SCFP Benchmark: Characterizing and Predicting LLM Self-Correction Failures

This repository contains the official dataset and code for the paper: **"Characterizing and Predicting the Failure Modes of Intrinsic LLM Self-Correction."**

Our work introduces a taxonomy of Large Language Model (LLM) self-correction failures, a new benchmark (SCFP v1.0) to study them, and a meta-model that can predict the likelihood and type of these failures in advance.

Repository Structure

```
SCFP-benchmark/
├── benchmark/
│   └── SCFP_v1.0.csv       # The annotated benchmark dataset.
├── scripts/
│   └── pipeline.py         # Python script for data generation and meta-model training.
├── .gitignore
└── README.md
```

- **/benchmark**: Contains the full SCFP v1.0 benchmark in CSV format.
- **/scripts**: Contains the Python pipeline for reproducing our data generation and training the failure predictor model.

The SCFP v1.0 Benchmark

The core of this repository is the **Self-Correction Failure Prediction (SCFP) v1.0** benchmark, located in benchmark/SCFP_v1.0.csv.

Dataset Schema

Each row in the CSV represents a single self-correction attempt by an LLM and is structured as follows:

Column	Type	Description
problem_id	string	A unique identifier for the source problem.
source_dataset	string	The original benchmark the problem was sourced from

		(e.g., 'GSM8K', 'StrategyQA').
problem_text	string	The full text of the problem presented to the LLM.
initial_response	string	The model's first attempt to solve the problem, including its chain-of-thought reasoning.
self_critique	string	The model's self-generated critique after being prompted to review its initial response.
final_response	string	The model's final, corrected answer generated after the critique.
is_correct	int	Binary label: 1 if the final_response is correct, 0 otherwise.
failure_mode	string	The annotated failure mode if is_correct is 0. One of: Justification Hallucination, Confidence Miscalibration, Bias Amplification, Over-correction, Reasoning Myopia, or N/A.
generating_model	string	The LLM used to generate the response triplet (e.g., 'GPT-4o', 'Claude-3.7-Sonnet').

Usage

1. Environment Setup

First, clone the repository and install the required Python packages.

```
git clone
[https://github.com/shahed-aut/SCFP-benchmark.git](https://github.com/shahed-aut/SCFP-benchmark.git)
cd SCFP-benchmark
pip install -r requirements.txt
```

(Note: A requirements.txt file should be created containing pandas, torch, scikit-learn, and transformers.)

2. Training the Failure Predictor Meta-Model

You can train the failure prediction model using the provided benchmark data.

```
python scripts/pipeline.py --mode train \
  --data_path benchmark/SCFP_v1.0.csv \
  --model_name "microsoft/deberta-v3-base" \
  --output_dir ./meta_model_output
```

This will:

1. Load the SCFP_v1.0.csv dataset.
2. Preprocess the text data.
3. Fine-tune the specified transformer model on the binary prediction task (Success/Failure).
4. Save the trained model and evaluation results to the --output_dir.

3. Generating New Self-Correction Data

The pipeline.py script also contains a placeholder function to demonstrate how new data can be generated. You will need to add your own LLM API logic to this function.

```
# In scripts/pipeline.py, update the `generate_correction_triplet` function
# with your own API calls to services like OpenAI, Anthropic, etc.
```

Once configured, you can run the data generation mode:

```
python scripts/pipeline.py --mode generate \
  --source_problems_path path/to/your/problems.csv \
  --output_path ./newly_generated_data.csv
```

Citation

If you use this benchmark, code, or the findings from our paper in your research, please cite our work:

```
@article{Author2025Characterizing,
  title   = {Characterizing and Predicting the Failure Modes of Intrinsic {LLM} Self-Correction},
  author  = {Shahed [Your Last Name]},
  journal = {Journal of Neural Networks},
  year    = {2025},
  note    = {In submission}
}
```

License