**✅ 1. Memory-Augmented Neural Networks (MANNs)**

MANN là một kiến trúc mạng nơ-ron được thiết kế **kết hợp với một bộ nhớ ngoài (external memory)** nhằm cải thiện khả năng lưu trữ và truy xuất thông tin trong quá trình học.

Không giống như mạng truyền thống chỉ dựa vào weights, MANN có thể **đọc và ghi dữ liệu từ bộ nhớ rời**, tương tự như cách máy tính thao tác với RAM.

MANN giúp xử lý các bài toán cần **ghi nhớ chuỗi dài**, như dịch máy, giải toán, suy luận logic, hay các tác vụ NLP phức tạp.

MANN là nền tảng của các mô hình như:

**Neural Turing Machine (NTM)**

**Differentiable Neural Computer (DNC)**

**✅ 2. Neural Turing Machine (NTM), DNC, Transformer + Memory**

**🧠 Neural Turing Machine (NTM)**

Được giới thiệu bởi DeepMind (2014), NTM là mô hình kết hợp giữa mạng nơ-ron và **bộ nhớ ngoài giống Turing Machine**.

Cấu trúc gồm:

**Controller (LSTM/MLP)**: điều khiển việc đọc/ghi

**Memory Matrix**: nơi lưu trữ dữ liệu

Có khả năng học cách **truy cập và thao tác dữ liệu** trong bộ nhớ bằng phương pháp gradient descent.

Dùng trong các tác vụ như **copying, sorting, associative recall**.

**🧠 Differentiable Neural Computer (DNC)**

Cải tiến của NTM, cũng từ DeepMind.

Tăng cường khả năng ghi nhớ lâu dài và **truy xuất chính xác hơn**.

Có các cơ chế truy xuất bộ nhớ nâng cao: Temporal Linkage, Memory Usage Vector…

DNC cho phép mô hình tự học cách cấu trúc và truy vấn dữ liệu như một cơ sở tri thức.

**⚡ Transformer + External Memory**

Một số biến thể Transformer hiện đại tích hợp external memory như:

**Memory Transformer**, **Compressive Transformer**, **RETRO**

Mục tiêu là khắc phục nhược điểm **context window giới hạn của Transformer** thông thường.

Mô hình có thể **truy cập thông tin đã gặp trước đó**, kể cả ở đoạn văn bản dài hay đã qua nhiều lượt hội thoại.

**✅ 3. External Knowledge giúp chatbot ghi nhớ tri thức lâu dài và có tổ chức hơn**

Chatbot truyền thống thường chỉ trả lời dựa trên ngữ cảnh hiện tại hoặc mô hình học sẵn, không có khả năng ghi nhớ lâu dài.

Khi tích hợp **external knowledge**, chatbot có thể:

Truy cập vào **bộ tài liệu bên ngoài** (file, CSDL, tài liệu nội bộ…)

**Ghi nhớ có chọn lọc** những thông tin quan trọng để phục vụ cho nhiều lượt đối thoại.

**Tổ chức tri thức một cách có cấu trúc** → giúp tăng độ chính xác và nhất quán trong câu trả lời.

Đây là cách kết hợp giữa **Generative AI + Retrieval-based QA**, tạo nên một chatbot **“biết học, biết nhớ”** thực sự.

### ****Ứng dụng thực tiễn của External Memory****

💡 **1. Trả lời câu hỏi (QA - Question Answering)**

* Bộ nhớ ngoài giúp mô hình truy xuất thông tin từ tài liệu dài hoặc dữ liệu chuyên ngành.
* Ứng dụng trong chatbot tư vấn, trợ lý học tập, trợ lý pháp lý…

💡 **2. Hội thoại đa lượt (Multi-turn Dialogue)**

* Ghi nhớ bối cảnh hội thoại trước đó → giúp trả lời mạch lạc, không rời rạc.
* Tạo cảm giác “trò chuyện tự nhiên” hơn với người dùng.

💡 **3. Hệ thống tư vấn chuyên sâu**

* Cho phép chatbot tư vấn dựa trên hồ sơ người dùng hoặc kiến thức chuyên môn lưu trong bộ nhớ.
* Ví dụ: chatbot y tế, tư vấn học tập, định hướng nghề nghiệp…

💡 **4. Hệ thống reasoning logic (chuỗi suy luận)**

* Hỗ trợ mô hình suy luận theo chuỗi logic nhờ khả năng lưu và nối kết thông tin.
* Giải quyết các bài toán phức tạp: toán học, luật, phản biện,…

**KIẾN TRÚC HỆ THỐNG CHAT BOT**

### 1. ****Frontend (Giao diện web)****

* Người dùng sẽ nhập câu hỏi trong ô chat trên trình duyệt.
* Khi bấm gửi, cái câu đó sẽ được gửi về backend thông qua đường dẫn /chat.

👉 Nói nôm na: bạn gõ "Trường XYZ có học bổng không?" → gửi phát là bay về server xử lý.

### 2. ****Backend (Server xử lý bằng Flask)****

Chính là cái file app.py.

* Flask nhận câu hỏi từ frontend.
* Gọi hàm call\_gemini\_api() để xử lý.

Trong hàm này nó làm mấy trò như:

#### ✅ Gắn thêm mấy thứ quan trọng vô:

* **Lần đầu chat:** nó dán sẵn một câu chào kèm hướng dẫn từ chatbot\_prompts.py, kiểu như:

"Chào bạn, mình là chatbot tư vấn trường XYZ..."

* **Nếu câu hỏi có liên quan tới trường:**  
  Nó sẽ kiểm tra qua hàm agent\_tracuu\_tailieu() bên agent\_docsearch.py.  
  Hàm này kiểu như: "À, câu này nói về học phí, học bổng, ngành học… → lấy nội dung từ file Word ra dán vô luôn cho Gemini biết mà trả lời."
* Rồi gom hết nội dung → gửi qua Google Gemini.

👉 Giống như bạn gói một cái đơn hàng: dán địa chỉ, kèm hướng dẫn → gửi đi.

### 3. ****Google Gemini API****

* Nó nhận được nội dung đầy đủ.
* Xử lý bằng trí tuệ nhân tạo.
* Trả lời lại bằng tiếng Việt, theo kiểu đã được gợi ý trước (ví dụ không được trả lời lan man, chỉ nói về trường XYZ thôi).

### 4. ****Trả kết quả về frontend****

* Flask nhận câu trả lời từ Gemini.
* Gửi lại lên trình duyệt.
* Người dùng thấy chatbot trả lời trên giao diện.