

# Formation OpenAI

Génération de contenus avec ChatGPT et DALL-E  
& reconnaissance vocale avec Whisper



**Alan GUYOLLOT**  
Formateur OpenAI | Développeur



# Et vous?

- Nom Prénom
- Âge
- Parcours
- Aspirations professionnelles



# Objectifs de la formation

---

## Formation

### Pré-requis :

- Maîtrise des bases du langage Python

### Objectifs :

- Découvrir les ressources et outils d'OpenAI disponibles pour les développeurs.
- Développer un chatbot utilisant les API d'OpenAI, notamment ChatGPT, DALL-E 2 et la reconnaissance vocale.
- Développer une application "intelligente" en intégrant ChatGPT et DALL-E.

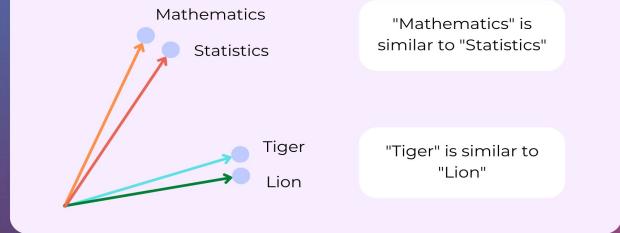
# Planning de la formation

## Module 01

### Introduction: Comprendre l'IA et ses applications avec OpenAI

- Principes de fonctionnement de l'IA (Intelligence Artificielle)
- Présentation d'OpenAI
- Traitement du langage naturel (NLP)
- Grand modèle de langage (LLM)
- Plateforme développeur OpenAI

### Word Embeddings



## Module 02

### ChatGPT: Concept & Implémentation

- ChatGPT c'est quoi?
- Les différents modèles de GPT, fonctionnalités, tarification
- Outil de développement: Playground
- Prompt Engineering
- Implémentation Python
- Intégration dans une application web (Streamlit)



# Planning de la formation

## Module 03



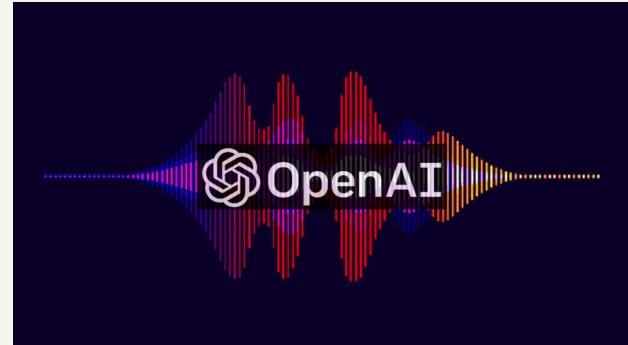
### DALL-E: Génération d'images

- Présentation de DALL-E 2 et son fonctionnement
- Génération d'images à partir de descriptions en langage naturel
- Exploration des fonctionnalités de DALL-E 2
- Atelier : Exemples d'utilisation de DALL-E 2

## Module 04

### Whisper & TTS: Transcription & generation audio

- Whisper: introduction
- Whisper: implémentation
- Conversion texte en audio (TTS)
- Atelier : Implémentation d'un assistant vocal avec ChatGPT



# Planning de la formation



## Module 05

### GPT4: Personnalisation et Optimisation

- Fine-tuning: Introduction
- Fine-tuning: Modèles et coûts
- Fine-tuning: Configuration
- Fine-tuning: Implémentation
- Optimisation: Prompt caching
- Optimisation: Batch API

## Module 06

### Mise en situation

- Développement d'une application intelligente
- Intégration combinée des différentes notions de la semaine

# 01

# Premiers pas dans l'IA

- Expliquez l'IA en quelques mots
- Avez vous déjà utilisé Chat GPT?



Le Parlement européen définit l'intelligence artificielle comme « *tout outil utilisé par une machine capable de "reproduire des comportements liés aux humains, tels que le raisonnement, la planification et la créativité".* »



## Il existe 3 types d'apprentissage d'IA :

**Apprentissage supervisé** : l'IA est entraînée sur des données étiquetées pour établir des relations entre les entrées et sorties.

*Exemple : classification d'images ou prédition de prix.*

**Apprentissage non supervisé** : l'IA identifie des motifs ou des regroupements dans des données non étiquetées.

*Exemple : segmentation de clients ou détection d'anomalies.*

**Apprentissage par renforcement** : l'IA apprend à prendre des décisions grâce à un système de récompenses/punitions.

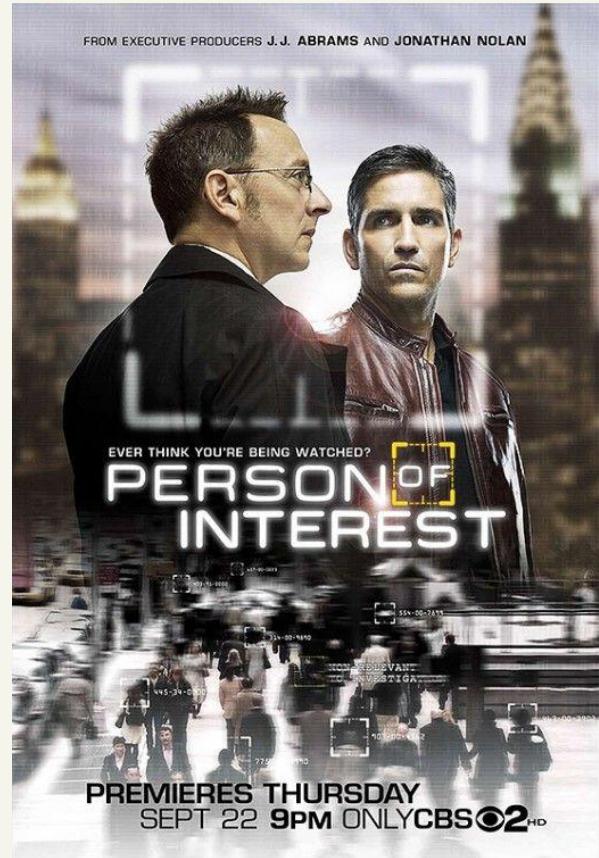
*Exemple : contrôle de robots ou stratégies de jeu.*

