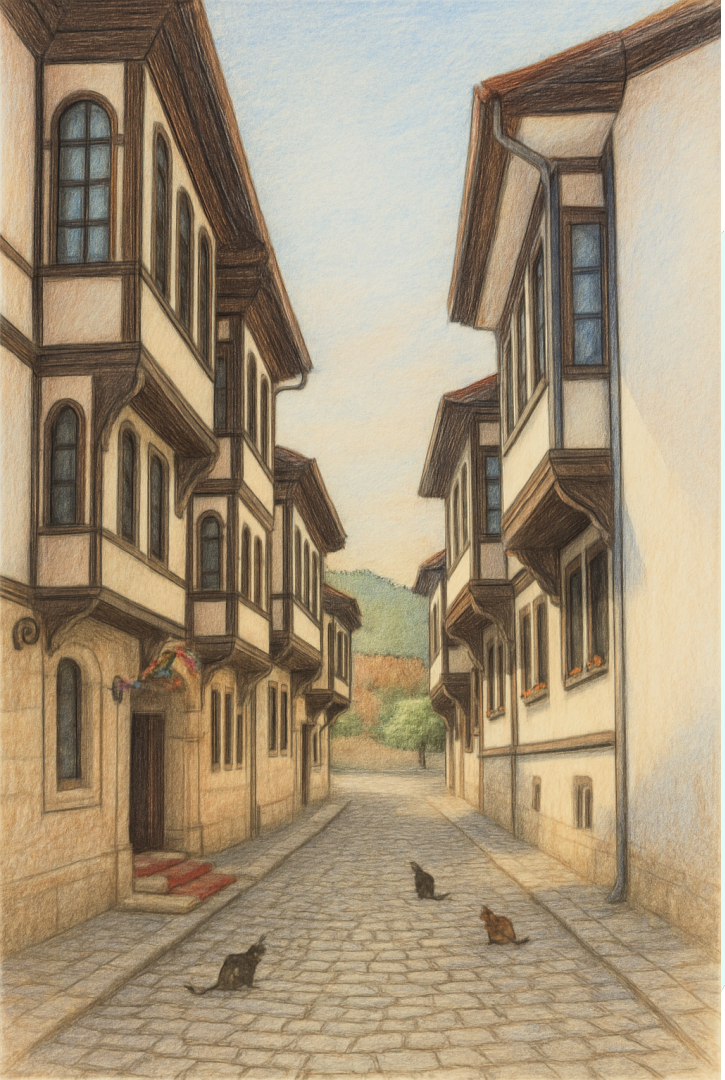
Hi!

Agentic Search: A New Way of Doing Old Things...

...That Requires Nuclear Powerplants :)

Mustafa Kıraç, CTO@GengageAI



Who am i?

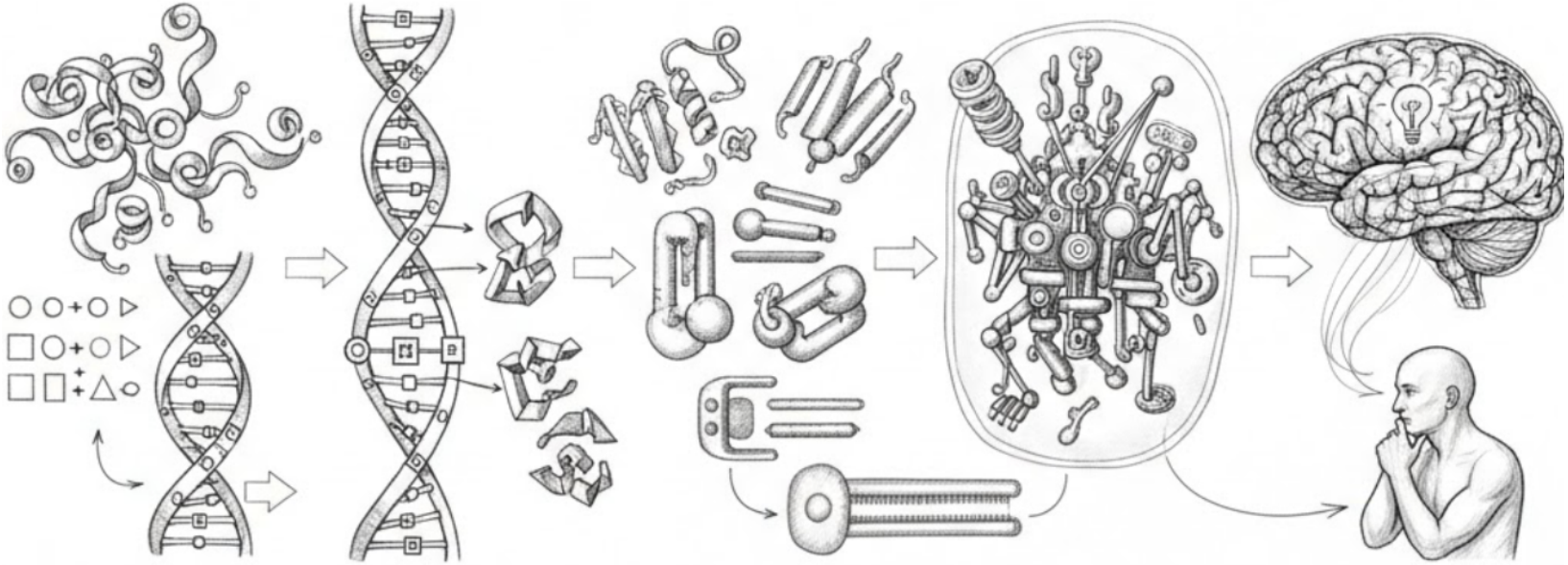
BS, CS @ Bilkent
PhD, Bioinformatics @ Case
DBMS Kernel @ Oracle
Data Mining @ Turkcell
Data Science @ iyzico
AI @ Afiniti
Agents @ GengageAI

