# ToTRM : Tree-of-Thought Tiny Recursive Model

*Junhyeok Lee*[1], *MyeongHyun Kim*[1]
[1]Department of Software, College of IT, Soongsil University, Seoul, South Korea

## 주제

- 본 프로젝트에서 제안하는 **ToTRM(Tree-of-Thought Tiny Recursive Model)**은 LLM 의 CoT 기법 중 하나인 ToT(Tree-of-Thought) 기법을 TRM 아키텍처에서 구현한다.
- 현재의 LLM(Large Language Model)들은 CoT(Chain-of-Thought)와 같은 프롬프팅 기법을 통해 복잡한 추론 문제를 해결하지만, 수십억~수천억 파라미터의 거대한 크기와 높은 계산 비용이 필요하다.
- 삼성 SAIL 몬트리올의 연구진이 2025년 10월 arxiv에서 제안한 **TRM(Tiny Recursive Model)**은 7M 파라미터의 작은 크기로 기존 LLM의 성능을 몇 가지 어려운 task에서 능가하지만, 선형적 추론 경로만 탐색하여 복잡한 문제 해결 능력에 한계가 있다.
- ToTRM은 Sudoku Extreme 데이터셋에서 **단 9%의 추가 파라미터를 사용해 TRM 대비 75%에서 80%로 5%p의 accuracy 개선**을 달성한다. 또한 ablation study를 통해 각 component의 효과를 검증하였다. 이를 통해 작은 모델도 올바른 구조 설계로 복잡한 추론이 가능함을 입증하고, LLM 없이도 효율적인 추론 시스템 구축 가능성을 제시한다.

## 배경

- TRM

  **TRM(Tiny Recursive Model)**은 7M 파라미터의 작은 크기의 신경망으로, 복잡한 추론 문제를 해결하기 위한 재귀적 추론 모델이다. sudoku extreme, ARC-AGI 등에서 HRM과 deepseek r1, gemini 2.5 pro 등 cloud 기반 LLM을 능가했다.

  TRM은 transformer block으로 구성된 하나의 network를 사용해, z와 y를 recursive하게 계산한다. 이때 y는 실제 정답을 예측하는 embedding이고, z는 reasoning을 위한 중간 embedding이다.

- Tree-of-Thought

  CoT(Chain-of-Thought)는 "단계 별로 설명해줘"와 같은 말을 프롬프트를 함께 입력하는 것으로 언어 모델이 단계별로 추론하도록 유도해 문제 해결 능력을 향상시키는 프롬프팅 기법이다.

  **ToT(Tree-of-Thought)**(NeurIPS 2023)는 CoT 기법의 하나로, ToT에서는 추론 중간에 분기(branch)하여 트리 구조로 여러 추론 경로를 동시에 탐색하고 평가하여 최적의 답을 도출한다.

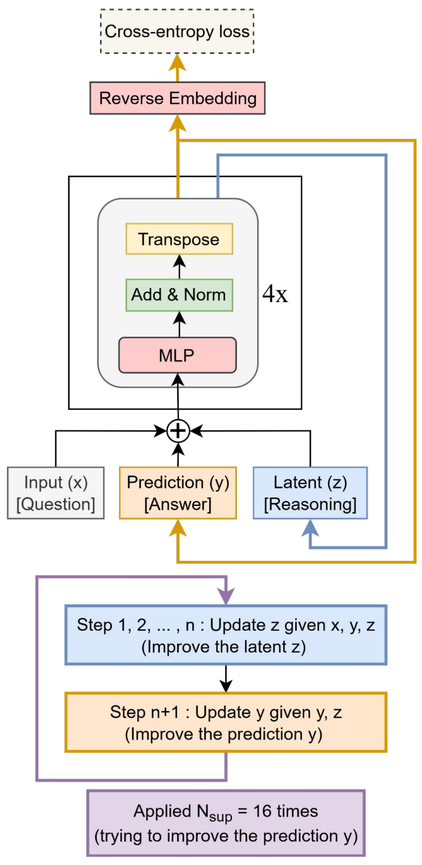


***Figure 1.***
*TRM architecture*

## ToTRM

- **ToTRM(Tree-of-Thought Tiny Recursive Model)**은 TRM에 ToT의 아이디어를 적용하여, TRM의 선형적 추론을 트리 구조로 확장한 모델이다.
- ToTRM의 method는 branching과 merging으로 나누어 볼 수 있다. ToTRM은 이런 아키텍처로 layer 수준의 ToT를 구현하여 병렬적으로 다양한 추론 경로를 탐색하고 가장 좋은 경로를 선택한다.

