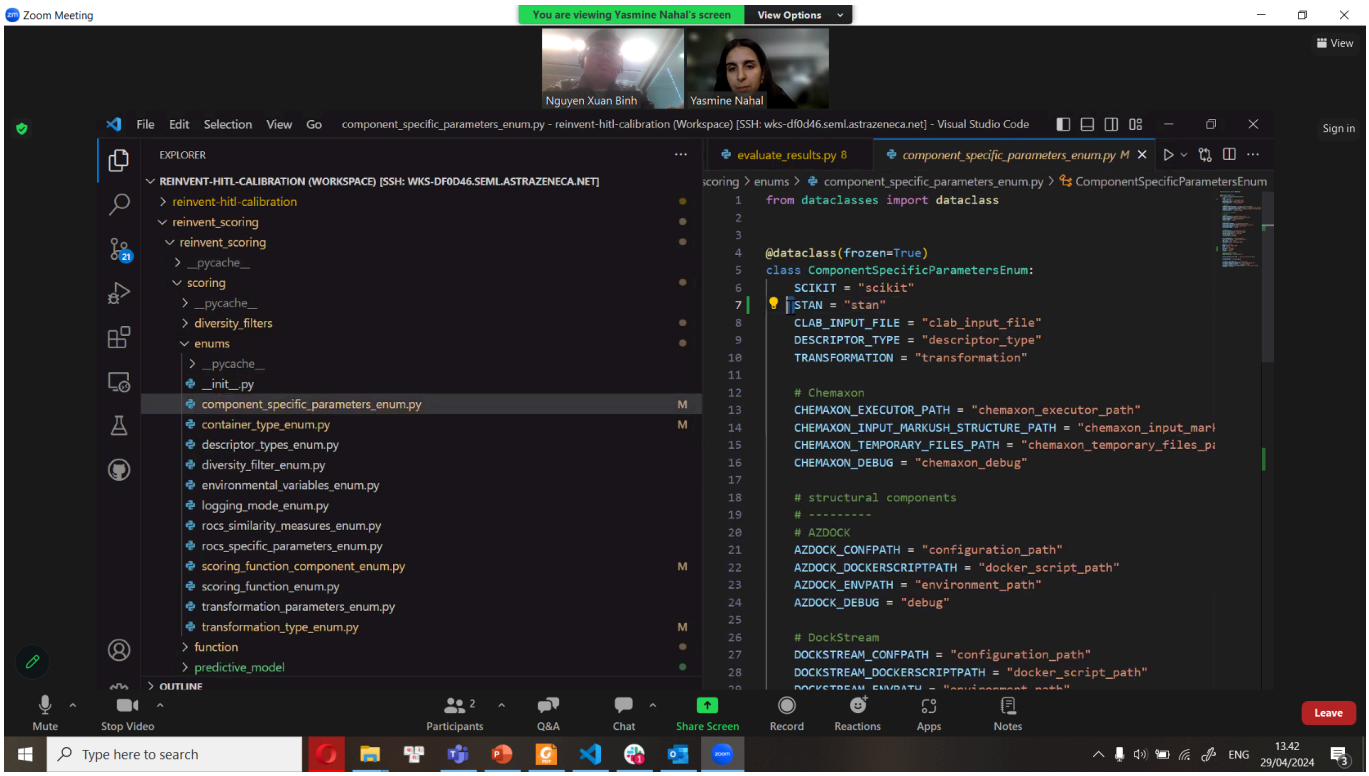
4. How to implement this BT model on your own as Jasmine.

