Chain-of-Thought (CoT), Büyük Dil Modellerinin (LLM) karmaşık akıl yürütme görevlerindeki performansını artırmak için kullanılan bir tekniktir. LLM'leri, nihai bir cevaba ulaşmadan önce ara mantıksal adımları üretmeye teşvik ederek çalışır. Bu, genellikle modelin bir sorunu nasıl parçalara ayıracağını gösteren örneklerin (few-shot CoT) sağlanması veya "Adım adım düşünelim" gibi genel bir talimatın (zero-shot CoT) eklenmesi yoluyla, özenle hazırlanmış prompt'lar aracılığıyla gerçekleştirilir. CoT, model geliştirme süreçlerine, eğitim veya ince ayar sırasında ara akıl yürütme adımlarını içeren veri kümeleri kullanılarak entegre edilebilir. Mimari olarak, CoTFormer gibi özel modeller veya ara denetim (teacher forcing) ve konumsal kodlamadaki (positional encoding) iyileştirmeler gibi teknikler, CoT yeteneklerini geliştirmeyi amaçlar.

Chain-of-Thought (CoT) in LLMs: Mechanisms, Integration, and Architectural Enhancements

1. Understanding Chain-of-Thought (CoT)

1.1. Definition of CoT

Chain-of-Thought (CoT) prompting, Büyük Dil Modellerinin (LLM) akıl yürütme yeteneklerini geliştirmek için kullanılan bir tekniktir. Bu yöntem, modeli, nihai bir cevap vermeden önce bir dizi ara mantıksal adımı veya "düşünceyi" üretmeye teşvik eder . Esas olarak, modelin karmaşık bir sorunu daha küçük, yönetilebilir parçalara ayırmasını ve bu parçaları sırayla çözmesini sağlayarak, insan benzeri bir problem çözme yaklaşımını taklit etmeyi amaçlar . CoT, modelin "işini göstermesini" sağlayarak, onun karar verme sürecini daha şeffaf ve yorumlanabilir hale getirir . Bu teknik, aritmetik, sembolik mantık ve karmaşık karar verme gibi çok adımlı akıl yürütme gerektiren görevlerde özellikle değerlidir. CoT prompting, genellikle modelin boyutu ve karmaşıklığı arttıkça etkililiği de artan, "ortaya çıkan bir yetenek" olarak kabul edilir . Bu, daha büyük modellerin geniş veri kümeleri üzerinde eğitilirken daha nüanslı akıl yürütme kalıplarını öğrenme eğiliminde olmasından kaynaklanır.

CoT prompting'in temel prensibi, modele, adım adım bir akıl yürütme sürecini gösteren örnekler (örnekçeler) veya talimatlar sağlamaktır . Bu örnekler, bir sorunun ara düşüncelere veya hesaplamalara nasıl ayrıştırıldığını ve bunların nihai cevaba nasıl yol açtığını açıkça gösterir . Model bu örnekleri gözlemleyerek, kendisine sorulan asıl soruyu işlerken ve yanıtlarken benzer bir adım adım yaklaşım uygulamayı öğrenir. Bu yapılandırılmış düşünme biçimi, genellikle açık akıl yürütme adımları içermeyen geleneksel prompt yöntemlerine kıyasla daha doğru çıktılar üretilmesini sağlar . CoT,

sıfır-atış CoT (prompt'a "Adım adım düşünelim" gibi bir ifade eklemek), birkaç-atış CoT (elle yazılmış akıl yürütme zinciri örnekleri sağlamak) veya otomatik CoT (Auto-CoT, LLM'leri örneklemeler için akıl yürütme zincirleri oluşturmak üzere kullanmak) gibi çeşitli prompt stratejileriyle uygulanabilir . Bu teknik, matematik problemleri, mantık bulmacaları veya çok atlamalı soru cevaplama gibi doğal olarak çok adımlı akıl yürütme gerektiren görevler için özellikle faydalıdır .

1.2. Purpose and Benefits of CoT

Chain-of-Thought (CoT) prompting'in temel amacı, Büyük Dil Modellerinin (LLM) karmaşık, çok adımlı problemleri çözerken akıl yürütme doğruluğunu ve yorumlanabilirliğini artırmaktır . Modeli, akıl yürütme sürecini adım adım ifade etmeye yönlendirerek, CoT, doğrudan cevap üretiminden kaynaklanabilecek düşünmeden verilmiş veya yanlış cevapların olasılığını azaltmaya yardımcı olur . Bu, aritmetik, sağduyu akıl yürütme, sembolik manipülasyon, planlama ve karar verme gibi, doğru bir çözüme ulaşmak için net, mantıksal bir düşünce ilerleyişinin gerekli olduğu görevler için özellikle önemlidir . Adım adım ayrıştırma, potansiyel hataların tanımlanabileceği ve düzeltilebileceği ara kontrol noktaları sağlar, bu da daha güvenilir nihai cevaplara yol açar . Ayrıca, akıl yürütme adımlarının açıkça üretilmesi, modelin karar verme sürecinin şeffaflığını artırır, kullanıcıların bir sonuca nasıl ulaşıldığını anlamasına ve mantıktaki kusurları ayıklamasına olanak tanır . Bu şeffaflık, modelin çıktılarına olan güveni artırır ve denetlenebilirliğin kritik olduğu uygulamalarda özellikle değerlidir .

CoT prompting'in en önemli faydalarından biri, özellikle daha büyük modellerde ortaya çıkan bir yetenek olarak kabul edildiği için, LLM'lerde sofistike akıl yürütme yeteneklerini ortaya çıkarma becerisidir . Bu, modellerin standart prompt tekniklerinin zorlandığı daha karmaşık problemleri ele almasını sağlar . CoT, genellikle ince ayar gerektirmez, çünkü akıl yürütme örneklerini veya talimatlarını içeren özenle hazırlanmış prompt'larla uygulanabilir . Bu da onu çeşitli karmaşık görevlere uyarlamak için nispeten verimli bir yöntem haline getirir. Bu teknik aynı zamanda, karmaşık problemleri yönetilebilir, ara adımlara bölerek ve bunları sırayla nihai bir cevaba ulaşacak şekilde birleştirerek daha iyi problem çözümüne katkıda bulunur . Bu yapılandırılmış yaklaşım, yalnızca doğruluğu değil, aynı zamanda modelin yanıtlarının netliğini ve mantıksal tutarlılığını da artırır . Örneğin, matematik problem çözmede, CoT prompting'in PaLM gibi modellerin GSM8K gibi kıyaslamalarda en iyi performansı elde etmesine yardımcı olduğu gösterilmiştir . Gelişirilen yorumlanabilirlik aynı zamanda model sınırlamalarını anlamaya ve daha fazla gelişirme alanlarını belirlemeye yardımcı olur.

