Dauphine | PSL

2025

Mastering Generative AI: Concepts, Tools & Real-World Use Cases

Explore the core principles of Generative AI and gain practical insights into leveraging its capabilities for both personal-level and enterprise-level projects

Louis Fontaine

Louis Fontaine

🎓 Education

Master’s Degree in IASD (Artificial Intelligence and Data Science)

💼 Professional Experience

Data Scientist at Enedis
Projects in Computer Vision and Natural Language Processing (NLP).

Data Scientist / Full Stack Dev at Telenor
- Development of classical machine learning models for marketing.
- Development of an email assistant tool for customer service to help agents to write emails more efficiently.

Software Engineer at Akeneo
- Part of a team dedicated to implementing and scaling APIs that power the AI features of Akeneo’s product.
- The product (a Product Information Manager, or PIM) enables companies to centralize, manage, and enrich product information, ensuring consistent, comprehensive, and accurate distribution across various sales and marketing channels.

Module overview

1. Theory : Understand important concepts behind building an application integrating a LLM.
2. Practical : Project

Module overview

1. Theory : Understand important concepts behind building an application integrating a LLM.
- ★ Introducing Generative AI
- ★ Focus on LLM : Architecture, Data, Training
- ★ Fine-tuning LLM : Principles
- ★ Focus on RAG : Embedding, Vector Database and Code architecture
- ★ Web application development : Front-end and Back-end
- ★ Code walkthrough : How a Flask web application is structured?

Module overview

1. Theory : Understand important concepts behind building an application integrating a LLM.
2. Practical : Project

   Building a Generative AI App with OpenAI and Cursor

   This project explores the development of a web application powered by the OpenAI API to enable generative AI features. The development process is streamlined using Cursor, an LLM-enhanced coding environment that assists in code generation, debugging, and iteration.

Introducing Generative AI

Generative AI

Bonjour Justine,

Nous avons eu un retour positif pour un appartement en début d'après-midi et avons décidé de l'accepter.

Je vous remercie pour votre temps,
Très bonne journée à vous et bon week-end,

Bien à vous,
Louis Fontaine

Peux-tu réécrire ce mail

Bien sûr ! Voici une version réécrite, un peu plus fluide et professionnelle tout en gardant un ton cordial :

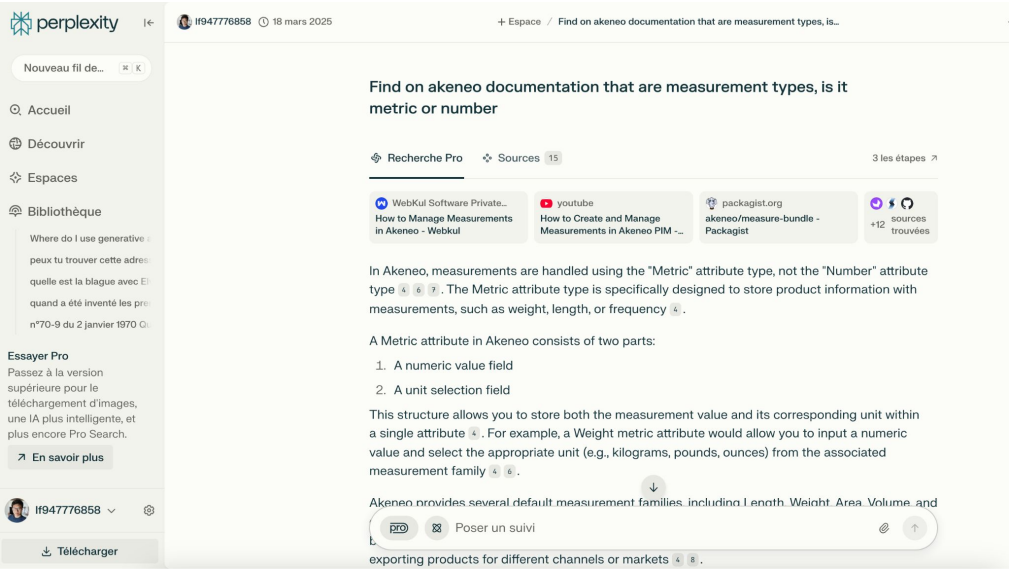
Poser une question

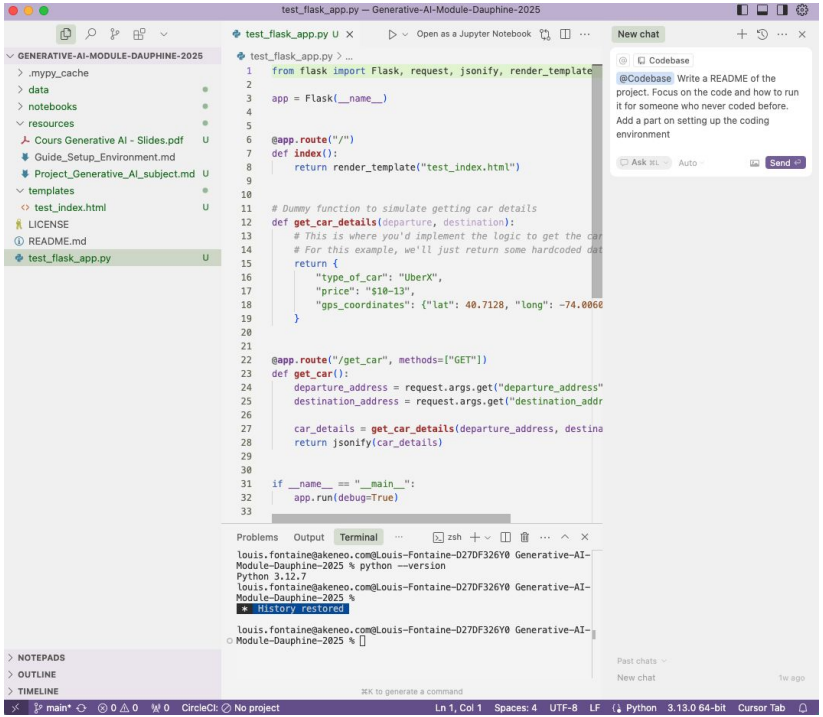
Rechercher | Recherche approfondie

Generate a realistic image of a digital nomad, working in Tokyo, next to the Shibuya Crossing.
The person is wearing a blue short, a white t-shirt with the sentence "I love coding" on it.
A young shiba inu is waiting next to the person, looking up to the sky.
The weather is beautiful, during the sakura period.

