Improving Support for Drug Formulation

Hailey Cheng (Cheng Hei Lam)

1. Introduction

Drug formulation is the process of combining an active drug (API) with excipients to create a final medicinal product like a tablet, capsule, or injection, that delivers the drug safely and effectively. The goal is to maximize bioavailability, maintain stability over the product's shelf life, and ensure patient convenience.

Key Challenges in Drug Formulation

Solubility and Bioavailability
Many drugs are poorly water-soluble, which limits their absorption in the body. Enhancing solubility (through methods like salt formation or nanoparticles) is critical for achieving therapeutic blood levels.

Stability and Shelf Life
Drugs can degrade over time (via oxidation, hydrolysis, or polymorphic changes). Ensuring long-term chemical and physical stability often requires the use of stabilizers and specialized packaging.

Dose and Dosage Form Selection
Choosing the correct dose and dosage form is complex. The formulation must ensure the proper therapeutic concentration is delivered at the right rate while also being acceptable for patient use (e.g., considering pill size or taste).

Traditionally, formulation development involves numerous trial-and-error experiments, which can be both time-consuming and costly. However, computational tools like DeepChem offer the potential to predict formulation performance in silico. This can streamline the development process by guiding researchers, providing actionable insights such as suggesting solubility-enhancing techniques or recommending an optimal formulation strategy.

2. Relevant Experience and Interest

Background
I am a Math and Computer Science sophomore at CityUHK, with research interest in Computational Biology. Based in Hong Kong, UTC+08:00.

Experience in Machine Learning (Python) and Github

- Reproduced and benchmarked machine learning models for calcium imaging, including converting CASCADE from TensorFlow to PyTorch, with data handling, retraining, and performance evaluation in Prof. Tin Chung's Lab.

Improving Support for Drug Formulation

Hailey Cheng (Cheng Hei Lam)

- Developed a 51-parameter ODE-based model in Python to simulate the efficacy of nine drugs for an inflammatory disease in Prof. Lo Wing Cheong's Lab (Research in progress.)
- Implemented YOLOv8 for object recognition using PyTorch in CityUHK Robotics Team
- Experience in using Git & Github, in closed-source projects for multiple hackathons.

Interest
I'm particularly motivated by the chance to reduce the costly trial-and-error process in drug formulation. By DeepChem's predictive capabilities, we can identify promising drug candidates more quickly, lower development costs, and ultimately bring effective treatments to patients faster. That potential for real-world healthcare impact is what truly excites me about this project.

LinkedIn: https://www.linkedin.com/in/heilcheng/
Github: https://github.com/heilcheng

3. Work Plan

3.1 Overall Project Scope

Introduce a set of drug formulation tutorials in DeepChem that guide users step-by-step through:

1. Loading multiple relevant datasets (ESOL, Lipophilicity, Tox21, BBBP, and optionally custom dataset).
2. Featurizing these datasets (using ECFP, GraphConv, or other advanced featurizers).
3. Training baseline models (e.g., `RandomForestRegressor`, `GraphConvModel`, `dc.models.MultitaskClassifier`).
4. Evaluating performance (RMSE for regression tasks such as ESOL or Lipophilicity; ROC-AUC or confusion matrices for classification tasks such as Tox21 or BBBP).
5. Visualizing and interpreting results (scatter plots, molecule grids, confusion matrices)—using existing DeepChem or RDKit functions directly in the notebooks.

This work will help reduce the trial-and-error burden in drug formulation research and highlight how DeepChem can streamline data handling, modeling, and result interpretation.

3.2 Design and Pseudocode

Purpose

A top-to-bottom tutorial demonstrating how to load, featurize, train, evaluate, and visualize these datasets in a formulation context.