2. How CoT Works in LLMs (Mechanisms)

2.1. CoT via Prompt Engineering

Chain-of-Thought (CoT) prompting, Büyük Dil Modellerini (LLM) nihai bir cevaba varmadan önce ara akıl yürütme adımları üretmeye yönlendirmek için özenle tasarlanmış prompt'lar aracılığıyla çalışır . Bu, birkaç farklı prompt stratejisiyle gerçekleştirilebilir. Yaygın bir yaklaşım, girdi prompt'una birkaç örnek (örnekçe) eklenmesini içeren **birkaç-atış CoT (few-shot CoT) prompting**'dir. Bu örnekler, problemler ve bunlara karşılık gelen, adım adım akıl yürütme zincirlerini içerir ve ardından hedef problem gelir . Model bu gösterimlerden öğrenerek yeni problem için benzer bir düşünce zinciri oluşturur. Örneğin, bir prompt şöyle olabilir: "S: Bir mağazada 12 elma var. 5 tanesini satıyor. Kaç elma kaldı? C: Mağazada başlangıçta 12 elma vardı. 5 tanesini sattı. Yani, 12 - 5 = 7 elma kaldı." Ardından kullanıcının sorusu gelir. LLM, örnekçeyi gördükten sonra, sonraki problem için benzer bir adım adım açıklama üretme olasılığı daha yüksektir . Bu yöntem, modelin bağlamdan öğrenme ve sağlanan örneklerdeki kalıpları tanıma yeteneğinden yararlanır .

Bir diğer önemli teknik ise **sıfır-atış CoT (zero-shot CoT) prompting**'dir. Burada açık örnekler sağlamak yerine, prompt'a "Adım adım düşünelim" gibi genel bir talimat eklenir . Bu basit ipucu, yeterince büyük ve yetenekli LLM'lerde bir CoT ortaya çıkarmada şaşırtıcı derecede etkili olabilir. Model, özünde kendi kendine bir akıl yürütme süreci oluşturmaya teşvik edilir. Örneğin, "Bir fırın 18 cupcake'e sahip, 12 tane daha pişiriyor, sonra 15 tanesini satıyor. Kaç tane kaldı? Adım adım düşünelim" diye sormak, modelin şunu üretmesine yol açabilir: "Fırın başlangıçta 18 cupcake'e sahipti. 12 tane daha pişirdikten sonra 18 + 12 = 30 cupcake'leri oldu. 15 sattılar, yani 30 - 15 = 15 cupcake kaldı" . Bu yöntem, elle örnek oluşturma gerektirmediği için uygulaması daha basittir, ancak kendi kendine üretilen CoT'nin kalitesi, birkaç-atış prompting'e göre daha az güvenilir olabilir . **Otomatik Chain-of-Thought (Auto-CoT)**, soruları kümeleyerek ve her kümeden temsili örnekler için sıfır-atış CoT kullanarak akıl yürütme zincirleri oluşturmayı ve bunları birkaç-atış prompt'ta kullanmayı amaçlayarak, her ikisinin de avantajlarını birleştirmeye çalışır . Bu, birkaç-atış CoT'nin manuel çabasını azaltırken, sıfır-atış CoT'den daha iyi güvenilirlik elde etmeyi hedefler .

CoT prompt'larının yapısı da değişiklik gösterebilir. **Yapılandırılmış şablonlar**, modelin atması gereken adımları açıkça ana hatlarıyla belirtebilir . **Etkileşimli prompt'lar**, modeli her adımı açıklamaya veya her aşamada geri bildirim vermeye teşvik ederek, insanın döngüde olduğu bir yaklaşımı benimseyebilir . **Geri bildirim döngüleri**, modelin ara

adımlarının değerlendirildiği ve gerekirse rafine edildiği, çok aşamalı prompting olarak bilinen bir sürece dahil edilebilir . Anahtar nokta, prompt'un hangi biçimde olursa olsun, LLM'nin sorunu ayrıştırması ve akıl yürütmesini sıralı bir şekilde ifade etmesi için gerekli rehberliği sağlamasıdır. Bu, tipik olarak basit girdi–çıktı örneklerinden oluşan ve açık akıl yürütme adımları içermeyen standart prompting ile tezat oluşturur; bu da modellerin çok adımlı görevler için gerekli mantığı çıkarmasını zorlaştırır . CoT prompting'in etkinliği, genellikle örtük akıl yürütme sürecini açık hale getirme ve modelin öğrenilmiş bilgisini ve kalıplarını karmaşık problem çözme için daha iyi kullanmasına izin verme yeteneğine atfedilir .

2.2. Theoretical Underpinnings of CoT in Transformers

Chain–of–Thought (CoT) prompting'in, Transformer tabanlı Büyük Dil Modellerinin (LLM) akıl yürütme yeteneklerini neden geliştirdiğine dair teorik anlayış, aktif bir araştırma alanıdır. Bir anahtar bakış açısı, CoT'nin, özünde paralel işlemciler olan Transformers'ın, özünde seri olan hesaplamaları gerçekleştirmesini sağladığıdır . Sabit derinliğe ve sonlu hassasiyete sahip standart Transformers'ın, CoT olmadan genellikle AC^0 veya TC^0 gibi karmaşıklık sınıfları içindeki problemleri çözmekle sınırlı olduğu gösterilmiştir . Bu sınıflar, sınırsız fan–in AND/OR/NOT kapılarına veya (TC^0 için) çoğunluk kapılarının eklenmesiyle sabit derinlikteki devreler tarafından çözülebilen problemleri temsil eder. Bu, geleneksel Transformers'ın kolayca paralelleştirilemeyen sıralı, çok adımlı akıl yürütme gerektiren görevlerde zorlanabileceği anlamına gelir. Bununla birlikte, modelin bir dizi ara akıl yürütme adımı (token) ürettiği CoT'nin tanıtılması, Transformer'ın etkin bir şekilde daha güçlü bir hesaplama modelini simüle etmesine izin verir .

Araştırmalar, T adım CoT ile, sabit–bit hassasiyeti ve $O(\log n)$ gömme boyutu kullanan sabit–derinlikli Transformers'ın bile, prensipte, T boyutunda Boole devreleri tarafından çözülebilen herhangi bir problemi çözebileceğini öne sürmektedir . Bu, ifade gücünde önemli bir artıştır. Kavramsal olarak, CoT sırasında üretilen ara token'lar bir "karalama defteri" veya bellek görevi görerek, modelin aksi takdirde uzun sıralı bağımlılıklarla uğraşırken sınırlı olacağı bir hesaplamanın ara sonuçlarını depolamasına ve manipüle etmesine olanak tanır . Bu seri hesaplama yapma yeteneği, permütasyon gruplarının oluşturulması, tekrarlanan kare alma ve devre değer problemleri gibi paralel hesaplama için zor olan görevler için çok önemlidir . Ampirik olarak, CoT'nin etkinleştirilmesinin, özellikle düşük derinlikli Transformers için bu tür görevlerde doğruluğu önemli ölçüde artırdığı gösterilmiştir . CoT mekanizması, esasen Transformer'ın karmaşık bir problemi daha basit, yönetilebilir alt problemlere ayırmasına, bunları sırayla çözmesine ve

sonuçları birleşţirerek nihai cevaba ulasm̧asına izin verir, böylece mimarinin belirli akıl yürütme türleri için bazı doğal paralellik sınırlamalarının üstesinden gelir .

Daha ileri teorik çalısm̧alar, CoT'nin örnek verimliliğini nasıl artırabileceğini arasţırmaktadır. Örneğin, esl̨ik öğrenme gibi görevlerde, CoT, bir Transformer'ın fonksiyonu polinom sayıda örnekle öğrenmesini sağlayabilir; oysa CoT olmadan, temsil gücü teorik olarak yeterli olsa bile, üstel sayıda örnek gerekebilir . Bu, CoT'nin yalnızca ifade gücünü artırmakla kalmayıp, aynı zamanda girdi token'ları arasında seyrek sıralı bağımlılıklar sunarak öğrenme sürecini basitlesţirdiğini ve daha seyrek ve potansiyel olarak daha yorumlanabilir dikkat kalıplarına yol açtığını düsş̧ündürmektedir . CoT'deki ara adımlar, problemin altında yatan yapısını model için eğitim sırasında daha belirgin hale getirebilir. Bazı çalısm̧alar, otoregresif Transformers ile CoT arasında paralellikler kurarak, CoT token'larının yinelemeli üretiminin yinelemeli hesaplamayı taklit edebileceğini, böylece modelin sıralı isl̨eme kapasitesini daha da artırabileceğini öne sürmektedir . Bu teorik çerçeve, modelleri ara akıl yürütme adımları üretmeye yönlendirmenin, çok adımlı problemleri doğru bir şekilde çözme yeteneklerini neden önemli ölçüde artırdığını açıklamaya yardımcı olur .

