

Interrogez vos documents avec des grands modèles de langage : découvrez la méthode RAG

ANF TDM & IA 2025

Pierre Poulain

`pierre.poulain@u-paris.fr`

Laboratoire de Biochimie Théorique



Loved by +5 million people 

Say Goodbye to Endless Reading, Get Instant Answers from Any PDF

Ask any question, summarize, extract key insights, and ace
your exams faster than ever.

[Start for free](#)



[Try Our GPT](#)

+ Upload PDF

📁 New Folder



Sign in for free to save
your chat history

Sign in



#1 PDF Chat AI
Original



Q's answered every day
1,000,000+



Gen AI apps of 2024
Top 50



Chat with any PDF

Join millions of students, researchers and professionals to instantly
answer questions and understand research with AI

DRAG + DROP YOUR
PDF FILE HERE



Click to upload, or drag PDF here



Upload PDF

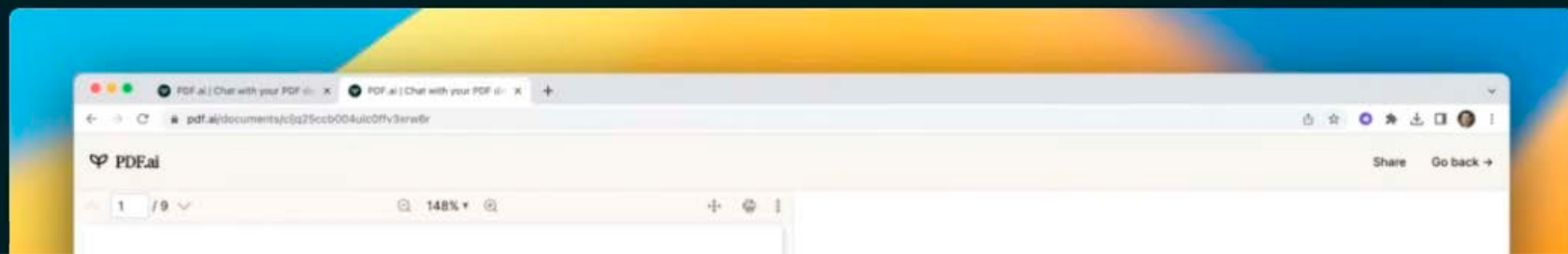


Chat with any PDF document

From legal agreements to financial reports, PDF.ai brings your documents to life. You can ask questions, get summaries, find information, and more.

[Get started for free](#)[Try the demo →](#)

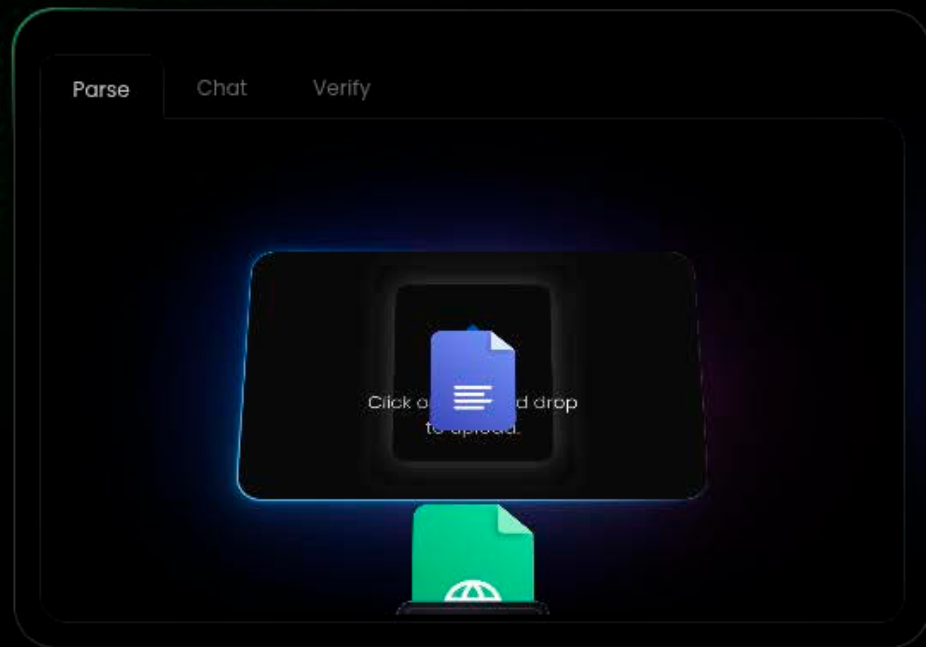
Loved by millions of happy users!