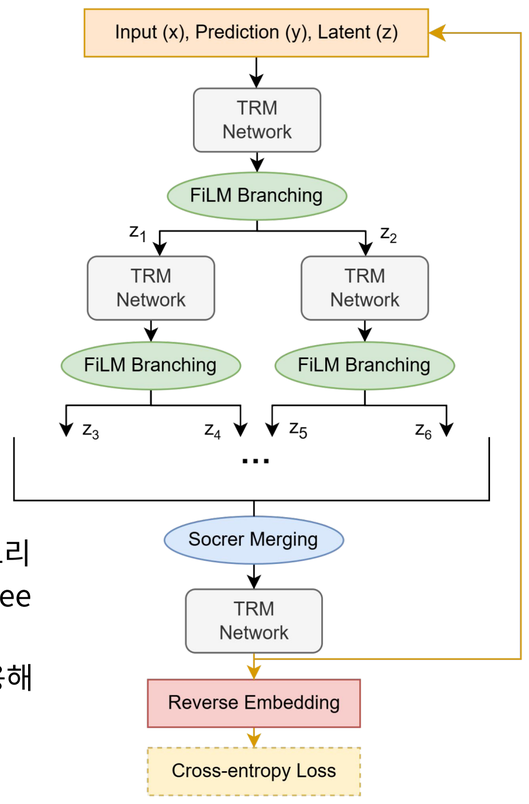


***Figure 2.***
*ToTRM architecture*

## Branching 기법

- Branching은 z에 대한 recursion step에서 이진 트리 형태로 수행한다. 즉, branching step이 n번이면 tree width는 $2^n$이 된다.
- 이때 z를 단순 복사하는 대신, FiLM의 method를 사용해 branching했다.
- FiLM

  **FiLM(Feature-wise Linear Modulation)**(AAAI 2018)은 신경망 내의 중간 feature map을 conditioning하는 기법으로, VQA나 style transfer 등에서 사용된다.

  z embedding 별로 동일한 크기의 vector γ, β를 사용해서, γ는 element wise하게 곱하고, β는 더한다.

  γ, β는 길이 n인 embedding에 대해 n x 2n 행렬을 곱하는 linear layer로 구했다.

- branch들이 서로 다른 예측을 하도록 branch 간 코사인 유사도로 **diversity loss**를, branch들이 정답을 예측하도록 각 branch가 예측한 결과와 정답 사이의 cross entropy로 **branch loss**를 사용했다.

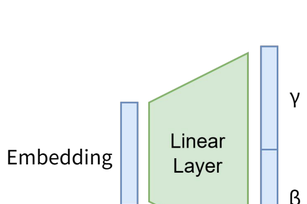


***Figure 3.***
*FiLM γ, β linear layer*

## Merging 기법

- 나뉘어진 각 branch들은 y를 계산하기 직전에 Merge된다.
- Merging에는 작은 MLP인 **scorer**를 사용해 각 branch에 대한 score를 계산하고, score에 따라 branch들을 가중합하는 method를 사용했다. 이를 통해 각 branch이 담고 있는 정보를 적절히 합쳤다.
- 이때 score를 활용한 가중합은 token level과 branch level에서 각각 수행했다.
  - **Token Level Scoring**: 각 branch에는 sequence length만큼의 embedding이 존재한다. 이에 따라 branch 내부의 각 embedding들에 대해 scorer로 score를 계산하고 가중합하여, branch에 대한 정보를 가진 하나의 **branch embedding**을 도출한다.
  - **Branch Level Scoring**: branch 별로 존재하는 branch embedding에 대해 scorer로 score를 계산하고, 해당 값으로 전체 embedding에 대해 가중합하여 branch를 합친다.
- Scorer가 branch를 잘 선택하도록 각 branch의 실제 accuracy와 scorer의 예측 score에 대한 cross entropy를 한 **scorer loss**를 사용했다.