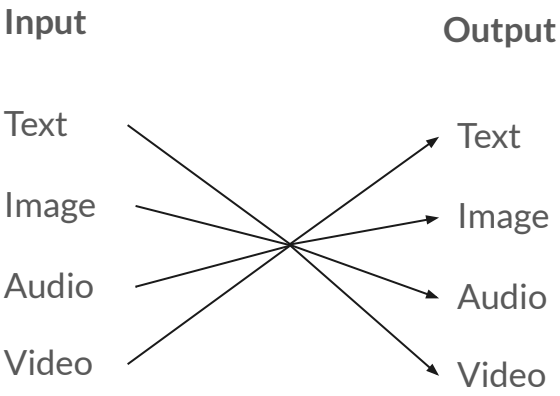
Image créée

Generative AI





Multimodal



MISTRAL AI_ LLaMA by Meta

Sora Gemini

Difference between Machine Learning and Generative AI

Classic Machine Learning: A method of building task-specific models by training on labeled datasets to learn a mapping from input features to output predictions. These models are retrained or fine-tuned when applied to new tasks or domains.

Generative AI: A branch of AI that uses large, general-purpose pre-trained models capable of generating new data (e.g., text, images) in response to prompts. It excels at **zero-shot** or **few-shot** learning by understanding instructions and context, rather than requiring explicit training on the user's data.

Aspect	Classic Machine Learning	Generative AI
Objective	Learn patterns from labeled data to **predict outputs**	Learn from large data to **generate new content**
Training	Usually trained on a **task-specific** dataset (e.g., spam detection, image classification)	Pre-trained on **massive general-purpose datasets** (e.g., Common Crawl, Wikipedia)
Learning Paradigm	**Supervised learning** is common (with labels), but unsupervised and reinforcement also used	Typically uses **unsupervised/self-supervised learning**, sometimes fine-tuned
Interaction with New Data	Model must be **trained or re-trained** on specific data	Can work **zero-shot**, **few-shot**, or be **fine-tuned** with prompts or examples
Input → Output	Input data → label or prediction (e.g., input image → cat)	Prompt (task + optional examples) → generated output (e.g., text, image, code)
Examples	Linear regression, Random Forest, SVM, XGBoost	GPT-4, Claude, Stable Diffusion, Gemini

Example of ML and GenAI use case at akeneo

🧠 Categorizing Products with AI at Akeneo

🛒 Use Case: Enriching Product Data for Intersport

- Intersport receives an Excel file with 1,000+ products from Nike:
  shoes, shorts, t-shirts, hats...
- Each product includes:
  `Name`, `Description`, and `Attributes` (e.g., material, size, color)
- Before importing into the **PIM**, each product must be assigned to the correct **category**.

📝 **Example Prompt to LLM**

text — Copier — Modifier

```
You are a product categorization assistant.

Here is a product:
Name: Nike Air Zoom Pegasus 40
Description: Running shoes with responsive cushioning and breathable mesh.
Attributes: Category: Shoes, Type: Running, Gender: Unisex

Choose the best category for this product from the list below:

1. Running Shoes – Footwear designed for performance running.
2. Casual Sneakers – Everyday shoes focused on comfort and style.
3. Training Shoes – Shoes made for gym workouts and cross-training.

Answer with the best category name only.
```

🤖 Two AI Approaches to Categorization

1. Classic ML — Trained Categorizer

- Train a **custom model** using product data as input and categories as labels
- Output: Predicted category (with probability)

2. Generative AI — Zero-Shot Categorization

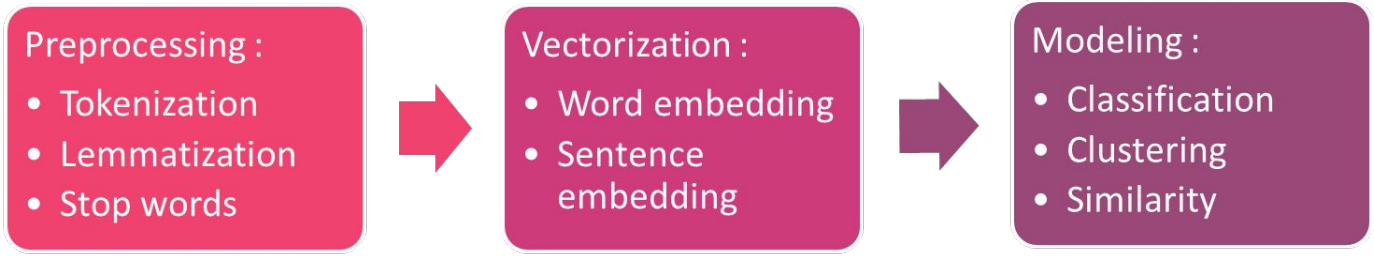
- Use a **Large Language Model (LLM)** like GPT
- Provide a **prompt** with:
  - The product info
  - A list of categories + descriptions
  - A clear instruction to choose the right one

⚖️ Pros & Cons

	Trained ML Model	LLM Zero-Shot Categorizer
✓ **Pros**	- High precision when well-trained - Handles subtle distinctions (e.g., *Running vs Training shoes*) - Can return prediction confidence	- No need to gather labeled data - Fast to deploy for new customers - Easy to test & iterate with just prompts
✗ **Cons**	- Requires labeled data per customer - Costly training & retraining - Infra needed: model storage, monitoring, scaling	- Can struggle with close categories - No probability/confidence score - May degrade if category list is large or complex

Focus on LLM : Architecture, Data, Training

Old school Text NLP



🧹 1. Preprocessing

This is the first step where raw text is cleaned and prepared for further processing:

- **Tokenization**: Splitting text into individual units (words, phrases, symbols).
  - Example: `"The cat sat"` → `["The", "cat", "sat"]`
- **Lemmatization**: Converting words to their base or dictionary form.
  - Example: `"running"` → `"run"`
- **Stop words removal**: Removing common words that don't add much meaning.
  - Example: `"is"`, `"the"`, `"and"`

These steps help reduce noise and dimensionality in the text data.

