Whitepaper: Customizing DeepSeek R1 for Scientific Discovery

Alexander Migdal Overleaf GPT

January 26, 2025

Abstract

This whitepaper outlines a comprehensive plan to develop a domain-specific large language model (LLM) based on Hugging Face's DeepSeek R1. The goal is to create a specialized AI system capable of parsing, reasoning, and interacting with scientific content in fields such as turbulence, number theory, and quantum physics. By fine-tuning the model on domain-specific datasets derived from recursive references in academic publications, this project aims to build an intelligent tool for scientific discovery. The proposed workflow includes data preprocessing, model fine-tuning, resource requirements, and cost estimates.

Contents

1 Introduction

Large Language Models (LLMs) have revolutionized natural language understanding and reasoning, but their generic nature often limits their utility in niche scientific fields. This project aims to customize DeepSeek R1 into a domain-specific LLM capable of:

- Parsing and reasoning about turbulence equations and their duality to string theory.
- Extracting and processing content from scientific papers.
- Answering domain-specific queries in turbulence, number theory, and quantum physics.

2 Dataset Preparation

2.1 Data Sources

The model will be trained on:

- Papers cited in publications by Alexander Migdal.
- Recursive references from the initial papers, focusing on turbulence and number theory.
- Authoritative works in number theory, quantum physics, and related fields.

2.2 Automated Data Collection

The dataset will be curated using APIs and libraries:

- **Semantic Scholar API**: For retrieving citation networks.
- **arXiv API**: To gather preprints in turbulence and related domains.
- **CrossRef API**: To access metadata and references from journals.

2.3 Data Preprocessing

Data preprocessing will involve:

- Text extraction from PDFs using `PyPDF2` or `Grobid`.
- Cleaning equations and tokenizing mathematical expressions.
- Formatting data for fine-tuning (prompt-response pairs).

3 Fine-Tuning DeepSeek R1

3.1 Model Overview

DeepSeek R1 is an open-source LLM optimized for tunability and symbolic reasoning. Fine-tuning will adapt the pre-trained model to the scientific domain.

3.2 Training Pipeline

The fine-tuning process includes:

1. Tokenizing the curated dataset using Hugging Face's `transformers` library.
2. Training the model on GPU instances using transfer learning.
3. Evaluating performance on domain-specific tasks.

4 Infrastructure Requirements

4.1 Compute Resources

The fine-tuning process requires:

- **Cloud Providers**: AWS, Google Cloud, or Paperspace.
- **GPU Requirements**: NVIDIA A100 or equivalent.
- **Estimated Costs**: \$200–\$800 for small models; \$10,000–\$50,000 for larger-scale fine-tuning.

4.2 Free and Academic Resources

- **Google Colab Pro**: Low-cost GPU access for small-scale experiments.
- **NVIDIA Academic Program**: Free GPU grants for research.

5 Implementation Plan

5.1 Project Structure

The GitHub repository will include:

- **data/**: Curated datasets (papers, tokenized files).
- **scripts/**: Python scripts for preprocessing and training.
- **models/**: Fine-tuned DeepSeek R1 models.
- **notebooks/**: Jupyter notebooks for experiments and visualizations.
- **README.md**: Documentation for contributors.

5.2 Development Steps

1. Automate data collection using APIs.
2. Preprocess the dataset for fine-tuning.
3. Fine-tune DeepSeek R1 on scientific datasets.
4. Deploy the fine-tuned model for interaction on GitHub.

6 Example Python Scripts

6.1 Data Collection Script

Listing 1: Data Collection with Semantic Scholar API

```
import requests

def fetch_references(paper_id, api_key):
    url = f"https://api.semanticscholar.org/graph/v1/paper/{paper_id}/references"
    headers = {"x-api-key": api_key}
    response = requests.get(url, headers=headers)
    if response.status_code == 200:
        return response.json()
    else:
        print("Error fetching references:", response.status_code)
        return None

paper_id = "arXiv:2301.12345"
api_key = "YOUR_API_KEY"
references = fetch_references(paper_id, api_key)
print(references)
```

6.2 Fine-Tuning Script

Listing 2: Fine-Tuning DeepSeek R1 with Hugging Face

```
from transformers import AutoTokenizer, AutoModelForCausalLM, Trainer, TrainingArgu

# Load pre-trained model and tokenizer
model_name = "huggingface/open-r1"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name)

# Prepare dataset
train_data = [{"input": "What is turbulence?", "output": "Turbulence is..."}, ...]

# Tokenize dataset
def tokenize_function(example):
    return tokenizer(example["input"], padding="max_length", truncation=True)

# Define training arguments
training_args = TrainingArguments(
    output_dir="./models",
    evaluation_strategy="steps",
    learning_rate=2e-5,
    per_device_train_batch_size=8,
    num_train_epochs=3,
    weight_decay=0.01,
)

# Trainer
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_data,
    tokenizer=tokenizer,
```

```
)
trainer.train()
```

7 Conclusion

This whitepaper provides a roadmap for developing a domain-specific LLM using DeepSeek R1. By combining curated datasets, advanced fine-tuning techniques, and efficient infrastructure, this project aims to build an intelligent assistant for scientific discovery.

References