MAGS: Learning LLMS

By: Pulkith, Pragya, Stanley, Shailesh

Humans vs LLMs

- Lack of Agency
  - Figure, Tesla, etc…
- Lack of True Understanding, Reasoning, Novelty
  - ???
- **Lack of Memory, Learning, and World Model**
  - **Our Goal**

Motivation & Problem

Memory and Learning

- LLMs cannot **learn and remember**
  - Test-time emulation with context windows is not viable
    - Quadratic scaling
    - Attention decay and ephemeral context
    - Stateless session
  - Retrieval Augmented Generation (RAG) can add information
    - But rigid, one way static transfer of knowledge
    - Cannot reconcile new or conflicting information permanently

Motivation & Problem

SoTA / Literature

- CAMELoT, MIT+IBM
  - Added Memory Blocks To Remember Information
  - Fixed Memory Size, Overwrites Past Memory, No relationships between memory, No reconciliation
- Continual Learning
  - Catastrophic Forgetting, Expensive, Very Slow, not optimal
  - Google Titans (RMTs): Updates Attention loses fine details, and slow + 2M token max
- AriGraph
  - Adding Dynamic Graph-Based RAG
  - LLM has no representation of memory, only adds information (no updating+consolidating), no importance of different episodes, memory not connection across episodes

Motivation & Problem

Technical Approach

M	A	G	S
Memory	**Augmented**	**Graph**	**Scaling**
Episodic and Semantic Memory Blocks (RAG) along with Long-term vs Working Memory Supports separation of general knowledge, experience, and importance.	Compatible with any vanilla frozen LLM. Fine-tuned for engram+recall steps, so LLM can interact with memory. LLM can query knowledge before answering and store novel experiences and learnings.	Liquid Knowledge Graphs, that can add+remove nodes, change edge weights, consolidate and forget knowledge. Replicates human Hebbian Plasticity, Neurogenesis, and Synaptic Pruning.	Allows for context scaling limited by memory constraints rather than attention decay. Similar time inference. Chat History is part of Working memory, so agents are more modular.

Technical Approach

Structure

Knowledge broken down into Long-Term (important info), and Working Memory (chat history). Memory is moved from Working to Long-Term based on usage, ‘surprise’, or importance.

Recall

Anchor Nodes selected from query, BFS over decaying edge weights. SCCs consolidated. Can also query for connection between nodes.

Engram

Updates nodes and edge weights based on query importance. Can add or update information.

Validation

Custom game with nonsensical rules (so mode cannot reason). LLM can store newfound experiences, rule changes, surprising things, general game rules, etc...

Current State

- A basic Graph RAG implementation is functional, allowing node querying, addition, and removal, but lacks the core dynamic and learning mechanisms envisioned for MAGS.
- The LLM's ability to autonomously update memory (the engram step) is simplistic, non-deterministic, often adds incorrect information, and the required structured output format is not consistently produced.
- The system's current inference speed is very slow (~1 minute per response) locally, despite low CPU utilization, indicating bottlenecks likely outside of graph density at this stage.
- Formal performance metrics for the integrated system are not yet established or measured, although basic functional tests for the graph library itself pass.
- Initial infrastructure, including custom dataloaders and a basic game environment for eventual evaluation, is in place.

Learning, Reflections, Progress

- Learnings & Reflections: Gained practical experience setting up and running local LLMs, revisiting RAG principles, and encountered unexpected challenges like LLM non-determinism at low temperatures and practical setup issues (like API rate limits). Realized the difficulty of setting specific test cases for unstructured LLM output.
- Key Immediate TODOs: The primary focus must be on implementing the foundational dynamic graph algorithms (decay, strengthening, anchor-based traversals), securing necessary GPU compute resources to enable LLM fine-tuning, and developing the required synthetic datasets for training.
- Implementation Gap: Acknowledge the significant gap between the detailed technical plan/abstract and the currently implemented basic RAG functionality. The next steps must prioritize bridging this gap by building out the unique MAGS components.