LLM Hallucination Prediction

I decided to do my project on the hallucination of LLMs because it's a problem that needs to be solved to take the next steps in the field. Specifically, I built a model to predict when an LLM (open-source Tiny Llama model off of HuggingFace since I have a CPU) will hallucinate.

Because hallucination is difficult to define, I decided to focus on a specific case: generating chess moves (I would prompt the model with the sequence of moves leading to the given position, the side whose turn it was to move, and ask it to generate a good move for them). This made it possible to define hallucination as an illegal move output.

Before doing any analysis on what affects the model's propensity to hallucinate, I decided to ensure that the model was actually taking the current position into account (wanted to make sure because I was using Tiny Llama and not a better LLM given computation constraints). To do this I used rejection sampling (with 1000 random positions from my games) to calculate the probability of outputting a legal move in a position and compared it to the probability that a randomly generated chess move (given by the starting square and ending square of the piece moved, so there are 64 choose 2 or 4032 different possibilities) was legal. The probability of the LLM generating a legal move was around 0.684 which was far higher than random generation (so low that it was practically 0), so this indicates that the LLM is taking into account the current position and that further analysis is useful.

After determining this, I decided to predict when the model would hallucinate using logistic regression. The parameters I ended up using were the mean and standard deviation of the logprobs of the model, the amount of pieces on the board, the amount of moves already played, the side whose turn it was to move, the amount of possible legal moves, and a variable indicating whether the side to move was in check, checkmate, or stalemate. I chose these parameters because there was a significant difference in the probability that the model generated a legal

move given some measure relating to the parameter versus its complement. These conditional probabilities were calculated using rejection sampling on the training data before the logistic regression was performed and the conditional probabilities are shown below.

Feature	Statistics
mean_logprob (legal)	mean $= -0.460$, std $= 0.108$
mean_logprob (illegal)	mean $= -0.433$, std $= 0.126$
std_logprob (legal)	mean $= 0.448$, std $= 0.083$
std_logprob (illegal)	mean $= 0.462$, std $= 0.101$
num_pieces (legal)	mean $= 21.1$
num_pieces (illegal)	mean $= 11.8$
num_moves_so_far (legal)	mean $= 33.5$
num_moves_so_far (illegal)	mean $= 95.9$
num_legal_moves (legal)	mean $= 27.060$
num_legal_moves (illegal)	mean $= 14.828$
$P(\text{legal} \mid \text{white to move})$	0.284
$P(\text{legal} \mid \text{black to move})$	0.387
$P(\text{legal} \mid \text{terminal\_status} = 1)$	0.000
$P(\text{legal} \mid \text{terminal\_status} = 0)$	0.370

As we can see from the data most of the features have a clear correlation with the legality of the move generated by the LLM. Especially notable, is the terminal status (check, checkmate, or stalemate) which indicates that the model doesn't know how to handle those (always gives illegal moves). Even though the logprobs had no noticeable difference in the means, I used Bayes' rule to find these probabilities which show that a higher logprob indicates that the model will generate a legal move and models have higher logprobs for legal moves.

Conditional Probability	Symbol	Estimate
Probability of high confidence	$P(\text{high confidence})$	0.633
Probability of legal move	$P(\text{legal})$	0.684
Legal given high confidence	$P(\text{legal} \mid \text{high confidence})$	0.920
High confidence given legal	$P(\text{high confidence} \mid \text{legal})$	0.852

LLM Hallucination Prediction

After choosing parameters, in the logistic regression, I used the same approach as used in the problem sets with a learning rate of 0.01 (found that this performed best after trials with higher and lower values) and 2000 steps. After being trained on the training data (many positions generated from my games with the sequences of moves being created by the python-chess library), the model's parameters were given as the following:

Feature	**Weight**
mean_logprob	−0.0658
std_logprob	−0.0621
num_pieces	0.1292
num_legal_moves	0.0104
num_moves_so_far	−0.1700
side_to_move_num	−0.4350
terminal_status	−0.0767
Intercept	0.0921

This was interesting to me, because it showed that the logprobs are not a great indicator of hallucination, which was counterintuitive since it seems like model confidence should definitely inform likelihood of making things up. Another thing that was interesting, but mostly expected was that as more moves were made and more pieces disappeared, the model would be more likely to hallucinate, and it was also interesting that the number of legal moves possible only had a small effect on how likely the model was to choose one of those. Finally, for some reason I really can't explain (especially since the dataset should be balanced because every game has an equal amount of positions where it's black's turn and white's turn, the model was much more likely to generate a legal move when it was black to move in a position.

After running the logistic regression, on the test data the regression performed with an accuracy of 0.779 which indicates that the regression model is actually picking up on patterns and that LLM hallucination may be reasonably predicted.