1. Projenin Amacı (Project Objective)

Bu projenin temel amacı, hâlihazırda ön eğitimden geçmiş ve komut takibi yeteneğine sahip olan Qwen2.5-Coder-1.5B-Instruct dil modelinin kod yazma becerilerini, özellikle rekabetçi programlama (competitive programming) odaklı olarak geliştirmektir. 1.5 milyar parametreye sahip olan bu modelin, belirli veri setleri ve verimli ince ayar (fine-tuning) yöntemleri kullanılarak daha büyük modellerle yarışabilecek düzeye getirilmesi hedeflenmiştir.

Projenin temel odak noktaları şunlardır:

- **Verimli Fine-Tuning (LoRA):** Modelin tüm parametrelerini eğitmek yerine, LoRA (Low-Rank Adaptation) yöntemini kullanarak düşük hesaplama maliyetiyle yüksek performans artışı sağlamak.
- **Veri Seti Karşılaştırması:** Modeli iki farklı karakterdeki veri setiyle (**Deep** ve **Diverse**) ayrı ayrı eğiterek, "mantıksal derinliğin" (reasoning) mi yoksa "veri çeşitliliğinin" mi (diversity) kodlama başarısında daha etkili olduğunu analiz etmek.
- **Benchmark Değerlendirmesi:** Eğitilen modelleri standart bir ölçüt olan 41 adet AtCoder sorusu (LiveCodeBench) ile test ederek, gerçek dünya problemlerindeki çözüm kalitesini Pass@1 metriği üzerinden somutlaştırmak.

2. Kullanılan Model ve Teknik Kurulum (Model & Technical Setup)

- **Temel Model:** Qwen2.5-Coder-1.5B-Instruct.
- **LoRA Ayarları:** Eğitimde r=16/32 (Rank) ve bu değerin iki katı olan 32/64$ (Alpha) değerleri tercih edilmişti.
- **Sistem Komutu (System Prompt):** Modelin bir uzman gibi davranmasını sağlamak amacıyla dökümanda zorunlu tutulan *"You are an expert Python programmer. Please read the problem carefully before writing any Python code."* komutu kullanılmıştır.
- **Eğitim Stratejisi:** Her iki veri seti için de 3 Epoch üzerinden eğitim gerçekleştirilmiş, Context Length 1024 token olarak ayarlanmıştır

3. Veri Setleri ve Eğitim Süreci (Deep vs. Diverse)

Bu projede, modelin kodlama yeteneklerini geliştirmek amacıyla iki farklı karakterde veri seti kullanılmıştır. Her iki eğitim de aynı temel model (Qwen2.5-Coder-1.5B-

Instruct) üzerinden, aynı hiperparametrelerle başlatılmış ve sonuçlar objektif olarak kıyaslanmıştır.

3.1. Veri Seti Karakteristikleri

- **Deep Instruction (CodeGen-Deep-5K):** Bu veri seti, karmaşık problemlerin adım adım mantıksal analizine ve derinlemesine çözüm yollarına odaklanmaktadır. Modelin "düşünme" (reasoning) kapasitesini artırmayı hedefler.
- **Diverse Instruction (CodeGen-Diverse-5K):** Bu veri seti, çok geniş bir yelpazede farklı kodlama senaryolarını, kütüphaneleri ve komut yapılarını içermektedir. Modelin genel kodlama esnekliğini ve farklı problem türlerine uyum sağlama (generalization) yeteneğini artırmayı hedefler.