## 1.1.1 IA - Annexe 1 (Person of Interest)

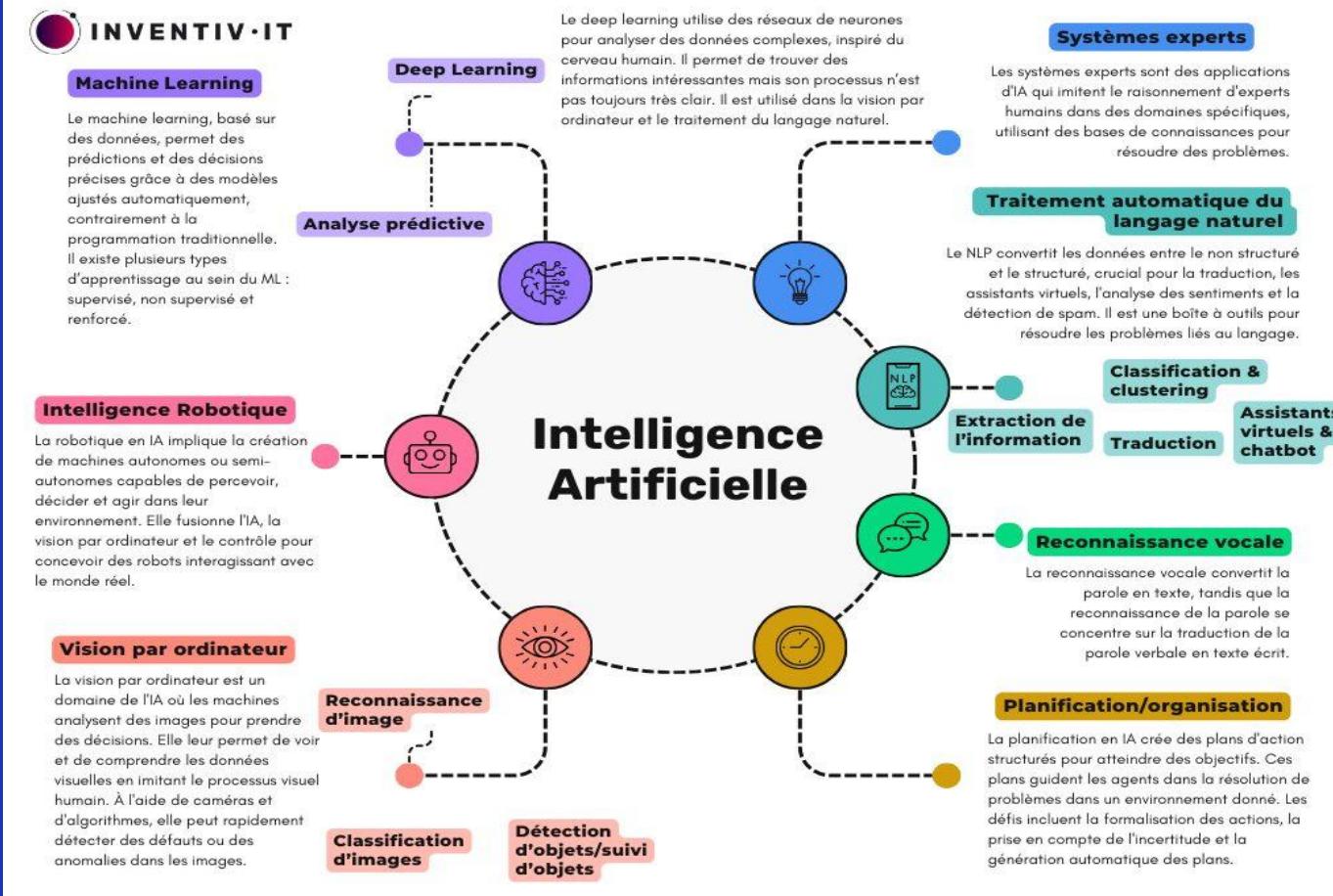
- **Person of Interest** est une série de science-fiction diffusée de 2011 à 2016.
- L'histoire suit **John Reese**, un ancien agent de la CIA, et **Harold Finch**, un milliardaire et génie informatique, qui utilisent une IA avancée, **la Machine**, pour prévenir des crimes avant qu'ils ne se produisent.
- **La Machine** analyse des millions de données (vidéos, appels, réseaux sociaux) pour prédire des événements, mais elle distingue les **menaces majeures** (terrorisme) des **menaces mineures** (criminels ordinaires).

L'IA dans la série :

- **Analyse prédictive et Big Data** : L'IA utilise des données massives pour identifier des menaces potentielles.
- **Éthique et surveillance** : Souligne les dangers de l'utilisation excessive de la surveillance et les questions de vie privée.



# 1.1.1 Applications de l'IA



[Lien de l'image](#)

## 1.1.2 OpenAI



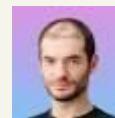
Fondée en 2015 par 5 entrepreneurs informaticien et chercheur en IA. OpenAI est une organisation à but non lucratif qui a mis en place une entité à but lucratif : [OpenAI LP](#).

L'objectif d'Open AI est de créer une IAG sûre qui profite à l'humanité.  
Leur approche met l'accent sur la transparence, la sécurité et l'accessibilité.

**IAG** : Intelligence Artificielle générale - capacité d'une machine autonome à réaliser toutes les tâches intellectuelles humaines



[Sam Altman](#) - homme d'affaires américain -  
CEO d'OpenAI



[Ilya Sutskever](#) - informaticien spécialisé  
dans l'apprentissage automatique  
(ancien de Google Brain)



[Greg Brockman](#) - Président d'Open IA  
entrepreneur, investisseur et développeur de  
logiciels américain



[John Schulman](#) - chercheur scientifique et dirige  
l'équipe de recherche apprentissage par  
renforcement.



[Elon Musk](#) - entrepreneur et chef d'entreprises  
de Tesla, X, Space X, un des premiers donateurs  
d'OpenAI.



[Wojciech Zaremba](#) - informaticien polonais,  
dirige les équipes de recherche et linguistiques  
et **Codex**.

## 1.1.2 OpenAI - Services

### Offres de services

Grand Public

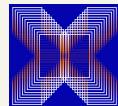


[ChatGPT](#) : Service basé sur la série de modèles GPT de OpenAI. Il a été optimisé pour des conversations interactives, et peut être utilisé pour diverses applications, allant des chatbots aux assistants virtuels et au-delà. (NLP)

### DALL·E 2

[DALL·E](#) : Système d'IA capable de créer des images et des œuvres d'art réalistes à partir d'une description en langage naturel basé sur GPT-3. (Computer Vision)

Développeurs



[Codex](#) : Un modèle dérivé de GPT-3 destiné spécifiquement à la génération de code. Il alimente également le produit GitHub Copilot, qui assiste les développeurs en suggérant des lignes de code pendant la programmation.

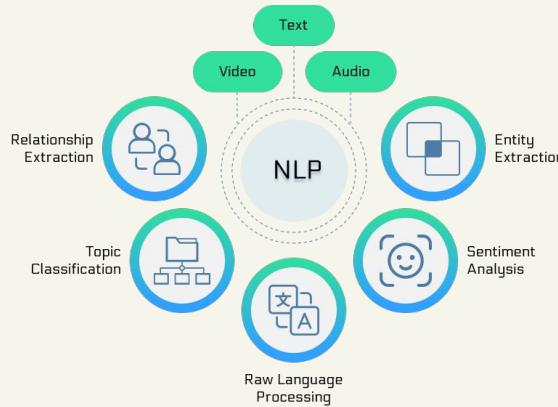


[OpenAI API](#) : Cette API permet aux développeurs d'accéder au modèle de traitement du langage naturel GPT-3 de OpenAI pour diverses applications, allant de la génération de contenu à la création d'applications de chatbot.

[Research](#) : OpenAI publie régulièrement des travaux de recherche sur des sujets d'IA, offrant à la communauté des insights et des avancées techniques.

## 1.2.1. Natural Language Processing (NLP)

### Définitions : Natural Language Processing (NLP)



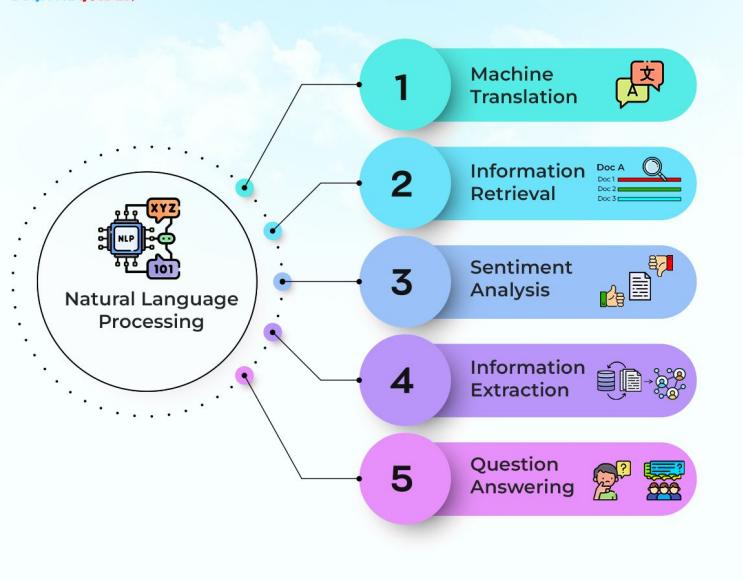
Le NLP, ou Traitement du Langage Naturel (Natural Language Processing en anglais), est un domaine de l'intelligence artificielle qui se concentre sur la manière dont les ordinateurs peuvent comprendre, interpréter et générer du langage humain de manière naturelle.

