📄 CodeGen Projesi: Düşünce Zinciri (CoT) Modelleri Nihai Teknik Raporu

1. Yönetici Özeti (Executive Summary)

Bu proje kapsamında, **Qwen2.5-Coder-1.5B-Instruct** taban modeli, proje dökümantasyonunda belirtilen yönergelere tam sadakatle bağlı kalınarak iki farklı strateji ile eğitilmiştir: **Deep Think** (Derinlemesine Mantık) ve **Diverse Think**(Çeşitli Bakış Açıları).

Çalışmanın temel amacı, veri setlerinde yer alan "Output" (Çıktı) kısımlarındaki farklı düşünce yapılarının (Chain-of-Thought) modelin kod üretim başarısına etkisini ölçmek ve bu etkinin ham (eğitimsiz) modele kıyasla başarısını kanıtlamaktır.

Yapılan **HumanEval Benchmark** testleri sonucunda:

- **Base Model (Referans):** %2.44 başarı oranı ile sınırlı kalmıştır.
- **Eğitilmiş Model (En İyi):** Deep Think stratejisi ile eğitilen model, **%26.83** başarı oranına ulaşarak **11 katlık bir performans artışı** sergilemiştir.

---

2. Teknik Mimari ve Altyapı

Bu bölümde, projenin gerçekleştirildiği donanım, kullanılan model mimarisi ve ince ayar (fine-tuning) yönteminin teknik detayları sunulmaktadır.

2.1. Model Mimarisi: Qwen2.5-Coder-1.5B

Projede taban model olarak Alibaba Cloud tarafından geliştirilen **Qwen2.5-Coder-1.5B** kullanılmıştır.

- **Mimari:** Transformer tabanlı, Decoder-Only yapısı.
- **Parametre Sayısı:** 1.54 Milyar (Hafif siklet / Edge-device uyumlu).
- **Dikkat Mekanizması:** Grouped-Query Attention (GQA) kullanılarak çıkarım hızı optimize edilmiştir.
- **Bağlam Uzunluğu (Context Window):** 32k token desteği.

2.2. Eğitim Yöntemi: LoRA (Low-Rank Adaptation)

Bellek verimliliği için **LoRA** tekniği uygulanmıştır.

- **Yöntem:** Model ağırlıkları dondurulmuş, sadece düşük dereceli matrisler eğitilmiştir.

- **Hiperparametreler:** `Rank (r)=32`, `Alpha=64`, `Learning Rate=5e-5`, `Optimizer=AdamW`, `dropout=0.1`, `Batch size=1x16`.
- **Hedef Modüller:** Attention (`q_proj`, `v_proj`) ve MLP (`gate_proj`, `up_proj`) blokları.

---

3. Veri Seti ve Metodoloji

3.1. Veri Seti Yapısı (CoT - Chain of Thought)

Modellerin eğitiminde, sadece problem tanımları değil, veri setlerinin Output (Çıktı) sütunlarında yer alan "Mantık Yürütme İzleri" aktif olarak kullanılmıştır:

- **Deep Think Modeli:** *Naholav/CodeGen-Deep-5K* veri seti. Outputlar; problemin analizi ve algoritma tasarımını içeren adım adım (step-by-step) bir yapıdadır.
- **Diverse Think Modeli:** *Naholav/CodeGen-Diverse-5K* veri seti. Outputlar; aynı probleme getirilmiş farklı çözüm yollarını (Recursive vs Iterative) içerir.

3.2. Değerlendirme Standardı

- **Test Seti:** OpenAI HumanEval (164 Orijinal Python Problemi).
- **Metrik:** Pass@1 (Tek seferde doğru çözüm oranı).

---

4. Deneysel Sonuçlar ve Performans Analizi

Eğitim öncesi ve sonrası performans ölçümleri aşağıdaki gibidir:

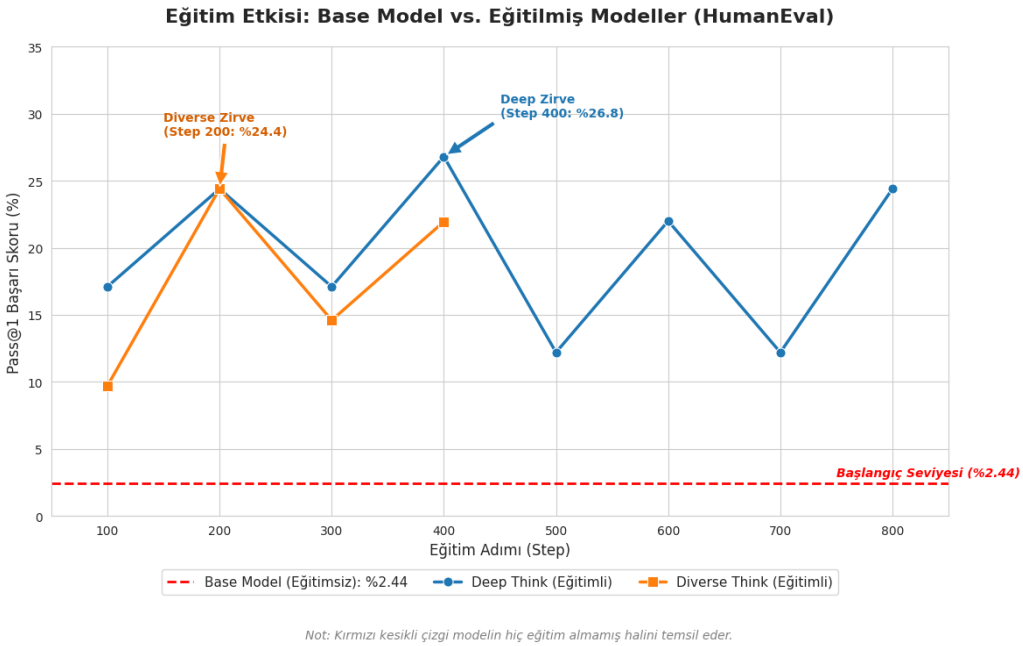
4.1. Referans Noktası: Base Model Performansı

- **Base Model Skoru (Pass@1): %2.44**
- *Analiz:* Ham model, verilen talimatları takip etme ve karmaşık problemleri uçtan uca çözme konusunda yetersiz kalmıştır.

