3. Bạn hiểu thế nào về kiến trúc phần mềm theo monolithic và microservice: Mô tả mỗi dạng (mono có bao nhiêu dạng), sự khác biệt, ưu/nhược điểm, kiểu tương tác giữa các lớp/gói trong mono và tương tác giữa các module/service trong micro, database trong 2 cách xây dựng

**1️ Monolithic Architecture**

**1.1. Định nghĩa:**

Monolithic (kiến trúc nguyên khối) là một cách xây dựng ứng dụng mà toàn bộ logic của hệ thống được đóng gói thành một **ứng dụng duy nhất**, triển khai dưới dạng một unit (thường là một file .war, .jar hoặc container).

**1.2. Các dạng phổ biến:**

* **Monolithic Layered Architecture (Kiến trúc phân lớp)**: Phổ biến nhất. Ứng dụng chia thành các lớp:
  + **Presentation Layer** (giao diện người dùng)
  + **Business Layer** (xử lý nghiệp vụ)
  + **Persistence/Data Access Layer** (truy cập dữ liệu)
* **Modular Monolith**: Vẫn là ứng dụng đơn khối, nhưng được chia thành các module nội bộ (module hóa), mỗi module vẫn deploy chung. Ví dụ: module quản lý user, module quản lý sản phẩm.

**1.3. Ưu điểm:**

✅ Dễ phát triển ban đầu (ít dịch vụ, ít giao tiếp).  
✅ Dễ dàng deploy (chỉ cần build một package).  
✅ Gỡ lỗi đơn giản vì mọi thứ nằm trong cùng một khối.

**1.4. Nhược điểm:**

❌ Khó mở rộng (scalability khó khăn).  
❌ Dễ dẫn tới tight-coupling giữa các lớp (khi ứng dụng lớn dần).  
❌ Mỗi lần cập nhật đều cần build và deploy toàn bộ ứng dụng, dễ gây downtime.  
❌ Dễ trở thành “Big Ball of Mud” nếu không có quy tắc module rõ ràng.

**2️ Microservice Architecture**

**2.1. Định nghĩa:**

Microservice chia ứng dụng thành nhiều **dịch vụ nhỏ (service)**, mỗi service đảm nhiệm một chức năng nghiệp vụ riêng biệt. Mỗi service chạy độc lập, deploy riêng.

**2.2. Đặc điểm:**

* Mỗi service có thể có:
  + **Giao tiếp HTTP REST (hoặc gRPC, messaging)**.
  + **Database riêng hoặc chia sẻ database nếu cần**.
  + Được phát triển, build, deploy độc lập.

**2.3. Ưu điểm:**

✅ Dễ mở rộng từng service độc lập (scale horizontal).  
✅ Đội ngũ phát triển có thể làm việc song song trên từng service.  
✅ Deploy một service không ảnh hưởng đến các service khác.  
✅ Dễ áp dụng công nghệ khác nhau cho từng service.

**2.4. Nhược điểm:**

❌ Triển khai phức tạp hơn (orchestration, service discovery, load balancing).  
❌ Tốn chi phí hơn (mỗi service có thể yêu cầu hạ tầng riêng).  
❌ Giao tiếp giữa các service (network latency, fault tolerance).  
❌ Yêu cầu quản lý phân tán: logging, monitoring, tracing.

**3️ Sự khác biệt chính**

| **Tiêu chí** | **Monolithic** | **Microservice** |
| --- | --- | --- |
| **Triển khai** | Một khối duy nhất | Nhiều service deploy riêng |
| **Mở rộng** | Mở rộng toàn khối | Mở rộng từng service |
| **Giao tiếp** | Nội bộ (method call) | Network (REST, gRPC, MQ) |
| **Database** | Thường dùng chung | Thường riêng cho từng service |
| **Tổ chức code** | Layered hoặc Modular | Service-based |
| **DevOps** | Đơn giản | Phức tạp hơn (Docker, Kubernetes, CI/CD) |

**4️ Tương tác giữa các lớp/gói**

**Trong Monolithic:**

👉 Thường dùng **method call trực tiếp** (hoặc dependency injection) để giao tiếp giữa các lớp/gói:

* Controller gọi Service.
* Service gọi Repository.
* Repository thao tác database.