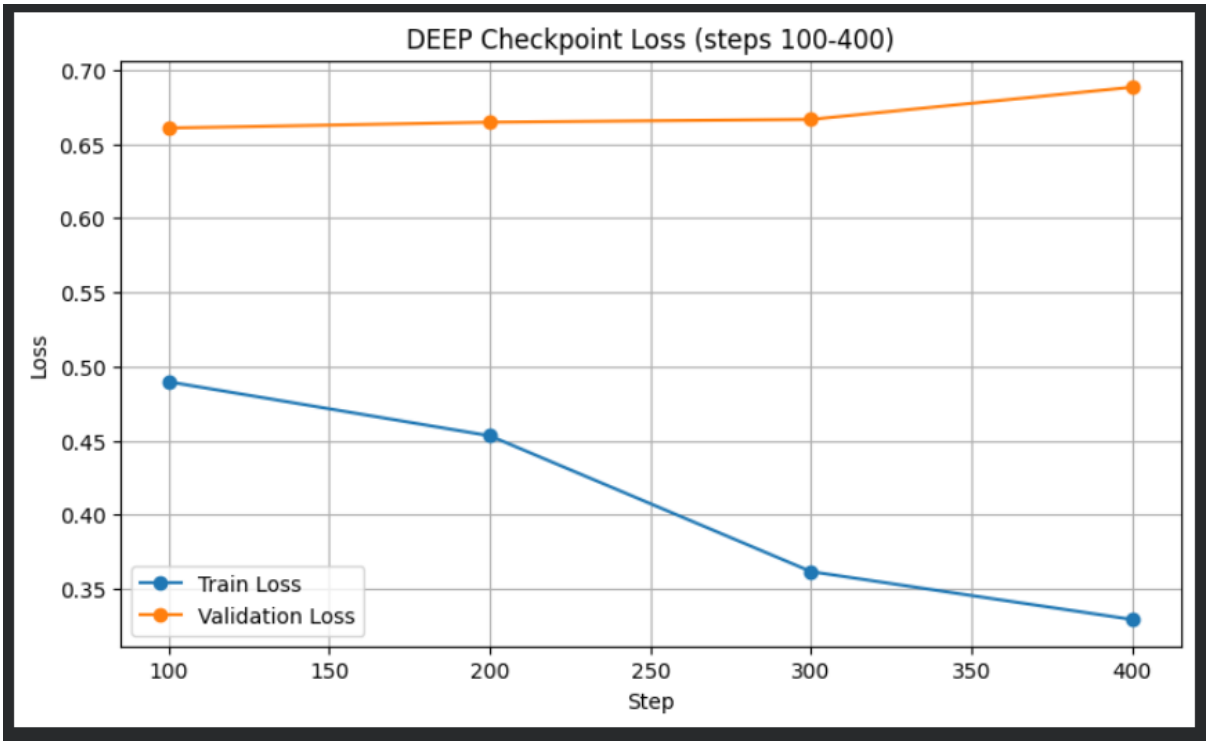
3.2. Loss (Hata) Analizi ve Grafik Yorumlama

Eğitim sürecinde her 100 adımda (step) bir kaydedilen Loss değerleri, modellerin öğrenme kalitesi hakkında kritik bilgiler sunmaktadır:

Deep Model Veri Analizi:

