Peyton Anton Rapo

Project Proposal

AdaptMark: An Adaptive Benchmark for Chart Reasoning Models

- **Research Problem:**
  - The research problem I am investigating is whether it is possible to create an evaluation dataset for chart reasoning models that is guaranteed to not have been seen during the training of the model. The motivation behind this is CharXiv and other papers such as Benchmark Self-Evolving have shown that tweaks to the evaluation data can lead to massive performance drops in these models.
- **Key References:**
  - Paper 1: Benchmark Self-Evolving: A Multi-Agent Framework for Dynamic LLM Evaluation
    - They expanded existing benchmarks by
      - creating alternate / more complex questions based on original context
      - making perturbations to the contexts of original questions
      - probing LLM's sub-abilities for solving different problems
  - Paper 2: EvoChart: A Benchmark and a Self-Training Approach Towards Real-World Chart Understanding
    - They set up a synthetic chart generation model that iterates over a chart trying to tweak it gradually until it reaches a certain threshold of quality
  - Paper 3: ChartLlama: A Multimodal LLM for Chart Understanding and Generation
    - They generate synthetic charts by first generating fake chart data. With this they then generate the chart visual via code using that data. Finally, then generate the instruction data. All of this is via an LLM.
  - Paper 4: CharXiv: Charting Gaps in Realistic Chart Understanding in Multimodal LLMs
    - They tested the robustness of closed and open source models by swapping charts and questions within some benchmark datasets with similar charts and questions that should still be applicable.
  - Paper 5: Automatic Generation of Benchmarks and Reliable LLM Judgment for Code Tasks
    - This paper talks about how to do automatic generation of a benchmark, but in the context of code generation.
  - Paper 6: LiveBench: A Challenging, Contamination-Free LLM Benchmark
    - They use new questions from current chart and question sources to facilitate the changing benchmark. They also don't use an LLM to evaluate the results.
  - Paper 7: Cheating Automatic LLM Benchmarks: Null Models Achieve High Win Rates

        - They show that LLMs as judges can be attacked by just having constant outputs which can end up having the model score way higher than it should.
- **Expected Contribution:**
    - There have been a few works to try to generate synthetic chart data to use as benchmarks since by definition the model will not have seen them, such as ChartLlama, but they tend to suffer from a lack of coverage. My goal is to blend the ideas of Benchmark Self-Evolving, EvoChart, and ChartLlama to generate code that creates the charts from the original dataset and then from there modify it in subtle ways such that the chart will appear mostly the same, thus increasing the coverage of the dataset.
    - I expect to contribute this code which takes in a dataset and will modify it such that it creates an evaluation dataset you can benchmark off of which has not been seen by the model. I will also contribute my evaluation of the models tested from CharXiv using my synthetic dataset.
- **Dataset:**
    - The datasets I will use as my base datasets will be subsets of DVQA, FigureQA, and ChartQA from MathVista as is done in CharXiv
- **Evaluation:**
    - I am going to evaluate my synthetic dataset by seeing if I can replicate the model performance dips of those evaluated in CharXiv. Specifically, some subset of Mini-Gemini (MGM), InternVL-XComposer2 (IXC2), InternVLXComposer2 4KHD (IXC2 4KHD), InternVL-Chat V1.5, SPHINX V2, LLaVA 1.6, and IDEFICS 2. I will also evaluate the performance of the same proprietary models they chose as well. Then I will run my evaluation a few times to make sure that it is consistently generating datasets that present good contamination-free coverage. If I have time, I may also try to train one of these smaller models on some of the generated charts to guarantee that even if the model is able to see one generated chart, the number of possible chart perturbations is large enough that it shouldn't matter.
- **Timeline:**
    - Nov 13th:
        - Have an experimental version of the chart generation code working.
    - Nov 20th:
        - Finalize the chart generation code and start to work on substituting this into the evaluation of models from CharXiv.
    - Nov 27th:
        - Finish up testing the models from CharXiv and try training a model with this dataset and begin to evaluate it.
    - Dec 4th:
        - Have everything done in terms of analysis and experimentation. Spend remaining time working on finalizing the write up.
    - Dec 11th?:
        - Turn everything in.