Le NLP comprend un large éventail de tâches, notamment la traduction automatique, la compréhension du langage, la génération de texte, la classification de texte et bien d'autres encore.

## 1.2.1. Natural Language Processing (NLP)

# Définitions : Natural Language Processing (NLP)

DIGITALGUIDER



## Classification de texte



## Chatbots



## Résumé et générateur de texte

## 1.2.1. Natural Language Processing (NLP)

### Exemple de l'analyse de sentiment

Dans le cas d'une tâche d'analyse de sentiment associé à un texte, la donnée est transformé sous forme numérique puis envoyé à un algorithme qui sortira une donnée binaire :

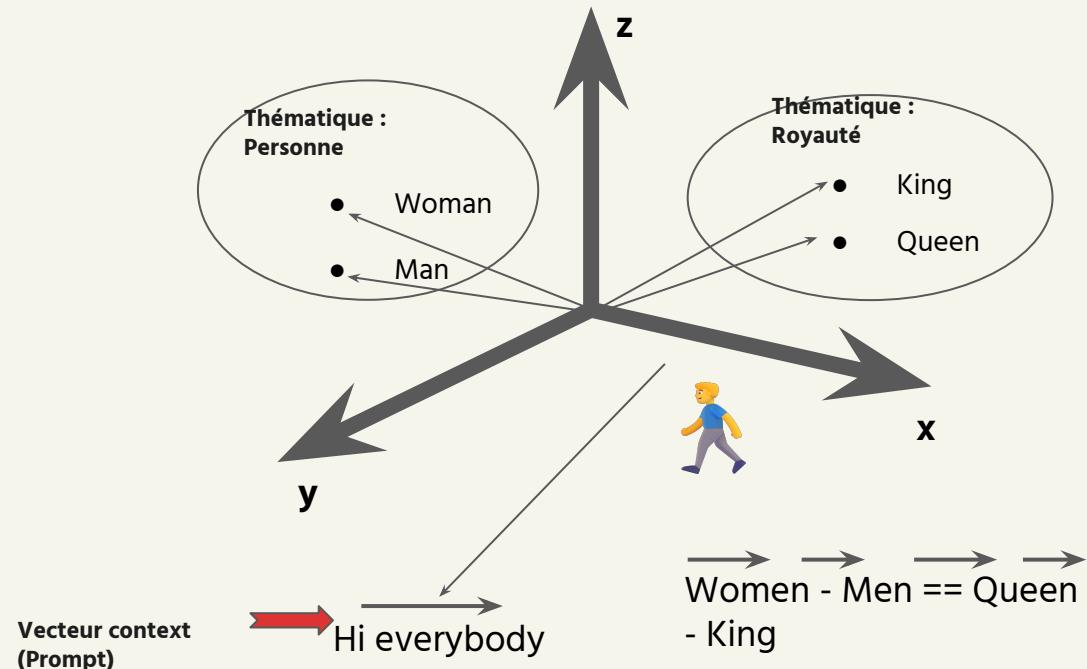
0 pour un sentiment négatif, 1 pour un sentiment positif.



## 1.2.1. NLP - WordEmbedding

### Espace de projection des tokens

**Word Embedding** : Cette technique permet de transformer des données textuelles en données numériques compréhensibles par les algorithmes d'apprentissage automatique tout en capturant leur sens et leurs relations.



## 1.2.2 NLP - WordEmbedding

### En quelques mots

Un **token** est une unité de texte, souvent un mot, une sous-partie d'un mot, ou parfois même un caractère.

**Tokenisation** : La tokenisation est l'étape cruciale de la division du texte en unités plus petites pour une analyse ultérieure. La tokenisation peut être utilisée pour diviser les données textuelles en mots, phrases, symboles et autres unités pertinentes.

Un **word embedding** est une représentation numérique (un vecteur) associée à chaque token. Ces vecteurs encodent les propriétés sémantiques et syntaxiques des tokens.

En résumé, les **tokens** sont les éléments de base du texte (comme des pièces de puzzle), et les **word embeddings** sont les représentations numériques (vecteurs) attribuées à ces tokens pour que les modèles puissent comprendre leur signification.

### Exemple

Texte : "Le chat mange"

Étape 1 : **Tokenisation** → ["Le", "chat", "mange"]

Étape 2 : **Word Embedding** →

- "Le" → vecteur [0.2, 0.1, 0.3, ...]
- "chat" → vecteur [0.5, 0.7, 0.6, ...]
- "mange" → vecteur [0.4, 0.3, 0.9, ...]

## 1.2.1. Natural Language Processing (NLP)

### Exemple de Tokenisation

Large Language Models (LLMs), such as GPT-3 and GPT-4, utilize a process called tokenization. Tokenization involves breaking down text into smaller units, known as tokens, which the model can process and understand. These tokens can range from individual characters to entire words or even larger chunks, depending on the model. For GPT-3 and GPT-4, a Byte Pair Encoding (BPE) tokenizer is used. BPE is a subword tokenization technique that allows the model to dynamically build a vocabulary during training, efficiently representing common words and word fragments. Although the core tokenization process remains similar across different versions of these models, the specific implementation can vary based on the model's architecture and training objectives.

## 1.2.3 LLM

### Large Language Model (LLM)

#### Apprentissage

Entraînés sur d'immenses corpus de textes, ils apprennent à reconnaître les patterns linguistiques et les relations sémantiques complexes.

#### Capacités

Ils excellgent dans la compréhension du contexte, la génération de texte, la traduction, et peuvent s'adapter à une grande variété de tâches linguistiques.

#### Définition

Les Grands Modèles de Langage (LLM) sont des systèmes d'intelligence artificielle conçus pour comprendre et générer du langage naturel à grande échelle.

Basés sur des architectures de transformers, ces modèles utilisent des milliards de paramètres pour traiter et générer du texte de manière contextuelle.

## Large Language Model (LLM)

### Capacités

Génération de texte cohérent (rédaction d'articles, réponses aux questions).

Résolution de problèmes complexes (mathématiques, logique).

Compréhension du contexte grâce à des milliers de paramètres.

### Limites

**Biais** : les modèles reproduisent parfois les biais présents dans leurs données d'entraînement.

**Compréhension imparfaite** : bien qu'impressionnantes, ils n'ont pas de véritable compréhension humaine.

**Ressources nécessaires** : l'entraînement et l'utilisation à grande échelle demandent des ressources significatives.

# 1.3. API OpenAI

**Présentation de la plateforme développeur OpenAI**

Overview Documentation API reference Examples Playground Forum Help Dawan

Welcome to the OpenAI platform

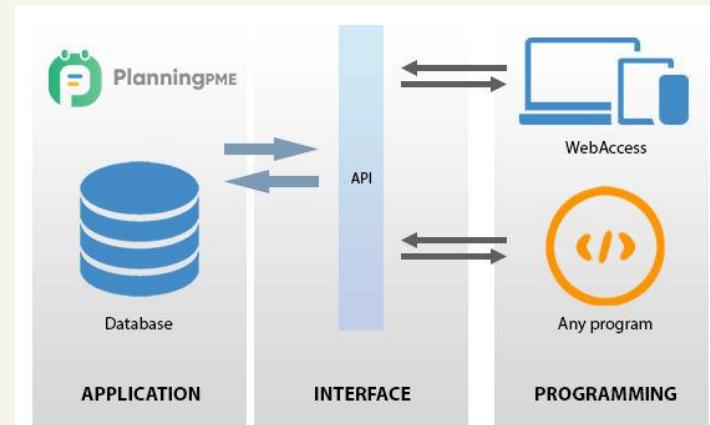
L'API OpenAI permet aux développeurs d'utiliser GPT-3 pour des applications comme la génération de contenu ou les chatbots.

