Irfan Khan

Abstract

In this paper, I address the limitations of our existing state of art artificial intelligence. I take the paper published by Apple called "Illusion of Thinking" which shows limitations in complex reasoning and state retention and address the issues there. While offering an alternative solution instead of just processing information, imagine an AI that can learn by trying, reflecting, and even backtracking, much like we do. Which works much better. In my theory I try to present a case to approach AGI, while trying to bridge multiple domains of education: Philosophy, computer science, Artificial intelligence, Machine learning, Data science etc. I try to prove it empirically while doing a falsifiability test.

Beyond just Attention: The AGI Solution

In this paper we will be building upon Apple's paper on "Illusion of thinking" and proposing a better alternative, let's quickly recap. The paper goes over "LLMs (Large language models) and "LRM" (Large reasoning model), which Apple demonstrate using towers of Hanoi, (For more info do check out the original paper called illusion of thinking) how both models fail when the reasoning gets more "complex" I've addressed this before why this happens

Reasoning in human works vastly different to AIs, we move on a dialectical progress of Trying, Reflect and Backtrack, but the rigid nature of Ai forces it to give outputs that does not go through this natural reasoning process. It's more of a dual nature of input and output, output often preceding input with prediction, while LRM works a bit differently with having actual structured process but it fails to address again similar nature. While it's really good at more mathematical and logical constructs which is due to its very nature, unfortunately this is not a form of intelligence, so Apple claims (mostly true) that reasoning models mimic human structure of intelligence, what happens at the end of the tower of Hanoi experiment is that at the end models break down due to its nature of reasoning

So what's the issue here? a very big one that stagnates our approach to AGI, spending billions of dollars in R&D of Reasoning models does not approach AGI, so all the investment in AI is very premature.

But I propose a different solution here, large intuitive models. The fundamental training of this model will be following human principle of "try" "reflect" "Backtrack" ie we'll process the reasoning and dataset with a dialectic approach instead of a strict mathematical one, instead of billions of token and output we use we use structured essays in training to move the language model in a similar fashion to humans, while this isn't strictly AGI, it's better than what LLMs and LRM does, and to keep a base to this structure since by itself it might pose the same threat as previous models we implement Hermeneutic reasoning what means here is that it works in similar a function of "attention mechanism" for context validation in paper ahead we will discuss why that's so important, so it's a multi layer training for datasets, this vastly improves the intuitive output of the AIs in a contrarian approach to present models.

Now one may ask how do we approach this? well there are a multiple ways to approach this, first is a curated dataset just using a common data set for pretaining won't do the job, i have prepared a dataset called TPS3 (triplet pair structure 3), here I take structure from CCPair from Microsoft but instead of query and passage system we expand it to 3 pair dialectical structure, now this is the base one can start from 300 billion to trillions of token depending upon the use. This database is VERY important as you'll see ahead that labelled data with two specific modes of reasoning and a synthesis for transformer architecture.

Now aside from dataset we also need to understand we need to make this model to be able to internally reason and not work on few shot or multiple shot prompts like how LLMs functions, in a similar function to LRM, now before that we must understand that what kind of transformer are we using? For reference GPT models like 4 and 5 uses only decoder while LRM like o1 and deepseek uses a Decoder plus CoT (chain of thought) due to the nature of our language model we have to use dual transformer ie encoder and decoder which separates it from nearly every LLM/LRM we have today, but this alone is not enough there are few NLP that does this. We need an intermediate like CoT used by LRM.

Now here I introduce a model called the chain of reasoning "CoR" which works in a similar manner to CoT aside from a few principle differences, it's basically a prompting style just like CoT. But it's more human like instead of simple "steps" of thinking we implement our traditional way of thinking, "try, reflect and Backtrack" but to avoid a rigid form we will not use it as intermediate we don't need some simulation so we will integrate into the transformer architecture itself, and along with CoR we will use another mode of reason we can use any philosophical dialectic as a placeholder for now I'll be going with hegelian dialectic so CoR does not become circular both will function as encoder, CoR $\rightarrow$ Encoder 1 & Dialectic $\rightarrow$ 2, feeding into a Decoder

Now to address the elephant in the room, what about attention? (for those who do not understand attention in this context, keep reading it's easy to understand with full context) In a transformer decoder there are two forms of attention. Single attention and Cross attention

Single attention : Each token in the decoder looks at other tokens in the decoder to build context. Example: When generating "the cat," the model looks at previously generated tokens.