Where **DOCX** Speaks Accurately

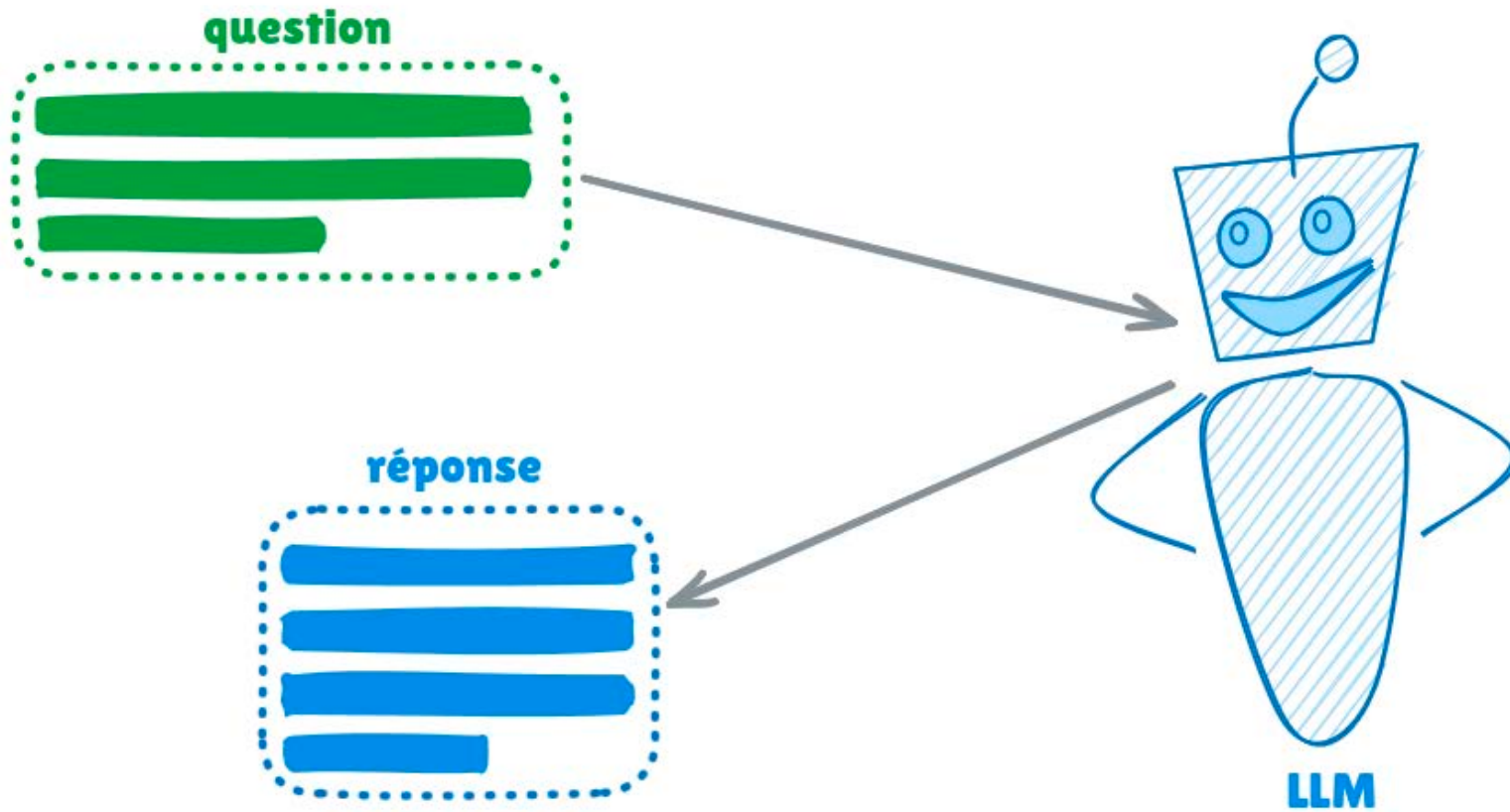
Answers You Can Trust. Sources You Can See.



