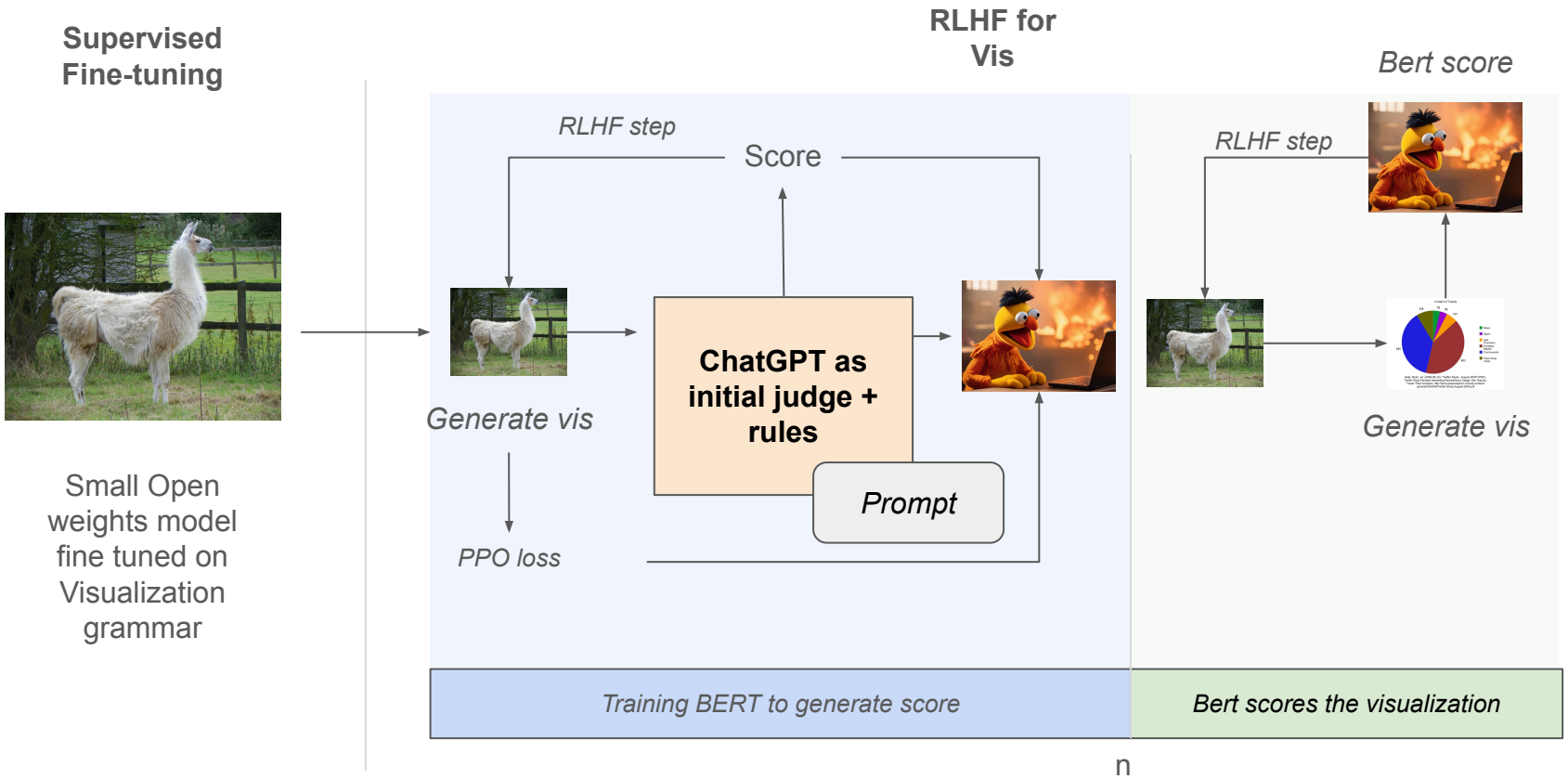
Adding chatgpt as a judge in the initial learning process



Lamma 3.0 Prompt Engineering

Solution 1: Simple Prompt

Instruction:
Bar chart x axis product name y axis how many product name , rank by the Y-axis in desc .
Input:
[('product_id', 'numeric'), ('product_type_code', 'categorical'), ('product_name', 'categorical'), ('product_price', 'categorical')]

Simple Prompt:	did compile: 193 / 200 data correct: 148 / 200 x correct: 25 / 200 y correct: 45 / 200
Chain of Thought:	did compile: 169 / 200 data correct: 127 / 200 x correct: 20 / 200 y correct: 45 / 200

Solution 2: Chain of Thought

Instruction:
Bar chart x axis product name y axis how many product name , rank by the Y-axis in desc .
Input:
[('product_id', 'numeric'), ('product_type_code', 'categorical'), ('product_name', 'categorical'), ('product_price', 'categorical')]
Thought Process:
1. **Understand the data**: Identify numerical and categorical columns from the input.
2. **Determine the chart type**: Choose an appropriate mark (`bar`, `line`, `point`, `arc`) based on the instruction and data type.
3. **Define encoding**: Assign x-axis, y-axis, aggregation function (if applicable), and any color mapping.
4. **Identify transformations**: Determine whether filtering, binning, grouping, sorting, or top-k selection is required.\n 5. **Flatten into a Vega-Zero specification**: Convert the reasoning into the keyword-based format required by Vega-Zero.

Chat GPT implementation

- Using chatgpt we can try and gain more insight into how the value and policy loss are affecting the model training.
- During the training process I changed the rewards that BERT was getting to learn from to rewards that chatGPT was outputting based on the prompt I engineered for it
- This looks very promising, however the baseline needs to be fixed before any solution can be presented

ChatGPT prompt, this is within the initial 200 iterations

{"messages": [{"role": "system", "content": "You are a reward generator for PPO Learning that outputs a reward between 0 and 1. Your input will be the model's prediction, the corresponding ground truth, the discrete reward that was generated, and some corresponding PPO Metrics that came out of that discrete reward. Your job is to output a reward that is better than the discrete reward given."},

{"role": "user", "content": "PREDICTION: {*PPO Model Prediction*}\nGROUND TRUTH: {*Corresponding Ground Truth*}\nDISCRETE REWARD: {*simple discrete reward*}\nPOLICY LOSS: {*Corresponding policy loss from discrete reward*}\nVALUE LOSS: {*corresponding value loss*}"}]}

Discrete vs BERT vs chat-gpt

Discrete Baseline

did compile: 329 / 400
data correct: 209 / 400
x correct: 52 / 400
y correct: 89 / 400

SON (BERT trained with discrete reward only)

did compile: 384 / 400
data correct: 268 / 400
x correct: 51 / 400
y correct: 94 / 400

BERT trained with Chat-GPT Modified rewards

did compile: 382 / 400
data correct: 281 / 400
x correct: 49 / 400
y correct: 84 / 400

- Good news:
  - 14% increase in outputs compiling
  - 19% increase in chosen data being correct
- Bad news:
  - X and Y column prediction stays almost exactly the same