Une entreprise peut utiliser l'API OpenAI pour intégrer à la fois GPT (pour répondre à des questions) et DALL-E (pour générer des visuels) dans une même application.

Accessible sur [platform.openai.com](https://platform.openai.com), elle propose :

- Une **page d'accueil** avec les fonctionnalités principales.
- Une **documentation** avec des tutoriels et guides.
- Une **référence API** pour les points de terminaison.
- Des **exemples** d'utilisation.
- Un **Playground** pour tester les modèles.

La plateforme permet aussi de gérer les comptes utilisateurs (Reader et Owner), d'utiliser un outil de tokenisation et de générer des clés API.



API : interface logicielle qui permet de « connecter » un logiciel ou un service à un autre logiciel ou service afin d'échanger des données et des fonctionnalités.

# 02 ChatGPT

# 02. ChatGPT

## Introduction

ChatGPT est un modèle de langage basé sur l'architecture GPT (Generative Pre-trained Transformer) développé par OpenAI et rendu accessible au public en novembre 2022.

Il s'agit d'une intelligence artificielle capable de comprendre et de générer du texte en langage naturel. ChatGPT peut répondre à des questions, résoudre des problèmes, écrire des textes, coder et bien plus encore.



### Architecture:

L'architecture GPT est basée sur des réseaux neuronaux de type "transformer" en couches, un concept introduit en 2017 par Ashish Vaswani et ses collègues dans l'article "[Attention is All You Need](#)".

Cette approche utilise un mécanisme d'attention pour attribuer de l'importance aux différentes parties du texte.

Les modèles GPT sont formés sur de vastes ensembles de données textuelles provenant d'Internet, sans supervision explicite.

### Entrainement:

Pré-entraînement : Le modèle est exposé à une grande quantité de données textuelles pour apprendre les règles du langage, les relations contextuelles et les structures grammaticales.

Fine-tuning : Le modèle est affiné avec des exemples étiquetés par des humains pour améliorer sa capacité à répondre de manière utile et sûre.

## Offres

### Offre Gratuite:

- OpenAI propose un accès gratuit à ChatGPT, permettant à tout un chacun d'utiliser l'outil pour générer du texte, répondre à des questions, obtenir de l'aide pour la rédaction, et plus encore.
- Les utilisateurs de la version gratuite peuvent rencontrer certaines limitations, telles que des temps de réponse plus lents pendant les périodes de forte demande, ou parfois l'indisponibilité du service si les serveurs sont saturés.

### ChatGPT Plus :

- **Coût** : Abonnement mensuel payant (environ 20\$ par mois)
- **Temps de Réponse Plus Rapides** : Les utilisateurs de ChatGPT Plus bénéficient de temps de réponse plus rapides, ce qui améliore l'expérience utilisateur, surtout en période de forte affluence.
- **Accès Prioritaire** : Pendant les périodes de forte demande, les abonnés ChatGPT Plus ont un accès prioritaire, réduisant ainsi le risque de voir le service indisponible.
- **Accès aux Améliorations** : Les abonnés peuvent également avoir accès à des fonctionnalités améliorées ou à des versions avancées du modèle, comme des mises à jour de capacité ou de performance.

## 02. ChatGPT

### Versions

| Modèle        | Date de sortie | Particularités                          | Prix par 1M token (input) | Prix pour 1M token (output) | Coupe de Connaissances |
|---------------|----------------|---|---------------------------|-----------------------------|------------------------|
| o1-preview    | Septembre 2024 | Raisonnement                            | \$15                      | \$60                        | Octobre 2023           |
| GPT-4o-mini   | Juillet 2024   | Plus petit<br>Moins cher<br>Plus rapide | \$0.15                    | \$0.60                      | Octobre 2023           |
| GPT-4o        | Mai 2023       | Plus grand<br>Audio et image            | \$2.50                    | \$10                        | Octobre 2023           |
| GPT-3.5-turbo | Novembre 2022  | Classique<br>Moins intelligent          | \$3                       | \$6                         | Septembre 2021         |

La liste complète des différentes versions et de leurs caractéristiques est disponible sur la [plateforme développeur](#).

# 02. ChatGPT

## Playground

La page **Playground** qui permet de tester les modèles de génération de texte.

The screenshot shows the ChatGPT Playground interface. On the left, there's a 'SYSTEM' section with the text 'You are a helpful assistant.' Below it, a red box highlights step 1: '1. Définition du Rôle Système que prendra le chatbot'. In the center, there's a 'USER' section with a placeholder 'Enter a user message here.' Below it, a red box highlights step 2: '2. Configuration des premiers échanges d'exemple entre l'**assistant** chatgpt un **utilisateur**'. On the right, there are several configuration options: Mode (set to Chat), Model (set to gpt-3.5-turbo), Temperature (set to 1), Maximum length (set to 256), Stop sequences, Top P (set to 1), Frequency penalty (set to 0), and Presence penalty (set to 0). A note at the bottom states: 'API and Playground requests will not be used to train our models. [Learn more](#)'. Red arrows point from each numbered step to its corresponding configuration option in the interface.

3. Sélection du mode d'interaction :
  - **Chat** : Conversation User/Assistant
  - **Complete** : CompléTION de texte
  - **Edit** : Reformulation de texte
4. Sélection du modèle de langue
5. Sélection de la température : réponses plus ou moins consensuelles.
6. Autres réglages concernant les pénalités et la fréquence des tokens.

## 02. ChatGPT

### Prompt Engineering

L'**ingénierie de requête**, est une technique qui consiste à fournir des instructions détaillées aux modèles de traitement du langage naturel (Natural Language Processing, ou NLP) afin d'améliorer leurs performances. Il existe énormément d'astuces pour améliorer et personnaliser le comportement d'un modèle, pour maximiser l'efficacité il est recommandé de suivre les bonnes pratiques suivantes :

1. **Invoquer un comportement** ou une façon de répondre : Formulez votre prompt de manière précise pour indiquer clairement le comportement attendu de l'IA.
2. **Donner un maximum de contexte** et des exemples de réponse ou de contenu : Fournissez des informations contextuelles et des exemples concrets pour guider l'IA dans sa réponse.
3. **Prévenir les hallucinations** et les textes trop longs : Soyez spécifique dans votre requête pour éviter que l'IA ne génère des informations incorrectes ou excessivement longues.
4. **Demander de poser des questions** sur la compréhension du sujet : Encouragez l'IA à poser des questions de clarification si elle ne comprend pas complètement votre demande.
5. **Effacer et recommencer vos conversations** si les premiers résultats ne sont pas concluants : Si les réponses initiales de l'IA ne sont pas satisfaisantes, n'hésitez pas à reformuler votre prompt ou à recommencer la conversation pour obtenir des résultats plus pertinents.