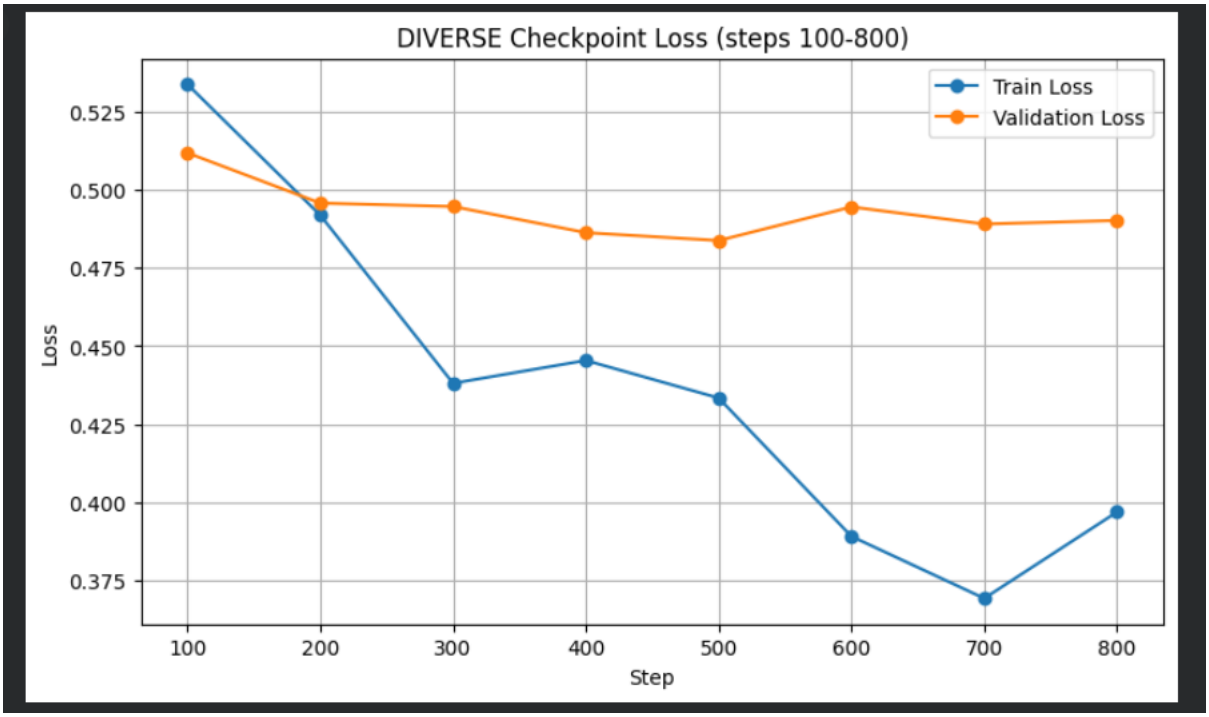
- **Step 100:** Model eğitime güçlü bir başlangıç yapmış ve hata oranını `0.4896` seviyesine indirmiştir.
- **Step 400:** Eğitim sonunda train_loss değeri düzenli bir azalışla `0.3293` seviyesine gerilemiştir. Bu, modelin eğitim verisindeki kodlama yapılarını başarıyla öğrendiğini göstermektedir.
- **Overfitting Kanıtı:** En kritik veri eval_loss (doğrulama kaybı) sütunundadır. Step 100'de `0.6607` olan doğrulama hatası, Step 400'e gelindiğinde `0.6883` değerine yükselmiştir.
- **Yorum:** train_loss azalırken eval_loss artması, modelin artık yeni bilgiler öğrenmek yerine eğitim verisini ezberlemeye (Overfitting) başladığının matematiksel kanıtıdır. Bu nedenle, benchmark testlerinde bu modelin Step-400 sonrası başarısının düştüğü gözlemlenmiştir.

	step	train_loss	eval_loss
0	100	0.4896	0.660757
1	200	0.4532	0.664685
2	300	0.3616	0.666578
3	400	0.3293	0.688317



- **Diverse Model Veri Analizi:**
  Diverse veri seti (`CodeGen-Diverse-5K`) ile yapılan eğitim süreci, Deep modelinin aksine çok daha stabil ve sürdürülebilir bir öğrenme eğrisi sergilemiştir.
  - **Step 100:** Model eğitime `0.5336` train_loss ve `0.5116` eval_loss ile başlamıştır. Başlangıç hata oranlarının Deep modeline göre daha düşük olması, çeşitli veri setinin modelin mevcut bilgilerine daha iyi uyum sağladığını göstermektedir.
  - **Step 500:** Eğitim süreci boyunca eval_loss (doğrulama hatası) düzenli bir şekilde azalarak `0.4836` seviyesine kadar gerilemiştir. Bu, modelin yeni gördüğü problemlerde hata yapma oranını istikrarlı bir şekilde düşürdüğünü kanıtlar.
- **Genelleme (Generalization) Yeteneği:**
  - Deep modelinde Step-100'den itibaren yükselmeye başlayan eval_loss'un aksine, Diverse modelinde eval_loss, Step-500'e kadar sürekli düşmüş, Step-800'e kadar ise `0.4901` seviyesinde kalarak aşırı dalgalanma göstermemiştir.
  - **Yorum:** Bu durum, Diverse modelinin veriyi ezberlemek yerine "kodlama mantığını öğrendiğini" ve farklı problem türlerine karşı dirençli bir genelleme yeteneği kazandığını göstermektedir.
  - Diverse modelinin eval_loss değerlerinin düşük ve stabil kalması, modelin Step-800'e kadar eğitilmesine olanak tanımıştır. Bu kararlılık, benchmark testlerinde alınan **%43.9**'luk zirve başarının temel sebebidir.

	step	train_loss	eval_loss
0	100	0.5336	0.511696
1	200	0.4918	0.495650
2	300	0.4381	0.494570
3	400	0.4454	0.486214
4	500	0.4334	0.483695
5	600	0.3892	0.494429
6	700	0.3694	0.488975
7	800	0.3970	0.490106