2.3. Internal Mechanisms and Serial Computation

Chain-of-Thought (CoT)'nin, özellikle Transformers gibi Büyük Dil Modellerinde (LLM) akıl yürütmeyi kolaylasţıran iç mekanizmaları, mimarinin doğal paralelliğini tamamlayan bir seri hesaplama biçimini etkinlesţirmeyi içerir. Transformers, bir dizi içindeki girdi token'larının paralel isl̨enmesinde ustadır, ancak karmasık̨ akıl yürütme genellikle bir adımın çıktısının bir sonraki adımın girdisi olduğu sıralı, adım adım bir yaklasım̧ gerektirir . CoT, modelin otoregresif bir şekilde bir dizi ara "düsş̧ünce" token'ı üretmesine izin vererek bunu ele alır. CoT dizisinde üretilen her token, akıl yürütme sürecindeki bir ara durum veya kısmi sonuç olarak görülebilir. Model, girdi prompt'unu ve daha önce üretilen CoT token'larını isl̨emek için standart Transformer kodlayıcı (veya kodlayıcı-kodçözücü) mimarisini kullanarak, akıl yürütme zincirindeki bir sonraki token'ı tahmin eder . Bu yinelemeli üretim süreci, modelin bir insanın bir karalama defterinde bir problem üzerinde çalısm̧ası gibi, birden çok adımda gelisen̨ hesaplamalar yapmasına olanak tanıyan etkili bir şekilde zamansal bir boyut yaratır .

Mekanistik yorumlanabilirlik çalısm̧aları, Transformers'ın CoT ile akıl yürütmeyi nasıl uyguladığına ısık̨ tutmaya basl̨amıştır. Örneğin, arasţırmalar, Transformer+CoT modellerinde, genellikle geç katman Çok Katmanlı Algılayıcı (MLP) nöronlarını içeren belirli devrelerin, "dünya durumunu" veya akıl yürütme sürecinin ara durumlarını izlemede çok önemli bir rol oynadığını göstermektedir . Bu nöronlar, temsil ettikleri

durumlara göre gruplandırılabilir ve aktivasyon kalıpları, farklı durumlar arasında ayrım yapmada yüksek doğruluk sergileyebilir, bu da modelin içine gömülü örtük bir Sonlu Durum Otomatı (FSA) olduğuna dair kanıt sağlar . Bu FSA benzeri davranış, modelin CoT'nin her adımını üretirken problem durumunu korumasına ve güncellemesine olanak tanır. Transformers'daki dikkat mekanizması da CoT üretimi sırasında uyum sağlar. Ara adımların sayısı arttıkça, dikkat kalıplarındaki değişimler gözlemlenebilir, bu da modelin bir sonraki adım için gerekli hesaplamaları yapmak üzere girdinin ve üretilen CoT'nin farklı kısımlarına nasıl odaklandığını dinamik olarak ayarladığını gösterir .

CoT'nin seri hesaplamayı etkinleştirme yeteneği özellikle önemlidir çünkü bu, özellikle sınırlı derinliğe sahip standart Transformers'ın, doğası gereği sıralı mantık gerektiren problemleri çözmedeki temel bir sınırlamasının üstesinden gelir . CoT olmadan, bir Transformer'ın hesaplaması büyük ölçüde sabit katman sayısı tarafından belirlenir, bu da doğrudan sıralı işleme derinliğini sınırlar. Buna karşılık CoT, modelin keyfi sayıda ara token üreterek hesaplama derinliğini genişletmesine izin verir, böylece daha uzun bir "hesaplama yolu" oluşturur . Bu nedenle CoT'nin, aritmetik, sembolik manipülasyon ve planlama gibi, sorunu daha basit işlemlere ayırmanın gerekli olduğu görevlerde performansı önemli ölçüde artırdığı gösterilmiştir . Süreç, modelin problemi kodlamasını, ardından akıl yürütmenin her adımını temsil eden token'ları yinelemeli olarak üretmesini ve son olarak, nihai cevaba ulaşmak için üretilen tüm CoT dizisini kullanmasını içerir . Probleme ve tüm önceki adımlara bağlı olan bu adım adım üretim, Transformers'ın aksi takdirde yapamayacakları kadar karmaşık, seri yapılandırılmış problemleri ele almasını sağlayan temel mekanizmadır.

3. Integrating CoT into Model Development

3.1. Integration via Training

Chain-of-Thought (CoT) akıl yürütme yeteneklerini, Büyük Dil Modellerinin (LLM) eğitim aşamasında doğrudan entegre etmek, yalnızca girdi-çıktı çiftlerini değil, aynı zamanda açık, adım adım akıl yürütme zincirlerini de içeren veri kümeleri kullanmayı içerir. Bu yaklaşım, modelin ara akıl yürütme adımları üretme sürecini, yalnızca eğitim sonrası prompt tekniklerine güvenmek yerine, temel işlevselliğinin bir parçası olarak içselleştirmesini öğretmeyi amaçlar. Araştırmalar, açık CoT açıklamalarıyla eğitilen modellerin iç temsillerini yeniden şekillendirebileceğini ve hem dağılım içi (ID) hem de dağılım dışı (OOD) akıl yürütme genellemesini iyileştirebileceğini göstermiştir . Bir anahtar fikir, CoT eğitiminin modelin içinde, aşamaların eğitim sırasında görülen açık akıl yürütme adımlarına karşılık geldiği bir "iki aşamalı genelleme devresi" gelişimine yol

açabileceğidir. Bu tür modellerde, ara sonuçlar, CoT olmayan muadillerine kıyasla daha sığ katmanlarda çözülme eğilimindedir, bu da daha derin katmanların sonraki akıl yürütme adımlarında uzmanlaşmasını sağlar . Bu yapısal avantaj, modelin karmaşık problemleri daha etkili bir şekilde ayrıştırmasına olanak tanır.

Süreç tipik olarak, her örneğin bir problem, ayrıntılı bir CoT ve nihai cevaptan oluştuğu veri kümelerinin oluşturulmasını veya üretilmesini içerir. Örneğin, matematiksel akıl yürütmede, bir eğitim örneği şöyle olabilir: "Problem: John'un 5 elması var. Mary'ye 2 tane veriyor. Geriye kaç tane kaldı? CoT: John başlangıçta 5 elmaya sahipti. 2 elma verdi. Yani, 5 − 2 = 3 elması kaldı. Cevap: 3". Model daha sonra, problemi verilen tüm diziyi, CoT ve nihai cevabı tahmin etmek üzere eğitilir. Bu, standart otoregresif dil modelleme hedefleri kullanılarak yapılabilir. Çalışmalar, bu tür açık CoT eğitiminin yakınsamayı hızlandırabileceğini ve genellemeyi artırabileceğini göstermiştir . Örneğin, 3B görü–dil hizalaması bağlamında, CoT yapılandırılmış açıklamaların model eğitimine entegre edilmesi, özellikle olanak tanıma ve etkileşim tahmininde, modellerin metinsel açıklamalar ve 3B yapılar arasında gelişmiş hizalamalar sergilemesiyle akıl yürütmeyi önemli ölçüde iyileştirmiştir . Eğitim protokolü, önce çok modlu özellikleri hizalamak ve ardından dil modelinin belirli katmanlarını çok adımlı akıl yürütme sinyallerini entegre etmek için ince ayar yapmak gibi birden çok aşama içerebilir .