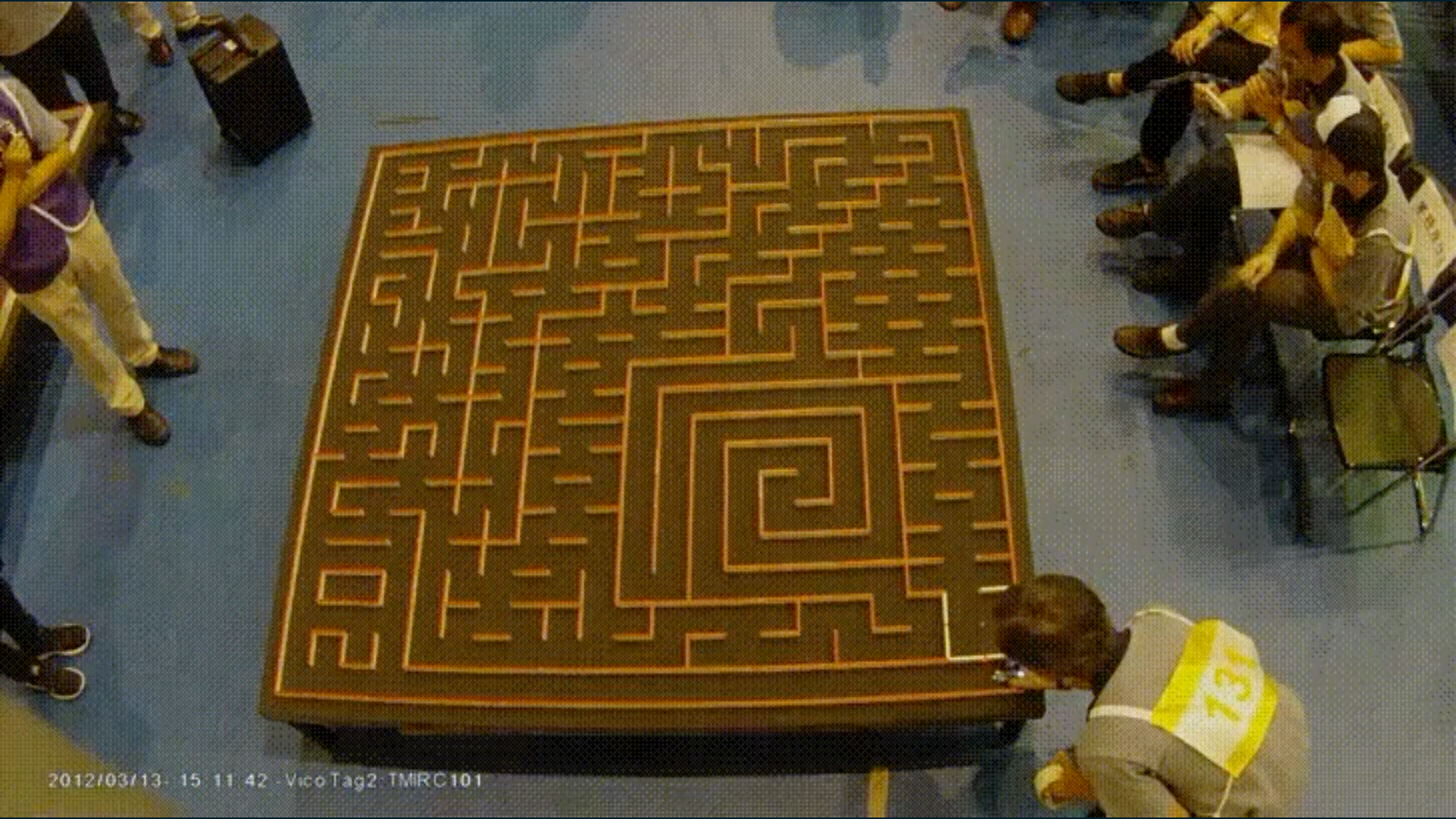
@muskirac

/mustafakirac

Biological Agents

agent(n.) late 15c., "one who acts," from Latin *agentem* (nominative *agens*) "effective, powerful," present participle of *agere* "to set in motion, drive forward; to do, perform; keep in movement".