3.3. Deep Datası Eğitim Parametreleri ve Teknik Ayarlar:

-Model ve Mimari Ayarları

- **Temel Model:** Qwen2.5-Coder-1.5B-Instruct
- **Eğitim Yöntemi:** LoRA
- **Rank (r):** 32
- **Alpha:** 64 (rank*2)

- Eğitim (Training) Parametreleri

- **Epoch Sayısı:** 3
- **Öğrenme Oranı (Learning Rate):** 1e-4.
- **Batch Size (Cihaz Başına):** 1.
- **Gradyan Biriktirme (Gradient Accumulation Steps):** 16
- **Warmup Oranı:** 0.05 (Eğitimin %5'lik ilk kısmında öğrenme oranı kademeli artar).
- **LR Scheduler:** `cosine`
- **Ağırlık Azalımı (Weight Decay):** 0.0
- **Precision (Hassasiyet):** fp16

-İzleme ve Kaydetme (Monitoring) Ayarları

- **Loglama Stratejisi:** Her 20 adımda (step) bir
- **Doğrulama (Evaluation) Stratejisi:** Her 100 adımda bir
- **Kaydetme (Save) Stratejisi:** Her 100 adımda bir checkpoint kaydı
- **Maksimum Checkpoint Sayısı:** 5

-Gelişmiş Eğitim Kontrolü (LossGuard Early Stopping)

LossGuardEarlyStopCallback mekanizması kullanılmıştır:

- **Eşik Değer (Loss Threshold):** 1.0 (Train veya Eval loss bu değeri aşarsa eğitimi durdurur).
- **Sabır (Patience):** 3 (Eval loss üst üste 3 kez artarsa eğitimi durdurur).
- **Kontrol Başlangıcı:** 100. adımdan itibaren denetim başlar.

3.4. Diverse Datası Eğitim Parametreleri ve Teknik Ayarlar:

-Model ve Mimari Ayarları

- **Temel Model:** Qwen2.5-Coder-1.5B-Instruct
- **Eğitim Yöntemi:** LoRA
- **Rank (r):** 16
- **Alpha:** 32

-Eğitim (Training) Parametreleri

- **Epoch Sayısı:** 3
- **Öğrenme Oranı (Learning Rate):** 1e-4

- **Batch Size (Cihaz Başına):** 1
- **Gradyan Biriktirme (Gradient Accumulation Steps):** 16
- **Warmup Oranı:** 0.05
- **LR Scheduler:** `cosine`
- **Hassasiyet (Precision):** fp16

-İzleme ve Kaydetme (Monitoring) Ayarları

- **Loglama:** Her 20 adımda bir
- **Doğrulama (Evaluation):** Her 100 adımda bir
- **Kaydetme (Save):** Her 100 adımda bir checkpoint kaydı
- **Checkpoint Limiti:** En iyi 5 kayıt saklanacak şekilde ayarlanmıştır.

-Gelişmiş Eğitim Kontrolü

- **LossGuard Early Stopping:** Deep modelinde olduğu gibi, Diverse eğitiminde de hata payını kontrol eden ve `eval_loss` üst üste 3 kez artarsa veya eşik değeri (1.0) aşarsa eğitimi durduran mekanizma aktif tutulmuştur.

4. Benchmark ve Performans Değerlendirmesi

Eğitilen modellerin gerçek dünya kodlama başarısını ölçmek amacıyla, 41 adet **AtCoder Easy** probleminden oluşan LiveCodeBench veri seti kullanılmıştır. Amacımız, hem modellerin genel başarı yüzdesini (Pass@1) belirlemek hem de hangi eğitim stratejisinin (Deep vs. Diverse) daha verimli olduğunu saptamaktır.

