Chain-of-Thought (CoT) Prompting Engine Guide

1. Prompting Techniques

There are three primary methods for implementing CoT prompting, each with its own advantages.

2.1. Few-Shot CoT

This is the standard approach where you provide the model with a few examples (demonstrations) that include a question, a step-by-step reasoning process (the chain of thought), and the final answer.

When to Use: Use this method for complex tasks where the reasoning structure is consistent and providing diverse, high-quality examples can significantly guide the model. This is the most powerful method but requires manual effort to create the demonstrations.

Example Prompt Structure:

```
Q: [Question 1]
A: [Step-by-step reasoning for Question 1]. The answer is [Answer 1].

Q: [Question 2]
A: [Step-by-step reasoning for Question 2]. The answer is [Answer 2].

Q: [New Question]
A:
```

The file `demos/multiarith_manual` provides a practical example of the JSON structure for these hand-crafted demonstrations.

2.2. Zero-Shot CoT

A surprisingly effective and simple method that requires no examples. By appending the phrase **"Let's think step by step"** to the end of a question, the model is triggered to generate a reasoning chain before giving the final answer.

When to Use: This is an excellent starting point for any reasoning task. It's highly effective for its simplicity and is particularly useful when you don't have time to create few-shot examples.

Example Prompt Structure:

□
```
Q: [New Question]
A: Let's think step by step.
```
□

The `api.py` script in the repository shows how this is implemented by setting a `cot_trigger` argument. The Jupyter notebooks (`try_cot.ipynb` and `try_cot_colab.ipynb`) demonstrate its application and output.

2.3. Automatic CoT (Auto-CoT)

Auto-CoT is an advanced technique designed to automate the creation of diverse and effective demonstrations for Few-Shot CoT, eliminating the manual effort. As detailed in the project's `README.md`, it works in two main stages.

Stage 1: Question Clustering

- The system takes a dataset of questions and groups them into several clusters based on semantic similarity.

Stage 2: Demonstration Sampling

- It selects a representative question from each cluster.
- It then uses **Zero-Shot CoT** to automatically generate a reasoning chain for each selected question.

This process, detailed in `run_demo.py`, ensures that the examples are both diverse (by sampling from different clusters) and accurate, creating a robust set of demonstrations for the model to learn from. The output of this process can be seen in the `demos/multiarith_auto` file.

When to Use: Use Auto-CoT when you need the high performance of Few-Shot CoT on a large dataset of questions but want to avoid the time-consuming and potentially suboptimal process of manually writing demonstrations.

---

3. Implementation in the Repository

The provided repository contains a full implementation of these techniques.

- **`api.py`**: A core file that defines the `cot` function, which can be called with different methods: `"zero_shot"`, `"zero_shot_cot"`, `"manual_cot"`, and `"auto_cot"`.

- **`run_inference.py`**: The main script for running experiments. It loads a dataset, constructs prompts based on the chosen method, and generates answers.
- **`run_demo.py`**: This script implements the Auto-CoT process by clustering questions and generating demonstrations.
- **`try_cot.ipynb`**: A Jupyter Notebook that provides a quick and clear way to test and compare the outputs of each CoT method.

To get started, refer to the `README.md` and the `try_cot_colab.ipynb` for a guided walkthrough.

4. References

- Amazon Science Repo on CoT
- CoT Example