## 02. ChatGPT

### Implémentation API (1 / 2)

Le package OpenAI est disponible en python (pip) et en javascript (npm). Il est toujours recommandé d'utiliser un environnement virtuel en python.

```
pip install openai
```

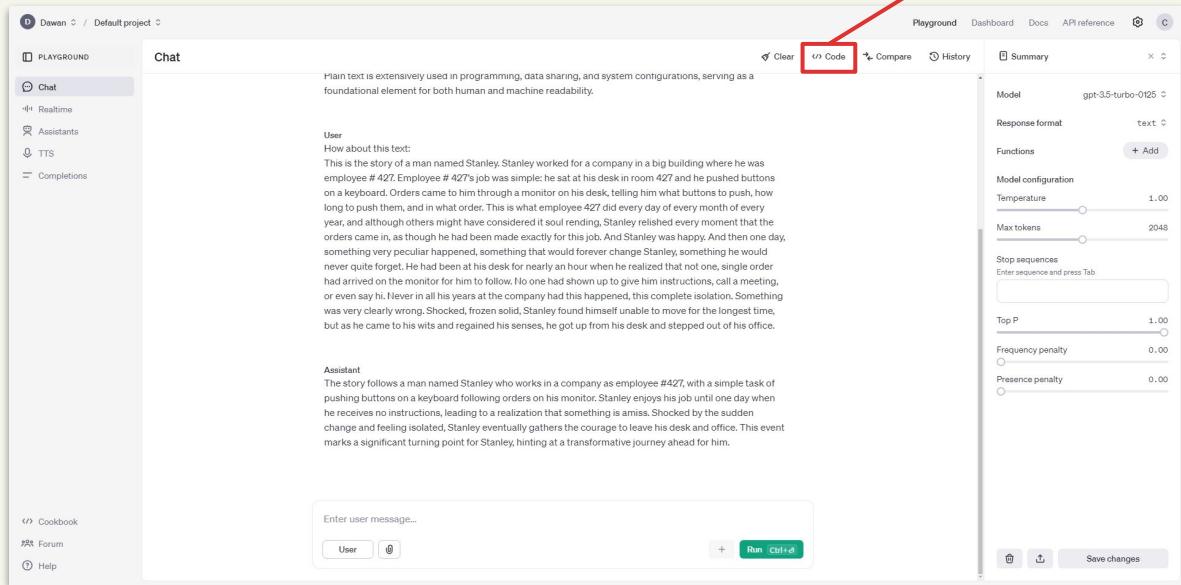
- Import du package openai
- Initialisation du client avec clé d'API
- Envoi d'une requête à l'API demandant une réponse qui sera stockée dans une variable
- Le modèle que l'on souhaite utiliser
- Une représentation objet d'une discussion comme sur le playground
- L'API retourne un objet contenant des informations sur l'opération. Ici on extrait le message de réponse pour l'afficher

```
from openai import OpenAI  
client = OpenAI(api_key="[secret key]")  
  
completion = client.chat.completions.create(  
    model="gpt-4o-mini",  
    messages=[  
        {"role": "user", "content": "write a haiku about ai"}  
    ]  
)  
  
print(completion.choices[0].message.content)
```

# 02. ChatGPT

## Implémentation API (2 / 2)

Heureusement il n'est pas nécessaire de transcrire notre assistant en code nous même!



The screenshot shows the ChatGPT playground interface. On the left, there's a sidebar with 'PLAYGROUND', 'Chat' (selected), 'Runtime', 'Assistants', 'TTS', and 'Completions'. The main area has sections for 'User' (containing a story about Stanley) and 'Assistant' (containing a summary). At the top right, there are tabs for 'Playground', 'Dashboard', 'Docs', 'API reference', and 'Code'. A red box and arrow point to the 'Code' tab. Below it are buttons for 'Clear', 'Compare', 'History', and 'Summary'. To the right of the main area is a sidebar with 'Model' set to 'gpt-3.5-turbo-0125', 'Response format' set to 'text', and various configuration sliders for 'Temperature', 'Max tokens', 'Stop sequences', 'Top P', 'Frequency penalty', and 'Presence penalty'. At the bottom is a message input field with placeholder 'Enter user message...', a 'User' button, and a 'Run Ctrl+R' button.

**View code**

You can use the following code to start integrating your current prompt and settings into your application.

```
POST /v1/completions
from openai import OpenAI
client = OpenAI()
response = client.chat.completions.create(
    model="gpt-3.5-turbo-0125",
    messages=[
        {
            "role": "system",
            "content": [
                {
                    "text": "Your role is to extract the most essential informations from"
                }
            ]
        },
        {
            "role": "user",
            "content": [
                {
                    "text": "Can you help me summarize this article?\nhttps://en.wikipedia.org/wiki/Stanley_Brown_(character)"
                }
            ]
        },
        {
            "role": "assistant",
            "refusal": False,
            "content": [
                {
                    "text": "Certainly! Plain text refers to unformatted text that consists of characters such as letters, numbers, and symbols, typically displayed on a computer screen or printed on paper. It is often contrasted with structured data formats like XML or JSON, which have a defined structure and meaning beyond just the characters themselves."
                }
            ]
        },
        {
            "role": "user",
            "content": [
                {
                    "text": "How about this text:\nThis is the story of a man named Stanley. Stanley worked for a company in a big building where he was employee #427. Employee #427's job was simple; he sat at his desk in room 427 and he pushed buttons on a keyboard. Orders came to him through a monitor on his desk, telling him what buttons to push, how long to push them, and in what order. This is what employee #427 did every day of every month of every year, and although others might have considered it soul rendering, Stanley relished every moment that the orders came in, as though he had been made exactly for this job. And Stanley was happy. And then one day, something very peculiar happened, something that would forever change Stanley, something he would never quite forget. He had been at his desk for nearly an hour when he realized that not one, single order had arrived on the monitor for him to follow. No one had shown up to give him instructions, call a meeting, or even say hi. Never in all his years at the company had this happened, this complete isolation. Something was very clearly wrong. Shocked, frozen solid, Stanley found himself unable to move for the longest time, but as he came to his wits and regained his senses, he got up from his desk and stepped out of his office."
                }
            ]
        }
    ],
    temperature=1.0,
    max_tokens=2048,
    stop_sequences=["<|endoftext|>"],
    top_p=1.0,
    frequency_penalty=0.0,
    presence_penalty=0.0
)
```

[Voici](#) un lien vers le playground au cas où

## 02. ChatGPT

### Intégration 1 / 3

Pour simplifier le développement, nous utilisons [Streamlit](#), un module python permettant de créer des applications webs axées data/ia très rapidement sans connaissances frontend nécessaires

- Import du package streamlit → `import streamlit as st`
- Création d'un élément texte avec comme contenu "Hello World" → `st.write("Hello World")`

```
pip install streamlit
```

Pour lancer l'application il faudra passer par la commande streamlit.  
Il ne restera plus qu'à accéder à la page web

```
streamlit run ./file.py
```

Si jamais votre terminal ne reconnaît pas cette commande malgré l'installation du package, essayer de passer par python avec le flag **-m**

```
python -m streamlit run ./file.py
```

# 02. ChatGPT

## Intégration 2 / 3

Même si une visite de la documentation **Streamlit** est recommandée, les différents éléments mentionnés sur cette page sont plus que suffisant pour commencer.

```
# Input
st.button("Button")
st.checkbox("Checkbox")
st.toggle("Toggle")
st.radio("Radio", ["Option 1", "Option 2"])
st.text_input("Text Input")
st.audio_input("Audio Input")
st.file_uploader("File Uploader")
st.slider("Slider", 0, 100)
```

```
# Media
st.image("image.webp")
st.audio("audio.wav")
```

