Mixture of Parrots: Experts improve memorization more than reasoning

Samy Jelassi Clara Mohri David Brandfonbrener Alex Gu
Nikhil Vyas Nikhil Anand David Alvarez-Melis Yuanzhi Li Sham M. Kakade Eran Malach
presenter: Shen Yuan

中国人民大學 RENMIN UNIVERSITY OF CHINA
高瓴人工智能学院 Gaoling School of Artificial Intelligence

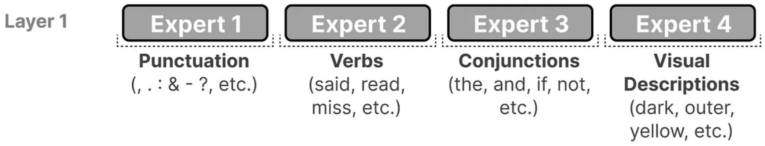
Outline

Outline

Introduction

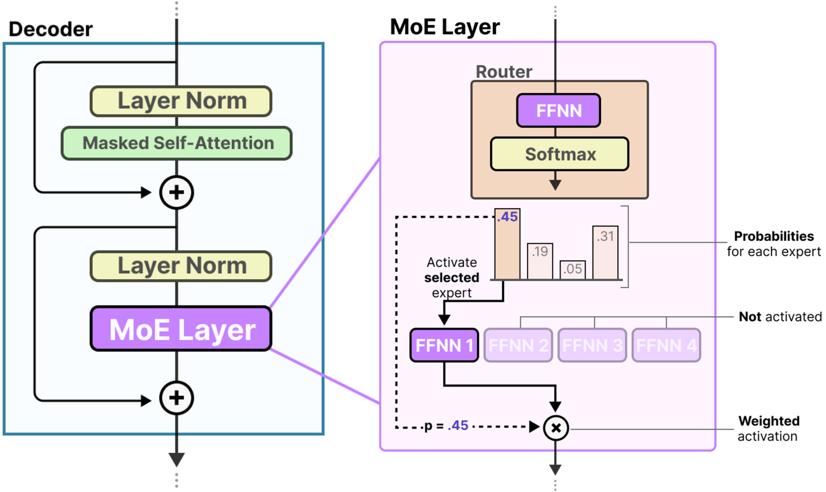
Mixture of Experts (MoE) is a technique that uses many different sub-models (or "experts") to improve the quality of LLMs.
Two main components define a MoE:

- **Experts:** Each FFN layer has a set of "experts" of which a subset can be chosen. These "experts" are typically FFNNs themselves. It can learn syntactic information on a word level.
- **Router or gate network:** Determines which tokens are sent to which experts.



Introduction

A common MoE-based Transformer decoder architecture:



Outline

Preliminary

Define the parameters as Q_h, V_h, $K_h \in \mathbb{R}^{m \times m}$, $\phi : \mathcal{X} \to \mathbb{R}^m$, $\psi : \mathbb{R}^m \to \mathbb{R}$. The output of a one-layer Transformer f is:

$$f(\boldsymbol{x} - 1, ..., \boldsymbol{x}_N) = \psi\Big([\mathrm{softmax}\big(\phi(x_N)^\top Q_h K_h^\top \phi(X)\big)\phi(X)\, V_h]_{h \in [H]}\Big) \tag{1}$$

- **Dense Transformer:** $\psi(\boldsymbol{x}) = \boldsymbol{u}^\top \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$ for $\boldsymbol{W} \in \mathbb{R}^{m' \times m}$, $b \in \mathbb{R}^{m'}$, $u \in \mathbb{R}^{m'}$
- **Sparse (MoE) Transformer with K experts:**
  $\psi(\boldsymbol{x}) = \boldsymbol{u}_i^\top \sigma(\boldsymbol{W}_i \boldsymbol{x} + \boldsymbol{b}_i)$ for $i = \arg\max_j \boldsymbol{r}_j^\top \boldsymbol{x}$

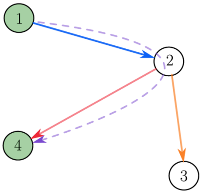
  where $\boldsymbol{W}_1, ..., \boldsymbol{W}_k \in \mathbb{R}^{m' \times m}$, $\boldsymbol{b}_1, ..., \boldsymbol{b}_k \in \mathbb{R}^{m'}$, $\boldsymbol{u}_1, ..., \boldsymbol{u}_k \in \mathbb{R}^{m'}$ are the parameters of each expert and $r_1, ..., r_k$ define the routing function (we use top-1 routing).