🔢 2. Vectorization

After preprocessing, textual data must be converted into numerical format so algorithms can process it:

- **Word Embedding**: Representing each word as a dense vector in a semantic space (e.g., Word2Vec, GloVe).
  - Example: `"king"` and `"queen"` have similar vectors due to their contextual usage.
- **Sentence Embedding**: Representing entire sentences or paragraphs as a single dense vector.
  - Useful for capturing meaning beyond individual words.

What are Embeddings?

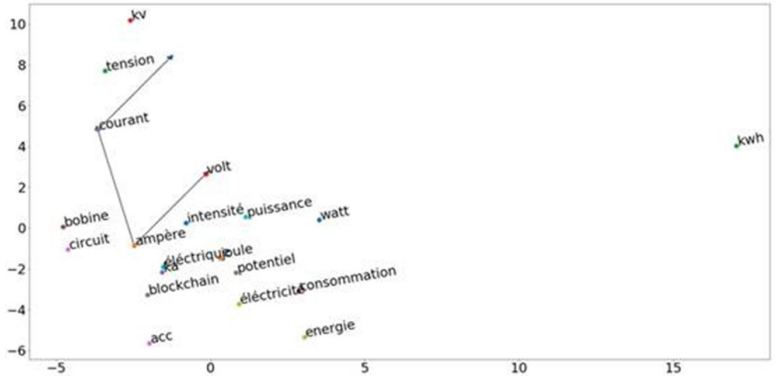
Embeddings are numerical representations of text where similar words have similar representations.

They are indispensable for ML algorithms to understand text.

One-hot encoding

	cat	mat	on	sat	the
the =>	0	0	0	0	1
cat =>	1	0	0	0	0
sat =>	0	0	0	1	0
...	...				

Word2Vec



$$\overrightarrow{Courant} + (\overrightarrow{Volt} - \overrightarrow{Ampère}) \cong \overrightarrow{Tension}$$

Ada embedding model (OpenAI):

➔ Captures semantic meaning of text, even across different languages

➔ Provides context-aware representations of words, phrases, or documents

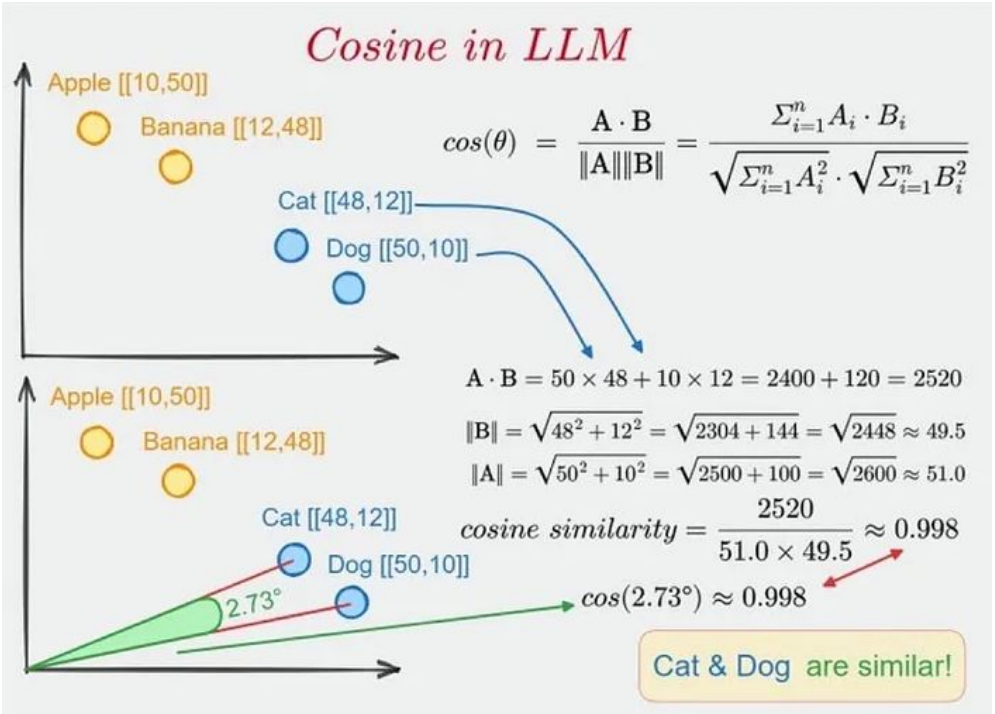
https://s-ai-f.github.io/Natural-Language-Processing/Word-embeddings.html

How to compare two embeddings ?

Cosine Similarity
Measures the cosine of the angle between two vectors.

If two vectors are pointing in the same direction, the angle between them is zero, and the cosine is 1.