Ví dụ:

java

CopyEdit

UserController -> UserService -> UserRepository

**Trong Microservice:**

👉 Giao tiếp chủ yếu qua **network**:

* **REST API** (JSON/HTTP)
* **gRPC** (protobuf)
* **Message Queue** (Kafka, RabbitMQ)

Ví dụ:

rust

CopyEdit

UserService <--> OrderService <--> PaymentService

Mỗi service có thể có API Gateway hoặc Service Registry (Eureka, Consul).

**5️ Database**

**Monolithic:**

* Thường dùng **một database chung** cho toàn bộ ứng dụng.
* Database có thể được chia theo bảng, schema hoặc module, nhưng vẫn deploy một cụm database.

**Microservice:**

* **Database per service**: Mỗi service sở hữu một database riêng (giảm coupling, tránh tranh chấp dữ liệu).
* Một số service vẫn có thể chia sẻ database nếu cần giao tiếp tight-coupling (ít dùng, vì dễ vi phạm bounded context).
* Dùng **event sourcing hoặc API** để đồng bộ dữ liệu giữa các service.

**Tóm tắt:**

| **Tiêu chí** | **Monolithic** | **Microservice** |
| --- | --- | --- |
| **Code Base** | Một codebase lớn | Nhiều codebase/service |
| **Database** | Chung | Tách riêng (hoặc chia sẻ với rủi ro coupling) |
| **Tương tác** | Method call (nội bộ) | Network call (REST/gRPC/MQ) |
| **Triển khai** | Deploy nguyên khối | Deploy từng service độc lập |
| **Ưu điểm** | Dễ phát triển ban đầu, đơn giản | Dễ mở rộng, team phát triển độc lập |
| **Nhược điểm** | Khó mở rộng, dễ coupling | Quản lý phức tạp, network overhead |

5. Các kỹ thuật (model) deep learning và đặc trưng + ứng dụng cho mỗi model. Các dạng tiến hóa của CNN, RNN, LSTM (sinh viên tìm trên google 5 Bài báo khoa học mới nhất sau 2022 và hiển thị link bài báo đó bài viết của mình). Cho ví dụ minh họa

**2. Các dạng tiến hóa của CNN, RNN, LSTM**

**2.1 CNN**

* **CNN hiện đại**: VGG, Inception, ResNet, DenseNet, MobileNet...
* **Edge-optimized CNN**: SqueezeNet, MobileNet thiết kế cho thiết bị IoT [arxiv.org+3researchgate.net+3egusphere.copernicus.org+3](https://www.researchgate.net/publication/379367981_Exploring_the_development_and_application_of_LSTM_variants?utm_source=chatgpt.com)[arxiv.org+1researchgate.net+1](https://arxiv.org/pdf/2311.12816?utm_source=chatgpt.com)
* **Tự động kiến trúc (NAS/Evolutionary)**: CNN phát triển qua thuật toán tiến hóa, tối ưu FLOPS/bộ nhớ

**2.2 RNN / LSTM / GRU**

* **Bi-directional RNN/LSTM**: học theo cả 2 chiều chuỗi.
* **Peephole LSTM**: tham số kết nối trực tiếp với cell state [mdpi.com+7sciencedirect.com+7sciencedirect.com+7](https://www.sciencedirect.com/science/article/pii/S1319157824001575?utm_source=chatgpt.com)[mdpi.com](https://www.mdpi.com/2078-2489/15/9/517?utm_source=chatgpt.com)
* **Variants (minLSTM, minGRU, Mamba, S4...)**: đơn giản hóa để train song song [mdpi.com+2arxiv.org+2arxiv.org+2](https://arxiv.org/pdf/2410.01201?utm_source=chatgpt.com)
* **Hybrid + attention**: kết hợp CNN/LSTM hoặc với attention để tăng hiệu quả và khả năng đồng bộ hóa [link.springer.com](https://link.springer.com/article/10.1007/s10791-025-09526-0?utm_source=chatgpt.com)