Overall Implementation

Improving Support for Drug Formulation

Hailey Cheng (Cheng Hei Lam)

- **New or Updated Dataset Loaders**
    - Adapt existing loaders for ESOL, Lipophilicity, Tox21, and BBBP if needed.
    - If any new dataset is introduced, create a specialized loader in deepchem/molnet/load_function/.
    - Each loader (e.g., load_delaney, load_lipo, load_tox21, load_bbbp) will be shown in the tutorials with instructions on how to select featurizers and splits.
- **New Tutorial Notebook(s)** (in examples/tutorials/)
    - **formulation_tutorial.ipynb** (or multiple notebooks):
        - **Introduction**: Explains drug formulation challenges, dataset context, and tutorial goals.
        - **Data Loading**: Demonstrates how to call the load_* functions.
        - **Featurization & Splitting**: Shows how to select ECFP, GraphConv, etc. and do random/scaffold splits.
        - **Model Training**: Trains baseline models for each dataset.
        - **Evaluation**: Reports RMSE or ROC-AUC.
        - **Visualization**: Provides scatter plots (pred vs. true), confusion matrices, or RDKit-based molecule grids.
        - **Interpretation**: Discusses how the results might guide formulation decisions (e.g., how predicted solubility or permeability informs excipient choice).
    - All visualization can be done directly in the notebook using standard Python libraries (e.g., matplotlib, seaborn, or RDKit's MolsToGridImage) without additional helper files.
- **Testing**
    - A new test file (e.g., tests/test_formulation_datasets.py) to ensure each loader (or any newly introduced code) works correctly. (Optional)
    - For tutorials, add a "smoke test" if needed, verifying the notebook runs without errors.

Which Files Will Be Added/Modified?

1. **`deepchem/molnet/load_function/formulation_datasets.py`**
    - *(Only if new loaders are needed or if we add special parameters to existing loaders.)*
2. **`examples/tutorials/formulation_tutorial.ipynb`**
    - Main tutorial notebook with step-by-step instructions, code, and visual outputs.

Pseudocode (for figures) can be viewed at:
https://colab.research.google.com/drive/1crNAewSYKcwNT6U-NePKnKV6sZrNvxRe?usp=sharing

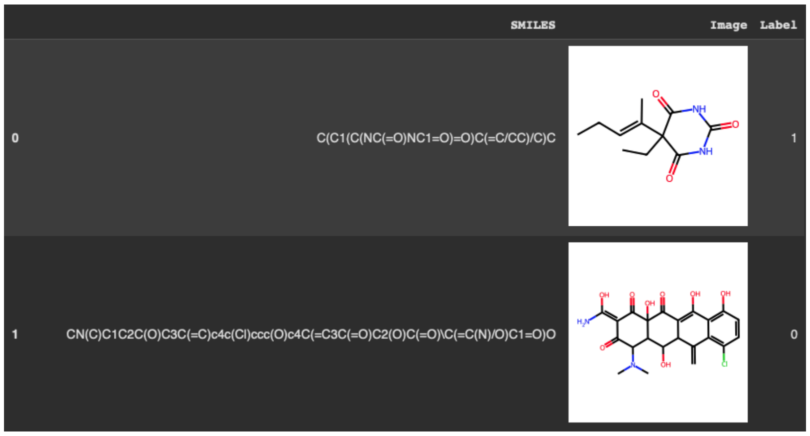
Improving Support for Drug Formulation

Hailey Cheng (Cheng Hei Lam)

- Code to load each dataset (`load_delaney`, `load_lipo`, `load_tox21`, `load_bbbp`), featurize, train a model, evaluate with RMSE/ROC-AUC.

Dataset Name	Formulation Challenge Addressed	Prediction Task	DeepChem Load Function (if applicable)
ESOL	Solubility	Regression	dc.molnet.load_delaney
Lipophilicity	Distribution	Regression	dc.molnet.load_lipo
Tox21	Toxicity	Classification	dc.molnet.load_tox21
BBBP	Permeability	Classification	dc.molnet.load_bbbp
FreeSolv	Hydration Free Energy	Regression	dc.molnet.load_freesolv

Table 1: Proposed and Suggested Datasets for Drug Formulation Tutorials



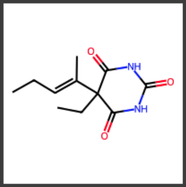
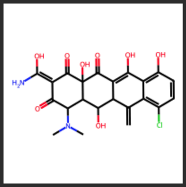
	SMILES	Image	Label
0	C(C1(C(NC(=O)NC1=O)=O)C(=C/CC)/C)C		1
1	CN(C)C1C2C(O)C3C(=C)c4c(Cl)ccc(O)c4C(=C3C(=O)C2(O)C(=O)\C(=C(N)/O)C1=O)O		0

Figure 1 (Loading BBBP): Show how to call load_bbbp(...), highlight label distribution (0 = non-permeable, 1 = permeable).

