# Adaptive Temperature for Mathematical Reasoning in LLMs - Colab Implementation

This repository contains two Google Colab notebooks implementing an inference-time adaptive temperature scaling mechanism for improving mathematical reasoning in Large Language Models (LLMs).

## Notebooks

### 1. Small-Scale Evaluation (n=50)

- Uses 50 samples from GSM8K dataset
- Useful for quick experimentation and parameter tuning
- Shows better performance with adaptive temperature in some runs

### 2. Large-Scale Evaluation (n=200)

- Uses 200 samples from GSM8K dataset
- Provides more robust evaluation metrics
- Current implementation shows baseline outperforming adaptive temperature

## Getting Started

### Quick Setup

Run these commands in either notebook:

```
!pip install datasets transformers torch tqdm
!huggingface-cli login
```

### Requirements

- Google Colab (GPU runtime recommended)
- Hugging Face account and access token
- Access to required models

## Hyperparameters

Both notebooks use this configuration with different sample sizes:

```python
class Config(NamedTuple):
    model_name: str = "google/gemma-2-2b-it"  # Model to use
    entropy_threshold: float = 0.3    # Triggers adaptive
    temperature
    poly_coeffs: Tuple[float, ...] = (
        -1.791, 4.917, -2.3, 0.481, -0.037
    )                                  # Temperature control
    max_new_tokens: int = 500         # Max generation length
    max_tokens: int = 2048            # Total token limit
    top_p: float = 0.9               # Nucleus sampling
    top_k: int = 40                  # Top-k sampling
    num_samples: int = 50 or 200     # Varies by notebook
    min_beta: float = 0.5            # Min inverse temperature
```

### Key Parameters

- **entropy_threshold**: Controls when adaptive scaling activates ($H > \theta$)
- **poly_coeffs**: Define temperature adaptation curve: $\beta(H) = \sum_{i=0}^{4} a_i H^i$
- **min_beta**: Sets minimum scaling factor ($\beta_{min}$)

## Running the Notebooks

1. Choose the appropriate notebook based on your needs:
    - Use n=50 for quick experiments and parameter tuning
    - Use n=200 for more thorough evaluation

2. Execute setup cells 3. Log in to Hugging Face when prompted 4. Run main experiment 5. Check results in output

## Output

Both notebooks generate:

- Solutions using baseline and adaptive methods
- Performance comparisons
- Token-level statistics
- Saved results in JSON format

## Known Results

- **n=50 Notebook**:
  - Shows promising results for adaptive temperature
  - Useful for parameter exploration
- **n=200 Notebook**:
  - Currently shows baseline outperforming adaptive temperature
  - Provides more statistically significant results

## Troubleshooting

If you encounter issues:

- Verify Hugging Face authentication
- Enable GPU runtime
- Confirm package installation
- Check model access permissions

## Notes

- Token metrics appear at execution end
- Results vary by model and dataset
- Uses GSM8K dataset for evaluation
- See code comments for detailed explanations
- Performance differences between sample sizes suggest need for further investigation