Let's formalize this idea. We will call the architecture SOME: Self Organized Mixture of Experts.

The core premise of SOME is to separate the expert's function (its internal weights) from its address (its routing key). The function is stable, but the address is plastic and evolves during inference. This "Knowledge Gravity" is the force that moves the addresses.

The Point: How SOME Works
SOME introduces a "synaptic consolidation" phase that occurs during inference. After a token or sequence is processed, the routing keys of the activated experts are updated based on two forces:

- Query Pull (Relevance Attraction): An expert's key is pulled closer to the query vector of the token that activated it. This moves an expert's address toward the "conceptual space" of the problems it is good at solving.
- Peer Pull (Hebbian Attraction): Experts that are frequently activated together for the same tokens have their keys pulled closer to each other. This is the "Knowledge Gravity" that forms self-organizing neighborhoods or "knowledge galaxies."

Experts' internal weights remain frozen, preventing catastrophic forgetting. Only their discoverability—their position in the routing space—changes. The model actively reorganizes its own address book in near real-time.

The Path: Architectural Design and Mechanisms
Here is a breakdown of the components and how they differ from a standard MoE.

1. Foundation: Static Experts, Dynamic Keys
This is the most critical design choice.
Experts: A vast pool of experts (potentially millions of tiny ones, like in PEER) are trained in a conventional pre-training phase. Their internal parameters (w_down, w_up) are then frozen. They are stable, reliable specialists.
Keys: Each expert is assigned a routing key k, a vector in the same space as the router's query vector q. These keys are plastic and are stored in a dynamically updatable key-value store (e.g., a FAISS index or a custom GPU structure).
Why this is crucial: Updating the multi-million parameter weights of an expert at test time is computationally suicidal and would lead to immediate catastrophic forgetting. Updating a small key vector is cheap and only affects routing, not function.

2. The Dual-Update Mechanism: The Forces of Gravity
After a batch of data is processed, the model enters a brief "synaptic consolidation" phase where the keys are updated.
Mechanism 1: Query Pull (k-means-like Drift)
For every token that produced a query vector q and activated an expert e with key k_e, we apply a small update:
$k\_e\_new = k\_e + \alpha * (q - k\_e)$
α is a small "plasticity rate" (e.g., 0.001).

This nudges the expert's key slightly from its current position toward the query that just successfully used it. An expert that consistently gets activated by queries related to Python syntax will see its key drift into the center of the "Python syntax" region of the vector space.
Mechanism 2: Peer Pull (Hebbian Co-activation)
When a single token q activates a pair of experts, e1 and e2 (standard in Top-K routing where K>1), we strengthen their association:
$k1\_new = k1 + \beta * (k2 - k1)$
$k2\_new = k2 + \beta * (k1 - k2)$
β is a "bonding rate," typically smaller than α.
This pulls the two co-activated experts' keys slightly closer together. If a "for-loop expert" and a "variable-assignment expert" are constantly used together, they will form a tight gravitational bond, creating a "programming basics" neighborhood. This makes them easier to find together in the future.

3. Maintaining Stability: Inertia and Decay
A system with only attractive forces would collapse. We need mechanisms to ensure stability.
Gravitational Mass (Usage Inertia): An expert's "mass" is proportional to its overall activation frequency. More frequently used experts have higher inertia, meaning their learning rates (α and β) are scaled down. This prevents popular, generalist experts from being erratically yanked around by every query. They become the stable "galactic centers" around which smaller, more niche experts can orbit.
Dark Energy (Repulsive Force / Decay): To prevent all keys from collapsing into a single point, a small, constant repulsive force could be applied, or keys of inactive experts could slowly decay back towards their origin point. This encourages the exploration of the space and allows "dead" or useless experts to be eventually recycled.