In the chemist component, we have

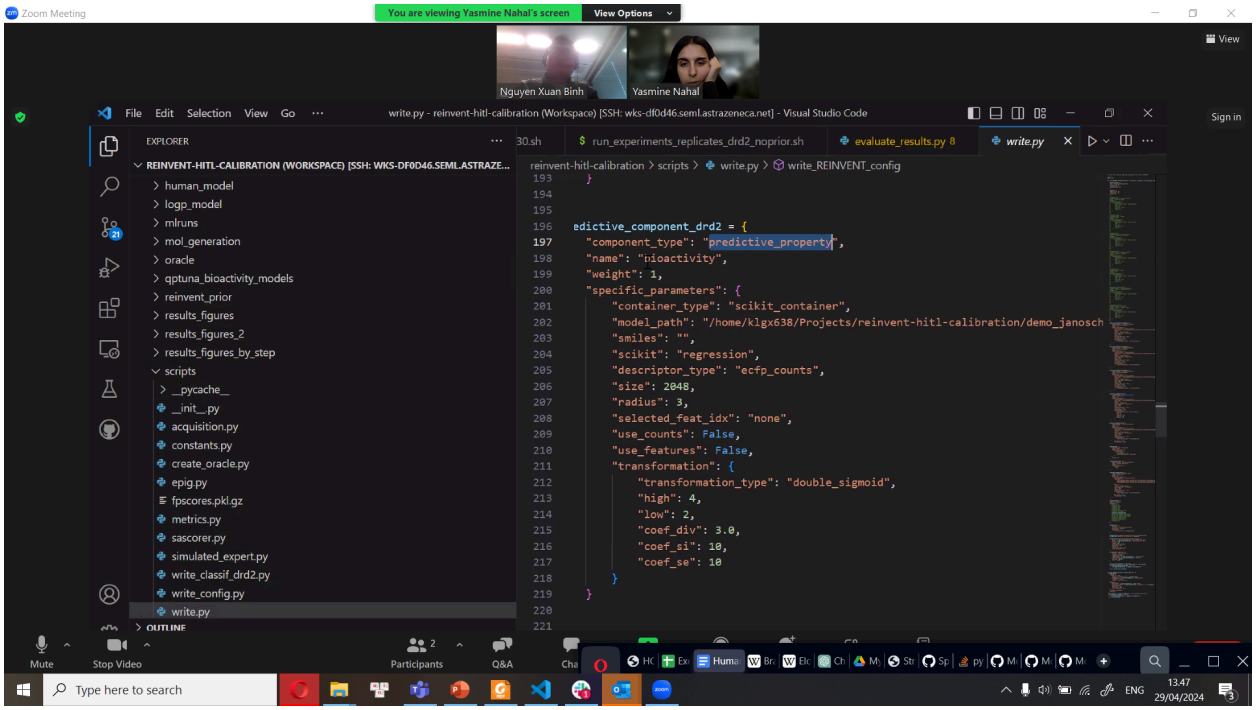
R = 1 (only one step)
T = 10 (ten iterations with humans)
K = 1 (only 1 property DRD2)