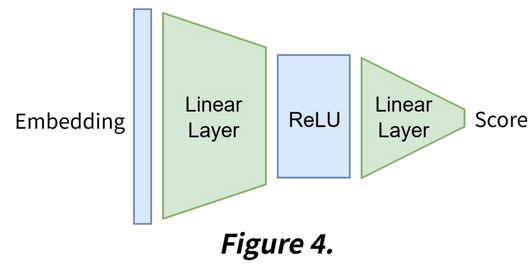


***Figure 4.***
*Scorer MLP architecture*

## 결과

- Sudoku Extreme 데이터셋에서 (train 1,001,000개, test 2,113개) 실험했다. ToTRM method에 의한 부분은 pretrain하고, pretrain이 완료된 TRM checkpoint를 가져와 함께 fine-tuning하는 식으로 학습시켰다. Branching step은 3으로 했다(8개의 branch).
- Accuracy

  다음과 같이 ablation study를 하며 accuracy를 찍어봤고, ToTRM이 sudoku extreme에서 TRM에 비해 5%p 정도의 accuracy 향상을 보였다. 또한 FiLM, scorer, 새로 추가한 loss들의 유의미함을 확인했다.

| Model | Accuracy | Branch cosine similarity |
|---|---|---|
| (TRM) Base | 0.750 | - |
| Base + add-only FiLM + mean merge | 0.761 | 0.899 |
| Base + FiLM + diversity loss + mean merge | 0.761 | 0.187 |
| Base + FiLM + diversity loss + mean merge + branch loss | 0.763 | 0.394 |
| **(ToTRM) Base + FiLM + diversity loss + scorer merge + branch loss + scorer loss** | **0.801** | **0.385** |

***Table 1.***
*Accuracy on sudoku extreme dataset*

- Model Size

  Model size는 다음과 같다. ToTRM method에 의한 추가 파라미터는 725,378 (0.73M) 개로, ToTRM 전체의 9% 수준이다. ToTRM이 TRM에 비해 소폭 증가했지만, LLM에 비하면 여전히 아주 작은 수치이다.

  TRM: 6.83M (6,829,058)
  ToTRM: 7.55M (7,554,436)
  GPT-OSS: 120B (120,000,000,000)
  Deepseek R1: 671B (671,000,000,000)

- Latency

  ToTRM, TRM 각각 10개의 샘플에 대해 latency를 측정해 평균낸 결과, ToTRM은 sudoku 샘플 하나당 173ms, TRM은 sudoku 샘플 하나당 133ms의 latency를 확인했다. 마찬가지로 ToTRM이 TRM에 비해 증가했지만, LLM에 비하면 여전히 아주 작은 수치이다.

- 성능 향상의 선형대수학적 이해
  - 모델의 추론 과정은 embedding vector가 vector space에서 이동하는 것으로 생각할 수 있다. TRM는 하나의 추론 경로만 가지므로 활용하고 탐색할 수 있는 정보의 범위가 span{z}이지만, ToTRM은 branch가 n개인 경우 span{$z_1$, $z_2$, ..., $z_n$}이다. 즉, embedding의 표현 공간이 확장된 것으로 이해할 수 있다.
  - 코사인 유사도를 활용한 diversity loss로 각 branch에서 표현의 다양성을 확보했다.

## 결론

- ToTRM은 TRM에 LLM CoT 기법인 ToT를 적절히 적용하였고, 9% 정도의 model size 증가로 sudoku extreme에 대해 accuracy를 5%p 가량 향상시켰다.
- 이를 통해 작은 모델로도 복잡한 다중 경로 추론이 가능함을 보였다. 즉, LLM의 거대한 크기와 비용 없이도, 특정 task에 대해서 추론 능력을 향상시킬 수 있는 아키텍처적 접근의 가능성을 제시한다.
- 향후 연구 방향으로는 ToT를 넘어선 Graph-of-Thought로 ToTRM을 확장하거나, 다른 task에 대해서 결과를 뽑아 보는 시도 등이 가능할 것으로 보인다.

## Reference

1. **Jolicoeur-Martineau, Alexia. "Less is more: Recursive reasoning with tiny networks." arXiv preprint arXiv:2510.04871 (2025)**
2. **Yao, Shunyu, et al. "Tree of thoughts: Deliberate problem solving with large language models." Advances in neural information processing systems 36 (2023)**
3. **Perez, Ethan, et al. "Film: Visual reasoning with a general conditioning layer." Proceedings of the AAAI conference on artificial intelligence. Vol. 32. No. 1. 2018.**