**3. 5 bài báo khoa học mới nhất sau 2022**

1. **"Evolution of Convolutional Neural Network"** (arXiv Nov 2023): tập trung phân tích tiến hóa CNN về FLOPS/memory cho Edge AI [arxiv.org+1researchgate.net+1](https://arxiv.org/pdf/2311.12816?utm_source=chatgpt.com)
2. **"Evolution under Length Constraints for CNN Architecture design"** (May 2023): sử dụng thuật toán tiến hóa giới hạn độ dài để tìm kiến trúc CNN tối ưu [arxiv.org+2dl.acm.org+2arxiv.org+2](https://dl.acm.org/doi/fullHtml/10.1145/3585542.3585546?utm_source=chatgpt.com)
3. **"A Critical Review of RNN and LSTM Variants in Hydrological Time‑Series Predictions"** (Jul/Sep 2024): phân tích các biến thể RNN/LSTM/GRU dùng cho dự báo thủy văn [researchgate.net+3researchgate.net+3sciencedirect.com+3](https://www.researchgate.net/publication/383985786_A_Critical_Review_of_RNN_and_LSTM_Variants_in_Hydrological_Time_Series_Predictions?utm_source=chatgpt.com)
4. **"Were RNNs All We Needed?"** (arXiv Oct 2024): giới thiệu minLSTM, minGRU, Mamba, S4 để overcome training limit [arxiv.org+1arxiv.org+1](https://arxiv.org/pdf/2410.01201?utm_source=chatgpt.com)
5. **"An enhanced CNN with ResNet50 and LSTM deep learning"** (Nature Apr/May 2025): mô hình lai CNN‑ResNet50‑LSTM dự báo biến đổi khí hậu đến 2030 [nature.com](https://www.nature.com/articles/s41598-025-97401-9?utm_source=chatgpt.com)

**4. Ví dụ minh họa**

* **CNN**: sử dụng ResNet để phân loại ảnh y khoa (X-quang, MRI), model tinh chỉnh ResNet50 phủ thêm layer FC cuối cùng.
* **Hybrid CNN‑ResNet50‑LSTM**: bài báo Nature (2025) dùng CNN‑ResNet50 để trích xuất đặc trưng không gian rồi LSTM để học đặc trưng thời gian, dự báo nhiệt độ và công suất gió đến năm 2030 [nature.com](https://www.nature.com/articles/s41598-025-97401-9?utm_source=chatgpt.com).
* **RNN/LSTM cải tiến**: trong dự báo chuỗi thủy văn, biến thể LSTM/GRU đặc biệt và attention cho hiệu quả tốt hơn báo cáo trước trong bài review .
* **MinLSTM, Mamba**: đơn giản hóa mô hình cho training song song, giảm thời gian tính toán so với LSTM/GRU truyền thống .

**🔍 Tổng kết**

* CNN mạnh về học đặc trưng không gian; tiến hóa qua NAS, lightweight CNN cho edge.
* RNN/LSTM/GRU phù hợp chuỗi thời gian, ngôn ngữ; biến thể giúp giảm gradient, đào tạo nhanh hơn.
* Hybrid và attention tiến hóa giúp nâng cao khả năng học đặc trưng đa chiều.
* Các bài báo mới sau 2022 tập trung vào tối ưu hóa hiệu quả tính toán, cấu trúc nhẹ, ứng dụng thực tiễn như dự báo khí hậu, thủy văn và edge deployment.

6. Phân biệt Deep learning, GNN, AI agent, reinforcement learning: Cho ví dụ code và chạy ra kết quả minh họa

**Phân biệt các khái niệm**

**🔹 Deep Learning**

* **Định nghĩa**: Một nhánh của machine learning tập trung vào các mạng nơ-ron nhiều lớp (deep neural networks) để học tự động đặc trưng phức tạp từ dữ liệu lớn.
* **Ứng dụng**: phân loại ảnh, nhận diện giọng nói, dịch máy, phân tích video.