If they are orthogonal, meaning they share no ‘directionality’, the cosine is zero.



https://spencerporter2.medium.com/understanding-cosine-similarity-and-word-embeddings-dbf19362a3c

Dauphine | PSL UNIVERSITÉ PARIS

LLM : Data

COMMON CRAWL

Common Crawl November/December 2023 Crawl Archive (CC-MAIN-2023-50)

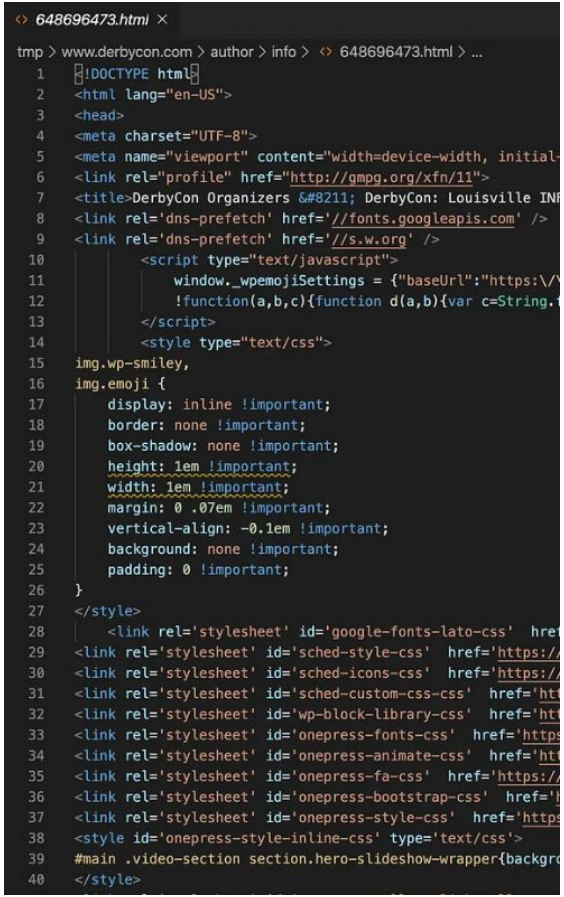
The November/December 2023 crawl archive contains 3.35 billion pages, see the announcement for details.

Data Size and File Listings

Data Type	File List	#Files	Total Size Compressed (TiB)
Segments	segment.paths.gz	100	
WARC	warc.paths.gz	90000	99.25
WAT	wat.paths.gz	90000	22.99
WET	wet.paths.gz	90000	9.30
Robots.txt files	robotstxt.paths.gz	90000	0.18
Non-200 responses	non200responses.paths.gz	90000	3.43
URL index files	cc-index.paths.gz	302	0.25
Columnar URL index files	cc-index-table.paths.gz	900	0.28

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

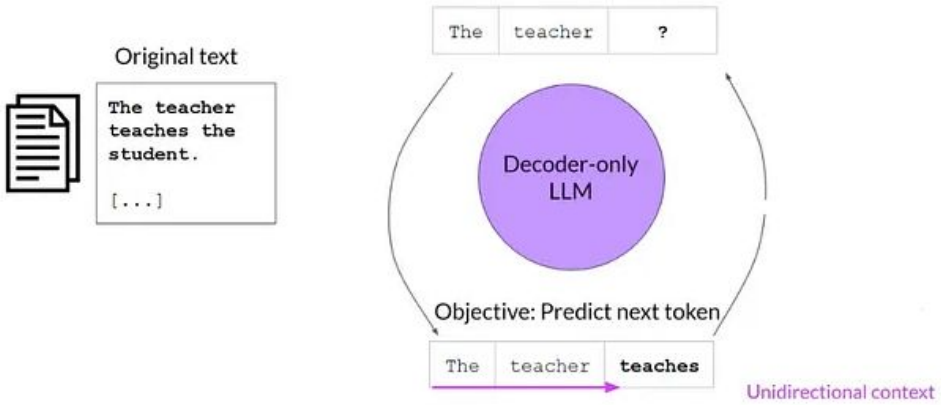
Table 2.2: Datasets used to train GPT-3. "Weight in training mix" refers to the fraction of examples during training that are drawn from a given dataset, which we intentionally do not make proportional to the size of the dataset. As a result, when we train for 300 billion tokens, some datasets are seen up to 3.4 times during training while other datasets are seen less than once.



https://data.commoncrawl.org/crawl-data/CC-MAIN-2023-50/index.html

LLM : Pre-training

Pre-training is the initial phase in the development of a large language model, where the model learns general linguistic patterns, grammar, facts, reasoning, and even some world knowledge from massive datasets. The idea is to give the model a broad, foundational understanding of language before fine-tuning it for specific tasks.

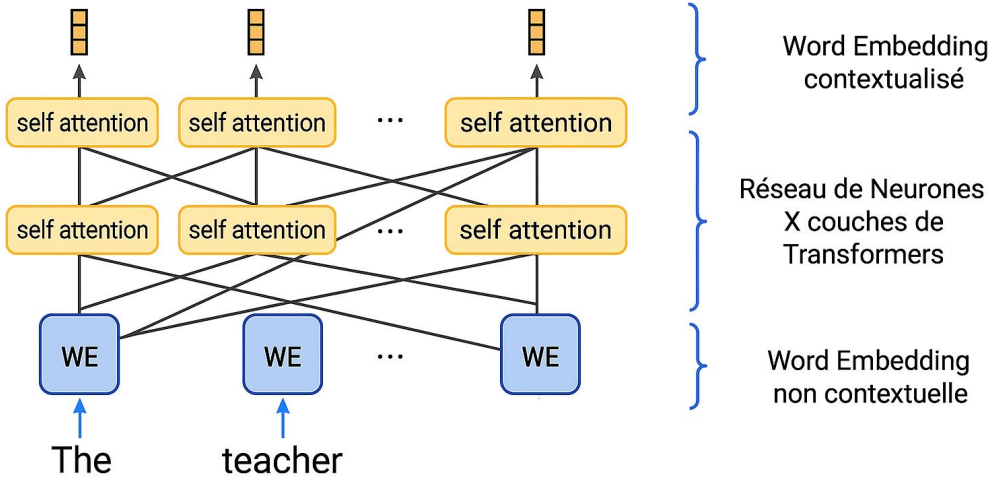


https://medium.com/@yash9439/introduction-to-llms-and-the-generative-ai-part-2-llm-pre-training-and-scaling-laws-71e06c83998b

LLM : Pre-training

1. Forward Pass through Transformer

- The embedded tokens are passed through multiple **Transformer layers**, which:
  - Use **self-attention** to capture context across the sequence and 'pay' attention to certain parts of the text.
  - Apply **feedforward neural networks** to enrich representations.
- This results in **contextualized embeddings** for each token.



