Project Proposal

Project Title: Modular soft actor-critic framework for synthetic pathway generation as an approach to goal-driven de novo drug design

Author: Aaron Tian

Date: 12/3/2021

Project Definition

Deep learning-based approaches to molecular design are ineffective at simultaneously generating synthesizable molecules and optimizing their properties. The objective of this project is to create a goal-driven reinforcement learning model to generate synthetic pathways to synthesizable drug compounds. The expected outcome of this project will be developed in Python using the PyTorch library.

Background

Deep learning-based approaches to molecular design are ineffective at simultaneously generating synthesizable molecules and optimizing their properties.

The objective of this project is to create a goal-driven reinforcement learning model to generate synthetic pathways of synthesizable drug compounds.

Introduction

The discovery of new drugs is a time-consuming and costly process, costing over $2.5 billion and taking 10 - 15 years on average (DiMasi et al., 2016). It is, therefore, vital for pharmaceutical companies to replace current techniques with more streamlined, automated solutions to accelerate the drug discovery process. A particular sub-task of interest within this process is lead optimization, in which the efficacy of a hit molecule acquired through screening is optimized into a drug candidate by making modifications to its chemical structure. In the past, lead optimization has been performed manually by industry professionals, but recent research shows promising results in regard to the use of deep learning for molecule generation and optimization.

The Advent of Deep Learning in Drug Discovery

The use of deep learning in chemistry-related tasks was popularized by Gilmer et al., (2017), who formulated molecules as mathematical graphs—with atoms representing nodes and bonds representing edges—and developed the Message-Passing Neural Network framework (MPNN) to learn abstract relationships between graph-structured data. MPNN achieved state of the art results on the QM9 dataset (Ramakrishnan et al., 2014), a dataset for the prediction of quantum chemical properties. The MPNN framework was particularly groundbreaking due to its independence of feature engineering, a costly process of calculating structural properties of a given molecule. Since then, various modifications have been made to improve the performance of MPNN, including XGraphBoost, which replaces the prediction layer with a gradient-boosting algorithm (Deng et al., 2021), and TrimNet, which develops a multi-head attention mechanism to reduce unnecessary parameters in the model (Li et al., 2021).

Deep Learning in Molecular Generation and Optimization

Deep learning frameworks for graph-structured data have now been extended to generative models, which are capable of goal-driven molecular design. These approaches can be divided into two categories, namely autoregressive and non-autoregressive. Autoregressive models consider molecule generation as an iterative process, starting with an empty set and repeatedly adding one atom/bond at a time based on the previous state of the compound until a terminal state is reached. Non-autoregressive methods, on the other hand, generate the entire molecular graph in a single forward pass.

The widely accepted criteria for the evaluation of these generative models are validity, novelty, and uniqueness. Validity refers to a generated compound's compliance with chemical rules, novelty refers to the variance between the set of generated molecules and the set of known molecules, and uniqueness refers to the variance between molecules within the generated set. Autoregressive models are capable of achieving high validity scores due to their ability to explicitly incorporate chemical rules; for example, Zhou et al. (2019) formulates molecule optimization as a Markov decision process and implements a deep reinforcement learning framework. The model iteratively builds molecules by selecting modifications to a base structure from a set of actions and can ensure 100% validity by automatically removing invalid modifications from the action set. Autoregressive models, however, suffer from lower scores for novelty and uniqueness due to a tendency for the outputs to converge (Xiong et al., 2021). Conversely, non-autoregressive models achieve improved novelty and uniqueness scores but often generate invalid molecules (Xiong et al., 2021).

A shared drawback of both approaches is the inability to account for chemical synthesis; Gao and Coley (2020) estimate that more compound-generating models have been developed than synthesizable molecules produced from these models. Synthetic accessibility is arguably the most important priority when designing compounds, as a structure that cannot be synthesized has no practical use on its own. Recent literature in molecule generation has pivoted toward models that can more effectively account for chemical synthesis.