Dahası, CoT eğitiminin teorik faydaları da araştırılmaktadır. Örneğin, ID hatasının CoT'ye bakılmaksızın yeterli eğitimle azalabileceği, ancak OOD hatasının kritik olarak buna bağlı olduğu öne sürülmüştür: CoT olmayan eğitim, görülmemiş akıl yürütme kalıpları nedeniyle OOD örneklerine genelleyemeyebilir, oysa CoT eğitimi, eğitim sırasında alt görevlere ve akıl yürütme kompozisyonlarına hakim olarak daha iyi OOD genellemesi elde edebilir . Bu, CoT ile eğitimin, onları tanıdık akıl yürütme adımlarına ayırarak yeni problemleri ele alabilen daha sağlam modellere yol açabileceği anlamına gelir. IBM'in Granite Instruct gibi modellerde görüldüğü gibi, talimat ince ayarının CoT verileriyle birleştirilerek daha küçük modellerin bile CoT akıl yürütmesini etkili bir şekilde gerçekleştirmesinin sağlanması, talimat ince ayarının CoT ile nasıl birleştirilebileceğinin bir örneğidir . Bu yaklaşım, basit prompting'in ötesine geçer ve akıl yürütme yeteneğini modelin parametrelerine daha derinden gömer.

3.2. Integration via Fine-tuning

Önceden eğitilmiş Büyük Dil Modellerini (LLM), Chain-of-Thought (CoT) akıl yürütme adımlarını içeren veri kümeleri üzerinde ince ayar yapmak, çok adımlı akıl yürütme yeteneklerini geliştirmek için yaygın bir stratejidir. Bu yaklaşım tipik olarak, büyük bir metin külliyatı üzerinde önceden eğitilmiş bir modeli almayı ve ardından örneklerin açık

ara mantık içerdiği daha küçük, özelleşţirilmis¸bir veri kümesi üzerinde daha fazla eğitmeyi içerir. Amaç, modeli, ilgili problemlerle karsı¸laşţığında bu CoT adımlarını doğal olarak üretecek şekilde uyarlamak veya prompt verildiğinde üretilen CoT'nin kalitesini artırmaktır. Örneğin, modeller, her problem için adım adım bir çözümün esli¸k ettiği GSM8K gibi matematiksel kelime problemleri için veri kümeleri kullanılarak ince ayarlanabilir . Iṅce ayar süreci, modelin ağırlıklarını, nihai cevap token(ler)inden önce gelen akıl yürütme token'larının dizisini daha iyi tahmin edecek şekilde ayarlar. Bu, mantıksal çıkarım ve sıralı isl¸eme gerektiren karması¸k görevleri çözme yeteneğinde önemli iyileşţirmelere yol açabilir.

Ancak, ince ayarın CoT yetenekleri üzerindeki etkisi nüanslıdır. Bazı çalısm̧alar, ince ayarın, ince ayar için kullanılan belirli görev veya veri kümesindeki performansı iyileşţirebileceğini, ancak bazen diğer görevlerde veya hatta farklı değerlendirme ortamlarında aynı görevde CoT doğruluğunda veya bağlılığında bir azalmaya yol açabileceğini, özellikle de daha küçük modeller için bulmuşţur . Örneğin, Llama–3–8B–Instruct gibi modellerin GSM8K, CosmosQA ve MedQA gibi akıl yürütme veri kümeleri üzerinde ince ayarlanmasının, bazen önceden eğitilmis¸sürümlerine kıyasla daha düşţük CoT doğruluğu ile sonuçlandığı, özellikle matematiksel akıl yürütme veri kümelerinde gözlemlenmişţir . Bu azalma, GPT–4 gibi daha büyük modellere kıyasla Llama–3–8B–Instruct gibi daha küçük LLM'lerde daha belirgindi, muhtemelen daha büyük modeller daha iyi genelleme yeteneklerine sahip olduğundan ve daha az önemli ağırlık ayarı gerektirdiğinden . Üretilen CoT'nin bağlılığı –yani akıl yürütme adımlarının modelin cevaba giden gerçek yolunu ne kadar doğru yansıttığı veya cevap için ne kadar gerekli olduğu– da etkilenebilir. Daha küçük LLM'leri minimal akıl yürütme gerektiren veya hiç gerektirmeyen veri kümeleri üzerinde ince ayarlamak, ürettikleri CoT'lerin bağlılığını daha da azaltma eğilimindeydi .

Bu potansiyel tuzaklara rağmen, ince ayar CoT akıl yürütmesi için uzmanlasm̧ıs¸LLM'ler için çok önemli bir yöntem olmaya devam etmektedir. Kendi Kendine Öğreten Akıl Yürütücü (STaR) gibi teknikler, LLM'nin kendi akıl yürütme verilerini ürettiği ve ardından bunları ince ayar için kullandığı yinelemeli bir süreci içerir, böylece modelin akıl yürütme yeteneklerini kendi kendine gelisţirmesine etkin bir şekilde izin verir . DeepSeek–R1 tarafından örneklendirildiği gibi basḳa bir yaklasım̧, çok turlu akıl yürütmede artan beceri edinimini kolaylasţırmak için görevlerin kademeli olarak düzenlendiği müfredat öğrenimi ile modelleri eğitmeyi içerir . Iṅce ayar sürecinin kendisi, Adafactor gibi optimize edicilerin kullanılması, belirli öğrenme oranı programları ve verimliliği artırmak için paketleme (birden çok eğitim örneğini tek bir diziye birlesţirme) gibi teknikleri içerebilir . Iṅce ayar için kullanılan hesaplama miktarı genellikle ön eğitim için

kullanılanın küçük bir kısmıdır, bu da onu model yeteneklerini geliştirmenin nispeten uygun maliyetli bir yolu haline getirir . Anahtar, ince ayar verilerindeki belirli kalıplara aşırı uyumu teşvik etmek yerine sağlam ve genellenebilir CoT akıl yürütmesini teşvik edecek şekilde ince ayar veri kümesini ve protokolünü dikkatlice tasarlamaktır.

3.3. Instruction Tuning for CoT

Instruction tuning (talimatla ince ayar), Büyük Dil Modellerini (LLM) kullanıcı niyetiyle uyumlu hale getirmek ve karmaşık talimatları izleme yeteneklerini geliştirmek için güçlü bir tekniktir; bu, Chain-of-Thought (CoT) akıl yürütmesini ortaya çıkaranlar da dahildir. Bu süreç, talimatların genellikle bir görevi tanımlayan doğal dil prompt'ları olduğu ve çıktıların istenen yanıtlar veya akıl yürütme süreçleri olduğu bir (talimat, çıktı) çiftleri veri kümesi üzerinde önceden eğitilmiş bir LLM'nin ince ayarını içerir. Talimatla ince ayar, çok adımlı akıl yürütma gerektiren örnekleri içerdiğinde, model, eğitim verisi ona bu tür problemlere adım adım akıl yürütmeyle yaklaşmanın en iyisi olduğunu örtük olarak öğrettiyse, "Aşağıdaki problemi çözün" gibi nispeten basit prompt'larla bile CoT üretmeyi öğrenebilir. Örneğin, IBM'in Granite Instruct modelleri, CoT görevleri için özelleştirilmiş eğitim veri kümeleri kullanılarak ince ayarlanır, bu da onları CoT akıl yürütmesini etkili bir şekilde gerçekleştirebilmelerini sağlar . Bu, talimatla ince ayarın modellere CoT yetenekleri aşılamanın bir yolu olabileceğini ve onları mantıksal, sıralı düşünce süreçleri gerektiren görevleri ele almada daha usta hale getirebileceğini göstermektedir.