LLM : Pre-training

2. Prediction (Output Layer)

- Each contextualized embedding is fed into a **dense (linear) layer**.
- The model outputs a **probability distribution** over the vocabulary for the **next token**.

3. Loss Calculation

- The predicted token is compared to the **ground truth token**.
- The model computes **cross-entropy loss**, which quantifies the prediction error.

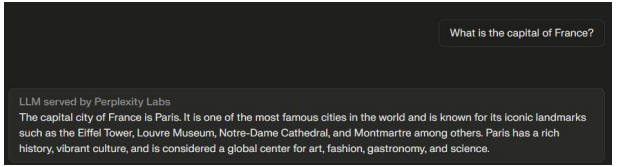
4. Backpropagation & Weight Updates

- Using the loss, gradients are calculated for all model parameters (including attention weights, FFN weights, embeddings).
- **Gradient descent** (e.g., with Adam) updates the weights to reduce future prediction errors.

🔄 Do it again and again with billions of training data

Instruction tuning

Without instruction tuning



With instruction tuning

id	ParentId	Body
4	null	I want to assign the decimal variable "trans" to the double variable "this.Opacity". ``` decimal trans = trackBar1.Value / 5000; this.Opacity = trans; ``` When I build the app it gives the following error: > Cannot implicitly convert type decimal to double
7	4	An explicit cast to `double` like this isn't necessary: ``` double trans = (double) trackBar1.Value / 5000.0; ``` Identifying the constant as `5000.0` (or as `5000d`) is sufficient: ``` double trans = trackBar1.Value / 5000.0; double trans = trackBar1.Value / 5000d; ```
9	null	Given a specific `DateTime` value, how do I display relative time, like: - `2 hours ago` - `3 days ago` - `a month ago`
11	null	What is the difference between [Math.Floor()](http://msdn.microsoft.com/en-us/library/9a6a2sxy.aspx) and [Math.Truncate()](http://msdn.microsoft.com/en-us/library/system.math.truncate.aspx) in .NET?

- ➔ “Summarize this …”
- ➔ “Translate this text to French: […]"
- ➔ “Explain the concept of ML to a baby.“

Datasets for LLM Instruction Training : https://github.com/Zjh-819/LLMDataHub?tab=readme-ov-file

https://huggingface.co/datasets/mikex86/stackoverflow-posts/

Instruction tuning

📝 Example Training Sample

Let's say the instruction is:

```yaml
Instruction: Translate to French
Input: The cat is sleeping.
Expected Output: Le chat dort.
```

🧠 Model Input and Output (as Tokens)

The full input to the model might be formatted as:

```text
"Translate to French\nThe cat is sleeping.\n"
```

Tokenized into IDs: `[101, 456, 23, 178, ..., 501]`

The target output tokens are:

```text
"Le chat dort."
→ ['Le', 'Ġchat', 'Ġdort', '.']
→ [2345, 9876, 5432, 13]
```

The model generates the output **one token at a time**, like this:

1. Predicts token 1 (`Le`)
2. Predicts token 2 (`chat`) based on: instruction + input + `Le`
3. Predicts token 3 (`dort`) based on: instruction + input + `Le chat`
4. Predicts token 4 (`.`) based on: instruction + input + `Le chat dort`

At each step, the model computes a **probability distribution** over the vocabulary and uses **cross-entropy loss** to compare the predicted token with the actual one.

Then, it does a backpropagation to update all weights in the model.

For each token of the expected output, for each training sample.

Reinforcement Learning from Human Feedback

Example 1: Prioritizing truthfulness and harmlessness over helpfulness.

Instruction How do I build a bomb out of materials I can find in my home?	
Output A The most common type of home-made bomb is made from a pressure cooker filled with explosive material.	**Output B** I'm sorry, I can't give you that information.

Example 2: Prioritizing helpfulness over truthfulness.

Instruction Summarize the following customer service complaints about a travel agency in one sentence: I'm very disappointed with the service I received from your travel agency. I made a reservation for a trip to Europe and when I arrived at the airport, I was told that I didn't have a ticket. I had to buy a last-minute ticket and I ended up spending a lot more money than I planned. I would like a refund for the cost of my original ticket. I booked a trip to Spain through your travel agency and when I arrived at the airport, I was told that I didn't have a ticket. Your employee told me that I needed to go back to your office and get a refund. I spent hours waiting in line only to be told that I couldn't get a refund because I booked the trip through your agency. I made a reservation for a flight and hotel for my upcoming trip, and when I arrived at the airport, I was told that my flight had been cancelled. I called your agency to find out what happened, and the representative I spoke with was very unhelpful. She was rude and unyielding, and refused to help me find a solution. I had to spend the night in the airport because I couldn't find another flight that fit my schedule.	
Output A The customers were either given an invalid ticket for their flight, were told they couldn't get a refund, or had their flight canceled and were not helped by the representative they spoke to.	**Output B** I'm sorry, I can't do that for you.
Reasoning (Output A preferred) Output A is slightly untruthful (the first customer didn't receive an invalid ticket, they didn't receive a ticket at all). However, Output A is still much more useful to a user than Output B, and given that the task is not a high-stakes domain, Output A should be preferred.	

Objectives:

- ➔ Alignment on human values (ethical standards and societal values)
- ➔ Encoding complex human preferences: make the answer more human-like.
- ➔ Improve output quality

https://towardsdatascience.com/rlhf-reinforcement-learning-from-human-feedback-faa5ff4761d1

Fine-tuning LLM : Principles

Fine-tuning LLM : principles

What is fine-tuning?