Chemical Synthesis in Molecule Generation

Chemical synthesis prediction as a standalone task has been addressed using computational methods in the past. AiZynthFinder, developed by Genheden et al., (2020), uses Monte Carlo tree search to recursively search for synthetic pathways of an input molecule. The algorithm identifies full synthesis routes with reasonable accuracy, but takes 7.1 seconds on average to find a solution. Analyzing chemical compound libraries with such a tool is clearly inefficient: computation time is strictly dependent on the size of the library, and the produced results cannot be conditionally selected with respect to desired chemical properties.

Gao and Coley (2019) summarize the various approaches that can be used to increase the rate of synthesizability in generative models. *Post hoc* filtering simply evaluates the outputs after being generated and isolates the synthesizable compounds. *A priori* biasing calls for the selective design of a train dataset such that the model only trains on synthesizable molecules in the hope of biasing the algorithm toward synthesizable products. Similarly, heuristic biasing and CASP oracle biasing also attempt to bias the model by incorporating synthesizability as an additional optimization parameter using heuristic synthetic accessibility scorers and retrosynthesis tools, respectively.

Since these methods often lack consideration of chemical knowledge, a new archetype of approaches attempts to incorporate chemical synthesis as an inductive bias of the model. Bradshaw et al. (2019) propose a generative model that takes reactants from a chemical catalog and applies reaction templates to predict the products of chemical reactions between these molecules. Since a molecule proposed by the model is derived from purchasable compounds, its synthesizability is implicitly guaranteed.

Experimental Design/Research Plan Goals

List IDV, DV, standardized variable/controls, experimental/control groups, iterations, etc., process of product design. Materials List Procedure

The described framework will connect to several models and data sources:

- A **chemical catalog** for validation of reactant molecules
- A **database of known chemical reactions** to provide synthesis templates
- A deep learning-based **molecular property predictor**
- A **central generation model** which creates synthesizable molecules with respect to the tools listed above.
- A **retrosynthetic planner** for validation of output molecules

The created framework will be evaluated an account of its scalability with respect to the connected model:

IDV: The molecular property predictor that is connected to the generation model.

DV: The quality of the generated molecules, evaluated on their synthesizability and how well they optimize for given target molecular properties.

Control: hyperparameter tuning method, data sources used.

Materials:

- Programming language: Python
- Additional libraries: NumPy, MatPlotLib, PyTorch, pytorch_geometric, Pandas, rdkit, networkx
- Databases:
    - Chemical catalog: ZINC
    - Chemical reaction database: USPTO
    - Molecular property predictors:
        - TrimNet, XGraphBoost, MPNN

Procedure:

1. Develop a framework of data inputs and models, using a modular design which allows for data sources and models to be substituted for other ones.
2. Develop model
3. Use evaluation criteria above to score the model
4. Propose adjustments and repeat the process.

Risk/Safety Concerns

There are no anticipated risks or conflicts of interest associated with this project.

Data Analysis

Generated data, such as loss curves and training metrics, will be processed and analyzed graphically/statistically in Python using the matplotlib and numpy libraries, respectively.

Potential Roadblocks

1. Code-related bugs are inevitable and will be procedurally addressed in PyCharm using the breakpointing method. The machine learning library of choice, PyTorch, supports this method due to its use of eager execution.
2. Computational limitations will be addressed by migrating the code to a cloud computing service such as Amazon AWS.

References

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural Message Passing for Quantum Chemistry. *ArXiv:1704.01212 [Cs]*. http://arxiv.org/abs/1704.01212

Ramakrishnan, R., Dral, P. O., Rupp, M., & von Lilienfeld, O. A. (2014). Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data, 1*(1), 140022. https://doi.org/10.1038/sdata.2014.22

Deng, D., Chen, X., Zhang, R., Lei, Z., Wang, X., & Zhou, F. (2021). XGraphBoost: Extracting Graph Neural Network-Based Features for a Better Prediction of Molecular Properties. *Journal of Chemical Information and Modeling, 61*(6), 2697–2705. https://doi.org/10.1021/acs.jcim.0c01489