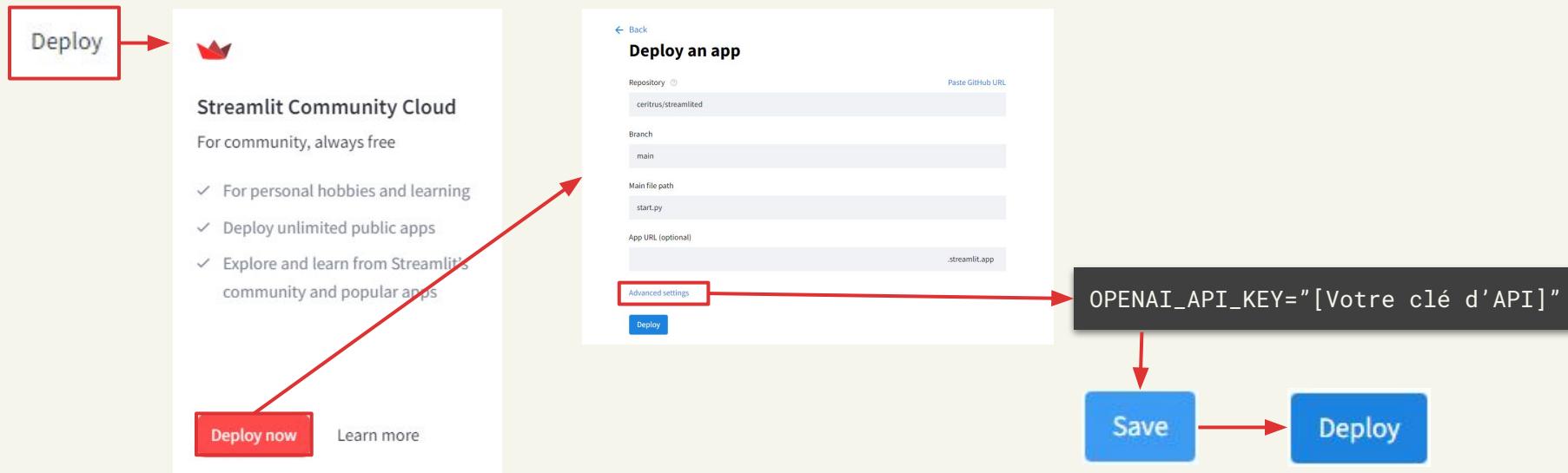
```
# Navigation
pg = st.navigation([
    st.Page("page1.py", title="First page", icon="🔥"),
    st.Page("page2.py", title="Second page", icon="😊"),
])
pg.run()
```

```
# Texte
st.write("Write")
st.text("Text")
st.header("Header")
st.subheader("Subheader")
```

## 02. ChatGPT

### Intégration 3 / 3

Un autre avantage de **Streamlit** est la facilité avec laquelle il est possible de publier vos projets. Assurez-vous que votre projet est publié sur un repo public Github.



# 03 DALL-E

# 3. DALL-E

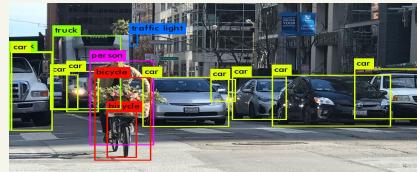
## Plateforme DALL-E

**La vision par ordinateur**, également appelée "computer vision" en anglais, est un domaine de l'informatique qui s'intéresse à la réalisation de tâches visuelles par des ordinateurs. Cela inclut la capture, l'analyse et la compréhension d'images et de vidéos par des ordinateurs.

### Classification



### Détection

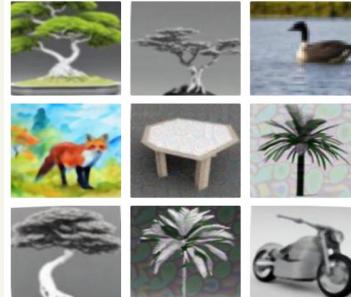


### Identification



### Entraînement via les captcha

Veuillez cliquer sur toutes les images contenant un meuble  
S'il n'y en a aucune, cliquez sur Passer



### Segmentation



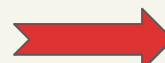
# 3. DALL-E

## Plateforme DALL-E

Entrainement d'un algorithme de débruitage.



Chat



Chat



???

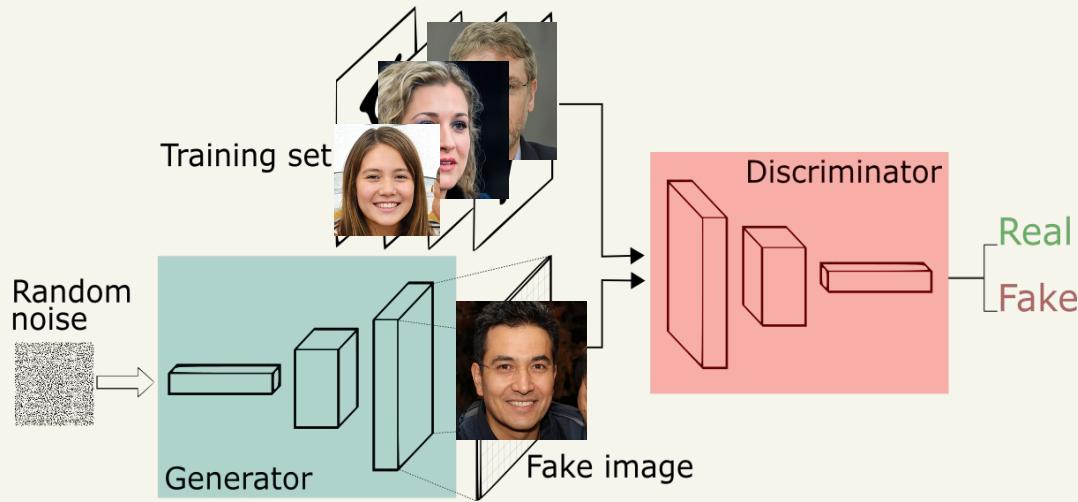


Chat

## 3. DALL-E

### GAN : Generative Adversarial Network

\*Réseaux antagonistes génératifs

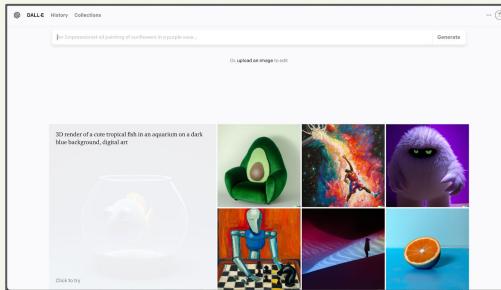


Ressources : [Dall-E : Open IA](#) | [This person does not exist](#) | [Deepfakesweb](#) | [Generative.fm](#) | [Midjourney](#)

## 3. DALL-E

### Plateforme DALL-E

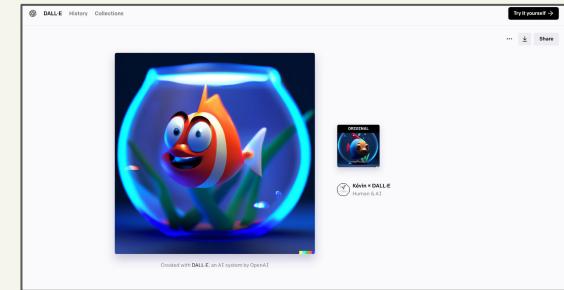
**La plateforme DALL-E** (<https://labs.openai.com/>) offre de nombreuses fonctionnalités de partage et d'édition de contenu.



**Génération d'images uniques basée sur des modèles d'apprentissage profond** : Transformez des concepts en images époustouflantes en utilisant des descriptions textuelles.



**Retouche d'image** : Personnalisez et éditez des images existantes pour les adapter à votre vision artistique.



**Partage de contenu Cratif** : Diffusez vos créations avec le monde entier et inspirez d'autres esprits créatifs.

## 3. DALL-E

### Plateforme DALL-E

Les utilisateurs gratuits et payants qui n'ont jamais eu de crédits (gratuits ou payants) ne peuvent pas acheter de crédits. Les utilisateurs de Labs qui ont déjà eu des crédits (même si vous n'avez actuellement aucun crédit) peuvent toujours utiliser Labs normalement.