**🔹 GNN (Graph Neural Network)**

* **Định nghĩa**: Mô hình deep learning xử lý dữ liệu có cấu trúc dạng đồ thị (graph-structured data), ví dụ mạng xã hội, mạng giao thông, protein network.
* **Ứng dụng**: phát hiện cộng đồng trong mạng xã hội, dự đoán liên kết (link prediction), phân loại node, khuyến nghị.

**🔹 AI Agent**

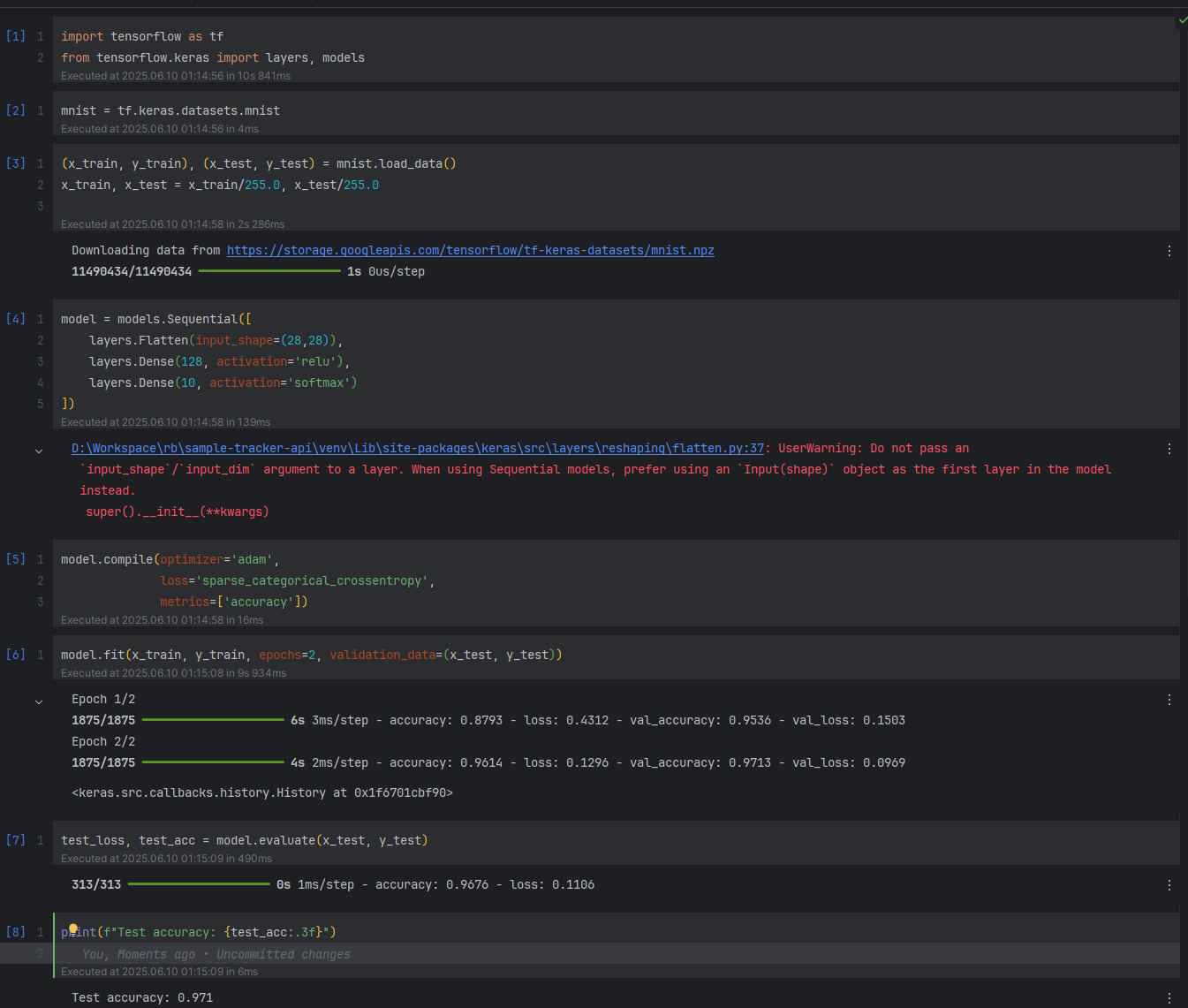
* **Định nghĩa**: Một thực thể phần mềm có khả năng quan sát môi trường, đưa ra quyết định và thực hiện hành động. Thường gắn với lý thuyết về agents trong trí tuệ nhân tạo.
* **Ứng dụng**: chatbot, trợ lý ảo, robot tự động, hệ thống gợi ý.