Graph reasoning

Problem Definition:

Definition 2.1

(Length-2 Path Problem). The input is a graph $G = (V, E)$. The source $s \in V$ and a destination $d \in V$ are fixed for all tasks as the 0 and $|V|$ vertex. The length-2 path problem asks whether there is a path of length 2 from s to d.



Graph reasoning

Theorem 2.1

(Length-2 path lower-bound on sparse (MoE) transformers). *For some input sequence* $G = (V, E)$*, fix two disjoint subsets* $A, B \subset [N-1]$*, and consider a single-layer transformer* $f \in \mathrm{Transformer}^N_{m,H,1,K}$ *with* $O(\log N)$*-bit precision that solves length-2 path for any input* X *where* X_A *is a function of edges with the source* s*,* X_B *is a function of edges with the destination* d*. Then,* f *has width satisfying* $mH = \Omega(|V| / \log N)$*.*

Theorem 2.2

(Length-2 path width upper bound for dense transformer). *There exists a transformer of width* $m = |V|$*,* $H = 1$*, and* $O(\log N)$*-bit precision that solves length-2 path problem for any input.*

Graph reasoning

Corollary 2.1

Consider a sparse transformer (with K experts) and a dense transformer with the same number of parameters. There exists a number of experts K so that the the sparse model is not able to solve the reasoning task, but the dense transformer solves the task.

Proof.

- **Dense Transformer** with the width m_d and $O(m_d^2)$ parameters;
- **MoE** with the width m_s and $O(Km_s^2)$ parameters , and K experts;

In order to match the number of parameters, it must be the case that $m_s = m_d/\sqrt{K}$. Suppose we let $m_d = |V|$, as this is sufficient to solve the above problems. For any $K \geq \Omega\big((\log N)^2\big)$, the sparse model is not sufficiently wide to solve the problem.

Outline

Memory-intensive tasks

Dataset: $\{(X^i, y_i)\}_{i=1}^n$ where $X^i \in \mathbb{R}^{N\times m}$ and $X^i[j]$ is sampled from $\mathcal{N}(0, I_m)$. We assume $y_1, ..., y_N \in \{\pm\mathbf{1}\}$ are arbitrary labels for the n sequences.
Objective: Given X^i, the goal is to predict y_i.

Upper-bound on MoE for memorization

Theorem 3.1

With probability at least 0.99, there exists a one-layer MoE transformer with K experts, using $\tilde{O}\big(\frac{n}{K} + Km\big)$ active parameters and $\tilde{O}(n + Km)$ total parameters stored in $\tilde{O}(1)$ bits that, when applied to each sequence X_i, outputs at the last token a value whose sign matches y_i, i.e., $sign\big(f(X_i)\big) = y_i$ for all $i = 1, \ldots, n$.

Specifically, if we choose $K = \sqrt{n/m}$ we get that an MoE architecture can solve the memorization problem with $\tilde{O}(\sqrt{nm})$ active parameters.

Lower-bound on dense Transformer for memorization

Theorem 3.2

Given the same task as above, a dense Transformer requires $\tilde{\Omega}(n)$ parameters to solve the memorization task.

- **Dense Transformer** requires $\tilde{\Omega}(n)$ parameters;
- **MoE** requires $\tilde{O}(\sqrt{nm})$ active parameters and $\tilde{O}(n + Km)$ total parameters;

So, for $n \gg m$, MoEs are significantly more efficient.

Outline

Synthetic Experiments

We mainly focus on two tasks:the **shortest path** task and the **phone-book** task.

- **Shortest path task:** The training set size is $1e6$ and the test set size is $1e3$.
- **Phone-book task:** We vary the training set size over $\{1e5, 5e5, 1e6, 1.5e6, 2e6, 2.5e6, 3e6\}$ and the test set consists of $1e3$ queries from the training set.