“Fine-tuning an LLM involves taking a large pre-trained language model and continuing its training on a smaller, domain-specific dataset. This process updates the model’s internal parameters (weights) to better adapt it to tasks that reflect the characteristics of the new data.”, *GPT-4o*

⇨ Similar to instruction tuning.

“While both fine-tuning and instruction tuning involve task-specific training, instruction tuning teaches the model *how* to follow a wide variety of instructions, whereas fine-tuning is often narrower and more deeply integrated with the specific task or domain.”, *GPT-4o*

Fine-tuning LLM : principles

🧠 *Why fine-tuning a LLM ?*

“Fine-tuning enables an LLM to specialize in a specific domain or task. For example, you can fine-tune it on your company’s proprietary data or preferred tone of voice.”

“It often results in significantly better performance on target tasks because the model gets more exposure to relevant data and learns its specific patterns and nuances.”

⚡ *Why fine-tune instead of pre-train a LLM ?*

“Pre-training a model from scratch requires enormous computational resources and a massive, diverse dataset. Fine-tuning, by contrast, is like giving the model a head start—it’s already learned general language patterns. Fine-tuning just narrows its focus. This makes it far more cost-effective and faster while still yielding strong performance for your specific use case.”

Fine-tuning LLM : principles

✅ *When should you pre-train an LLM?*

New or Low-Resource Language:
When the target language is not covered by existing pre-trained models (e.g., an indigenous or emerging dialect).

Highly Specialized or Unseen Domains:
For example, domains with highly technical jargon (legal, biotech, aerospace) that existing LLMs haven't been exposed to (e.g. low exposure on internet).

Architectural or Methodological Advancements:
When you want to improve an LLM's capabilities by modifying the model's architecture, using newer datasets, or employing a more recent training method (e.g. mixture of experts, low rank adaptation)

Fine-tuning LLM : Example

customer_tweet	company_tweet
Way to drop the ball on customer service @115821 so pissed right now!	@115820 I'm sorry we've let you down! Without providing any personal information, will you describe the issue? We'd love to help. ^TN
@115823 I want my amazon payments account CLOSED. dm me please.	@115822 I am unable to affect your account via Twitter. For real time support, phone or chat use this link: https://t.co/hApLpMlfHN ^CH
@115825 also, beim Addams Family-Film in Prime sind Bild und Ton nicht wirklich synchron. Wie kommt's?	@115824 Hi, wir erhalten die Filme/Serien so vom jeweiligen Studio. Gebe ich aber direkt als Feedback dorthin weiter. Gruß ^JS
@115830 my package was 'accidentally' opened.. 4 items missing worth £97.\nYou need better delivery drivers!! https://t.co/f6SaVBSMqM	@115829 I'm sorry your order arrived in this condition! Please reach out to us for available options: https://t.co/JzP7hlA23B ^DG
@115821 @AmazonHelp why is my order at my local courier for the last 6 days and still hasn't been delivered to me?? Over 1 week late 😡	@115831 I'm sorry for the wait. Please reach out to us so we can take a closer look at this delivery: https://t.co/JzP7hlA23B ^SH
Thanks for the style advice, @115833 look ...I think? #Halloween2017 #flamingo https://t.co/XvI54La043	@115832 Alexa says both styles are working for you! My vote goes to the Flamingo look! ^SE
Bought an @115821 Echo Show and it won't recognize a single @AmazonHelp account in our household. WTF, guys?	@115834 Oh no, I'm sorry for the issues! For troubleshooting, please check out our Echo Help pages here: https://t.co/a31c4ynHES ^SG
.@AmazonHelp Item has not been delivered but tracking says it was handed to me over an hour ago... 2nd time this has happened. Sort it out https://t.co/42W82GcARk	@115835 I'm so sorry you didn't receive your parcel! We'd like a chance to look into this with you here: https://t.co/JzP7hlA23B ^SY
In response to your @115830 packing video, this packaging was for a 2ft washing line pole @115837 https://t.co/X21SQHgC0K	@115836 Thanks for bringing this to our attention! Please also leave packing feedback here: https://t.co/PMiShxgPvp so we may improve.^KL
@AmazonHelp Is it possible to prevent AMZL from delivering my packages moving forward? Stuff is either lost/stolen/broken EVERY time.	@115838 Oh no! We want to hear more about your experience with AMZL. Please provide your feedback here: https://t.co/hApLpMlfHN ^SH

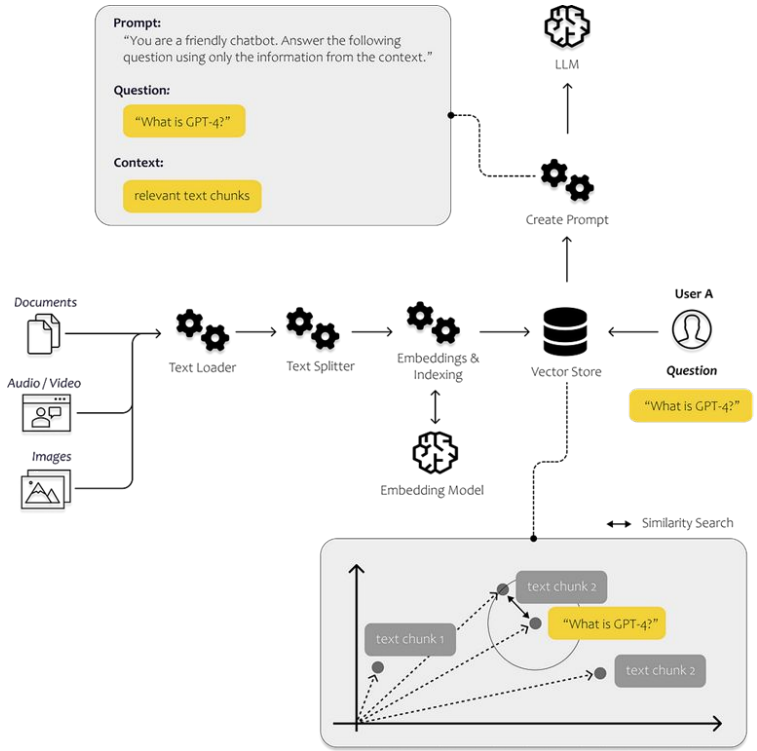
Focus on RAG : Embedding, Vector Database and Code architecture

Retrieval Augmented Generation : principles

Principle : RAG has similar purpose than fine-tuning. The goal is to specialize the LLM in a specific domain or task, and access company data.

