NLP - LORA EĞİTİMİ PROJESİ YOL HARİTASI

Bu projede amacımız, pretrain ve fine-tune edilmiş hazır bir LLM (Instruct) modelini alıp, ona daha iyi kod yazmayı öğretmektir. Bunu **LoRA** adı verilen verimli bir yöntemle yapacağız.

Kaynaklar ve İndirme Linkleri

Projeye başlamadan önce aşağıdaki kaynakları hazır edin:

- **Temel Model (Base Model):** Qwen2.5-Coder-1.5B-Instruct [1]
- **Veri Seti 1 (DEEP):** CodeGen-Deep-5K [2]
- **Veri Seti 2 (DIVERSE):** CodeGen-Diverse-5K [3]
- **Rehber Doküman:** HuggingFace LoRA Training Guide [4]

Adım Adım Yapılacaklar Listesi

1. Adım: Modeli Tanıyın

- Temel modeli indirin ve kurun.
- Ona birkaç Python sorusu sorun (inference yapın). Modelin eğitim öncesindeki cevap kalitesini not edin. (Örn: Prepare a code to sum two input value.)

2. Adım: Verileri İnceleyin

- **Önemli:** Eğitimi solution (sadece kod) alanını kullanarak yapacaksınız.
- *Ekstra:* Deneyim kazanmak isterseniz output (düşünme adımları + kod) alanını da deneyebilirsiniz ama zorunlu değil.

3. Adım: Eğitimi Başlatın

İki ayrı eğitim yapacaksınız. İkisi de aynı temel modelden sıfırdan başlayacak:

Eğitim 1: Temel Model + DEEP Dataset

Eğitim 2: Temel Model + DIVERSE Dataset

4. Adım: Checkpoint Seçimi ve Test

Eğitim sırasında modeliniz belirli adımlarda (step) kaydedilecektir. Hangi kaydın en iyisi olduğunu anlamak için:

- Kaydedilen checkpoint'i yükleyin.
- Modele, eğitim verisinde **olmayan** yeni bir soru (prompt) gönderin.

- Ürettiği cevabın kalitesini gözle kontrol edin. En mantıklı ve çalışan kodu üreten checkpoint'i "Final Model" olarak seçin.

⚙️ Teknik Ayarlar (Önerilen Başlangıç)

⚠️ Uyarı: Parametre ayarlarının "mükemmel" kombinasyonunu bulmak kolay olmayabilir. İlk denemenizde başarısız olursanız endişelenmeyin; deneme-yanılma (trial-and-error) ile birkaç farklı ayar denemeniz gerekebilir.

- **Epoch:** 3 (Öneri)
- **Rank (r):** 16, 32 veya 64
- **Alpha:** Rank değerinin 2 katı (Örn: r=16 ise alpha=32)
- **Context Length:** 1024 token (Solution alanı için)
- **System Prompt** (Zorunlu): "You are an expert Python programmer. Please read the problem carefully before writing any Python code."

Hata Çözümleri (OOM - Out of Memory)

Ekran kartı hafızası yetmezse sırasıyla şunları uygulayın:

1. Flash Attention 2 aktif edin.
2. Gradient Checkpointing aktif edin.
3. Context Length'i 800'e düşürün.
4. Hala hata alıyorsanız modeli **8-bit** modunda yükleyin (Son çare).

BENCHMARK GÖREVİ

Eğittiğiniz modellerin gerçekte ne kadar başarılı olduğunu görmek için onları **41 adet AtCoder sorusu** ile test edeceğiz. Amacımız en yüksek puanı alan "Checkpoint"i bulmak.

1. Hazırlık: Test Ortamını Kurun

Önce testleri yapacak programı bilgisayarınıza indirin. Terminali açın ve şu komutları sırasıyla girin:

```
git clone https://github.com/naholav/CodeGen.git
cd CodeGen
pip install -r requirements.txt
```

---

2. Klasör Düzeni (⚠️ En Kritik Adım)

Testin çalışması için bilgisayarınızdaki klasör yapısı **birebir** aşağıdaki gibi olmalıdır. Eğittiğiniz modelleri bu yapının içine taşıyın:

```
CodeGen/
+-- models/
    +-- deep_instruction/                    <-- 1. egitim klasorunuz
    |   +-- checkpoints/
    |        +-- checkpoint-step-300-epoch-1/ <-- LoRA dosyalari burada
    |       +-- checkpoint-step-400-epoch-2/
    |       +-- checkpoint-step-500-epoch-2/
    |       +-- ...
    |
    +-- diverse_instruction/                 <-- 2. egitim klasorunuz
        +-- checkpoints/
            +-- checkpoint-step-300-epoch-1/
            +--    checkpoint-
            step-400-epoch-2/
            +-- ...
```