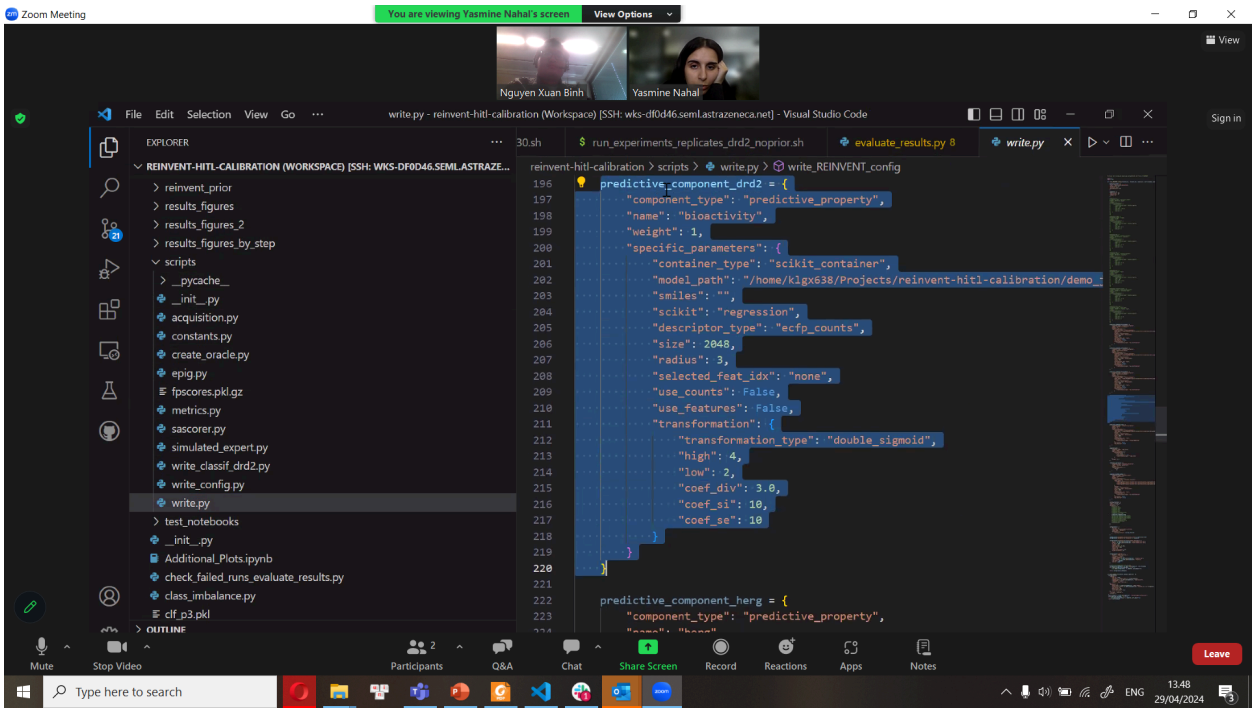
==========

Note meeting 22/04/2024



scoring/predictive_model/scikit_model_container.py which contains the base code of scikit learn model





In the predictive_component_drd2, we use scikit_container to refer to the scikit learn model (originally composed by Reinvent). We can also use torch container or stan model (created by yourself). The model_path is the human component according to Yasmine. If we use different container_type then the model_path should be the corresponding compatible model, like pytorch or stan models. Finally, Binh is supposed to train the bradley-terry model as pytorch

==========

Note meeting 29/04/2024

Modified files in reinvent_scoring you may need to modify:

reinvent_scoring -> scoring -> enums ->
component_specific_parameters_enum.py, ✅
container_type_enum.py, ✅
scoring_function_component_enum.py ✅
(*you may need to modify these files to enable activating your model via the config.json*)

reinvent_scoring -> scoring -> predictive_model -> model_container.py + *here you create your own container*, *for example*
"bradley_terry_model_container.py" ✅
and implement how your model will score the molecules

reinvent_scoring -> scoring -> score_components -> standard ->
predictive_property_component.py
(*here you may need to modify the* ***_load_container*** *method to correctly load your pickled model inside Reinvent. I have already implemented loading torch pickled models.*)

Once you have modified reinvent_scoring, you can go in your reinvent environment, remove the pip installed reinvent_scoring and replace it with your modified reinvent_scoring package by doing ``pip install -e path_to_your_modified_reinvent_scoring_package``.

$ pip install -e reinvent_scoring_bradley_terry

Then you can specify in the config.json that you want to use your bradley_terry_model as a predictive_property_component to score the molecules, by specifying the container_type as "bradley_terry_model_container".

You need to put there a model_path also, so it's the path to your pre-trained bradley terry model. That will probably be a pickle file of the model you have trained on some prior dataset, the small DRD2 data that I will send you.

The smiles column contains the molecule and the target is the label (0: not active for DRD2, 1: active for DRD2).