Talimatla ince ayarın CoT için etkinliği, modele yalnızca *ne* çıktı vereceğini değil, aynı zamanda bir soruna *nasıl* yaklaşacağını da öğretme yeteneğinde yatar. Model, istenen çıktının bir akıl yürütme zinciri içerdiği çok sayıda örneğe maruz bırakılarak, benzer karmaşık görevlerle karşılaştığında bu tür zincirleri önceliklendirmeyi ve üretmeyi öğrenir. Bu, her bir sorgu için son derece mühendislik ürünü prompt'lara olan güveni azaltabileceğinden özellikle faydalıdır. Bunun yerine, model, belirli problem türleri için CoT tercihini içselleştirir. Örneğin, talimatla ince ayarlanmış bir model, "Bir çiftçinin 12 elması var. 8 tanesini satıyor ve sonra 5 tane daha alıyor. Kaç elması olur?" gibi bir prompt alabilir ve ardından şu şekilde bir CoT üretebilir: "Çiftçinin başlangıçta 12 elması vardı. 8 tanesini sattı, yani 12 − 8 = 4 elması kaldı. Sonra 5 tane daha aldı, yani 4 + 5 = 9 elması oldu. Cevap 9'dur." Bu, modelin problemi adım adım çözdüğünü gösterir. Bu yaklaşım, modelin öğrenilmiş dil modellerini ve kalıplarını, sorunu daha küçük, yönetilebilir parçalara ayırmak ve her adımı sırayla ele almak için kullanmasını sağlar. Talimatla ince ayar, modelin yalnızca sonucu değil, aynı zamanda o sonuca ulaşmak için

kullanılan mantıksal süreci de üretmesini teşvik eder, bu da çıktıları daha şeffaf ve güvenilir hale getirir.

4. Improving CoT in Transformer Architectures

4.1. Architectural Embedding of CoT

Transformer modellerine yapılan mimari değişiklikler, Chain-of-Thought (CoT) akıl yürütme yeteneklerini gömmek ve geliştirmek için prompt tabanlı veya ince ayar yaklaşımlarının ötesine geçen doğrudan bir yol sunar. Bu tür bir yenilik, CoT'yi belirteç düzeyinde taklit etmek üzere tasarlanmış **CoTFormer** mimarisidir . CoTFormer, ara "düşünme" belirteçlerinin yinelemeli olarak üretildiği ve işlendiği bir mekanizma sunar. Temel bir özellik, önceki "tekrarlar" veya "düşünce adımlarından" elde edilen bu ara temsillerin, sonraki adımlardaki dikkat mekanizmasına sunulmasıdır. Bu, modelin CoT benzeri bir şekilde anlayışını ve akıl yürütmesini içsel olarak rafine etmesine olanak tanır. Mimaride genellikle, Evrensel Transformers'a benzer şekilde bu yinelemeli adımlar arasında ağırlık paylaşımı yer alır, ancak kritik olarak, geçmiş "düşünce" temsillerinin daha sonraki adımların dikkatine nasıl dahil edildiği konusunda farklılık gösterir . Bu tasarım, modelin önceki akıl yürütme durumları üzerine açıkça inşa etmesine izin verir. Başka bir mimari yaklaşım, kablosuz sembol tespiti gibi belirli uygulamalar için tasarlanmış ancak örtük CoT'nin genel bir ilkesini gösteren **CHOOSE (Latent CHain Of-thOught SElf-training)** gibi modellerde görülür . CHOOSE, sığ bir Transformer'ın gizli alanına otoregresif gizli akıl yürütme adımlarını doğrudan dahil eder. Sorgu konumunda, bu gizli alan içinde yinelemeli olarak otoregresif geri bildirim yoluyla bir dizi "düşünce" yerleştirmesi üretilir. Son "düşünce" yerleştirmesi daha sonra çıktıyı üretmek için kullanılır. Bu, hatta sığ Transformers'ın (örneğin, 1-2 katman) sembolik ara belirteçler açıkça üretilmeden, bu sıkıştırılmış gizli alanda çok adımlı akıl yürütme gerçekleştirerek çok daha büyük modellerin akıl yürütme derinliğini taklit etmesine olanak tanır . Bu tür mimari gömmeler, CoT'yi modelin işlemesinin ayrılmaz bir parçası haline getirmeyi amaçlar. Bu değişiklikler genellikle temsillerin yinelemeli olarak rafine edilmesi için yollar yaratmaya odaklanır, böylece modelin çıktıya bağlanmadan önce birden çok adım "düşünmesine" etkin bir şekilde izin verir.

4.2. Role of Intermediate Supervision (Teacher Forcing)

Ara denetim, genellikle "teacher forcing" (öğretmen zorlaması) olarak uygulanır, özellikle CoT adımları çıktı dizisinin bir parçası olarak açıkça üretildiğinde, Transformers modellerini Chain-of-Thought (CoT) akıl yürütmesini kullanacak şekilde etkili bir şekilde eğitmede çok önemli bir rol oynar . Teacher forcing, modelin kendi önceden ürettiği (ve

potansiyel olarak yanlış) ara belirteçleri, akıl yürütme zincirindeki sonraki adımları tahmin etmek için girdi olarak kullanmasına izin vermek yerine, eğitim sırasında modeli doğru ara adımlarla beslemeyi içerir . Bu teknik, erken CoT adımlarındaki hataların hızla yayılabileceği ve birleşebileceği, böylece yanlış nihai cevaplara yol açabileceği ve modelin doğru akıl yürütme yörüngesini öğrenmesini zorlaştırabileceği için kritiktir . Doğru ara adımları sağlayarak, teacher forcing eğitim sürecini stabilize eder, modelin her adımı doğru önceki bağlama dayanarak tahmin etmesini sağlar ve karmaşık görevleri yönetilebilir alt görevlere ayırmaya yardımcı olur . Örneğin, k−parite problemini çözmede, teorik sonuçlar, teacher forcing ve ara paritelerin kayıp fonksiyonuna dahil edilmesiyle eğitildiğinde, Transformers'ın herhangi bir parite fonksiyonunu tek bir gradyan güncellemesinde çözmeyi öğrenebileceğini göstermektedir . Bu, ara denetimin sağladığı öğrenme verimliliğindeki önemli artışı vurgular. Teacher forcing olmadan, modelin tüm akıl yürütme zincirini uçtan uca kendi çıktılarına dayanarak üretmesi gerekir, bu da çok daha yavaş ve daha az kararlı bir öğrenme sürecine yol açabilir ve genellikle hata yayılımını azaltmak için veri artırma ve kendi kendine tutarlılık kontrolleri gibi daha karmaşık teknikler gerektirir . Bu nedenle, teacher forcing, özellikle karmaşık, çok adımlı akıl yürütme görevleri için LLM'lerin eğitim rejimine açık CoT üretimini başarıyla entegre etmek için önemli bir bileşendir.