Synthetic Experiments

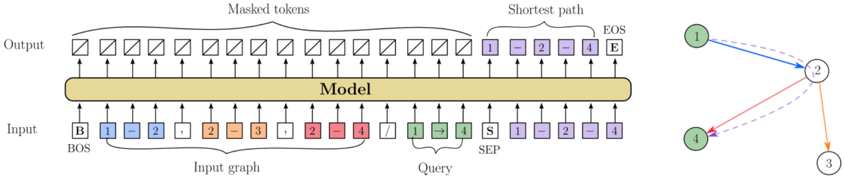
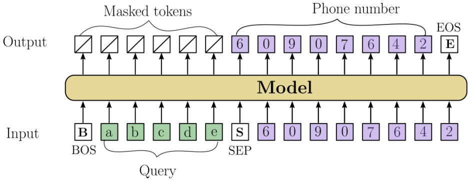


Figure 2: Illustration of the shortest path task. We feed the model with a sequence that lists all the edges in the input graph and ends with the query (in green) which asks the model to find a shortest path between two vertices (from vertex 1 to vertex 4 in the figure). The model then autoregressively returns the shortest path (in purple).

Shortest path task We vary $n \in \{25, 30, 50, 40, 45, 50, 55\}$ and set p such that the average length of the shortest path is 3.5.

Synthetic Experiments



Name	☎
abcde	60907642
fghij	40652457
klmno	20826343
pqrst	53143138
uvwxy	80506194

Figure 3: Illustration of the phone-book task for closed-book retrieval. The model is first trained to memorize a phone-book (illustrated on the right). Then, we randomly select a name in the phone-book (in green) and ask the model to return their phone number (in purple) without access to the phone-book.

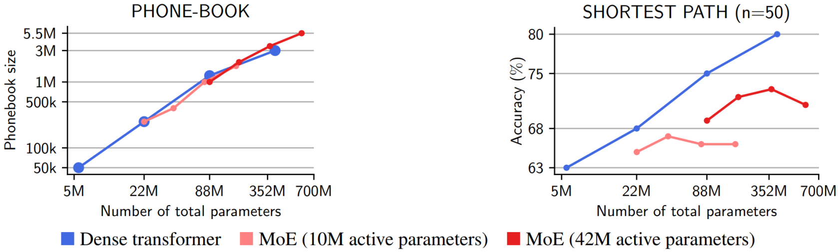
Phone-book task We generate phone-books where the names consist of 5 letters and the phone numbers of 8 digits. We ensure that both the names and numbers are unique.

Synthetic Experiments

Architecture:

- **Dense:** Mistral with the number of layers $L = 12$, and the width $d \in \{256, 512, 1024\}$;
- **MoE:** Mixtral with the number of layers $L = 12$, the width $d \in \{256, 512\}$, and the number of experts $E \in \{8, 16, 32, 64\}$;

We use token-choice routing, and each token is routed to the top-2 experts.



(a) Phone-book memorization

(b) Shortest path (50-node graphs)

Figure 4: **(a) Phone-book memorization**: We train a series of dense transformers and MoEs on phone-books of varying sizes and then evaluate their memorization capacity. We report the maximal phone-book size where the model obtains more than 90% accuracy. The maximal phone-book size correlates with the total (and not active) number of parameters. **(b) Shortest path (total parameters)**: We train models to find the shortest path in 50-node graphs and report the test accuracy. Here, increasing the number of experts provides limited improvements and the performance rather correlates with the number of active parameters.

Outline

Experiments

Architecture:

- **Dense:** Transformer with the number of layers $L = 20$, and the width $d \in \{256, 512, 1024, 2048, 4096\}$;
- **MoE:** Transformer with the number of layers $L = 20$, the width $d \in \{256, 512, 1024\}$, and the number of experts $E \in \{8, 16, 32, 64\}$;

We use token-choice routing, and each token is routed to the top-2 experts.

Pre-training datasets:

We train two collections of models by using these two dataset:

- The **natural language** dataset is a mixture constituted of FineWeb-edu, Cosmopedia, Wikipedia and the training sets of the downstream tasks we evaluate on;
- The **math** dataset is a mixture made of Proof-Pile 2 and instruction datasets such as OpenMathInstruct and MetaMathQA.

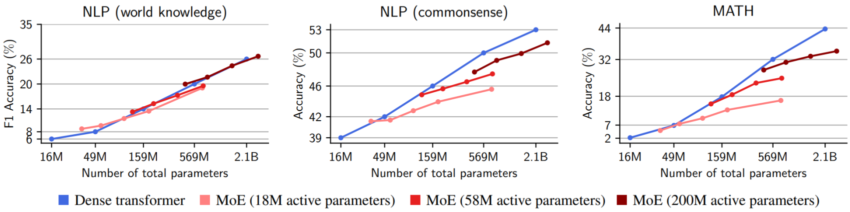
Each of the two training mixture approximately totals 65B tokens.