DALL-E fonctionne sur un système de crédits pour générer des images à partir de descriptions textuelles. Voici les détails des forfaits disponibles :

**Crédits de base** : Les utilisateurs reçoivent un certain nombre de crédits gratuits chaque mois (souvent 50 crédits pour les utilisateurs standard et plus pour ceux disposant d'abonnements premium comme ChatGPT Plus).

**Achat de crédits supplémentaires** : Si vous dépassez le quota mensuel, vous pouvez acheter des crédits en lots de 115, coûtant environ 15 USD. Chaque génération d'image consomme un crédit (ou plus selon les options choisies, comme les variations ou les retouches).

**Utilisation commerciale** : Les images créées peuvent être utilisées commercialement, que ce soit pour des livres, des conceptions graphiques ou des illustrations.

**DALL-E via ChatGPT Plus** : Avec un abonnement ChatGPT Plus (20 USD/mois), vous pouvez intégrer les capacités de DALL-E directement dans ChatGPT, incluant l'accès à DALL-E 3 et ses nouvelles fonctionnalités telles que l'inpainting et des améliorations dans la génération des détails.

Pour les développeurs et entreprises, OpenAI propose également une **API DALL-E** qui permet une intégration dans des applications ou services à des tarifs spécifiques liés à l'usage commercial.

04

# Whisper & TTS

# 05. Whisper & TTS

## Whisper: introduction

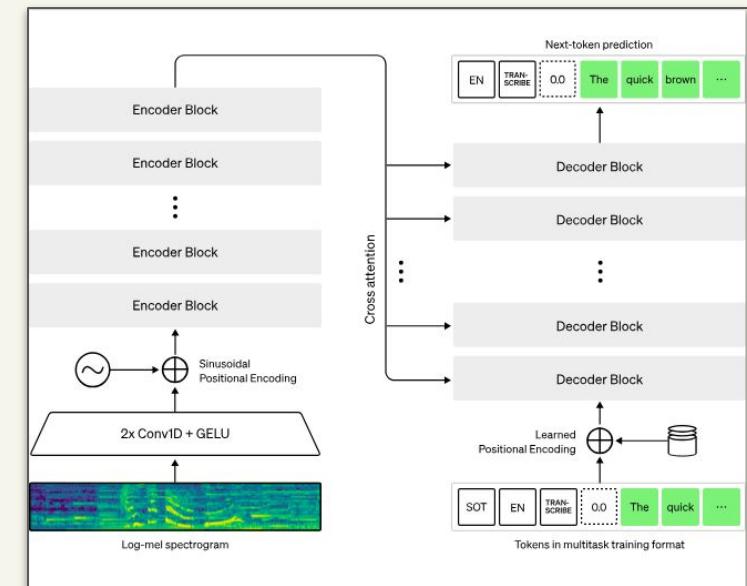
Whisper est un système de reconnaissance vocale automatique (ASR) formé sur 680 000 heures de données supervisées multilingues et multi-tâches collectées sur le Web.

L'API de Whisper propose 2 modes de fonctionnement, transcriptions et traduction. Ils peuvent être utilisés pour :

- Transcrire l'audio en texte dans sa langue originale.
- Traduire l'audio vers l'anglais (toujours en texte).

Les téléchargements de fichiers sont actuellement limités à 25 Mo et les types de fichiers d'entrée suivants sont pris en charge mp3, mp4, mpeg, mpg, m4a, wav et webm.

Voir aussi la solution de [ElevenLab](#), de [Microsoft](#) ou de [MetaAI](#).



La documentation de Whisper est disponible [ici](#)

## 02. Whisper & TTS

### Whisper: implémentation

Le package OpenAI est disponible en python (pip) et en javascript (npm). Il est toujours recommandé d'utiliser un environnement virtuel en python.

```
pip install openai
```

- Import du package openai → `from openai import OpenAI`
- Initialisation du client avec clé d'API → `client = OpenAI(api_key="[secret key]")`
- Ouverture du fichier contenant votre audio → `audio_file = open("/path/to/file/audio.mp3", "rb")`
- Envoi d'une requête à l'API demandant une réponse qui sera stockée dans une variable (.translations pour une traduction) → `transcription = client.audio.transcriptions.create(  
 model="whisper-2",  
 file=audio_file  
)`
- Le modèle que l'on souhaite utiliser → `model="whisper-2"`
- Le fichier contenant votre audio → `file=audio_file`
- L'API retourne un objet contenant des informations sur l'opération. Ici on extrait le message de réponse pour l'afficher (par défaut au format json) → `print(transcription.text)`

## 02. Whisper & TTS

### TTS

Un TTS (text to speech) est un système permettant de générer une représentation audio d'un texte, simulant la parole humaine. Whisper est également disponible hors ligne.

- Import du package pathlib
  - Import du package openai
  - Initialisation du client avec clé d'API
  - Envoi d'une requête à l'API demandant une réponse qui sera stockée dans une variable (.translations pour une traduction)
  - Le modèle que l'on souhaite utiliser
  - La voix que l'on souhaite utiliser
  - Le texte que vous souhaitez convertir en audio
  - Création d'un objet représentant le fichier de sortie
  - La réponse est une représentation binaire de votre audio. Ici on l'écrit dans un fichier pour pouvoir l'écouter
- ```
from pathlib import Path
from openai import OpenAI

client = OpenAI(api_key="[secret key]")

response = client.audio.speech.create(
    model="tts-1",
    voice="alloy",
    input="Bonjour, je suis votre assistant vocal"
)

file_path = Path(__file__).parent / "filename.mp3"

response.stream_to_file(file_path)
```

Voir aussi la solution de [ElevenLab](#) ou de [Google](#)

05

# Personnalisation & Optimisation

## Fine-tuning: introduction 1 / 2

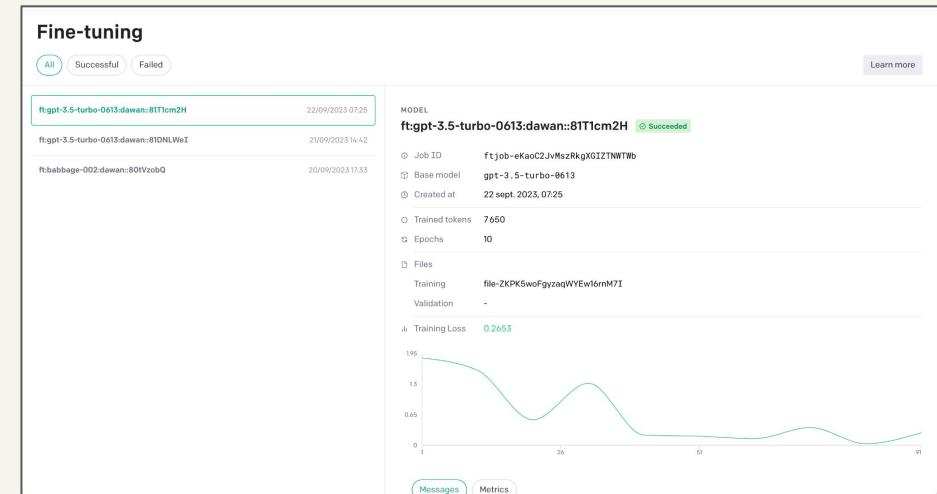
# 05. Personnalisation & Optimisation

Le fine-tuning permet d'entraîner un large modèle de langue tels que GPT3.5 sur des données spécifiques. Le réglage fin est disponible à partir de la version 0.28.0 de l'[API d'OpenAI](#).

Une fois qu'un modèle a été affiné, vous n'aurez plus besoin de fournir autant d'exemples manuellement. Cela permet de réduire les coûts et de permettre des requêtes à faible latence.