4.3. Impact of Positional Encoding and Scratchpad CoT

Konumsal kodlamanın (Positional Encoding – PE) ele alınma biçimi ve "karalama defteri" (scratchpad) kavramı, özellikle etkili seri hesaplamayı etkinleştirmede, Transformers'daki Chain−of−Thought (CoT) işleyişinin ayrılmaz bir parçasıdır. Transformers'daki standart konumsal kodlamalar, girdi dizisindeki belirteçlerin mutlak veya göreli konumları hakkında bilgi sağlar. CoT, ara akıl yürütme adımları metin olarak üretildiğinde uygulandığında, bu adımlar sonraki üretimler için girdi dizisinin bir parçası haline gelir. Bu nedenle, model, orijinal girdi belirteçleri ile yeni üretilen CoT belirteçleri arasındaki ve ayrıca CoT belirteçlerinin kendi aralarındaki konumsal ilişkileri doğru bir şekilde yorumlamalıdır. Konumsal kodlamalar bu dizinin farklı bölümlerini yeterince ayırt etmezse veya CoT adımlarının sıralı doğasını iletmezse, modelin tutarlı çok adımlı akıl yürütme yapma yeteneği engellenebilir . CoT'nin etkinliği genellikle modelin bir sonraki "düşünceyi" üretirken doğru önceki CoT adımına dikkat etmesine bağlıdır ve konumsal kodlamalar bunu kolaylaştırmada rol oynar. CoT ile Transformers'ın ifade gücüne ilişkin bazı araştırmalar, CoT belirteçlerinin iç içe geçtiği veya eklendiği bu genişletilmiş dizileri işleme yeteneğine örtük olarak güvenir .

CoT'nin "karalama defteri" yönü, modelin ara hesaplama adımlarını veya düşüncelerini, genellikle insan tarafından okunabilir bir biçimde, nihai cevaba varmadan önce açıkça üretme uygulamasını ifade eder . Bu, bir insanın karmaşık bir problemi çözerken bir parça kağıda ara sonuçlar yazmasına benzer. Transformers bağlamında, bu karalama defteri, bu ara adımları temsil eden belirteçlerin otoregresif olarak üretilmesiyle gerçekleştirilir. Model standart üretim mekanizmasını kullanır, ancak prompt ve sonraki üretim bu adım adım çıktıyı teşvik edecek şekilde yapılandırılmıştır . Model daha sonra bu adımları birer birer üretir, her yeni adım orijinal girdiye ve daha önce üretilen tüm karalama defteri belirteçlerine bağlı olarak oluşturulur. Bu yinelemeli süreç, modelin, sabit derinlikli bir Transformer'ın tek bir ileri geçişteki doğal paralel işleme sınırlamalarını aşan hesaplamalar yapmasına olanak tanır. Karalama defteri etkin bir şekilde modelin kullanabileceği "hesaplama bandını" genişletir. Bu karalama defteri tabanlı CoT'nin sağlamlığı, gürültünün tanıtılması veya eğitim verilerindeki ara adımların atlanması gibi faktörlerden etkilenebilir. Araştırmalar, eğitim verisi dağılımı uygun şekilde rafine edilirse, Transformers'ın bazı ara "karalama defteri durumları" bozulmuş veya eksik olsa bile doğru durum izlemeyi öğrenebileceğini göstermiştir . Örneğin, karalama defterindeki bazı ara adımlar kasıtlı olarak atlanmışsa veya hatalar içeriyorsa, model bu boşlukları "atlamayı" veya gürültüyü düzeltmeyi öğrenebilir. Bu, dikkat başlıklarının karalama defterinin daha önceki, doğru kısımlarına veya orijinal girdiye odaklanmayı öğrenmesini içerebilir . Bu uyarlanabilirlik, akıl yürütme zincirlerinin her zaman mükemmel şekilde doğrusal veya hatasız olmayabileceği gerçek dünya uygulamaları için çok önemlidir. Transformer+CoT mimarisinin, modelin içinde örtük sonlu durum otomatları (FSA'lar) olarak temsil edilebilen sağlam algoritmalar öğrenme yeteneği, sıralı bilgileri ve durum izlemeyi işlemede bu karalama defteri yaklaşımının gücünü daha da vurgular . Karalama defterinin tasarımı, ara değişkenlerin veya durumların metinsel olarak nasıl temsil edildiği de performansı etkileyebilir.

4.4. Other Potential Architectural Enhancements

CoTFormer veya CHOOSE gibi doğrudan mimari gömmelerin ötesinde, Transformer mimarilerinde Chain–of–Thought (CoT) akıl yürütmeyi geliştirebilecek başka potansiyel mimari değişiklikler ve eğitim stratejileri bulunmaktadır. Bir alan, üretim sürecinin kendisinin değiştirilmesidir. Wang & Zhou (2024) tarafından detaylandırılan "CoT Reasoning without Prompting" (Prompt Kullanmadan CoT Akıl Yürütme) yaklaşımı, özenle hazırlanmış prompt'lara güvenmeye bir alternatif sunar . Açgözlü kod çözme yerine, bu yöntem her kod çözme adımında en iyi *k* alternatif belirteci araştırır. Bu alternatif yollara dallanarak, model aksi takdirde gizli kalacak akıl yürütme dizilerini ortaya çıkarabilir. Üretilen yollar daha sonra, ara adımlarda net bir düşünce süreci

gösteren, CoT akıl yürütmesi içerenlerin tanımlanması için analiz edilir. Bu yaklasım, modelin standart kod çözme ile etkinlestirilmeyebilecek ancak mevcut olabilecek içsel akıl yürütme yeteneklerinden yararlanır ve matematiksel, sağduyu ve sembolik akıl yürütme kıyaslamalarında gelismis¸performans göstermistir . Bu, bir modelin çıktı üretme seklinin, iç mimarisi kadar CoT'yi ortaya çıkarmak için de önemli olabileceğini düsündürmektedir.

İyilestirme için bir diğer yol, yalnızca kodçözücülü Transformers'ın akıl yürütme yetenekleri için sınırlayıcı bir faktör olarak tanımlanan ara katmanlardaki temsil çökmesini (representation collapse) ele almaktır. "Dummy pause tokens" (sahte duraklatma belirteçleri) kullanımıyla birlikte önerilen **Seq-VCR (Sequential Variance-Covariance Regularization)** yöntemi, ara temsillerin entropisini artırmayı ve çökmelerini önlemeyi amaçlar . Açık CoT denetimi gerektirmeyen bu teknik, zorlu bir 5x5 tamsayı çarpma görevinde %99,5 tam eslesme doğruluğu elde ederek, aritmetik akıl yürütmede önemli iyilestirmeler göstermis¸ve hatta bes¸-atıs¸CoT prompt'lu GPT-4'ü geride bırakmıstır . Bu, etkili çok adımlı akıl yürütme için Transformer'ın katmanları boyunca zengin ve çesitli temsillerin korunmasının önemini vurgular. Ayrıca, çıkarım sırasında keyfi sayıda adım için yinelemeli bir blok açan ve böylece model parametrelerini artırmadan veya belirteç üretimine güvenmeden CoT benzeri yetenekleri gelistiren, yinelemeli gizli alan akıl yürütmesine izin veren teknikler, gelistirmenin baska bir boyutunu sunar . Bu tür modeller, gizli alanda daha fazla hesaplama adımı gerçeklestirerek, daha derin bir "tartısma" yapmalarına olanak tanıyarak, karmasık akıl yürütme görevlerinde çok daha büyük modellerle karsılastırılabilir performans elde edebilir . CoTFormer'da olduğu gibi eski ve yeni belirteç temsillerinin iç içe geçtiği dikkat mekanizmalarındaki değisiklikler veya uzun bağlam gerçekliği ve CoT için "alım baslıkları" gibi belirli dikkat baslığı islevlerinin analizi , Transformers'da CoT'yi mimari olarak iyilestirmek için zengin bir tasarım alanına isaret etmektedir.