Experiments

Evaluation:
We measure the validation perplexity on 5,000 held-out sequences sampled from the training distribution. And we evaluate our models on a series of natural language and math benchmarks. Explicitly, we divide them into three categories:

- World-knowledge tasks: TriviaQA, Natural Questions, HotpotQA , WebQuestions, ComplexWebQuestions.
- Commonsense tasks: ARC-C and ARC-E, CommonsenseQA, HellaSwag, OpenbookQA, PIQA, SciQ, SIQA, WinoGrande.
- Math benchmarks: SVAMP, GSM8k, GSM-Hard, Hendrycks-MATH and Minerva-MATH.

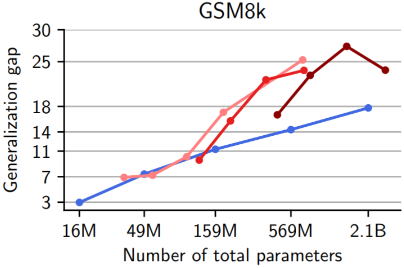
Experiments



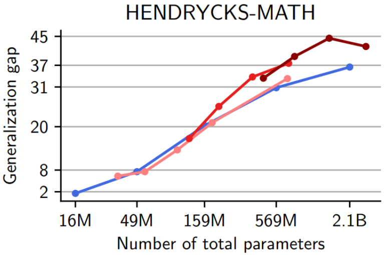
(a) Evaluation: world knowledge (b) Evaluation: commonsense (c) Evaluation: math

Figure 1: **(a) Evaluation: world knowledge.** We train a series of dense transformers and MoEs on 65B tokens from a corpus essentially made of Fineweb-edu, Cosmopedia and Wikipedia (see Section 5 for details). We then evaluate the models on several world knowledge benchmarks (e.g., TriviaQA (Joshi et al., 2017), Natural Questions (Kwiatkowski et al., 2019)) and report the average F1 accuracy. Surprisingly, at a fixed number of total parameters, MoEs with substantially fewer active parameters approximately match the performance of dense models. This highlights the importance of experts in tasks that require memorization. **(b) Evaluation: commonsense.** Here we evaluate the aforementioned pre-trained models on natural language commonsense benchmarks (e.g., HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021)). On these reasoning tasks, we observe that MoEs perform worse than dense models and more significant benefits are obtained by increasing the number of active parameters. **(c) Evaluation: math.** Here we train a series of dense transformers and MoEs on 65B tokens from a corpus essentially made of Proof-Pile2 (Azerbayev et al., 2023) (see Section 5 for details). The results are consistent with the ones in (b): MoEs perform worse than dense

Experiments



(a)



(b)

- Dense transformer
- MoE (18M active parameters)
- MoE (58M active parameters)
- MoE (200M active parameters)

Figure 5: Generalization gap i.e., difference between the training and test accuracies, when the test set is GSM8k (a) and Hendrycks-MATH (b).

Experiments

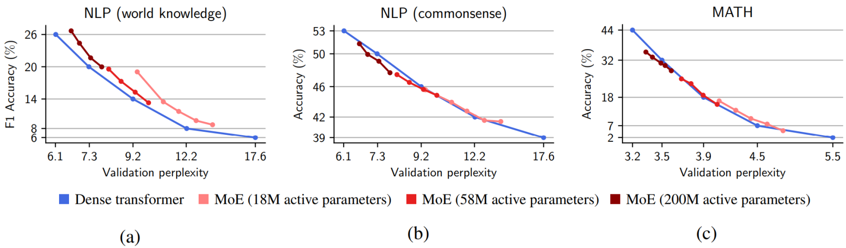


Figure 6: (a) On world knowledge benchmarks, MoEs consistently outperform dense transformers in downstream performance when fixing the validation perplexity. (b-c) In reasoning benchmarks, dense transformers perform about the same as MoEs at a fixed validation perplexity. MoEs can achieve these perplexities with less active parameters, but may require substantially more total parameters.

Validation perplexity measures how well these models fit the training distribution.

Outline

Conclusion

- MoEs are highly effective at memory-intensive tasks and not good at reasoning tasks.
- For reasoning tasks, increasing the model width is inevitable.