- **Visualization:** code using RDKit (`MolsToGridImage` or table with embedded PNG).

Improving Support for Drug Formulation

Hailey Cheng (Cheng Hei Lam)

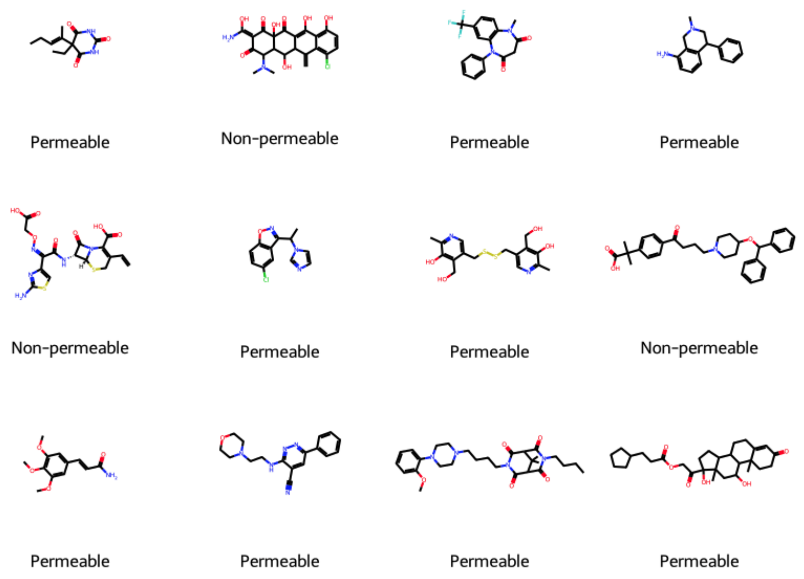
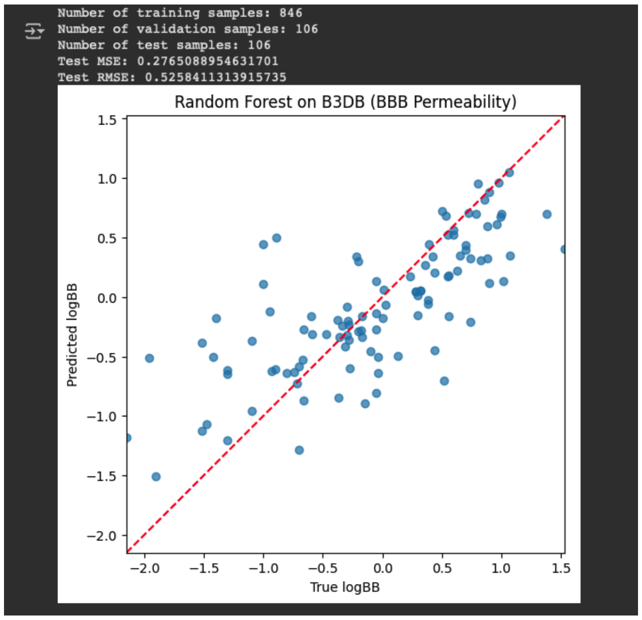


Figure 2 (Visualization of BBBP): Use RDKit's MolsToGridImage or a quick table in the notebook.