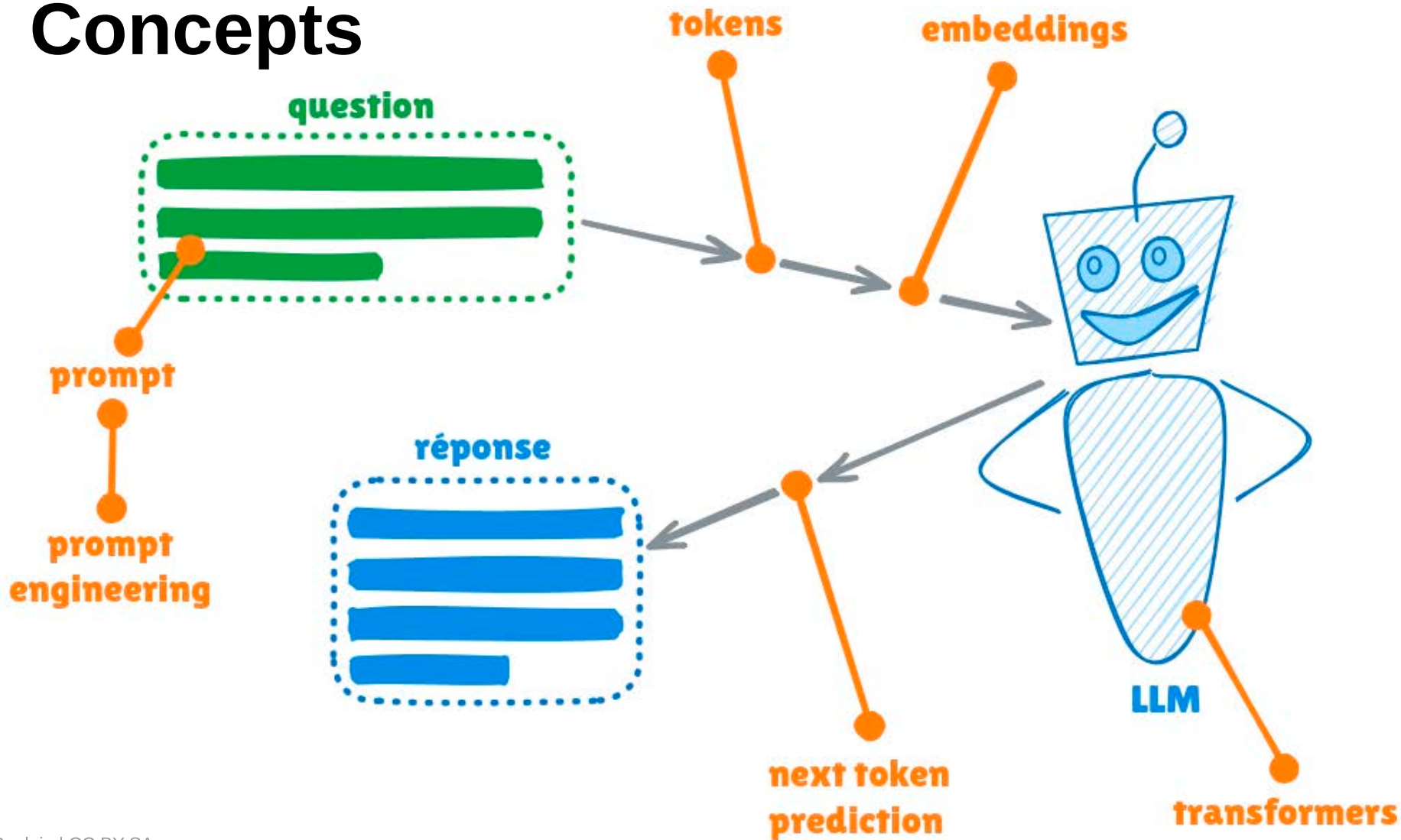
Au programme

1. La méthode RAG
2. Construction d'un RAG
3. RAG vs fine-tuning
4. Cas d'usage pédagogique

Concepts



Concepts



Transformers

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez*[†]
University of Toronto
aidan@cs.toronto.edu

Lukasz Kaiser*
Google Brain
lukaszkaizer@google.com

Illia Polosukhin*[‡]
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

*Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

[†]Work performed while at Google Brain.