First, you need to calculate molecular fingerprints or numerical descriptors for every smiles to get the features you will train your model on. You can do this with RDKit. Second, we may need to modify the dataset by comparing every two molecules together and have a different label (0: molecule 1 is not better than molecule 2, 1: molecule 1 is better than molecule 2). For this I have an idea, we can use an Oracle to predict the probability of every molecule to be active, then, compare every two molecules, and if probability of molecule 1 is higher than probability of molecule 2, we consider it as better and the pair (molecule 1, molecule 2) gets label 1. Otherwise, if probability of molecule 1 is lower than probability of molecule 2, then the pair (molecule 1, molecule 2) will get label 0. And on this new data, you can train a bradley terry model and we can use it as a starting point for the human component.

For the second point, to get probabilities of being active for all molecules in the dataset you can do this:

First install pyTDC by doing pip install pyTDC.

Then:

```
  import pandas as pd
  from tdc import Oracle
  oracle = Oracle(name = 'DRD2')

  small_drd2_data = pd.read_csv("small_drd2_data.csv")
proba_molecules = [oracle(smiles) for smiles in small_drd2_data.smiles]
```

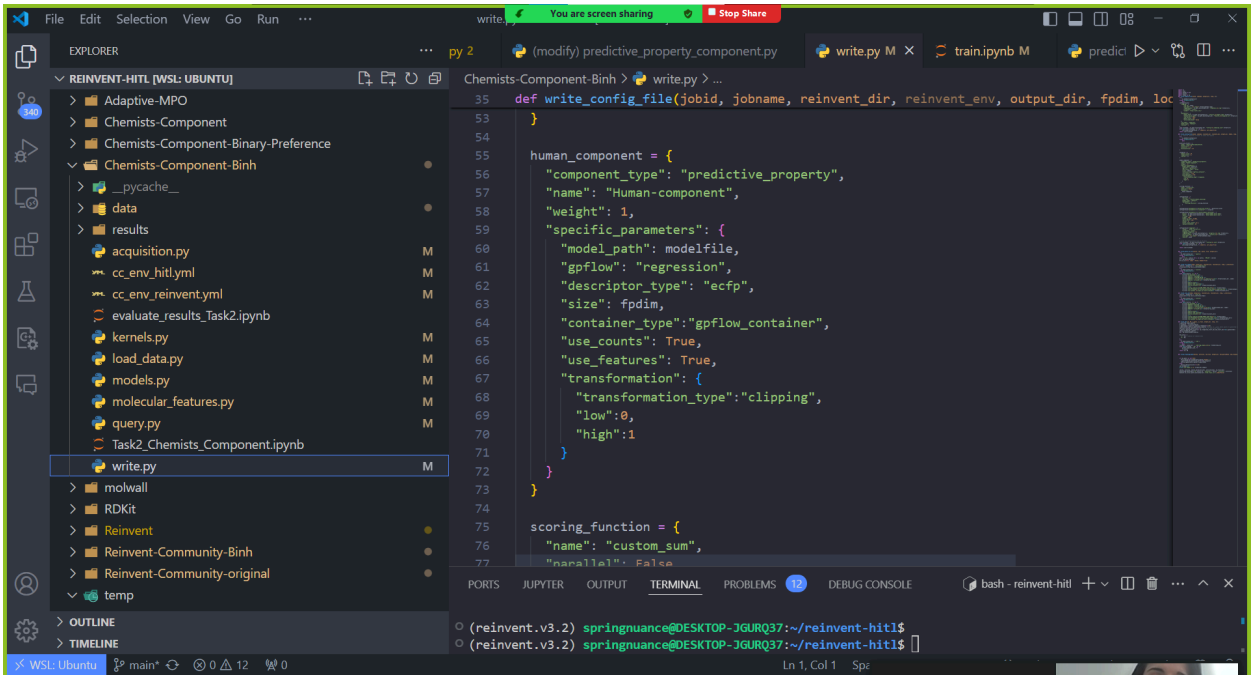
Then for example, you can take 10 pairs of molecules, compare their probabilities and create the labels

1. REINVENT produces thousand of molecules.
2. The bradley terry model forward function takes 2 molecules
3. In the init method of bradley terry model, we save all the molecules.
4. In the forward, we would compare this molecule with n molecules that is saved in the init function

bradley terry model it produces one continuous value which tells the prob of molecule 1 better than molecule 2.

