# Meta-Reasoning Experiment

**Due:** Wednesday, Oct 15 at 11pm

**Overview:**
This assignment gives you hands on experience with using machine-learning to improve the performance of a traditional backward-chaining reasoner. I have provided you with a Jupyter notebook to demonstrate how it works. Since some of the steps in this homework will take over an hour to run, we have created a GPU environment hosted on Lehigh's MAGIC machines for you to use. You can access this environment via a web browser: http://magic01.cse.lehigh.edu/ . Details on using the environment are here: https://docs.cse.lehigh.edu/magic/. To accomplish the exercises below, you can either copy the notebook to the server and edit it or you can open a terminal on the server and run the commands there.

**Submission:**
Make a ZIP file that contains your source (.py) and any supporting files. For each KB you create, you should include a subdirectory with all of the files that were produced, including the KB itself, any loss diagrams (.png files), and the vocab.pkl, all_facts.txt, train_queries.txt, test_queries.txt, and mr_train_examples.csv files. Use Google Docs or a word processor to record the commands you executed, the resulting output (the first and last 20 lines are fine for output that is over 50 lines long) and answer any questions below. Save this as a PDF. Upload both the PDF and the ZIP file to Course Site.

**Exercises:**

Note, some steps (#5 and #6) are intended for CSE 498 students only. CSE 398 students will get extra credit if they do these steps.

1. Login to the MAGIC environment and upload the files I have provided. You can use unzip if you uploaded the zipped file. If you want to run commands in a MAGIC terminal, then issue `conda activate krclassp311` first to select the correct Python environment. If you get errors, try issuing `conda init`, closing the terminal and then running the conda activate command in a new terminal.
2. Open the kr_experiment.jpynb Jupyter notebook. Select Kernel→ Change Kernel… and choose krclassp311 from the dropdown list. Read through the notebook and run all of the code blocks. Note, if you want to run the examples in a local Python environment (or in the MAGIC terminal), you should remove the "%" and "!" from the front of commands. You do not need to include any results from this exercise in your final document.
3. Use the code to generate two random KBs of sizes 250 and 500. To keep from overwriting important files, it is recommended that you create each in its own subdirectory. For example, if you want to put the 250 statement KB in a directory named *size250*:
   **cd size250**
   **python ..\kbencoder.py --generate_kb --num_rules 250 --new_vocab --save_vocab**
4. For each random KB, train the embedding and scoring models and then compare reasoning using the scoring model to the standard reasoner.

5. **CSE 498 Students Only:** The architecture (layers, activation functions, etc.) of the scoring model are defined in the **NeuralNet** class of **nnreasoner.py.** Change this architecture in some interesting way that you think might lead to improved performance on answering queries. Note, when **evaluate.py** uses the scoring model, it creates an instance of the NeuralNet class, and then loads the learned parameters from a file (see **load_guidance()**). If you create a new class for your modified model, you'll need to write some code to load it properly.
6. **CSE 498 Students Only:** Train and evaluate your modified model on both KBs.
7. Create a table that summarizes the results of your experiments. For each combination of KB (the two different random KBs) and test condition (standard, default scoring model, and, if appropriate, the modified model from Ex. 5 and 6), it should display the mean and median nodes explored to answer the test queries, as well as the mean time.
8. In your document, reflect on what you have learned through these experiments. Did you see improvements from incorporating learning? What sorts of differences do you see in performance across the two KBs of different sizes? What sorts of experiments would you recommend to be able to draw more definitive conclusions?