[‡]Work performed while at Google Research.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Source : arXiv, 2017

LLM Visualization

<

Chapter: Overview

>

How to predict text tokens

LLM

pos embed

tok embed

transformer i

layer norm

multi-head, causal self-attention

layer norm

feed forward

layer norm

linear

softmax

Table of Contents

Intro

Introduction

Preliminaries

Components

Embedding

Layer Norm

Self Attention

Projection

MLP

Transformer

Softmax

Output

GPT-2 (small)

nano-gpt

GPT-2 (XL)

GPT-3

Q Bias

K Bias

V Bias

Q Weights

K Weights

V Weights

Attn Matrix Softmax

Attention

Projection Bias

Projection Weights

V Out

MLP Bias

Welcome to the walkthrough of the GPT large language model! Here we'll explore the model *nano-gpt*, with a mere 85,000 parameters.

Its goal is a simple one: take a sequence of six letters:

CBABBC

and sort them in alphabetical order, i.e. to "ABBBCC".

Press Space to continue

We call each of these letters a *token*, and the set of the model's different tokens make up its vocabulary:

token	A	B	C
Index	0	1	2

Source : LLM Visualization

Prompt engineering

Méthodes qui optimisent la réponse d'un modèle de langage

Prompt Engineer 20 ans d'expérience ⋮
Open Prompt IA
Télétravail à 75018 Paris 18e

 CDI

 Heures supplémentaires +1

Français non requis

 Postuler directement

 Urgent

◦ Nous recherchons un ingénieur **prompt engineering** talentueux et passionné pour rejoindre notre équipe en pleine croissance.

Dernière activité : il y a 18 jours · plus...

Source : X/Twitter, 2023

- Chain-of-Thought (CoT)
- Zero-shot
- Few-shot
- ...
- RAG

→ <https://www.promptingguide.ai/fr/>

Tokens

GPT-4o & GPT-4o mini

GPT-3.5 & GPT-4

GPT-3 (Legacy)

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques



Clear

Show example

Tokens

18

Characters

90

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

Text

Token IDs

Tokens

GPT-4o & GPT-4o mini

GPT-3.5 & GPT-4

GPT-3 (Legacy)

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques



Clear

Show example

Tokens

18

Characters

90

```
[4579, 24975, 272, 95645, 7756, 2894, 686, 59199, 198, 35418, 70240, 731, 129412, 11, 146156, 859, 66702, 65641]
```

Text

Token IDs

Tokens

GPT-4o & GPT-4o mini

GPT-3.5 & GPT-4

GPT-3 (Legacy)

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

Clear

Show example

Tokens

Characters

18

90

59199

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

35418

70240

Text

Token IDs

Tokens

~ 200k tokens

GPT-4o & GPT-4o mini

GPT-3.5 & GPT-4

GPT-3 (Legacy)