**🔹 Reinforcement Learning (RL)**

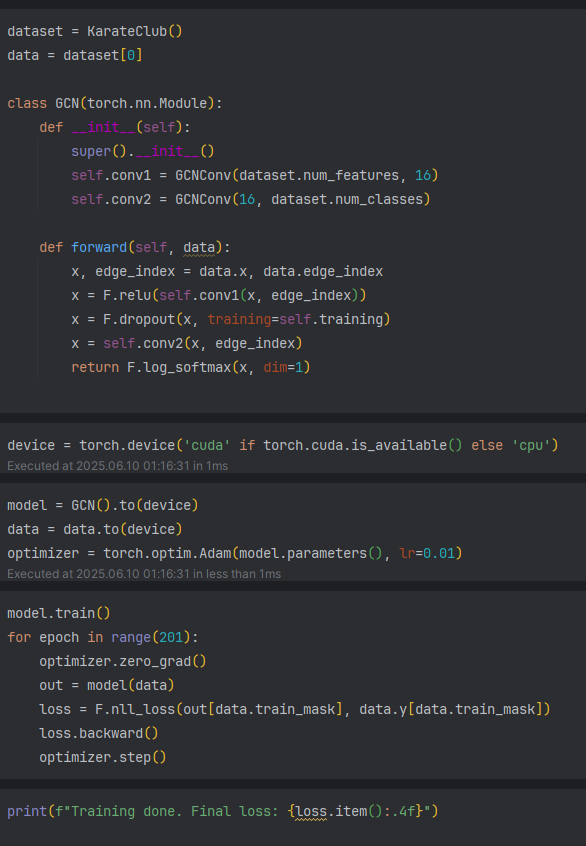
* **Định nghĩa**: Một nhánh của machine learning trong đó một agent học thông qua thử nghiệm–phản hồi (trial and error) bằng cách tối đa hóa phần thưởng nhận được từ môi trường.
* **Ứng dụng**: robot học cách đi, chơi game (Atari, cờ vây), tự động giao dịch chứng khoán, tối ưu hoá logistics.