131
2012/03/13 15:11:42 VicoTag2 TMIRC101

Reinforcement Learning

FMs as Universal Intelligence

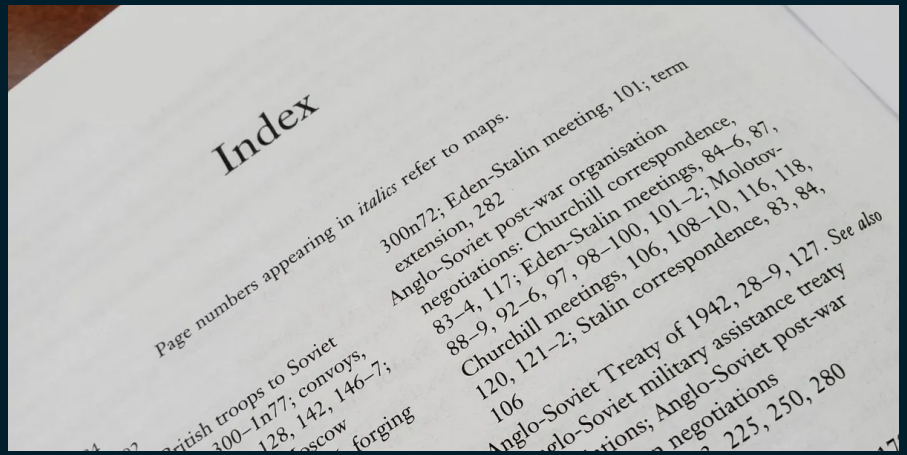
FMs can be "programmable" through simple prompts, they can understand context, make decisions, and perform complex reasoning.

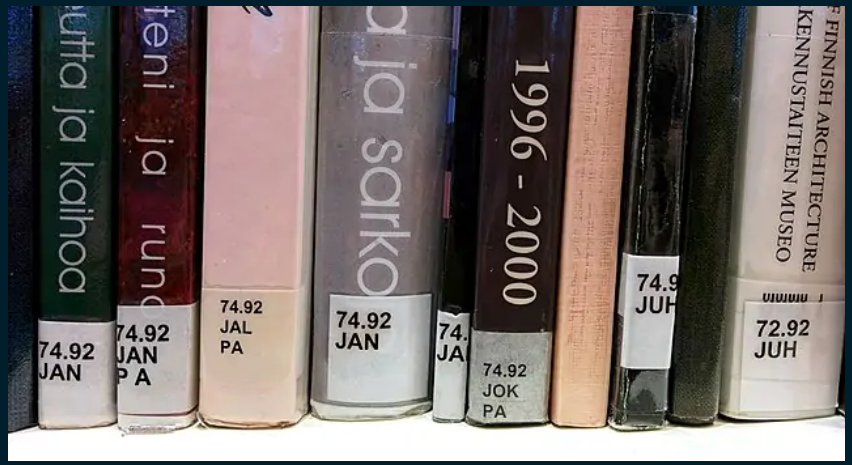
1. Neural Networks as Function Approximators: The Universal Approximation Theorem proves this capability exists, though in practice, deep networks with many layers are required to learn complex functions efficiently.

2. Few-Shot Learning: LLMs could adapt to new tasks without retraining. By simply showing the model a few examples in the prompt, it can "learn" a new function on the fly, without any weight updates. This is what makes prompts "programmable"

