CS336 Language Modeling from Scratch

Lecture 4: Mixture of Experts (MoE) Architectures

Stanford University, Spring 2025

Abstract

This lecture covers Mixture of Experts (MoE) architectures, a critical technique for scaling language models efficiently in 2025. Key topics include sparse activation patterns, routing mechanisms, load balancing strategies, expert configurations, and real-world implementations in systems like GPT-4, DeepSeek V3, and modern state-of-the-art models. The lecture demonstrates how MoE enables training larger models with constant computational cost through strategic parameter scaling.

Contents

1 Introduction and Course Context

1.1 The MoE Revolution in 2025

Mixture of Experts (MoE) has become a critical architecture in 2025, adopted by most modern high-performance language models including GPT-4, Grok, DeepSeek V3, and Llama 4.

Key Point: MoE Dominance

At compute scales used for training state-of-the-art models, MoE architectures consistently outperform dense models when implemented correctly. This represents a fundamental shift in how we scale language models.

Leading MoE Adopters:

- GPT-4 (potentially GPT-MoE-1.8T according to Nvidia leak)
- Grok, DeepSeek V3, Llama 4
- Most state-of-the-art systems in both East and West

Definition: Mixture of Experts

A mixture of experts is a sparse neural network architecture that replaces dense feed-forward networks (FFNs) with multiple smaller expert networks that are selectively activated based on input routing decisions.

Warning: Common MoE Misconception

Despite the name suggesting domain-specific experts, MoE is NOT about having specialized experts for coding, English, etc. It's purely an architectural optimization for computational efficiency.

Key Point: Core MoE Advantage

MoE enables more parameters without affecting FLOPs, allowing better memorization of world knowledge while maintaining computational efficiency.

1.2 Core Architecture Principle

The fundamental idea is to replace a single large FFN with:

1. Multiple smaller expert networks (copies or splits of the original FFN)
2. A router that selectively activates only a subset of experts
3. Sparse activation to maintain constant FLOPS

2 Technical Architecture

2.1 Basic MoE Structure

In a standard transformer:

- Dense model: Self-attention → One large FFN

- MoE model: Self-attention → Router + Multiple smaller FFNs

Advantage: More parameters without affecting FLOPS, enabling better memorization of world knowledge.

2.2 Router Mechanism

The router function determines which experts process each token:

$$S_i(t) = \text{softmax}(U \cdot E_i) \tag{1}$$

where:

- U is the residual stream input
- E_i are learned expert affinity vectors
- $S_i(t)$ represents expert-token affinity scores

Top-K Routing Process:

1. Compute affinity scores for all experts
2. Apply softmax normalization
3. Select top-K experts with highest scores
4. Gate outputs and compute weighted average
5. Add residual connection

$$\text{Output} = \sum_{i \in \text{top-K}} G_i(S_i(t)) \cdot \text{FFN}_i(U) + U \tag{2}$$

3 Routing Strategies

3.1 Routing Decision Types

1. **Token Choice:** Each token selects top-K experts
   - Most commonly used in practice
   - Better performance but potential load imbalance
2. **Expert Choice:** Each expert selects top-K tokens
   - Guarantees balanced expert utilization
   - Better for distributed systems
3. **Global Assignment:** Solve optimization problem for balanced mapping
   - Computationally expensive
   - Rarely used in practice

3.2 Alternative Routing Methods

Surprising Result: Even random hashing-based routing (no semantic information) provides significant gains over dense models.

Historical Approaches:

- Reinforcement Learning for routing decisions (early work, now abandoned due to computational cost)
- Linear assignment problems and optimal transport (elegant but impractical)

4 Training Challenges and Solutions

4.1 Non-Differentiable Routing Problem

Challenge: Routing decisions are discrete and non-differentiable, making standard gradient descent difficult.

Problem: Without proper constraints, models tend to route all tokens to a single ”good” expert, leaving others unused.

4.2 Load Balancing Loss

The critical training innovation is the auxiliary balancing loss from Switch Transformer:

$$L_{\text{balance}} = \alpha \sum_{i=1}^{N} f_i \cdot P_i \tag{3}$$

where:

- f_i = fraction of tokens routed to expert i
- P_i = average routing probability for expert i
- α = balancing coefficient
- N = number of experts

This loss encourages uniform expert utilization and prevents routing collapse.