==========

Note meeting 7/05/2024



In human_component in write.py in Chemist Component

We need to change the model_path as the path to the trained bradley_terry_model.pth

change “container_type” to “bradley_terry_model_container”

for transformation, change to

"transformation_type": "no_transformation"

and remove low and high as well

fpdim: fingerprint dimension. We can write 2048

remove gpflow

if we use "predictive_property", we dont need to change the file scoring_function_component_enum (dont need to add the scoring name)

modify the _load_container function in predictive_property_component.py

update the model_container.py in predictive_model to work with bradley terry model.

/home/springnuance/reinvent-hitl/reinvent-scoring/reinvent_scoring/scoring/score_components/standard/predictive_property_component.py

==========

Note meeting 20/05/2024

The container_type in the write.py file should match the name in the container_type_enum

The specific parameters keywords can contain many parameters that are specific to the trained model and can be passed to the _load_function data

Please read more about ModelContainer.py in predictive_model

REINVENT will firstly call the _load_model function at the beginning from reinvent_scoring_bradley_terry/reinvent_scoring/scoring/score_components/standard/(modify) predictive_property_component.py

After that, the _load_model construct the Bradley_Terry model from scratch, loaded the keys from .pth file and pass the loaded model to the template ModelContainer.py, which is an abstract class.

reinvent_scoring_bradley_terry/reinvent_scoring/scoring/predictive_model/model_container.py

class ModelContainer:

```
def __new__(cls, activity_model: Any, specific_parameters: Dict) -> BaseModelContainer:
    _component_specific_parameters = ComponentSpecificParametersEnum()
    _container_type = ContainerType()
```

```
        container_type =
specific_parameters.get(_component_specific_parameters.CONTAINER_TYPE,
                                        _container_type.SCIKIT_CONTAINER)

        if container_type == _container_type.BRADLEY_TERRY_CONTAINER:
            container_instance = BradleyTerryModelContainer(activity_model, specific_parameters)
```

which calls the BradleyTerryModelContainer.py

```
class BradleyTerryModelContainer(BaseModelContainer):
    def __init__(self, activity_model, specific_parameters):
        """
        :type activity_model: stan type of model object
        :type model_type: can be "classification" or "regression"
        """
        #self._model_type = model_type
        self._molecules_to_descriptors = self._load_descriptor(specific_parameters) #ecfp_counts
        self._selected_feat_idx = specific_parameters["selected_feat_idx"]
        self._activity_model = activity_model
```

the predict function calls predict_from_mols, which in turn calls the predict_from_fingerprints

==========

Note meeting 03/06/2024

Description of the workflow (actually not that small):

1. You have REINVENT that produces many molecules. Those molecules are then passed to a "scoring model". The scoring model or other scoring criteria are defined on the config.json. Basically, when you run a REINVENT experiment, you type python input.py config.json. The input.py will call the REINVENT process and all scoring model information, number of steps etc. are specified in the config.json. In your project, the molecules produced by REINVENT should be scored by a "human". We don't have access to a real human so instead, we will use a model of the human. That model is your Bradley Terry model, and you need to specify one in the config.json.
2. The Bradley Terry model needs to be trained on some data. You used the DRD2 activity data, which contains a number of molecules (SMILES) with their activity labels (1 and 0). A Bradley Terry model, by definition, compares a pair of objects and outputs a value that represents if object 1 is better than object 2. So we need to modify the DRD2 activity data so that each molecule in the dataset is compared to all others, and each pair then gets a label (1 if molecule1 is better than molecule2, 0 otherwise). Then you will have a Bradley model trained to compare 2 molecules. You will use it as "prior knowledge" of the human because then you will make it better during a REINVENT run.
3. Now you are inside REINVENT, thousands of molecules are being produced and passed to your Bradley Terry model. In your Bradley Terry container (the file you already started creating in reinvent_scoring), you have the function predict_from_fingerprints that will score the produced molecules based on the Bradley Terry model. So you have to take that list of produced molecules, and for each molecule in the list, you compare it to all others molecules in the list and output the mean of predicted preferences (for example, molecule 1 is better than molecule 2, better than molecule 3, not better than molecule 4 would give a mean of 1+1+0/3 = 0.66 and that's the score of molecule 1). Then you will have scores for all molecules and that is what you give to REINVENT so it can improve itself.