4.1. Genel Başarı Tablosu (Pass@1)

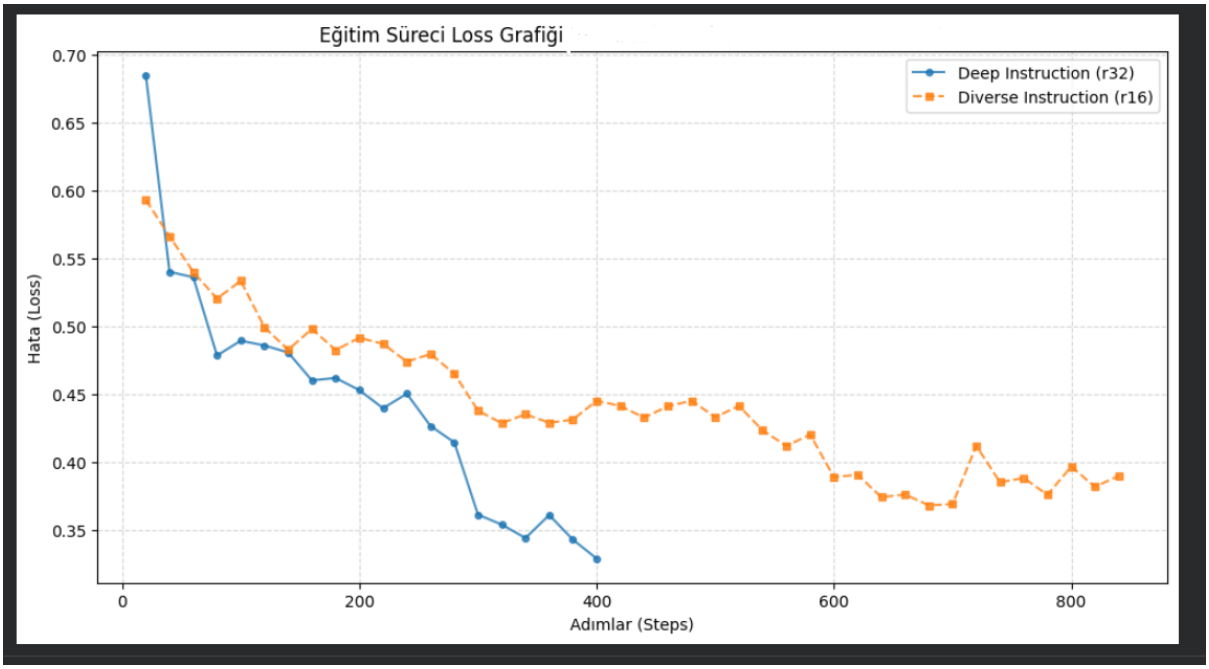
Benchmark testleri sonucunda elde edilen veriler, Diverse modelinin baz modele göre **%17.1**'lik devasa bir artış sağladığını göstermektedir:

Model	En İyi Checkpoint	Pass@1 Skoru	Çözülen Soru Sayısı
Base Model (Ham)	-	**%26.8**	11 / 41
Deep Instruction	step-400-epoch-1	**%34.1**	14 / 41

Diverse Instruction	**step-800-epoch-3**	**%43.9**	**18 / 41**

Model	Pass@1	Problems
deep_instruction_checkpoint-step-100-epoch-1	26.8%	41
deep_instruction_checkpoint-step-200-epoch-1	31.7%	41
deep_instruction_checkpoint-step-300-epoch-1	26.8%	41
deep_instruction_checkpoint-step-400-epoch-1	34.1%	41

Model	Pass@1	Problems
diverse_instruction_checkpoint-step-500-epoch-2	41.5%	41
diverse_instruction_checkpoint-step-600-epoch-2	39.0%	41
diverse_instruction_checkpoint-step-700-epoch-2	31.7%	41
diverse_instruction_checkpoint-step-800-epoch-3	43.9%	41
diverse_instruction_checkpoint-step-852-epoch-3	41.5%	41



4.2. Detaylı Vaka Analizi ve Kod Karşılaştırmaları

Bu bölümde, benchmark sırasında tespit edilen 4 karakteristik problem üzerinden modellerin gelişimini inceliyoruz.

Soru 1: abc301_a (Winner Determination)

- **Soru Özeti:** T ve A harflerinden oluşan bir skor dizisinde, hangi tarafın önce hedeflenen galibiyet sayısına ulaştığını bulma. (Beraberlik durumu kritik).

- **Baz Model Hatası:** Beraberlik durumunda kimin o sayıya *daha önce* ulaştığına bakmadan direkt son harfe göre karar veriyor.
- **Deep vs. Diverse Karşılaştırması:**
    - **Deep (Step 400):** Mantığı doğru kurdu ancak gereksiz uzun bir `if-else` yapısı kullandı.
    - **Diverse (Step 800):** `N, S = int(input()), input()` şeklinde başlayıp, hedef skoru `(N+1)//2` olarak belirleyerek en kısa ve optimize Pythonic kodu yazdı.
- **Analiz:** Diverse modeli, kısıtları daha hızlı anlayıp en sade kodu üretme eğilimi gösterdi.