Le fine-tuning permet de tirer le meilleur des modèles disponibles via l'API en fournissant:

- Des résultats de meilleure qualité et plus précis qu'avec les prompts
- Une capacité à s'entraîner sur plus d'exemples que ne peut en contenir un prompt
- Une économies de jetons grâce à prompts plus courts
- Des requêtes plus rapides



## Fine-tuning: introduction 2 / 2

# 05. Personnalisation & Optimisation

Même si le fine-tuning peut apparaître comme une solution parfaite, cette dernière peut demander beaucoup de temps et d'efforts en amont pour préparer les données d'entraînement.

Plus de 90% du temps il sera possible d'obtenir le comportement souhaité en travaillant un peu plus sur les prompts systèmes et utilisateur de l'assistant. OpenAI recommande de toujours essayer différentes approches de [prompt engineering](#) avant de décider d'utiliser le fine-tuning.

Quelques cas d'utilisation courants dans lesquels un réglage fin peut améliorer les résultats :

- Définir le style, le ton, le format ou d'autres aspects qualitatifs
- Améliorer la fiabilité pour produire le résultat souhaité
- Correction des échecs de gestion de requêtes complexes
- Gérer de nombreux cas extrêmes de manière spécifique
- Exécuter une nouvelle compétence ou une nouvelle tâche difficile à articuler dans un prompt

## Fine-tuning: modèles & coûts

# 05. Personnalisation & Optimisation

La tarification d'un modèle affiné est séparé en 2 parties, l'entraînement et l'utilisation.

| Modèle                | Prix / 1M token<br>(entraînement)* | Prix / 1M token<br>(input) | Prix / 1M token<br>(output) |
|-----------------------|------------------------------------|----------------------------|-----------------------------|
| gpt-4o-2024-08-06     | \$25                               | \$3.75                     | \$15                        |
| gpt-4o-mini-2024-07-1 | \$3                                | \$0.30                     | \$1.20                      |
| gpt-3.5-turbo         | \$8                                | \$6                        | \$6                         |
| davinci-002           | \$6                                | \$12                       | \$12                        |
| babbage-002           | \$0.40                             | \$1.60                     | \$1.60                      |

Non seulement l'entraînement peut engendrer un coût initial conséquent, mais il est également important de noter que le coût d'utilisation par token est plus élevé que pour les équivalents standards.

En effet, étant donné qu'OpenAI doit stocker et garder accessible un modèle personnel ils peuvent se permettre de charger plus.

Si vous optez pour le fine-tuning dans un but de faire des économies, il faudra faire bien attention à ce que cela vous fasse économiser suffisamment de token lors de l'utilisation à la longue.

Token dans les données d'entraînements \* epochs (itérations) = token d'entraînements  
Le nombre d'epochs par défaut est de 4

## Fine-tuning: configuration

# 05. Personnalisation & Optimisation

Grossièrement, la mise en place du fine-tuning comprend 4 étapes:

1. Préparer et uploader les données d'entraînement
2. Entraîner le nouveau modèle affiné
3. Évaluer les résultats et retourner à l'étape 1 si nécessaire
4. Utiliser votre modèle affiné grâce

La création de modèle affiné peut se faire de 2 manières, via l'[interface dédiée](#) disponible sur le tableau de bord de la plateforme développeur ou via code.

Étant un processus qu'il n'est pas nécessaire de répéter sauf problème nous opterons ici pour l'interface en ligne.

Lors de la création d'un modèle les champs à préciser sont les suivants:

- **Modèle de base:** sur quel modèle de langage souhaitez vous vous baser, le coût dépendra de cela
- **Données d'entraînements:** il faudra uploader un fichier .jsonl [formaté](#) contenant vos données
- **Données de validation (optionel):** des données spécifiques sur lesquelles tester le modèle, par défaut le système prendra des entrées aléatoires parmis données d'entrainements pour remplir ce rôle
- **Suffixe:** le nom de votre modèle qui sera utilisé en plus du nom de base pour le référencer plus tard
- **Seed (optionel):** un nombre qui influencera l'aléatoire de l'entraînement, 2 entraînements avec les mêmes données et la même seed produiront le même résultat, par défaut une seed aléatoire est générée
- **Hyperparamètres (optionnels):** des paramètres qui influencent la vitesse d'apprentissage du modèle ainsi que sa relation aux données, OpenAI essaye de fournir des paramètres optimaux par défaut

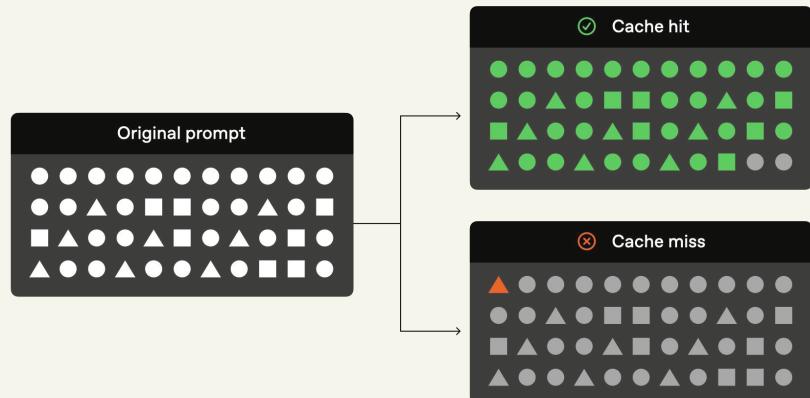
# 05. Personnalisation & Optimisation

## Optimisation: Prompt caching

Les modèles basés sur des prompts contiennent régulièrement du contenu répétitif, tel que les prompts systèmes ou des instructions communes. OpenAI redirige automatiquement les appels d'API vers des serveurs ayant récemment géré des prompts similaires, permettant de rendre ces requêtes plus rapides et moins coûteuses que si le prompt avait été géré de zéro. Ce mode de gestion peut réduire la latence des requêtes jusqu'à 80% et le coût de 50% (particulièrement pour les prompts long). Le prompt caching ou mise en mémoire de prompt fonctionne automatiquement sur toutes les requêtes d'API (sans modification du code) et n'a pas de coût supplémentaire associé.

Conseils pour en profiter:

- Placez les portions fixes ou répétées au début de votre prompt et le contenu dynamique à la fin.
- Jetez un oeil aux statistiques de taux de correspondance en mémoire, latence et pourcentage de tokens mis en mémoire pour optimiser votre stratégie de prompt caching.
- Pour augmenter votre taux de correspondance, prioriser les requêtes plus longues et faites vos appels d'API pendant les heures creuses, malheureusement pendant les heures de grande activité il arrivent plus souvent que la phase de prompt caching soit partiellement ou totalement ignorée.
- Les prompts qui n'ont pas été utilisés récemment sont automatiquement retirées de la mémoire. Pour éviter les omissions, maintenez un flux de requête constant avec les même préfixes.



## Optimisation: Batch API

# 05. Personnalisation & Optimisation

Même si la majorité des utilisations de la plateforme OpenAI requièrent que vos requêtes soient envoyées de manière synchrone, il existe pleins de cas où vous n'aurez pas besoin d'une réponse immédiate ou que les limites de flux vous empêchent d'exécuter un grand nombre de requêtes.

La gestion en groupe est souvent utiles pour des cas tels que:

- Évaluer des données
- Classifier des grandes quantités de données
- Remplir des archives de contenu.

L'API de groupe offre une collection d'endpoints directs qui vous permettent de préparer une groupe de requêtes en un seul fichier, lancer un gestionnaire de batch pour exécuter les requêtes, vérifier l'état d'un groupe pendant que les requêtes s'exécutent et enfin récupérer les résultats du groupe une fois complet.

Comparé à leurs équivalents standards, l'API de groupe a:

- Des coûts réduits: 50% de réduction comparé à l'API synchrone.
- De plus hautes limites de flux: Bien plus de marge que l'API synchrone.
- Des exécutions rapides: Chaque groupe sera complet sous 24h (souvent moins).