Dikkat: Checkpoint klasörlerinizin isminde mutlaka step ve epoch kelimeleri geçmelidir. (Örn: checkpoint-step-500-epoch-2)

---

3. Küçük Bir Kod Ayarı

İndirdiğiniz CodeGen klasöründeki **livecodebench_eval.py** dosyasında 70. satırı bulun: Buradaki isimlerin klasör isimlerinizle aynı olduğundan emin olun.

```
# Varsayilan: model_types: tuple = ("deep_think", "deep_instruction",
"diverse_think", "diverse_instruction

# Sizin klasor isimlerinize gore degistirin: model_types: tuple =
("deep_instruction", "diverse_instruction")
```

Not: Buradaki isimler models/ altindaki klasor isimleriyle **birebir ayni** olmali!

Aynı dosyada **100. Satırı Bulun:** Eğer eğitim sırasında System Prompt'u değiştirdiyseniz burayı da güncelleyin. Değiştirmediyseniz dokunmayın.

---

4. Testi Başlatma

Artık modelleri sınav yapabiliriz. Terminalde şu komutları çalıştırın:

DEEP için: `python livecodebench_eval.py --model_type deep_instruction --platform atcoder --difficulty easy`

DIVERSE için: `python livecodebench_eval.py --model_type diverse_instruction --platform atcoder --difficulty easy`

---

5. Sonuçları Okuma ve Raporlama

Test bittiğinde sonuçlar results/livecodebench/summary.json dosyasına yazılacaktır.

1. Dosyayı açın.
2. Her checkpoint için **Pass@1** değerine bakın.
3. En yüksek yüzdeye sahip olan checkpoint sizin **kazanan modelinizdir.**

Örnek Çıktı:

```
deep_instruction_checkpoint-step-300-epoch-1    19.5%
deep_instruction_checkpoint-step-400-epoch-2    22.0%
deep_instruction_checkpoint-step-500-epoch-2    26.8% <-- En iyi
deep_instruction_checkpoint-step-600-epoch-3    24.4%
```

🆘 Sık Yapılan Hatalar

- **"Checkpoint directory not found":** models klasörünün içindeki klasör isimleriniz yanlış. Yukarıdaki şemayı tekrar kontrol edin.
- **"No checkpoints found":** Checkpoint klasör isimleriniz checkpoint-step-... formatında değil. İsimleri düzeltin.

Teslim Edilecekler ve GitHub Reposu

Projenin en önemli çıktısı hazırlayacağınız **GitHub Reposu** ve **Proje Raporu**dur.

1. GitHub Reposu (Zorunlu)

Aşağıdaki dosyaları içeren düzenli bir repo oluşturmalısınız:

- train.py (Eğitim kodlarınız)
- eval.py (Test kodlarınız)
- requirements.txt (Gerekli kütüphaneler)
- Eğitim Logları (Loss değerlerini içeren dosyalar)

2. Proje Raporu ve Grafikler (Zorunlu)

Hazırlayacağınız raporda şu analiz kesinlikle yer almalı:

- **Loss Grafiği:** Train, Validation ve Test veri setlerindeki **Loss** (hata) değerlerinin değişimini gösteren bir grafik hazırlayın.
- **Veri Sıklığı:** Bu grafikte değerler her 20 adımda (step) bir işaretlenmiş olmalıdır.

- **Yorumlama:** Grafiğe bakarak modelin öğrenip öğrenmediğini, ezberleyip ezberlemediğini (overfitting) yorumlayın.
- **En iyi Checkpoint:** Eğitimde farklı adımlarda kaydettiğiniz modellerin (checkpoints) arasından en iyi olanı Benchmarka göre nasıl belirlediğinizi aşağıdaki tablo ile gösterin:

Model	En Iyi Checkpoint	Pass@1	Cozulen Soru
Base Model	-	?	?27?/41
Deep_instruction	step-120	?%	?33?/41
Diverse_instruction	step-180	?%	?35?/41