Cross attention : Each token in the decoder looks at the output of the encoder (the latent representations). This allows the decoder to incorporate information from the input or multiple input streams while generating the output.

A quick mathematical format [Let $Q =$ Query (decoder token embedding) / Let $K = \text{Key}, V = \text{Value}$ (encoder hidden states)]

Now, Let us assume two encoders: Encoder A: CoR reasoning $\rightarrow$ hidden states H_A Encoder B: Dialectic reasoning $\rightarrow$ hidden states H_B

At each decoding step, it decides how much to rely on CoR vs dialectic reasoning

Now below I'll discuss a mathematical solution, for those who are not interested in that can skip as I've tried to give a jargon free interpretation on the mathematical solution:

Input sequence : $X = [x_1, x_2, ..., x_n]$ Decoder output sequence : $Y = [y_1, y_2, ..., y_n]$

Dual encoder :

$$H^{\text{CoR}} = \text{Encode}^{\text{CoR}}(X) \in \mathbb{R}^{n\times d}$$

Dialectic encoder:

$$H^{\text{Dialectic}} = \text{Encoder}^{\text{dialectic}}(X) \in \mathbb{R}^{n\times d}$$

Decoder with attention fusion :

$$Q_t = W_Q \times y_{<t} \quad \text{(decoder query)}$$
$$K^{\text{CoR}}, V^{\text{CoR}} = H^{\text{CoR}} \times W_K, H^{\text{CoR}} \times W_V$$
$$K^{\text{Dial}}, V^{\text{Dial}} = H^{\text{Dial}} \times W_K, H^{\text{Dial}} \times W_V$$
$$\alpha_t^{\text{CoR}} = \text{softmax}\left(\frac{Q_t \times (K^{\text{CoR}})^T}{\sqrt{d}}\right)$$
$$\alpha_t^{\text{Dial}} = \text{softmax}\left(\frac{Q_t \times (K^{\text{Dial}})^T}{\sqrt{d}}\right)$$
$$C_t = \alpha_t^{\text{CoR}} \times V^{\text{CoR}} + \alpha_t^{\text{Dial}} \times V^{\text{Dial}} \quad \text{(fused context)}$$
$$y_t = \text{DecoderStep}(y_{<t}, C_t)$$

(Reader context Image 1 : Decoder Query : WQ projects it into query space for attention, Image 2 : Key value pairs from relative context : Hdial could be dialogue context embeddings. Each is projected into keys (K) and values (V) via learned matrices Wk and Wv, image 3 : Attention weight : Softmax converts similarity scores to probability, image 4 : fused context : simple additive fusion combines both encoders, image 5 : decoder step.)

C_y = combined latent representation of CoR + dialectic reasoning

Decoder generates output y_t conditioned on both streams

Training objective : Lambda : $\Lambda = \text{crossEntropy}(Y, Y')$, [where Y = True output sequence (the ground trut
& Y' = model's predicted output sequence]

Add intermediate supervision for each encoder to encourage specialization

$$\Lambda_{\text{total}} = \Lambda + \lambda(\Lambda_{\text{CoR}} + \Lambda_{\text{dial}})$$

Λ_{CoR} = loss for the CoR encoder output (how well the CoR module predicts its intermediate reasoning ste

Λ_{dial} = loss for the dialectic encoder output (how well the dialectic module predicts its intermediate reaso

λ = a weight to control how much you care about these intermediate losses relative to the final answer

Now, the mathematical solution is done. We will give a simpler jargon-free interpretation here.

1. Two "thinking modules" process the same input in parallel:

One thinks step-by-step (CoR)... One thinks dialectically (thesis $\rightarrow$ antithesis $\rightarrow$ synthesis)

2. The decoder acts like a smart writer:

At each word it's about to write, it "looks" at both reasoning modules. it decides how much to trust CoR vs dialectic reasoning for that word.

3. The output is a single answer that combines:

Logical clarity from CoR, Insightful synthesis from dialectic reasoning

Now this begs another question: what really separates these two we can of course understand the distinction between what about the machine?