5 Expert Configurations

5.1 Shared vs. Fine-Grained Experts

Shared Experts:

- Always activated for every token
- Provide stable base computation
- Typically 1-2 shared experts

Fine-Grained Experts:

- Smaller individual expert size (e.g., 1/4 of original FFN)
- More experts overall (e.g., 64 instead of 8)
- Better parameter efficiency
- DeepSeek innovation widely adopted

5.2 Modern Expert Configurations

Model	Total Experts	Active Experts	Expert Size Ratio
Mixtral	8	2	1.0x
DeepSeek V1	64	6	0.25x
DeepSeek V3	256+	8	0.125x
Llama 4	128	8	0.25x

Table 1: Expert Configuration Evolution

6 Systems and Parallelism

6.1 Expert Parallelism

MoE enables a natural parallelization strategy:

1. Place different experts on different devices
2. Route tokens to appropriate devices after router decision
3. Perform computation locally on each device
4. Gather results through collective communication

Benefits:

- Additional axis of parallelism beyond data/model parallelism
- Natural sharding point for very large models
- Efficient utilization when communication costs are manageable

6.2 Communication Optimization

For large-scale deployment, DeepSeek V3 introduces hierarchical routing:

1. First select top-M devices
2. Then select top-K experts within chosen devices
3. Reduces cross-device communication overhead

7 Performance Results

7.1 Empirical Evidence

Multiple studies demonstrate consistent advantages:

- **Fedus et al. (2022):** 7x speedup with equivalent FLOPS
- **AI2 OLMo study:** Consistent improvements across benchmarks
- **DeepSeek V2:** Superior activated parameter efficiency on MMLU

Key Finding: At fixed FLOPS, MoE consistently outperforms dense models across training loss, perplexity, and downstream tasks.

7.2 Scaling Trends

- More experts generally lead to better performance
- Fine-grained experts show clear benefits (8 → 32 → 64)
- Shared experts provide stability in some configurations

8 Modern Implementations: DeepSeek Evolution

8.1 DeepSeek V1 Architecture

- 16B parameters (2.8B active)
- 2 shared + 64 fine-grained experts
- 6 active experts per token
- Standard top-K routing with auxiliary balancing loss

8.2 DeepSeek V3 Innovations

- 671B parameters (37B active)
- Hierarchical routing for communication efficiency
- Advanced expert parallelism strategies
- Architecturally similar to V1 but massive scale engineering

Key Insight: DeepSeek V3's success comes primarily from engineering excellence rather than architectural novelty.

9 Implementation Considerations

9.1 Why MoE Isn't Standard Teaching

1. **Complexity:** Significant systems engineering required
2. **Scale Dependent:** Benefits primarily visible at multi-node scale
3. **Training Instability:** Requires careful auxiliary loss tuning
4. **Infrastructure:** Non-trivial distributed systems concerns

9.2 Practical Guidelines

When to Use MoE:

- Large-scale training with multi-device setups
- When parameter count matters more than FLOPS
- Sufficient engineering resources for systems complexity

Hyperparameter Choices:

- K=2 most common (exploration vs. exploitation balance)

- Fine-grained experts with 0.25x size ratio
- 1-2 shared experts for stability
- Auxiliary loss coefficient $\alpha = 0.01 - 0.1$

10 Future Directions and Variations

10.1 Sparse Attention MoE

- Applying MoE principles to attention layers
- More unstable than FFN-based MoE
- Limited adoption in major releases

10.2 Hardware Optimizations

- Modern sparse matrix multiply engines (MegaBlocks)
- Device-level sparsity support
- Efficient expert computation fusion

