SN40L and Composition of Experts (CoE)
The Emergence of Two Trends

#1: Larger LLMs are more capable

The “bigger is better” story is true

- OpenAI’s GPT-4 is the leading proprietary model today (rumored to be ~1.8T parameters) => the largest proprietary LLM

- TII’s Falcon-180B is the leading open source model sitting on the top of Hugging Face’s LLM leaderboard => the largest open source LLM

But larger LLMs are also...

- Extremely costly to train
  - Shortage of accelerators
- Even more costly to serve
- Non-trivial to fine-tune and maintain overtime
The Emergence of Two Trends

#2: A composition of models is required to solve complex tasks

LLMs are good at one task at a time
- When you combine more than one task in a prompt, the quality of result drops
- Make multiple LLM calls: First for task 1, then for task 2, etc.

The rise of application frameworks
- Various integration framework designed to simplify the creation of applications using LLMs have been gaining popularity
- Specialization becomes the natural delineation of the models in the chains

LangChain  
LlamaIndex  
Semantic Kernel
**SN40L: SambaNova’s new language-optimized RDU**

“Cerulean” Architecture-based Reconfigurable Dataflow Unit

- **5nm TSMC**
- **102B Transistors**
- **1,040 RDU Cores**
- **638 TFLOPS (bf16)**

**3-tier Dataflow Memory**
- **520 MB On-Chip Memory**
- **64 GB High Bandwidth Memory**
- **1.5 TB High Capacity Memory**

Generative AI Training and Inference

Copyright © 2023 SambaNova Systems
SN40L System - The Pillar for Both Emerging Trends

- **High throughput inference with caching**
- **On-Chip SRAM** [4 GB, PBs per sec]
- **Dataflow enabled by large On-Chip Memory**

12.8 TB/s

- **RDU High Bandwidth Memory [512 GB]**

800 GB/s

- **Up to 5 Trillion Parameters!!**
- **Super Low Latency Model Switching** (Eg. <0.02sec for llama V2 7B)

- **RDU High Capacity DDR Memory [12 TB]**

Copyright © 2023 SambaNova Systems
Confidential & Proprietary | Internal Use Only
A Fundamental Shift of Models Deployment at Scale

Traditional GPU Systems

Individual model endpoints

All models in memory
(Super low latency model switching)
Composition of Experts (CoE)

- Composition of Experts (CoE) is a system of ‘N’ experts wherein each expert is a full model.
- *Routing module* routes input to the appropriate expert.

CoE System at highest level of abstraction
Rule-based routing:
- Enterprise scale usage of LLMs
- Granular Access control/personalized LLMs
- Multimodality

Symbiotic relationship with open source models:
- Accelerates adoption
- No expensive pre-training just fine-tune

Open source community fine-tuned experts
CoE – A New Way to Build Powerful LLMs

Simpler to build:
- Classic ML based router
- Fine tuned small experts

Faster to build:
- Leverage open source
- Smaller model trains faster

Modular:
- Debuggable/Interpretability
- Improvements without regression

CoE with intelligent routing

input

Intelligent Routing

Expert Id

Load and Run Expert

Expert Model

output

Expert Models Store

Chat

Base

Fine-tuned

Open source community fine-tuned experts
Value of Composition of Experts

Best TCO for inference
- Best TCO for running many expert models (up to 5T parameters worth) and up to 256k sequence length on a single node at 300 tokens/second throughput (with 7B parameter experts)

Modular training
- Train expert-by-expert, meaning quicker time to value and ability to incrementally scale.

Avoids alignment tax
- Train on new domains, new tasks, new languages, new modalities, without becoming worse at in existing capabilities.

Granular access control
- This new paradigm provides much higher security, ensuring access control of information within the model. Eg. Accounting expertise should only be made available to finance dept. this lowers risk of adopting AI ubiquitously.
# Composition of Experts vs. Mixture of Experts

<table>
<thead>
<tr>
<th>Claim</th>
<th>Composition of Experts</th>
<th>Mixture of Experts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of Models</strong></td>
<td>Many models, each expert is a single model</td>
<td>Single model, experts made up of different layers</td>
</tr>
<tr>
<td><strong>Training Process</strong></td>
<td>Each model can be trained independently. Router can be optimized independently through human feedback</td>
<td>Each time model is trained, entire model must be trained. Router coupled with training process, cannot be optimized independently.</td>
</tr>
<tr>
<td><strong>Modularity</strong></td>
<td>Models can be added and removed from the composition incrementally</td>
<td>Model and number of experts is fixed</td>
</tr>
<tr>
<td><strong>Experts Definition</strong></td>
<td>Experts are defined i.e. I can trained each expert explicitly to align to a particular domain, task, language etc</td>
<td>Experts emerge i.e. I do not explicitly define experts, the model and training process will define the experts, thus I cannot say definitively what each layer is an expert in</td>
</tr>
<tr>
<td><strong>Granularity</strong></td>
<td>Each expert solves one particular sequence. I.e. if i am asking a finance question, the finance expert will respond to the whole query</td>
<td>Routing happens for each new token, so very hard to define which experts answered my question</td>
</tr>
<tr>
<td><strong>Access Control</strong></td>
<td>Easy to manage access for each expert e.g. so employees in marketing cannot see HR data</td>
<td>Not feasible to manage access, as model is still one single monolith.</td>
</tr>
</tbody>
</table>
Thank you

Q&A

www.sambanova.ai