Kod Üreten ve Öğreten Yapay Zeka

Proje Ekibi

- **Hasan Basri Gedik**
- **Toprak Henaz**

Rapor Özeti

Bu rapor, kod üretme ve öğretme yeteneklerine sahip bir yapay zeka modeli oluşturmak için yürüttüğümüz fine-tuning çalışmalarını özetlemektedir. Projemizde iki farklı yaklaşım denedik: önce yerel kaynaklarla bir fine-tuning çalışması, ardından bulut tabanlı Predibase platformu kullanarak daha büyük bir model üzerinde eğitim. Sonuç olarak, CodeLlama-70B-Instruct modelini Magicoder-OSS-Instruct-75K veri seti üzerinde eğitmenin en verimli yaklaşım olduğunu belirledik.

Kullanılan Veri Seti

Projemizde ana veri seti olarak Magicoder-OSS-Instruct-75K kullanılmaktadır. Bu veri seti 75.000 kod örneği içerir ve programlama problemleri, başlangıç kodları ve çözümlerden oluşmaktadır. Farklı programlama dillerinde kod örnekleri barındıran bu veri seti, kod üretme modeli eğitimi için idealdir.

Denenen Yaklaşımlar

1. Yöntem: Google Colab ile Fine-Tuning

İlk yaklaşımımızda, daha küçük bir CodeLlama modelini (7B) Google Colab ortamında eğitmeye çalıştık. Bu yöntemde bellek kullanımını optimize etmek için LoRA ve 4-bit kuantizasyon gibi teknikler kullandık:

```
# 1. Yöntem - LoRA ve Kuantizasyon Konfigürasyonu
config = {
    "model_name": "codellama/CodeLlama-7b-hf",
    "lora_r": 16,
    "use_4bit": True,
```

```
    "bnb_4bit_compute_dtype": "float16",
    "max_seq_length": 512,
    "batch_size": 4,
    "gradient_accumulation_steps": 4,
    "learning_rate": 2e-4,
    "num_train_epochs": 3
}

# LoRA Konfigürasyonu
peft_config = LoraConfig(
    task_type=TaskType.CAUSAL_LM,
    r=config["lora_r"],
    target_modules=["q_proj", "v_proj", "k_proj", "o_proj", "gate_proj", "up_pr
oj", "down_proj"],
    bias="none",
)
```

2. Yöntem: Predibase ile Bulut Tabanlı Fine-Tuning

İkinci yaklaşımımızda, Predibase platformu üzerinden çok daha büyük bir modeli (CodeLlama-70B-Instruct) eğitmeyi tercih ettik. Bu yöntem API tabanlı bir yaklaşım kullanarak altyapı zorluklarını ortadan kaldırdı:

```
# 2. Yöntem - Predibase API Konfigürasyonu
from predibase import Predibase

# Predibase istemcisini başlat
pb = Predibase(api_token=my_api_token)
deployment = pb.deployments.get(deployment_ref="codellama-70b-instru
ct")

# Fine-tuning için adapter oluştur
adapter = pb.finetuning.jobs.create(
    config={
        "base_model": deployment.name,
        "epochs": 5,
        "learning_rate": 0.0002,
    },
```

```
    dataset=dataset,
    repo_ref=repo_ref
)
```

İki Yaklaşımın Karşılaştırması

Özellik	1. Yöntem (Google Colab)	2. Yöntem (Predibase)
Model Büyüklüğü	7B parametre	70B parametre
Teknik Karmaşıklık	Yüksek	Düşük
Donanım Gereksinimleri	Colab GPU (yetersiz)	Bulut kaynakları
Eğitim Performansı	Bellek sınırlamaları nedeniyle başarısız	Başarılı
Uygulama Kolaylığı	Zor, çok sayıda parametre ayarlaması gerekli	Kolay, API tabanlı yaklaşım
Maliyet	Colab kullanım limitleri	Hizmet maliyeti

Sonuç

İlk yaklaşımımız, Google Colab'in donanım kısıtlamaları nedeniyle başarısız oldu. Özellikle bellek yetersizliği, optimizasyon zorlukları ve eğitim kararsızlıkları, küçük modellerde bile etkili bir fine-tuning yapmamızı engelledi.

İkinci yaklaşımda Predibase platformuna geçiş yaparak, çok daha büyük bir model olan CodeLlama-70B-Instruct üzerinde çalışabilme imkanı elde ettik. Bu yaklaşım, altyapı zorluklarını ortadan kaldırdı ve teknik karmaşıklığı azalttı.

Predibase ve CodeLlama-70B-Instruct modeli kullanarak, Magicoder veri seti ile eğitim yapmamız, kod üretme ve öğretme konusunda daha yüksek kaliteli sonuçlar elde etmemizi sağladı. Projemizin şu anki aşamasında, bu yaklaşımla fine-tuning sürecimizi sürdürmekteyiz.

Sonuç olarak, yapay zeka ile kod eğitimi alanında gelişmiş bir model oluşturmak için, büyük modellerin bulut tabanlı platformlar üzerinden eğitilmesinin daha verimli ve etkili bir yaklaşım olduğu sonucuna vardık.