Well this why i said we need hermeneutic reasoning

Let us assume α and β are dynamic weight decided by hermeneutic function

$$\text{decoder output} = \alpha \times \text{CoR latent} + \beta \times \text{dial latent}$$

Now, $\alpha_t = f(\text{CoR latent}_t \text{ context})$, $\beta_t = g(\text{Dial latent}_t \text{ context})$

here f and g are learned function that map latent vectors and current decoder state to attention weight

--

Here we begin the part two let's start

1 Experiment

I suggest everyone to read the paper themselves or at least have a grasp of towers of Hanoi experiment as i will be building in regards to that, nonetheless I do explain it here at least the core concepts

So the experiment gives the ai models 3 towers with disks, The goal is to move a stack of four disks from a starting peg (A) to a destination peg (C), using an auxiliary peg (B). There are two rules:

1. Only one disk can be moved at a time.

2. A larger disk can never be placed on top of a smaller disk.

A standard Large Language Model (LLM) approaches this by predicting the most plausible next step in a sequence. It has seen many examples of the puzzle but doesn't maintain a true "state" of the board or understand the rules in a functional way.

Its process might look like this :

Move 1: Move disk 1 from A to B. (Correct)

Move 2: Move disk 2 from A to C. (Correct)

Move 3: Move disk 1 from B to C. (Correct, but strategically questionable)

Move 4: Move disk 3 from A to B. (Correct)

Move 5: Move disk 1 from C to A. (Correct)

Move 6: Move disk 2 from C to B. (Correct)

Move 7: Move disk 1 from A to B. (Correct)

As this goes on the disk set increases we start to see failure of LLMs near the 7-8th disk. A 4-disk puzzle requires 15 moves, which is manageable for many models. However, a 7-disk puzzle requires 127 optimal moves. This leap in complexity is where standard AI models break down now this happens for multitude of reasons

1. Loss of State : Over a sequence of 127 steps, a standard LLM loses track of which disk is on which peg. Its attention mechanism cannot reliably

maintain an accurate "mental model" of the game board for that many moves. 2. Sequential Prediction Errors: The model's process is to predict the next plausible-sounding text. After dozens of moves, it is highly likely to generate a move that is syntactically correct ("Move disk X to peg Y") but is illegal based on the actual, forgotten state of the game (e.g., placing a larger disk on a smaller one). 3. No Error Correction: Once an LLM makes an illegal move, it has no built-in mechanism to recognize the error and backtrack. It simply continues generating the sequence, leading to a completely failed solution. (Bingo as we said no backtracking mechanics)

Now let's address how large intuitive Language addresses this

1. The Dialectic Encoder :

Thesis: Move 7 disks from peg A to peg C.

Antithesis: The rules (one disk at a time, no larger on smaller) create a complex sub-problem.

Synthesis: The overarching strategy is to first move the top 6 disks from A to B, then move the 7th disk from A to C, and finally move the 6 disks from B to C. This high-level plan, held by Encoder B, prevents the model from getting lost in strategically pointless moves.

2. CoR Encoder:

For every single one of the 127 moves, the system would execute its core reasoning loop:

Try: Propose a move (e.g., "Move disk 4 from A to B").

Reflect: Internally check the move against the current, actively tracked state of the pegs. "Is disk 4 the top disk on A? Yes. Is peg B empty or does it have a disk larger than 4? Yes." It also checks if the move aligns with the high-level strategy from the dialectic encoder.

Backtrack: If the reflection reveals an illegal move (e.g., proposing to move disk 5 onto disk 2), the system immediately discards that option before generating it as output. It then tries an alternative valid move. (Again! proves the claim how important the backtracking model is)

Because this validation happens for every single step, the system's integrity does not degrade over the 127-move sequence. It is not relying on a fragile, long-term memory of the entire text sequence but on a robust, state-aware validation loop that resets at each step.

Symbolic argument :

S_t = state of the pegs at time t

M_t = proposed move at time t

valid(M_t, S_t) = a function that that returns true if move M_t obeys hanoi rules

Then the "try $\rightarrow$ reflect $\rightarrow$ backtrack" loop can be written as :

```
while not goal(S_t):
    M_t = propose_move(S_t)
    if Valid(M_t, S_t):
        S_{t+1} = apply_move(M_t, S_t)
    else:
        discard M_t
```

```
backtrack()
```

Proof Sketch:

1. Base Case ($n = 1$): Only one disk, any move obeys rules $\rightarrow$ trivially correct.

2. Inductive Step: Assume the system is correct for k disks. For $k+1$ disks:

Any proposed move of disk $k+1$ is checked by Valid().