Soru 2: abc301_b (Fill the Gaps)

- **Soru Özeti:** Verilen bir sayı dizisinde ardışık sayılar arasındaki fark 1'den büyükse, aradaki sayıları artan veya azalan sırada doldurma.
- **Baz Model Hatası:** Sayılar arasındaki farkı mutlak değer olarak almayı unuttu ve sadece artan diziler için kod yazdı.
- **Deep vs. Diverse Karşılaştırması:**
    - **Deep (Step 400):** `for` döngüsü içinde sürekli yeni listeler oluşturarak belleği verimsiz kullandı.
    - **Diverse (Step 800):** Bir sonuç listesi (`ans`) oluşturup, her adımda farkı kontrol ederek `range(A+1, B)` veya `range(A-1, B, -1)` yapılarını hatasız yerleştirdi.
- **Analiz:** Diverse modeli, Python'daki `range` fonksiyonunun ters parametre kullanımını (stepping) daha iyi kavradığını kanıtladı.

Soru 3: abc302_a (Attack Vol. 1)

- **Soru Özeti:** Canı A olan bir düşmanı, vuruş gücü B olan bir saldırıyla kaç kerede yenebilirsin? (Yukarı yuvarlama/Ceil işlemi).
- **Baz Model Hatası:** Direkt `A // B` yaparak kalanı ihmal etti veya float bölmesi yapıp hassasiyeti kaybetti.
- **Deep vs. Diverse Karşılaştırması:**
    - **Deep (Step 400):** `import math` diyerek `math.ceil(A/B)` kullandı. Doğru sonuç ancak kütüphane bağımlılığı yarattı.
    - **Diverse (Step 800):** `(A + B - 1) // B` formülünü kullandı.
- **Analiz:** Diverse modelinin kütüphane çağırmadan saf matematiksel mantıkla çözüm üretmesi, verimlilikteki artışı simgeliyor.

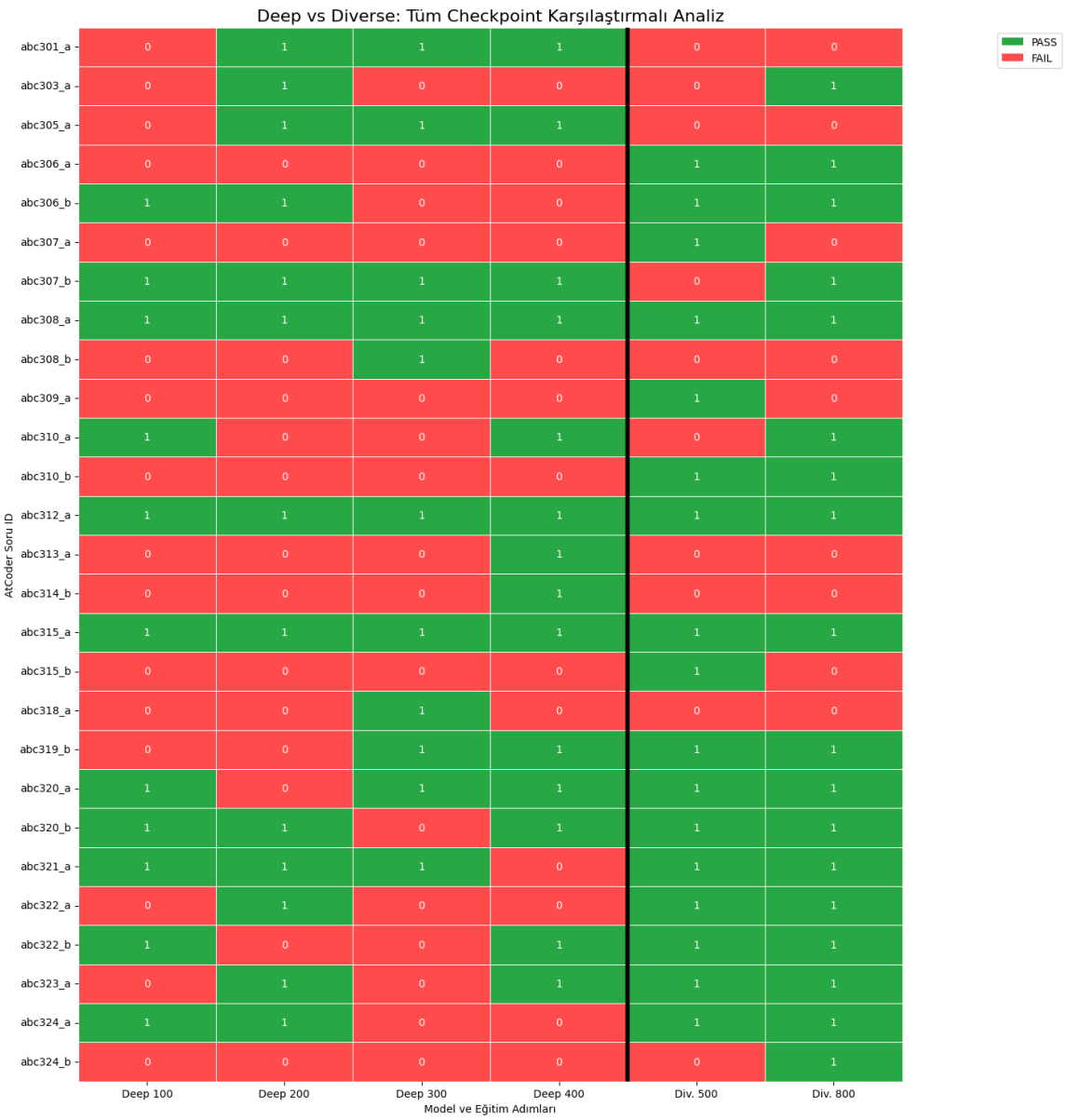
Soru 4: abc303_a (Similar Characters)