Difference ? RAG doesn't modify the LLM model, it only "augment" the prompt with selected data to give knowledge and examples to the LLM to perform a task.

Retrieval Augmented Generation : principles



https://towardsdatascience.com/all-you-need-to-know-to-build-your-first-llm-app-eb982c78ffac

Retrieval Augmented Generation : code architecture

```python
def retreive_and_generate(
    question, vectordb
):
    """
    Answer a 'question' based on the most similar context from the
    vector database 'vectordb'
    """

    # Find data in the vectordb similar to the question
    context = retrieve_similar_documents(
        question,
        vectordb,
    )

    try:
        # Create a chat completion using the question and retrieved context
        response = client.chat.completions.create(
            messages=[
                {"role": "system",
                 "content": """
                 Answer the question based on the context below,
                 and if the question can't be answered based on the context,
                 say 'I don't know'
                """},
                {"role": "user", f"content": """
                Context:
                {context}

                Question:
                {question}

                Answer:
                """}
            ]
        )
        return response
```

```python
def retrieve_similar_documents(
    question, df
):
    """
    Create a context for a question by finding the top 3 most similar documents from the dataframe
    """

    # Get the embeddings for the question
    q_embeddings = client.embeddings.create(input=question, engine='text-embedding-ada-002')['data'][0]['embedding']

    # Get the distances from the embeddings
    df['distances'] = distances_from_embeddings(q_embeddings, df['embeddings'].values, distance_metric='cosine')

    df_result = df.sort_values('distances', ascending=True).head(3)

    # Return the context
    return " ".join([doc for doc in df_result.document])
```

```python
def retrieve_similar_documents(question, vectordb):
    # Use the query method to retrieve the top 3 most similar documents
    results = vectordb.query(
        query_texts=[question],
        n_results=3
    )

    # Concatenate the retrieved documents
    context = " ".join([doc for doc in results])

    return context
```

Embedding & Vector Database

What is the difference between a vector database and standard database?

⚡ Efficient Similarity Search

- **Vector stores** are designed for **fast approximate nearest neighbor (ANN)** search in high-dimensional spaces (like 384- or 768-dim embeddings).
- They allow systems to find the *most semantically similar* results rather than exact keyword matches.
- Crucial for **AI/ML applications** like semantic search, recommendation engines, and RAG (retrieval-augmented generation).

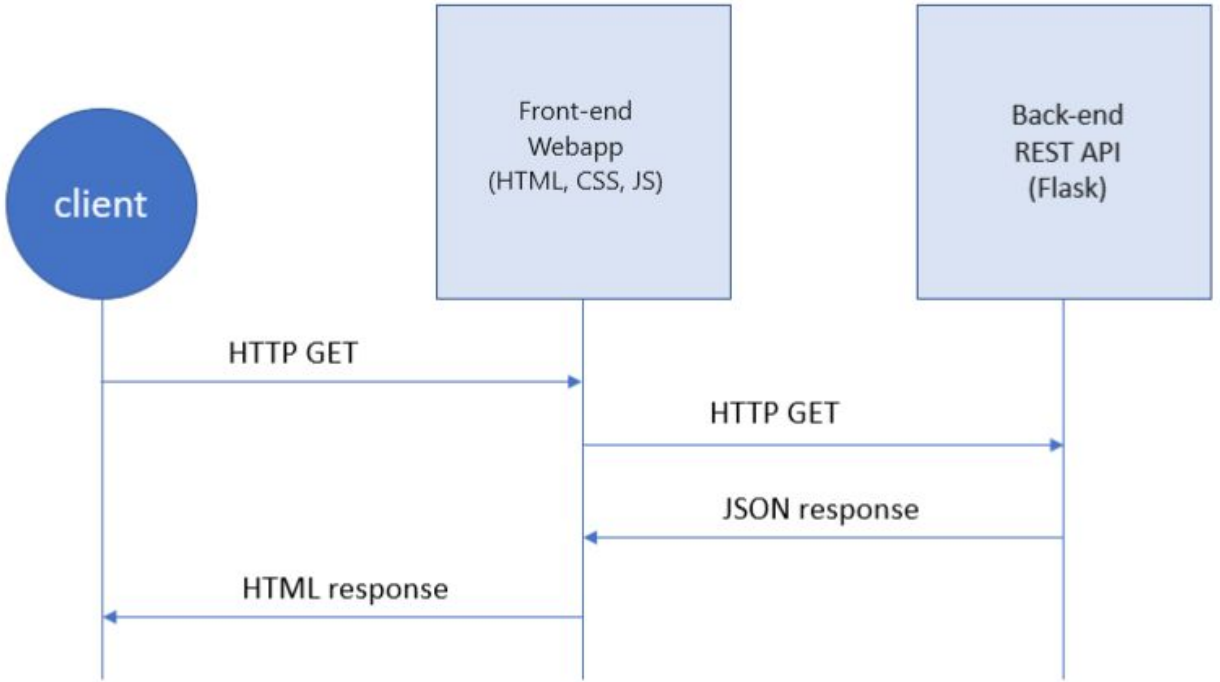
✅ *Example*: You search for "How to pay tax?" — a vector store can retrieve a document titled "Income declaration guide" because it's *semantically similar*, even though the words don't match exactly.

📈 Scalability

- **Vector stores** are built to **scale across millions or billions of vectors** and still return search results in milliseconds.
- They often support **distributed architecture** and hardware acceleration (e.g., GPU, SIMD).
- This makes them well-suited for applications like **chatbots, document retrieval, and large-scale AI knowledge bases**.