**Ví dụ code minh họa**

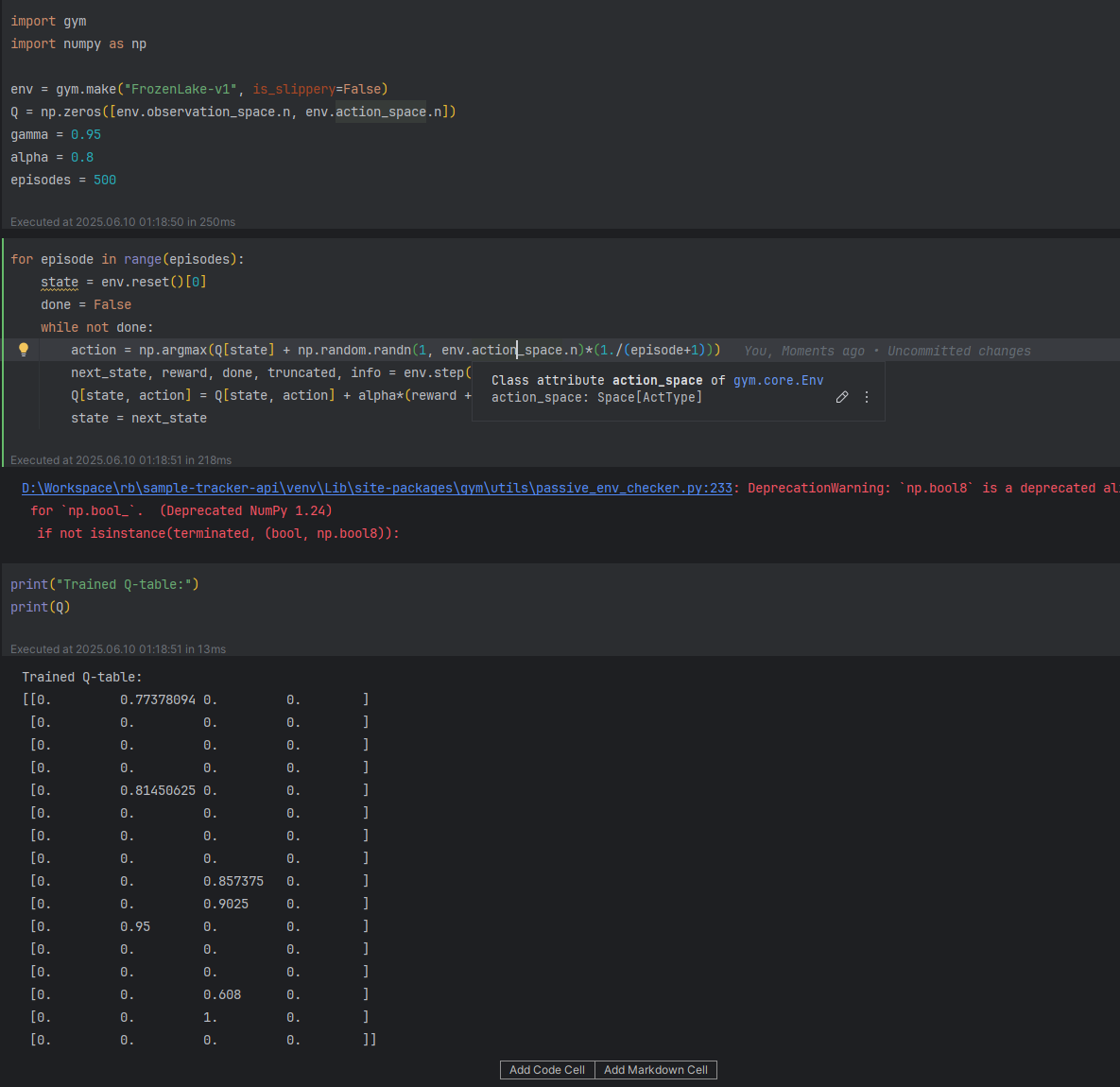
**1. Deep Learning (nhận dạng chữ số MNIST)**



2. GNN (phân loại nút trên Karate Club)



4. Reinforcement



7. Các mô hình ngôn ngữ tự nhiên và ứng dụng: N-gram, HMM, tf-idf, word2vec, GloVe, attention based model, BERT, GPT, Large Language Model. 1 ví dụ với code cho mô hình bất kỳ

**1️⃣ Các mô hình ngôn ngữ tự nhiên và ứng dụng**

**🔹 1. N-gram**

* **Mô tả**: Xây dựng mô hình ngôn ngữ dựa trên xác suất chuỗi liên tiếp gồm n từ (unigram, bigram, trigram, …).
* **Ứng dụng**: Dự đoán từ tiếp theo, kiểm tra chính tả, sinh văn bản đơn giản.

**🔹 2. HMM (Hidden Markov Model)**

* **Mô tả**: Mô hình xác suất cho chuỗi trạng thái ẩn và quan sát; thường dùng để gán nhãn chuỗi như gán nhãn từ loại (POS tagging).
* **Ứng dụng**: POS tagging, nhận dạng giọng nói, phân tích cú pháp.