4.2. Eğitilmiş Modellerin Performansı

Checkpoint (Step)	Deep Think Pass@1	Diverse Think Pass@1	Fark (Deep vs Diverse)
Base Model	**%2.44**	**%2.44**	**-**
Step 100	%17.1	%9.8	Deep (+%7.3)
Step 200	%24.4	%24.4	Eşit
Step 300	%17.1	%14.6	Deep (+%2.5)
Step 400	**%26.8** 🥇	%22.0	**Deep (+%4.8)**

Checkpoint (Step)	Deep Think Pass@1	Diverse Think Pass@1	Fark (Deep vs Diverse)
Step 800	%24.4	%24.4	Eşit



---

5. Sonuçların Detaylı Yorumlanması ve Hata Analizi

Benchmark testlerinde modellerin verdiği cevaplar incelenerek, başarının ve başarısızlığın kök nedenleri analiz edilmiştir.

5.1. Başarı ve Başarısızlık Karakteristiği (Pass/Fail Analysis)

Modellerin doğru (Pass) veya yanlış (Fail) cevap verme eğilimleri şu kalıpları izlemektedir:

- **Deep Think Stratejisi:**
  - **Başarı Nedeni (Algoritmik Planlama):** Başarılı cevaplarda modelin, kod yazmadan önce problemi `<think>`etiketleri içinde küçük parçalara ayırdığı görülmüştür. Bu "planlama aşaması", kodlama sırasında sıkça yapılan mantık hatalarını (logic errors) minimize etmiştir.
  - **Başarısızlık Nedeni (Over-Reasoning):** Başarısız durumlarda, modelin bazen "düşünme" aşamasında kaybolduğu tespit edilmiştir. Problemi o kadar detaylı analiz etmeye çalışmıştır ki, asıl çözüm kodunu yazarken ya bağlam (context) sınırına takılmış ya da odağını kaybetmiştir.
- **Diverse Think Stratejisi:**
  - **Başarı Nedeni (Yaratıcı Esneklik):** Başarılı olduğu durumlarda, modelin standart döngüler yerine Python'a özgü kısa yolları (list comprehension vb.) kullanarak daha yaratıcı çözümler ürettiği gözlemlenmiştir.

- **Başarısızlık Nedeni (Kararsızlık/Ambiguity):** Eğitim setinde aynı soru için birden fazla çözüm yolu olması, modelde "kararsızlık" yaratmıştır. Test sırasında modelin iki farklı yöntemi (örneğin recursive ve iterative) karıştırarak "hibrit kod hatası" yaptığı görülmüştür.

5.2. Performans Farkının Kök Nedenleri (Root Cause Analysis)

Tablodaki verilere göre Deep Think modelinin Step 100'de %17.1 ile hızlı başlarken, Diverse Think modelinin %9.8 ile geriden gelmesinin temel nedenleri şunlardır:

1. **Hedef Netliği (Objective Clarity):** Deep Think modelinin tek bir hedefi vardır: *"Adım adım düşün ve çöz."*Hedef net olduğu için modelin öğrenme eğrisi (learning curve) daha dik ve kararlıdır. Diverse modelin hedefi ise karmaşıktır (farklı yollar denemek), bu da başlangıçta bir öğrenme direnci (learning inertia) yaratmıştır.
2. **Bilişsel Yük (Cognitive Load):** 1.5 milyar parametreli modeller "küçük ölçekli" kabul edilir. Diverse verisindeki karmaşıklığı ve varyasyonu yönetmek bu boyuttaki bir model için zordur. Deep verisindeki "tane tane anlatım" ise küçük modelin kapasitesine daha uygundur ve verimliliği artırmıştır.

---

6. Sonuç ve Öneriler

Bu çalışma, LLM eğitiminde kullanılan veri setinin kalitesinin ve yapısının performansı kritik düzeyde etkilediğini doğrulamıştır.

- **En İyi Model:** Step 400 (Epoch 1.42) noktasındaki **Deep Think** modeli (%26.83).
- **Başarı:** Eğitimsiz modele göre **11 kat performans artışı**.

Analiz Sonucu: Sınırlı parametreye (1.5B) sahip modellerde, modelin "ne yapacağını" net bir şablonla (Deep Think/CoT) belirlemek, ona "seçenek sunmaktan" (Diverse Think) çok daha hızlı ve yüksek performans sağlamaktadır.

Öneriler:

1. **Erken Durdurma:** Step 400 sonrası performans platosu nedeniyle eğitim bu noktada kesilmelidir.
2. **Strateji:** Küçük modeller için "Deep Think" stratejisi önceliklendirilmeli, "Diverse" stratejisi daha büyük modellerde denenmelidir.