5. Practical Examples and Applications

5.1. CoT for Mathematical Reasoning

Chain-of-Thought (CoT) prompting, Büyük Dil Modellerinin (LLM) matematiksel akıl yürütme görevlerini, özellikle de matematiksel kelime problemlerini çözme yeteneklerini gelistirmede dikkate değer basarı göstermistir . Bu görevler genellikle çok adımlı hesaplamalar ve doğal dilde tanımlanan nicelikler arasındaki iliskilerin net bir sekilde anlasılmasını gerektirir. Modelden doğrudan bir cevap vermesi istendiğinde, modelin önemli ara adımları atlayabileceği veya problemin mantığını yanlıs¸yorumlayabileceği için

standart prompt teknikleri sıklıkla hatalara yol açar . CoT prompting, LLM'yi nihai cevaba varmadan önce akıl yürütme sürecinin her adımını açıkça ifade etmeye yönlendirerek bu sorunu ele alır . Örneğin, "Roger'ın 5 tenis topu var. Tenis toplarından oluşan 2 kutu daha satın alıyor, her birinde 3 top var. Şimdi kaç topu var?" problemi ele alındığında, CoT etkinleştirilmiş bir LLM şu şekilde bir akıl yürütme üretebilir: "Roger başlangıçta 5 topa sahipti. Her biri 3 toptan oluşan 2 kutu, 6 tenis topu eder. 5 + 6 = 11. Cevap 11'dir." Bu adım adım ayrıştırma, yalnızca doğru cevaba ulaşma olasılığını artırmakla kalmaz, aynı zamanda modelin "düşünce sürecini" şeffaf ve yorumlanabilir hale getirir .

CoT'nin matematiksel akıl yürütmeye uygulanması çeşitli kıyaslamalarda kapsamlı bir şekilde değerlendirilmiştir. Örneğin, Wei ve diğerleri (2022), GSM8K (ilkokul matematik kelime problemleri), SVAMP, ASDiv, AQuA ve MAWPS gibi veri kümelerinde elle hazırlanmış CoT gerekçelerini içeren birkaç-atış örnekleri sağlayarak önemli iyileştirmeler gösterdi . CoT prompting'in daha küçük modellerde matematiksel akıl yürütmeye minimal fayda sağladığını, ancak yaklaşık 100 milyar parametre veya daha fazlasına sahip LLM'lerde önemli kazançlar gösterdiğini, bu da CoT'yi daha büyük modellerde ortaya çıkan bir yetenek olarak işaret ettiğini buldular . Örneğin, CoT, GSM8K kıyaslamasında GPT-3 ve PaLM'in performansını iki katından fazla artırdı . CoT'nin matematikte kullanımını göstermek için kullanılan bir diğer yaygın örnek: "Bir tren 2 saatte 60 mil yol alıyor. Ne kadar hızlı gidiyor?" Bir CoT çıktısı şöyle olabilir: "Hız = Mesafe / Zaman. 60 / 2 = 30 mil/saat. Son cevap: 30 mil/saat" . Bu, CoT'nin modelin ilgili formülleri hatırlamasına ve uygulamasına ve hesaplamaları yapılandırılmış bir şekilde yapmasına nasıl yardımcı olduğunu gösterir.

Matematiksel akıl yürütme için farklı CoT prompting stratejileri mevcuttur. Birkaç-atış CoT, modele prompt'un bir parçası olarak birkaç örnek problem ve bunların ayrıntılı, adım adım çözümlerinin sağlanmasını içerir . Bu, modelin takip etmesi için net bir şablon sağlar. Sıfır-atış CoT ise, tipik olarak, modele açık örnekler olmadan kendi akıl yürütme zincirini oluşturması için "Adım adım düşünelim" veya "Adım adım ayrıştıralım" gibi basit bir talimatın problem ifadesine eklenmesini içerir . Örneğin, Kojima ve diğerleri (2022), "Adım adım düşünelim" prompt'unun LLM'lerden aritmetik ve sembolik akıl yürütme gibi görevler için CoT akıl yürütmesini ortaya çıkarmada son derece etkili olduğunu ve MultiArith gibi kıyaslamalarda performansı önemli ölçüde iyileştirdiğini buldu . Süreç genellikle iki adımlı bir prompt dizisini içerir: ilk olarak, CoT'yi üretmek için bir akıl yürütme prompt'u ve ikinci olarak, üretilen metinden nihai cevabı ayrıştırmak için bir cevap çıkarma prompt'u (örneğin, "Bu nedenle, cevap (Arap rakamlarıyla):") .

Matematiksel akıl yürütmede CoT için daha ileri gelismȩler arasında, LLM'nin doğal dil akıl yürütme yerine problemi çözen bir program (örneğin, Python'da) üretmeye yönlendirildiği Program Destekli Dil Modelleri (PAL) gibi teknikler yer alır. Bu program daha sonra cevabı almak için bir yorumlayıcı tarafından çalıştırılır . Pizza paylasım problemi ("Henry ve 3 arkadası̧, her biri 8 dilime kesilmiș7 pizza siparișediyor. Esiţ paylaştırılırsa kisi̧ basına kaç dilim düşer?") için bir PAL modeli söyle bir kod üretebilir: `numara_pizza = 7; dilim_pizza_basına = 8; toplam_dilim = numara_pizza * dilim_pizza_basına; kisi̧_sayısı = 1 + 3; kisi̧_basına_dilim = toplam_dilim / kisi̧_sayısı;` . Bu yaklasım, gerçek hesaplamayı güvenilir bir yorumlayıcıya aktararak, LLM'lerin sıklıkla yaptığı aritmetik hataları azaltır. Diğer yöntemler, ince ayar veya prompting için kullanılan CoT örneklerinin kalitesini ve çesitliliğini iyilestirmeye odaklanır, örneğin GPT-4'ü akıl yürütme zincirleri olusturmak için kullanmak veya Evol-Instruct gibi teknikleri kullanarak daha zorlu matematik problemleri olusturmak . DUP (Sorunları Derinlemesine Anlama) prompting yöntemi, LLM'yi önce ana soruyu ve problem çözme bilgilerini çıkarmaya yönlendirerek anlamsal yanlıșanlama hatalarını azaltmayı amaçlar ve ardından ayrıntılı CoT yanıtını üretir .