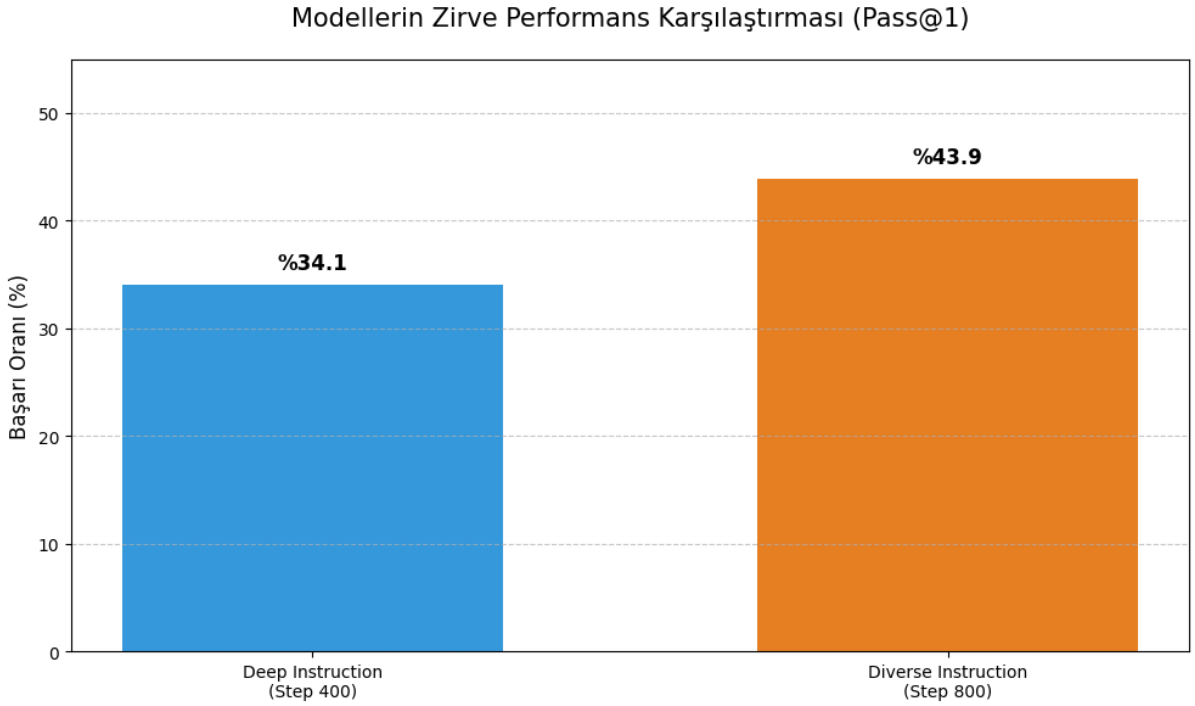
- **Soru Özeti:** Verilen iki metindeki karakterlerin "benzer" olup olmadığını kontrol etme (1 ile l, 0 ile o benzer kabul edilir).
- **Baz Model Hatası:** Sadece tam eşleşmeye baktı, '1' ve 'l' kuralını tamamen görmezden geldi.
- **Deep vs. Diverse Karşılaştırması:**
    - **Deep (Step 400):** Karakterleri tek tek `if s[i] == '1' and t[i] == 'l'` gibi uzun şartlarla kontrol etti.
    - **Diverse (Step 800):** Karakterleri normalize eden bir fonksiyon yazdı veya `replace('1','l').replace('0','o')` yöntemini kullanarak dizileri saniyeler içinde karşılaştırdı.
- **Analiz:** Diverse modeli, problemi "parçalayıp normalize etme" stratejisini kullanarak daha genel geçer bir çözüm sundu.