If invalid (placing on smaller disk), backtrack() prunes the move.

By induction, all moves for disks $\leq k$ are also validated.

thus the system never executes invalid move, unlike standard LLMs

Falsifiability test :

1. Falsifying the dialectic encoder.

- The Hypothesis: The dialectic encoder (Encoder B) is crucial for solving problems that require long-term strategy and avoiding "garden path" problems where locally optimal moves lead to a dead end.

- The Test (Strategic Trap Puzzles): Design puzzles with deceptive but tempting initial moves that lead to an unsolvable state. A classic example is a modified chess puzzle where an early, aggressive move appears advantageous but forces a loss 20 moves later. You would compare the full model against a version with only the CoR encoder active.

- Condition for Falsification: The full model, with its dialectic encoder, falls for these strategic traps at the same rate as the model without it. If the "strategic oversight" provides no measurable benefit in avoiding these long-term blunders, its essential role in the architecture would be

2. Falsifying the "Try, Reflect, Backtrack" Mechanism

- The Hypothesis: The try, reflect, backtrack cycle is essential for maintaining soundness and will prevent the model from making strategically poor or illegal moves.

- The Test (Ablation Study): Create two versions of your model: the full model and a "lesioned" version where the reflect and backtrack functions are disabled. This version can still "try," but it must commit to its first generated action, much like a standard LLM. Both models are then tasked with solving a series of complex logic puzzles that are prone to errors, like the 7-disk Tower of Hanoi or constraint-satisfaction problems (e.g., Sudoku) .... - Condition for Falsification: The performance of the full model is not significantly better than the lesioned model. If your complete model still makes illegal moves, gets stuck in loops, or performs comparably to the version without its core reasoning cycle, then the claim that this mechanism is the key to robust reasoning would be falsified

2 Sources and reference

Journals and Conferences:

- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Å., & Polosukhin, I. (2017). *Attention Is All You Need.* Advances in Neural Information Processing Systems (NeurIPS).

- Zhao, W. X., Zhou, K., Li, J., Liu, T., Chen, Y., Cheng, X., Wang, Y., Wu, H., Wang, W., Guo, A., Maul, X., Zhang, Y., Wang, F., Cheang, W. L., Liu, M., Liu, P., Wang, J., Liu, Q., Li, H., Yang, Y., & Liu, Z. (2023). *A Survey on Large Language Models.* arXiv preprint arXiv:2303.08774.

- Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Roberts, A., Clark, J., Lee, S., Gordon, A., Du, N., Zhou, Y., Bahri, D., Garcia, X., Abu-El-Haija, S., Edison, J., Roy, A., Omernick, M., Michalewski, K., Khan, M., Avati, A., Kelley, K., Ter-Sarkisov, G., Austin, J., Bieber, D., Farajidoust, A., Hwang, K., Robinson, K., Ng, A., Chen, A., Zheng, R., MacMahon, C., Lee, K., Kavukcuoglu, K., Liu, R., Salakhutdinov, R., Wei, J., Luan, D., Liu, H., Yuan, A., So, D., & Schmidhuber, J. (2022). *PaLM: Scaling Language Modeling with Pathways.* arXiv preprint arXiv:2204.05814.

Online Resources / Preprints:

- Apple Machine Learning Research. (2024). *The Illusion of Thinking.* Retrieved from `https://machinelearning.apple.com/research/illusion-of-thinking`

- Stanford Encyclopedia of Philosophy. (n.d.). *Hegel's Dialectics.* Retrieved from `https://plato.stanford.edu/entries/hegel-dialectics/`

- Stanford Encyclopedia of Philosophy. (n.d.). *Hermeneutics.* Retrieved from `https://plato.stanford.edu/entries/hermeneutics/`

- Sinha, K., Welleck, S., Xu, J., Wu, Z., Yuan, X., Lee, Y. S., Fang, H., Li, X. L., & Gao, J. (2023). *Text Embeddings by Weakly Supervised Contrastive Pre-training.* Microsoft Research. Retrieved from `https://www.microsoft.com/en-us/research/publication/text-embeddings-by-weakly-supervised-contrast`