**🔹 3. TF-IDF (Term Frequency-Inverse Document Frequency)**

* **Mô tả**: Trọng số phản ánh tầm quan trọng của một từ trong một tài liệu so với toàn bộ tập tài liệu.
* **Ứng dụng**: Trích xuất đặc trưng văn bản, tìm kiếm thông tin.

**🔹 4. Word2Vec**

* **Mô tả**: Học các vector biểu diễn từ bằng cách tối ưu hoá bài toán phân loại từ ngữ cảnh (Skip-gram hoặc CBOW).
* **Ứng dụng**: Tìm từ tương tự, tìm từ đồng nghĩa, cluster từ.

**🔹 5. GloVe (Global Vectors)**

* **Mô tả**: Học vector từ dựa trên ma trận đồng xuất hiện từ toàn tập văn bản, giúp biểu diễn ngữ nghĩa từ.
* **Ứng dụng**: Xử lý văn bản, tìm tương đồng từ.

**🔹 6. Attention-based Models**

* **Mô tả**: Học cách “chú ý” vào các phần quan trọng của câu đầu vào (ví dụ: dịch máy, tóm tắt).
* **Ứng dụng**: Dịch máy, tóm tắt, trả lời câu hỏi.

**🔹 7. BERT (Bidirectional Encoder Representations from Transformers)**

* **Mô tả**: Mô hình transformer encoder song hướng, tiền huấn luyện bằng Masked Language Modeling và Next Sentence Prediction.
* **Ứng dụng**: Fine-tuning cho các tác vụ NLP như phân loại văn bản, trích xuất thực thể.

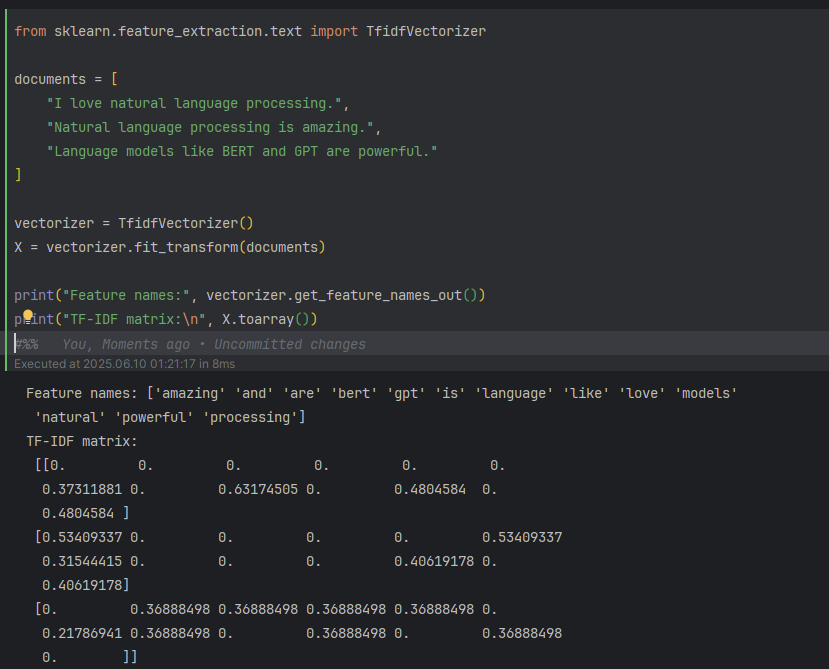
**🔹 8. GPT (Generative Pre-trained Transformer)**

* **Mô tả**: Mô hình transformer decoder uni-directional được huấn luyện để sinh văn bản.
* **Ứng dụng**: Sinh văn bản, trả lời câu hỏi, chatbots.

**🔹 9. Large Language Models (LLMs)**

* **Mô tả**: Các mô hình transformer quy mô lớn (GPT-3, GPT-4, PaLM) có hàng tỷ/thậm chí hàng ngàn tỷ tham số.
* **Ứng dụng**: Viết sáng tạo, hỗ trợ lập trình, trợ lý ảo, chatbots, tóm tắt văn bản.