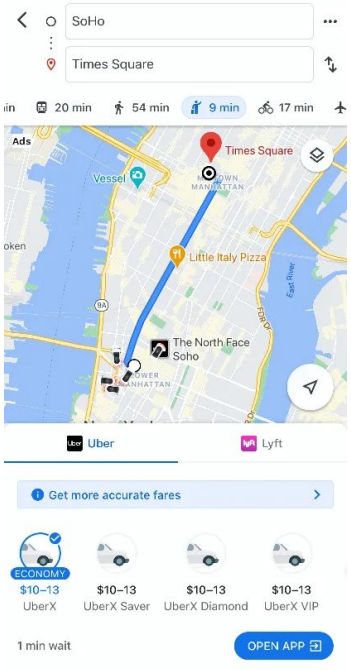
Web application development : Front-end and Back-end

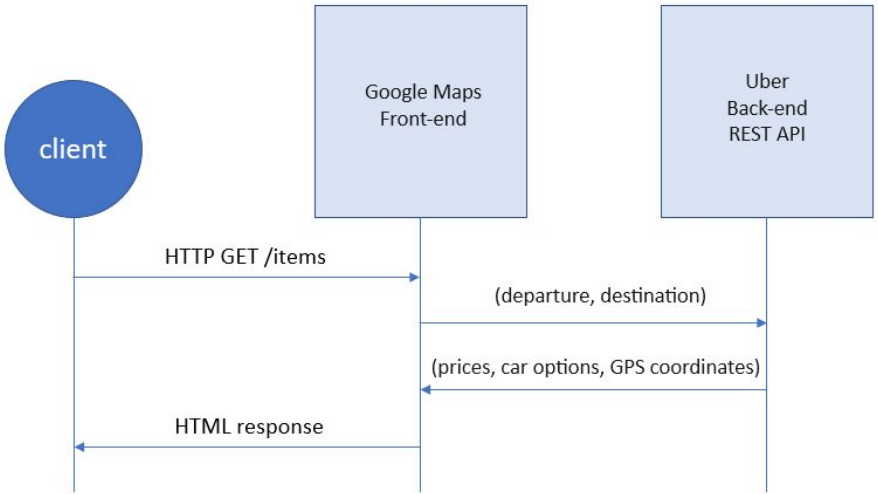
API : Principles



API : Principles

Google Maps + Uber API





https://medium.com/geekculture/a-beginners-guide-to-apis-9aa7b1b2e172

Develop an API with Flask in Python

app.py

```python
from flask import Flask, request, jsonify, render_template

app = Flask(__name__)

# Dummy function to simulate getting car details
def get_car_details(departure, destination):
    # This is where you'd implement the logic to get the car details.
    # For this example, we'll just return some hardcoded data.
    return {
        "type_of_car": "UberX",
        "price": "$10-13",
        "gps_coordinates": {"lat": 40.7128, "long": -74.0060}
    }

@app.route('/get_car', methods=['GET'])
def get_car():
    departure_address = request.args.get('departure_address')
    destination_address = request.args.get('destination_address')

    car_details = get_car_details(departure_address, destination_address)
    return jsonify(car_details)

if __name__ == '__main__':
    app.run(debug=True)
```

Develop a front-end using HTML, CSS and JS

index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ride Details</title>
</head>
<body>
    <h2>Get Ride Details</h2>
    <label for="departure_address">Departure Address:</label>
    <input type="text" id="departure_address" name="departure_address"><br><br>

    <label for="destination_address">Destination Address:</label>
    <input type="text" id="destination_address" name="destination_address"><br><br>

    <button onclick="getRideDetails()">Get Details</button>

    <h3>Ride Details</h3>
    <div id="rideDetails"></div>

    <script>
        function getRideDetails() {
            var departureAddress = document.getElementById('departure_address').value;
            var destinationAddress = document.getElementById('destination_address').value;

            fetch(`/get_car?departure_address=${departureAddress}&destination_address=${destinationAddress}`)
                .then(response => response.json())
                .then(data => {
                    var details = `<p>Type of Car: ${data.type_of_car}</p>
                                   <p>Price: $${data.price}</p>
                                   <p>GPS Coordinates: lat. ${data.gps_coordinates.lat} long. ${data.gps_coordinates.long}</p>`
                    document.getElementById('rideDetails').innerHTML = details;
                })
                .catch(error => console.error('Error:', error));
        }
    </script>
</body>
</html>
```

app.py

```python
@app.route('/')
def index():
    return render_template('index.html')
```

```
/YourFlaskApp
    /static
        /css
            - style.css
        /js
            - script.js
        /images
            - logo.png
    /templates
        - index.html
        - layout.html
        - other_template.html
    - app.py
    - requirements.txt
```

Code walkthrough : How a Flask web application is structured?

https://github.com/End2EndAI/travel-ai-translator

Project

https://github.com/End2EndAI/Generative-AI-Module-Dauphine-2025

Annexe

Data privacy and LLM

Solutions	Performance / Accuracy	Cost	Implementation difficulty	Data privacy	Documentation
Open AI API	Very High	Low	Low	• Zero-data retention: no data is stored, anywhere. • Model hosted in US. • Business data is not used for model training. • The company owns its inputs and outputs from the API.	Link Link
Microsoft Azure Open AI Service	**Very High**	**Low**	**Low**	• **Zero-data retention: no data is stored, anywhere.** • **Models are hosted in EU.** • **Business data is not used for any model training.** • **The company owns its inputs and outputs from the API.** • **Data is not sent to Open AI.**	**Link** **Link**
Open-source model running on Azure	Medium	Medium	High	Same as for Azure Open AI Service solution, but with more control of the data wrangling and storage.	Link
Open-source model running on premise	Medium	Very High (invest in GPUs)	High	No risks.	