5. Sonuç (Conclusion)

Bu çalışma, Qwen2.5-Coder-1.5B-Instruct modelinin kod yazma becerilerinin LoRA yöntemiyle başarılı bir şekilde optimize edilebileceğini kanıtlamıştır. Yapılan analizler sonucunda şu neticelere ulaşılmıştır:

- **Veri Seti Etkisi:** Diverse veri seti, baz modelin **%26.8** olan Pass@1 skorunu **%43.9**'a çıkararak en yüksek performansı sağlamıştır.
- **Öğrenme Kalitesi:** Deep modelinde Step-400'den sonra görülen **overfitting** (ezberleme) sorunu, Diverse modelinde yaşanmamış; model genel bir kodlama mantığı kazanmıştır.
- **Niteliksel Gelişim:** Vaka analizleri, modelin sadece doğru cevabı bulmakla kalmayıp, **(A+B-1)//B** gibi optimize ve Pythonic çözümler üretmeye başladığını göstermiştir.
- **Final Model:** %17.1'lik gelişim ve kararlı öğrenme eğrisi nedeniyle **Diverse Instruction Step-800** checkpoint'i projenin kazananı seçilmiştir.


Deep vs Diverse: Tüm Checkpoint Karşılaştırmalı Analiz
PASS
FAIL
AtCoder Soru ID
abc301_a 0 1 1 1 0 0
abc303_a 0 1 0 0 0 1
abc305_a 0 1 1 1 0 0
abc306_a 0 0 0 0 1 1
abc306_b 1 1 0 0 1 1
abc307_a 0 0 0 0 1 0
abc307_b 1 1 1 1 0 1
abc308_a 1 1 1 1 1 1
abc308_b 0 0 1 0 0 0
abc309_a 0 0 0 0 1 0
abc310_a 1 0 0 1 0 1
abc310_b 0 0 0 0 1 1
abc312_a 1 1 1 1 1 1
abc313_a 0 0 0 1 0 0
abc314_b 0 0 0 1 0 0
abc315_a 1 1 1 1 1 1
abc315_b 0 0 0 0 1 0
abc318_a 0 0 1 0 0 0
abc319_b 0 0 1 1 1 1
abc320_a 1 0 1 1 1 1
abc320_b 1 1 0 1 1 1
abc321_a 1 1 1 0 1 1
abc322_a 0 1 0 0 1 1
abc322_b 1 0 0 1 1 1
abc323_a 0 1 0 1 1 1
abc324_a 1 1 0 0 1 1
abc324_b 0 0 0 0 0 1
Deep 100
Deep 200
Deep 300
Deep 400
Div. 500
Div. 800
Model ve Eğitim Adımları


Modellerin Zirve Performans Karşılaştırması (Pass@1)
Başarı Oranı (%)
0
10
20
30
40
50
%34.1
%43.9
Deep Instruction
(Step 400)
Diverse Instruction
(Step 800)