5.2. CoT for Logical and Sequential Problems

Matematiksel akıl yürütmenin ötesinde, Chain-of-Thought (CoT) prompting, ara adımların doğru sonuca ulasm̧ak için çok önemli olduğu çesitli mantıksal ve sıralı akıl yürütme görevleri için de etkili olmustur . **Sembolik akıl yürütmede**, sembolleri manipüle etmeyi veya mantıksal kuralları izlemeyi (örneğin, cebir, mantık bulmacaları) içeren CoT, modelin sorunu daha küçük, daha yönetilebilir çıkarımlara ayırmasına yardımcı olur . Örneğin, "A veya B, C'yi ima eder" ve "D, E'yi ima eder" kuralları ile "A doğrudur" ve "D doğrudur" gerçekleri verildiğinde ve C'nin doğruluk değeri sorulduğunda, CoT kullanan bir model sụ şekilde akıl yürütebilir: "A doğrudur. A veya B, C'yi ima eder; bu nedenle, C doğrudur" . Bu mantıksal adımların açıkça ifade edilmesi, tutarlılığın korunmasına ve doğrudan cevap üretiminde ortaya çıkabilecek hatalardan kaçınılmasına yardımcı olur. Sembolik akıl yürütme kıyaslamalarındaki performans, CoT ile önemli ölçüde iyilesm̧istir; PaLM 540B'nin doğruluğunun bir vakada yaklasık %60'tan %95'e çıktığı bildirilmistir .

Sağduyu akıl yürütmede, CoT, modellerin gerçek dünya bağlamını daha iyi çıkarmasına ve makul çıkarımlar yapmasına yardımcı olur . Sağduyu QA kıyaslamalarından alınan görevler, modelin kullandığı örtük bilgiyi ve mantıksal bağlantıları açıkça belirtmesine izin verdiği için CoT'den yararlanır . Örneğin, "Bir penguen uçabilir mi?" diye sorulduğunda, bir CoT söyle olabilir: "Penguenler kusţur. Çoğu kușuçabilir. Ancak penguenlerin uçmaya uygun kanatları yoktur, yüzgeçleri vardır ve yüzmeye adapte

olmuşlardır. Bu nedenle, bir penguen uçamaz." Bu adım adım ayrıştırma, nüanslı gerçek dünya bilgisinde gezinmeye yardımcı olur. CoT, ayrıca, bir dizi adımın belirli bir sırayla takip edilmesini gerektiren **algoritmik görevler** için de faydalı olabilir. Örneğin, bir listeyi sıralamak veya bir ağaç veri yapısında gezinmek, her adımın bir öncekinin sonucuna bağlı olduğu net bir prosedür gerektirir. CoT, LLM'nin bu algoritmik adımları simüle etmesine ve her birinin doğruluğunu kontrol etmesine olanak tanır. Bu, modellerin yalnızca desen eşleştirmenin ötesine geçerek, daha yapılandırılmış ve prosedürel problem çözme yeteneklerini geliştirmelerine yardımcı olur. Bu çeşitli uygulamalar, CoT'nin LLM'lerin yalnızca matematiksel değil, aynı zamanda geniş bir mantıksal ve sıralı problem yelpazesini çözme kapasitesini genişletmedeki çok yönlülüğünü göstermektedir.

6. Challenges and Considerations in CoT Implementation

6.1. Need for Intermediate Supervision

Chain-of-Thought (CoT) akıl yürütmenin etkili bir şekilde uygulanmasındaki temel zorluklardan biri, **ara denetime olan ihtiyaçtır**. Modelin doğru bir nihai cevaba ulaşmak için bir dizi ara mantıksal adımı doğru bir şekilde üretmesi ve birleştirmesi gerektiğinden, bu ara adımların doğruluğu kritik öneme sahiptir. Eğitim sırasında, modelin kendi ürettiği (ve potansiyel olarak yanlış) ara belirteçlere dayanarak sonraki adımları öğrenmeye çalışması, hataların birikmesine ve yayılmasına neden olabilir, bu da öğrenme sürecini zorlaştırır ve modelin istenen akıl yürütme yolunu içselleştirmesini engelleyebilir . Bu nedenle, **teacher forcing** gibi teknikler, model eğitimi sırasında doğru ara adımları sağlayarak bu sorunu hafifletmek için çok önemlidir . Ancak, yüksek kaliteli, adım adım açıklamalı veri kümelerinin elde edilmesi zor ve maliyetli olabilir. Bu, özellikle çok çeşitli problem alanlarını kapsayan kapsamlı CoT veri kümeleri oluşturmaya çalışırken geçerlidir. Bu nedenle, ara denetim ihtiyacını en aza indirebilecek veya otomatikleştirebilecek yöntemler (örneğin, kendi kendine tutarlılık kontrolleri, veri artırma veya daha büyük modellerden öğrenme) aktif araştırma alanlarıdır.

6.2. Error Propagation in CoT

Chain-of-Thought (CoT) akıl yürütmede **hata yayılımı**, modelin ürettiği ara adımlardaki hataların, sonraki adımları etkileyerek nihai cevabın yanlış olmasına yol açabileceği önemli bir sorundur. CoT süreci doğası gereği sıralı olduğundan, erken bir adımda yapılan bir hata, zincir boyunca ilerleyebilir ve sonraki tüm çıkarımları ve hesaplamaları bozabilir . Bu, modelin doğru bir akıl yürütme zinciri oluşturma yeteneğini ciddi şekilde baltalayabilir. Örneğin, bir matematik probleminde, bir önceki adımda yanlış bir ara

toplam hesaplanırsa, bu yanlış toplam üzerine inşa edilen tüm sonraki işlemler de yanlış olacaktır. Bu sorunu ele almak için çeşitli stratejiler geliştirilmiştir. **Teacher forcing**, eğitim sırasında modelin doğru ara adımları görmesini sağlayarak hata yayılımını önlemeye yardımcı olur . Çıkarım sırasında, **kendi kendine tutarlılık (self-consistency)** gibi teknikler, modelden birden fazla akıl yürütme zinciri üretmesini ve ardından en sık görülen nihai cevabı veya en tutarlı görünen zinciri seçmesini isteyerek hata yayılımının etkisini azaltabilir. Ayrıca, modelin ara adımlarını doğrulamak veya düzeltmek için **geri bildirim döngüleri** veya **etkileşimli prompting** kullanılması da düşünülebilir.

6.3. Generalization and Sample Efficiency

Chain-of-Thought (CoT) akıl yürütmenin uygulanmasında **genelleme ve örnek verimliliği** önemli hususlardır. CoT'nin temel amaçlarından biri, modellerin eğitim verilerinde görmedikleri yeni ve daha karmaşık problemlere genelleme yapma yeteneğini artırmaktır. Ancak, CoT'nin kendisi de genelleme sorunlarıyla karşılaşabilir. Örneğin, bir model, eğitim sırasında gördüğü belirli CoT kalıplarına çok fazla bağımlı hale gelebilir ve farklı türdeki problemler için yeni akıl yürütme yolları oluşturmakta zorlanabilir. Ayrıca, CoT, özellikle daha büyük modellerde, genellikle daha fazla hesaplama kaynağı ve daha uzun prompt'lar gerektirir, bu da örnek verimliliğini etkileyebilir. Teorik çalışmalar, CoT'nin aslında belirli görevler için örnek verimliliğini artırabileceğini öne sürmektedir, çünkü problemleri daha küçük, öğrenilmesi daha kolay adımlara ayırarak öğrenme görevini basitleştirir . Bununla birlikte, pratikte, etkili CoT için gerekli olan yüksek kaliteli, adım adım açıklamalı eğitim verilerinin elde edilmesi zor olabilir. Bu nedenle, daha az örnekle veya daha az açık denetimle iyi genelleme yapabilen CoT yöntemleri geliştirmek (örneğin, sıfır-atış CoT veya otomatik CoT üretimi) aktif bir araştırma alanıdır. Ayrıca, CoT'nin farklı problem türlerine ve alanlarına ne ölçüde genelleşebileceği ve daha küçük modellerde CoT yeteneklerini geliştirmenin en iyi yolları da devam eden araştırma konularıdır.