3. Structured Output: LLMs can be fine-tuned to learn specific grammars.

Search and Ranking

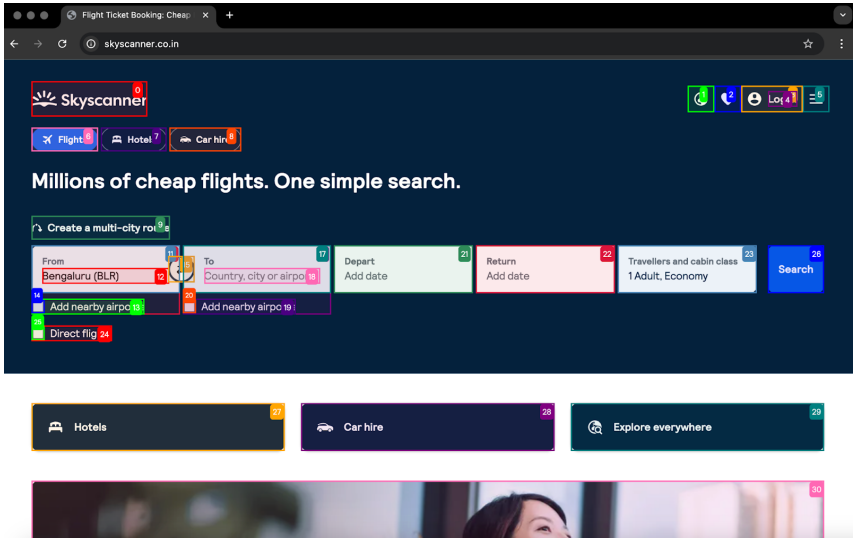




- `grep` for exact-match and regex scanning.
- `SQL` for structured data.
- `Gremlin` (for graph databases) to search for relationship patterns.
- GET-SET **K-V stores**
- **Bloom filters** for probabilistic "does this exist?" checks.

- For structured documents, this was easy: `ORDER BY`.
- **BM25** and **TF-IDF** normalize ranking.
- **PageRank** global “prestige” rank.
- **Mat. Factorization** ranking *per-user*.
- **Embeddings** for "semantics”
- **HNSW** for "similarity islands”

Agentic RAG via FMs



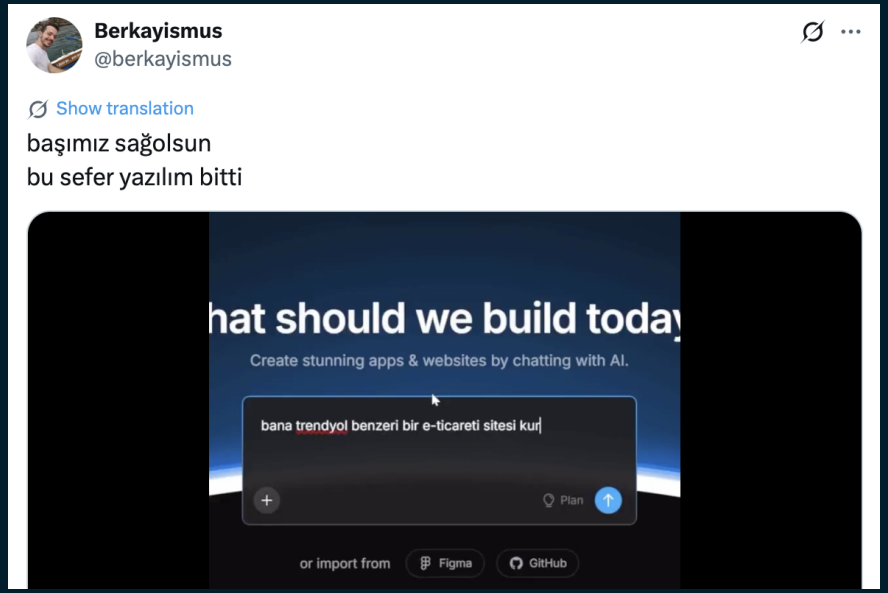
- **Analyze the goal** from the user's prompt.
- **Check what tools are available** in its "toolbox."
- **Create an execution plan.** Similar to cost-based query optimization in SQL. The agent "thinks" about which tool is best. *Should I do a fast vector search first to get general context, or a precise keyword filter to narrow the results?*
- **Execute the plan** by calling a tool and then **checking the results** (sensing the new environment state).
- **Change the plan if needed.** If the search returned no results, it can autonomously decide to retry with a different query or a different tool.

Halting Problem

For an agent that can "refine" its plan indefinitely, how does it ever know it's "done"?

Implement cost controls, hard token limits, budget circuit-breakers, and timeouts.

SOTA



Hierarchical Multi-Agent Systems: A manager agent delegates tasks to specialists.

ASearcher: Solves the "long-horizon" problem, where agents "get lost." It uses a scalable, asynchronous RL training on 40+ turn trajectories, making it robust for tasks that take days.

Agent Swarms: The agents use shared protocols and memory to coordinate, allowing complex, emergent behavior.

ADAS: AI Designing AI. This is the "Meta Agent Search" algorithm.

MongoDB

Solves Agent Amnesia: MongoDB provides persistent, long-term agent memory.

Provides the SOTA Tools: Atlas is the engine for Agentic RAG, allowing agents to dynamically orchestrate Hybrid Search (Vector + Full-Text) as a tool.

The future is not one giant "brain." It's a swarm of smaller, specialized agents. And the one thing they all need is a common, reliable, stateful memory and search tools to access to right data.

Thanks.