**2️⃣ Ví dụ code minh họa: TF-IDF với scikit-learn**



10. AI tổng quát – AGI (Artificial General Intelligence) và super AI là gì? Hiện nay các tập đoàn đang nghiên cứu phát triển. Suy nghĩ của bạn về tương lai này?

**1️ AI tổng quát – AGI (Artificial General Intelligence) là gì?**

AGI (Trí tuệ nhân tạo tổng quát) là khái niệm về một hệ thống AI có khả năng học, hiểu, và giải quyết bất kỳ nhiệm vụ trí tuệ nào mà con người có thể làm được — tức là nó có khả năng thích nghi và tự học một cách linh hoạt trong nhiều lĩnh vực khác nhau, thay vì chỉ giải quyết những bài toán hẹp (như chơi cờ, xử lý ngôn ngữ hay nhận dạng hình ảnh).

AGI khác với các hệ thống AI hẹp (narrow AI), vốn chỉ tập trung vào một nhiệm vụ cụ thể (như chatbot, xe tự lái hay chẩn đoán y học).

**2️ Super AI (Artificial Super Intelligence) là gì?**

Super AI là mức độ tiếp theo của AGI — khi hệ thống AI có năng lực trí tuệ vượt trội hơn con người trên mọi lĩnh vực, bao gồm sáng tạo, ra quyết định, lập kế hoạch, và các kỹ năng xã hội. Super AI không chỉ học mọi thứ giỏi hơn con người, mà còn có thể tự cải thiện và phát triển trí tuệ của mình liên tục.

**3️ Các tập đoàn lớn đang nghiên cứu phát triển AGI và Super AI**

Hiện nay, một số tập đoàn công nghệ hàng đầu đang tham gia nghiên cứu AGI, bao gồm:

* **OpenAI** (phát triển các mô hình như GPT-4, GPT-5, và định hướng AGI).
* **DeepMind (thuộc Google)** (với các dự án như AlphaFold, Gemini).
* **Anthropic**, **Meta AI**, **Microsoft AI**, **Tesla AI**, và nhiều startup khác cũng đang đầu tư mạnh mẽ vào các mô hình quy mô lớn hướng đến AGI.

Họ sử dụng các kiến trúc transformer, reinforcement learning, các mô hình ngôn ngữ cực lớn (LLM) và kết hợp với kỹ thuật robotics, thị giác máy tính, mô phỏng đa nhiệm để tiến gần hơn đến AGI.

**4️ Suy nghĩ về tương lai này**

👉 **Ưu điểm**:

* AGI/Super AI có thể giải quyết những vấn đề phức tạp mà hiện tại con người chưa thể giải quyết hiệu quả (ví dụ: dự đoán bệnh tật, tối ưu hoá năng lượng, biến đổi khí hậu).
* Tạo ra những công cụ trợ lý thông minh hỗ trợ con người làm việc sáng tạo, khoa học, và quản lý xã hội hiệu quả hơn.
* Đóng vai trò quan trọng trong việc phát triển kinh tế tri thức và tăng năng suất lao động.

👉 **Lo ngại**:

* Rủi ro về kiểm soát và an toàn nếu AGI phát triển vượt khỏi tầm kiểm soát của con người (ví dụ: mất kiểm soát mục tiêu, tác động xã hội tiêu cực).
* Vấn đề việc làm: Một phần lớn công việc hiện nay có thể bị thay thế, dẫn đến bất bình đẳng xã hội.
* Nguy cơ lạm dụng AI cho mục đích xấu (chiến tranh, deepfake, thao túng thông tin).

👉 **Kết luận cá nhân**:

* AGI/Super AI là một bước tiến lớn và cần được phát triển một cách có trách nhiệm, minh bạch và có các biện pháp kiểm soát rủi ro.
* Con người cần chuẩn bị trước về chính sách, giáo dục và đạo đức để đảm bảo rằng AI phục vụ lợi ích chung thay vì gây ra những tác động xấu đến xã hội.
* Sự đồng hành của các nhà khoa học, chính phủ và cộng đồng trong việc đặt ra các nguyên tắc an toàn và phát triển bền vững cho AGI là rất quan trọng.