For CS252R I worked on the code for the first assignment, the transformers’ presentation and the final project. The final project was continued as part of CS91R and resulted in the ICLR GEM publication.

# CS252R First Assignment (Bottom Up Synthesis)

**Summary:** I wrote a bottom up synthesis in TypeScript powered by a small scheme interpreter (also in TS) that was able to solve synthesis problems by taking in a DSL and a set of examples as input. The DSL was not fixed and the GitHub repository shows examples with arithmetic and a string manipulation DSL. It uses expression caching for faster interpretation of ASTs, and is able to use native TS implementations of the DSL for the same purpose, but can also fallback on the interpreter to obtain a value for the entire expression from scratch when the native TS implementation is not available for a node in the AST.

Link to github repository: <https://github.com/inakineitor/fall-2023-cs252r/tree/main/pset-1>

When I submitted, I suggested a couple of ideas for improvement:

* Using the cached types from previous interpretations to perform more aggressive narrowing by type (to avoid trying invalid expressions). This would not work in recursive cases, but this BUS cannot handle recursive cases without using the interpreter, so in the non-recursive mode, this would work and provide significant speed improvements.
* Parallelizing each expression size loop by batching all the expressions to be round and distributing their execution amongst multiple threads. These would yield at the end and the results captured by the main thread. Most of the program time is spent on expression execution, so this would also provide significant speed improvements.
* Implementing the machine learning portion of BUSLE.
* Adding another search guiding factor, like the cost to run each function.
* Caching of expression results under examples

For the assignment, Tyler provided the following feedback:

* The synthesizeProgram function adheres to bottom-up enumerative synthesis and uses a weight-based strategy for terms.
* Uses the seenEvaluations set for observation equivalence, reducing redundant computations.
* Accepts availableOperations and extraPrimitives as parameters, offering some DSL-agnosticism; however, weight computation is tightly coupled within the function. This integration could potentially limit the function's adaptability to different DSLs if those DSLs require different weight computation methods, suggesting room for modularization to achieve full DSL-agnosticism.
* Lengthy and complex structure suggests a need for refactoring, particularly for the loop generating argument tuples and evaluating expressions.
* Leaves placeholders for future improvements like caching and enhanced error handling to improve efficiency and robustness.
* Makes indiscriminate use of console.log for debugging, indicating a need for a more formal logging mechanism or varying verbosity levels.

# CS252R Presentation (Attention Is All You Need)

**Summary:** I presented on the transformers paper Attention Is All You Need, explaining their inner workings in an intuitive way and providing examples of what they could be used for.

Presentation link: <https://docs.google.com/presentation/d/1i9uLQ_D9Ufk9IuJMI7pWQyRKuUNLuVSMGV6g9tGI-B4/edit?usp=sharing>

# CS91R Paper (Multi-Objective Generative AI for Designing Novel Brain-Targeting Small Molecules - ICLR GEM 2024)

**Summary:** In our study, we focus on identifying small molecule drug candidates for central nervous system (CNS) diseases that currently lack effective treatments. The blood-brain barrier (BBB) poses a significant challenge in delivering drugs to the CNS, hindering both diagnosis and treatment. Computational methods that generate BBB-permeable lead compounds in silico could be valuable tools in the CNS drug design process. However, in practical applications, BBB penetration alone is not sufficient; molecules must also perform a desired function, such as binding to a specific target or receptor in the brain, while being safe and non-toxic for human use.

To address these challenges, we employed multi-objective generative AI to synthesize drug-like, BBB-permeable small molecules with high predicted binding affinity to a disease-relevant CNS target. We specifically focused on designing molecules with predicted bioactivity against dopamine receptor D2, which is the primary target for most clinically effective antipsychotic drugs. After training several graph neural network-based property predictors, we adapted SyntheMol (Swanson et al., 2024), a recently developed Monte Carlo Tree Search-based algorithm originally designed for antibiotic discovery, to perform a multi-objective guided traversal over an easily synthesizable molecular space.

As a result, we designed a library of 26,581 novel and diverse small molecules containing hits with high predicted BBB permeability, favorable predicted safety and toxicity profiles, and the potential for straightforward synthesis and experimental validation in the wet lab. We also validated top-scoring molecules using molecular docking simulation against the D2 receptor, demonstrating predicted binding affinity comparable to risperidone, a clinically prescribed D2-targeting antipsychotic. We hope that our SyntheMol-based computational approach will enable the discovery of novel neurotherapeutics for currently intractable CNS disorders in the future.

Conference funding proposal link: <https://docs.google.com/document/d/145axx0tQuakK1I6ftDy2G24t18Tw6L8LvHiWU6hMOrM/edit?usp=sharing>

Conference report link: <https://docs.google.com/document/d/1xdtudvtkxWCzudJVf7ynB7Rkp_KNfiZaY7ZFMMZsVi0/edit>

Arxiv link to paper: <https://www.arxiv.org/abs/2407.00004>