4. After running 100 steps of REINVENT (because you specified n_steps = 100 if I remember correctly), then you will query the human model aka the Bradley Terry model. So from the list of produced molecules by REINVENT, you select n_queries

pairs of molecules based on thompson sampling or uncertainty or whatever acquisition you choose, and present them to the human. Note that the acquisition needs to select pairs of 2 objects and not one object, maybe there exist some acquisition functions that deal with pairs and that you can implement directly. Then you should label those molecules, it's better to use the oracle here (the same one you used to generate the comparison labels to train the Bradley model) and that's how you label those molecules. Then you concatenate those new labelled molecules to your existing training set, and retrain the Bradley Terry model.

5. After this "human update", you save the retrained Bradley model, replace it in config.json, also replace the REINVENT model checkpoint (you put the new path to "Agent.ckpt" that is usually saved automatically after a REINVENT run ends) and then you start a new REINVENT run (that's your round 2).

I hope this gives you a clearer idea of how the process should work.

PRESENTATION notes

Settles (2009) mentioned three main sampling strategies [3]

Sampling process, also known as the query strategy, consists of selecting those instances to be labeled by the human expert.

• Membership query synthesis: The learner may request labels for any unlabeled instance in the input space, including queries that the learner generates de novo.

• Stream-based selective sampling: Also called sequential sampling, in which each unlabeled instance is drawn one at a time from the data source, and the learner must decide whether to query or discard it.

• Pool-based sampling: The entire collection of data (or a subset of it) is evaluated and ranked in order to select the best element to annotate.

Other two strategies are more interesting since they describe a well-known dilemma: exploitation vs. exploration

• Uncertainty sampling (Exploitation): It selects instances which have the least label certainty under the current trained model.

• Diversity sampling (Exploration): It selects unlabeled items that are rare or unseen in the training data to increase the picture of the problem space. Here, we found:

==========

Other aspect related with the AL process is how many new instances are labeled before training again the model

• Batch: several examples get labeled until the model is trained again

• Sequential: the system is retrained after each new element is labeled given immediate feedback to the user.

Summary of the differences between AL and IML (Interactive ML)

• AL is the basis for IML.

• The difference relies more in who has the control of the learning process and not in the interactivity of the approach. In AL the model retains the control and uses the human as an oracle; in IML there is a closer interaction between users and learning systems, so the control is shared.

• Since the interaction is closer in IML, we need to take into account Human-Computer Interaction techniques (HCI), something that is not so important in AL.

• In IML, humans perform more tasks other than labeling data in a freer and less structured process.

Data classification:

• Structured data: also known as fully-structured, is data that follows a predefined data model or schema. A typical example is relational database (or a similar structure like Excel tables or Pandas DataFrames).

• Unstructured data: is data that has no identifiable structure such as image, video and audio files, and certain types of text documents. Non-relational or NoSQL databases are the best fit for managing this data.

• Semi-structured data: is a middle category between the other two that does not conform with a data model or structure, but contains tags or markers that add semantics to that data. Examples: tagged text formats such as XML, JSON or YAML.