Li, P., Li, Y., Hsieh, C.-Y., Zhang, S., Liu, X., Liu, H., Song, S., & Yao, X. (2021). TrimNet: Learning molecular representation from triplet messages for biomedicine. *Briefings in Bioinformatics, 22*(4), bbaa266. https://doi.org/10.1093/bib/bbaa266

Xiong, J., Xiong, Z., Chen, K., Jiang, H., & Zheng, M. (2021). Graph neural networks for automated de novo drug design. *Drug Discovery Today, 26*(6), 1382–1393. https://doi.org/10.1016/j.drudis.2021.02.011

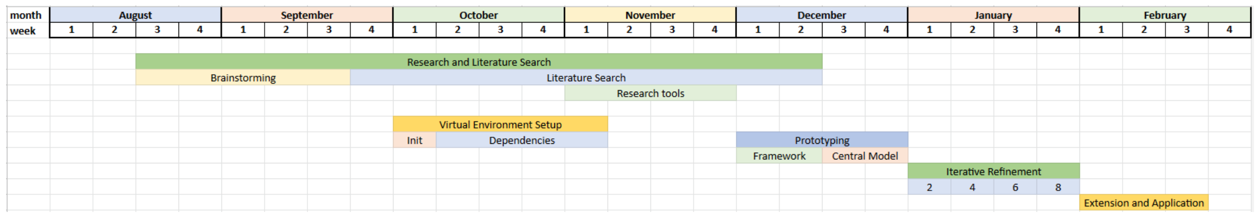
Zhou, Z., Kearnes, S., Li, L., Zare, R. N., & Riley, P. (2019). Optimization of Molecules via Deep Reinforcement Learning. *Scientific Reports, 9*(1), 10752. https://doi.org/10.1038/s41598-019-47148-x

Gao, W., & Coley, C. W. (2020). The Synthesizability of Molecules Proposed by Generative Models. *ArXiv:2002.07007 [Cs, q-Bio, Stat]*. http://arxiv.org/abs/2002.07007

Bradshaw, J., Paige, B., Kusner, M. J., Segler, M. H. S., & Hernández-Lobato, J. M. (2019). A Model to Search for Synthesizable Molecules. *ArXiv:1906.05221 [Physics, Stat]*. http://arxiv.org/abs/1906.05221

Genheden, S., Thakkar, A., Chadimová, V., Reymond, J.-L., Engkvist, O., & Bjerrum, E. (2020). AiZynthFinder: A fast, robust and flexible open-source software for retrosynthetic planning. *Journal of Cheminformatics, 12*(1), 70. https://doi.org/10.1186/s13321-020-00472-1

Timeline



Phase 1: Research

- ➢ Fill in remaining brainstorming components (5 whys, 1 mind map) by 2nd STEM meeting Sep 24, 2021
- ➢ Relevant model architectures, read by Oct 8, 2021
    - ✓ ~~Autoregressive flow models~~
    - ✓ ~~Graph attention mechanism~~
    - ✓ ~~Reinforcement learning~~
    - ✓ ~~RetroXPert~~
- ➢ 15 total papers read by October break Oct 15, 2021
- ➢ 25 total papers read by December break Dec 17, 2021
- ➢ Compile a potential list of databases from literature to incorporate into model Nov 15, 2021

Phase 2: Environment Setup

- ➢ Start initial project directory and connect to GitHub Oct 6, 2021
- ➢ Identify necessary dependencies and install in an Anaconda environment Nov 1, 2021

Phase 3: Initial Prototyping and Implementation

- ➢ Develop framework (no central model) by Dec 14, 2021 and test on dataset as baseline
- ➢ Implement and analyze 1st prototype for central generation model Jan 1, 2021

Phase 4: Iterative Refinement

- ➢ Make 2 refinements to prototype weekly, for a total of 8 iterations over the course of 1 month.

Phase 5: Extension and Application

- ➢ Package Code Feb 1, 2021
- ➢ Write GitHub documentation Feb 7, 2021
- ➢ Create user interface Feb 14, 2021