- **Interpretation**: Train baseline models (e.g., dc.models.SklearnModel for regression tasks, dc.models.torch_models.GraphConvModel or dc.models.MultitaskClassifier for classification). Compute metrics (RMSE for ESOL, ROC-AUC for Tox21, etc.). Scatter plots (pred vs. true for regression), confusion matrix or ROC curve for classification.



```
Number of training samples: 846
Number of validation samples: 106
Number of test samples: 106
Test MSE: 0.2765088954631701
Test RMSE: 0.5258411313915735
```

Figure 3 (Scatter plots): For regression tasks, produce a "predicted vs. true" scatter plot and discuss performance. For classification tasks, might show a confusion matrix or ROC curve.

3.3 Testing Plan

1. Unit Tests for tutorial code: ensure each snippet runs end-to-end.
2. Smoke Tests for dataset loaders (ESOL, Lipophilicity, etc.) verifying shapes, tasks.
3. Notebook runs in CI or local environment (Jupyter, Google Colab).
4. Cross-check predictions with known literature baselines (e.g., ESOL typical RMSE ~0.6–0.8).

3.4 Sources of Risk

1. **Data Availability**: Sets can be small or inconsistent.
2. **Overfitting**: Small datasets (ESOL, ~1k molecules) can cause inflated performance if not careful with splits.
3. **Time Constraints**: Integrating advanced GNN or multi-task might take longer than expected. Start with simpler approaches (Random Forest, ECFP). If time remains, expand to advanced models.

3.5 Milestones

1. **Milestone 1** (Community Bonding / Early June):
   - Finalize plan for tutorials. Confirm dataset approach. Prepare environment.
2. **Milestone 2** (Mid-June to Mid-July):
   - Implement ESOL, Lipophilicity, Tox21, BBBP, FreeSolv tutorials with code + visualization.
   - Provide test results.
3. **Milestone 3** (Mid-July to August):
   - Add custom data example.
   - Finalize integrated "Drug Formulation" notebook.
   - Polish documentation, code testing.

4. Timeline

- **May 8 – June 1**: **Community Bonding**
  - Join DeepChem calls, read docs, finalize design.
  - Gather any real data examples or references.
  - Possibly do small PRs to get comfortable with codebase.
- **June 2 – July 14**: **Coding Phase (First Half)**

- **Week 1–2 (June 2–June 15)**: Implement & refine ESOL tutorial with advanced visualization (tables, grids). Test end-to-end.
- **Week 3–4 (June 16–June 30)**: Implement Lipophilicity and FreeSolv tutorials. Evaluate performance, produce example plots.
- **Week 5–6 (July 1–July 14)**: Implement Tox21 and BBBP tutorials (multi-task classification). Add code to interpret results (ROC, confusion matrix). Prepare midterm demo.

- **July 14 – July 18**: **Midterm Evaluations**
  - Submit progress. Tutorials for 5 sets should be working & tested.
- **July 14 – August 25**: **Coding Phase (Second Half)**
  - **Week 7–8 (July 19–July 31)**: Add other real data example. Show CSV→featurize→model pipeline.
  - **Week 9–10 (August 1–August 14)**: Integrate all into a single "Drug Formulation with DeepChem" notebook or doc page. Add advanced tips (scaffold splits, GNN if time).
  - **Week 11–12 (August 15–August 25)**: Final polishing, code cleanup, unit tests, docstrings. Prep final deliverables.
- **August 25 – September 1**: **Final Week**
  - Submit final work product, final evaluation.
  - If time remains, address last-minute feedback from mentors.

4.2 Pull Requests (2 PR in total)

- **Tutorial Content PR:**
  Combine in a notebook
  1. Molecule visualization
  2. ESOL + Lipophilicity + BBBP + Tox21+ Custom data tutorials & plotting

     This PR will focus solely on adding the new tutorials and examples.

- **Refinements & Testing PR:**
  This PR will handle all final refinements, documentation improvements, and testing expansions.

5. Community

Office Hours: I can attend at least 2 sessions weekly to discuss progress and issues.
Interaction: Active on the Discord (ID haileychl)

6. Resources Required

Compute: Google Colab (free) should suffice for moderate dataset sizes.
Data: MoleculeNet is included in DeepChem. No large compute is required.