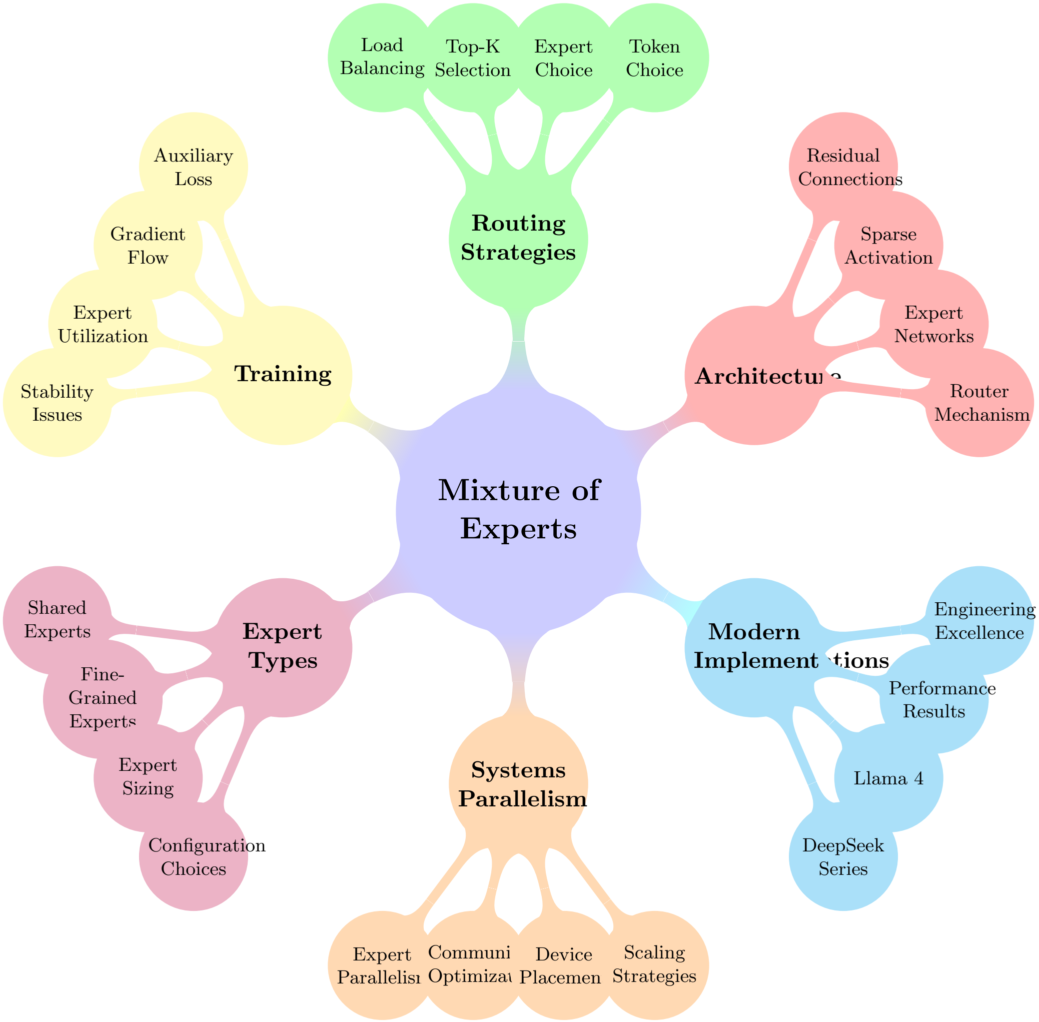
11 Conclusion

Mixture of Experts represents a fundamental shift toward sparse, efficient architectures in large language models. Key takeaways:

1. **Core Principle:** More parameters at constant FLOPS through sparse activation
2. **Routing:** Token-choice top-K routing dominates practical implementations
3. **Training:** Auxiliary balancing losses essential for stable training
4. **Architecture:** Fine-grained experts with shared experts becoming standard
5. **Systems:** Expert parallelism enables new scaling strategies
6. **Future:** Engineering excellence more important than architectural novelty

The success of DeepSeek V3 and adoption by Llama 4 indicates MoE is becoming the dominant paradigm for frontier language models in 2025.

12 Visual Mind Map



12.1 Mind Map Structure Description

The mind map centers on ”Mixture of Experts” with six main branches:

1. **Architecture (Red):** Core technical components including router mechanisms, expert networks, sparse activation, and residual connections.

2. **Routing Strategies (Green):** Different approaches to token-expert assignment including token choice, expert choice, top-K selection, and load balancing techniques.

3. **Training (Yellow):** Training-specific challenges and solutions including auxiliary losses, gradient flow management, expert utilization optimization, and stability considerations.

4. **Expert Types (Purple):** Different expert configurations including shared experts, fine-grained experts, sizing considerations, and configuration choices.

5. **Systems Parallelism (Orange):** Distributed computing aspects including expert parallelism, communication optimization, device placement strategies, and scaling approaches.

6. **Modern Implementations (Cyan):** Real-world applications focusing on the DeepSeek series, Llama 4, performance results, and the importance of engineering excellence.

The visual layout uses color coding to distinguish different conceptual areas and shows the hierarchical relationships between core concepts and their implementation details. This structure reflects the lecture's progression from basic principles through technical details to practical considerations and real-world applications.