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

~ 100k tokens

~ 50k tokens

Clear

Show example

Tokens

Characters

18

90

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

Text

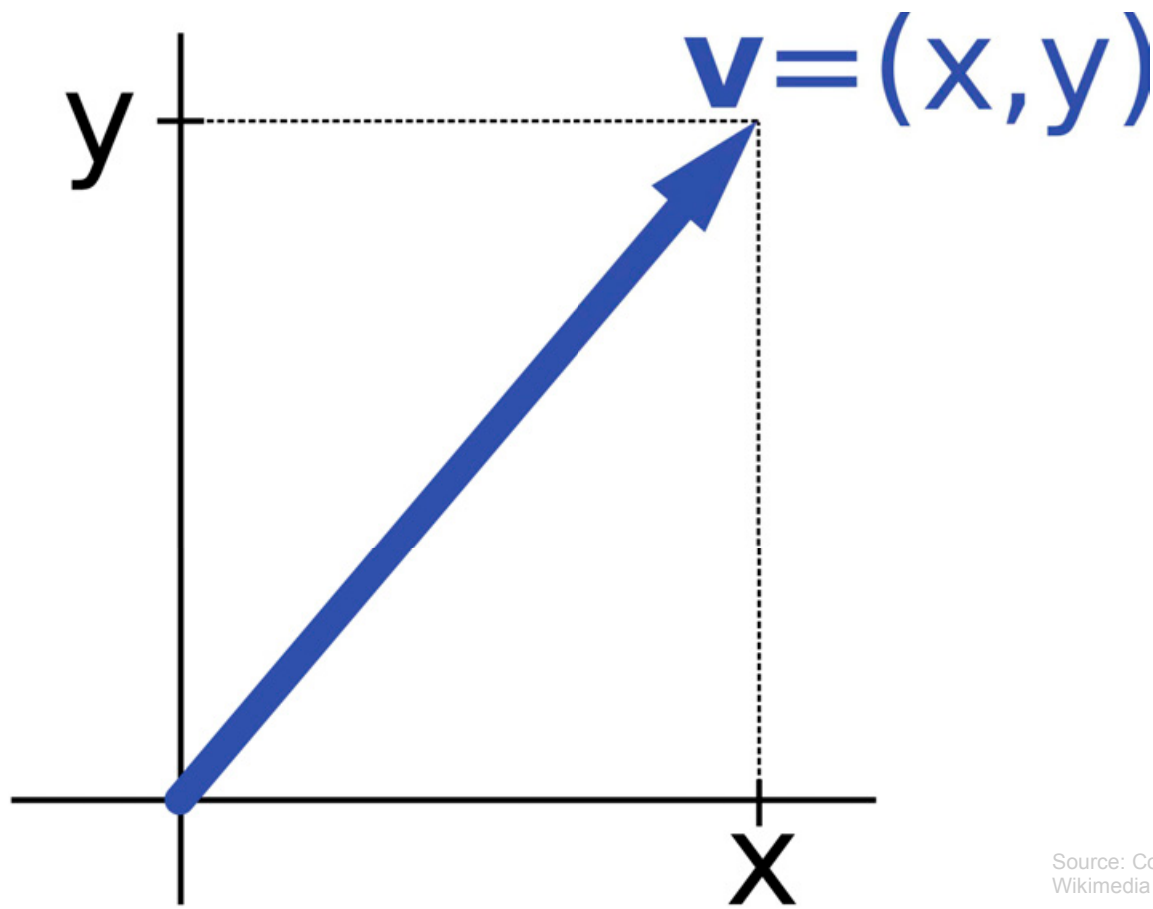
Token IDs

Embeddings = plongement lexical

1 token ► 1 embedding ► 1 vecteur

Embeddings = plongement lexical

1 token ► 1 embedding ► 1 vecteur (2 coordonnées)



Embeddings = plongement lexical

« IA » ► 1 token (n° 59199) ► 1 embedding ► 1 vecteur (3 072 coord.)

[0.009801761247217655, -0.023482611402869225, 0.012982317246496677, 0.002463444834575057, 0.0018708062125369906, 0.031240791082382202, -0.017597096040844917, 0.013331583701074123, 0.0337376743555069, 0.013212684541940689, -0.01622975431382656, -0.010589469224214554, 0.0069110216572880745, 0.008753960952162743, -0.009831485338509083, 0.016645902767777443, -0.038047775626182556, 0.013517363928258419, -0.025667386129498482, -0.01796865649521351, 0.024701328948140144, -0.044141363352537155, -0.005004917271435261, 0.0453898087143898, 0.01740388572216034, 0.03837474808096886, -0.0112657081335783, -0.010953597724437714, -0.03760190308094025, 0.011897360906004906, 0.02025746740400791, 0.04464668780565262, -0.005097807385027409, -0.0106935054063797, -0.03376740217208862, 0.020391229540109634, -0.00947478786110878, 0.0029278951697051525, 0.02025746740400791, 0.0022609445732086897, 0.022873252630233765, -0.010307082906365395, -0.013428189791738987, 0.01688370108604431, 0.024047384038567543, 0.04256594926118851, 0.013220115564763546, -0.003983126487582922, 0.0026585140731185675, -0.024939127266407013, -0.015709569677710533, 0.04176338016986847, -0.05107352389550209, 0.005944964961591024, 0.007553820963948965, 0.02483696987032988758, 0.004243218805640936, 0.08275391161441803, -0.012410114519298077, -0.0031508314423263073, -0.00858304277062416, 0.005996983498334885, 0.016200030222535133, 0.018533429130911827, 0.015308285132050514, 0.019588658586144447, 0.002818285021930933, 0.011154240928590298, 0.0023761282209306955, 0.04874871298670769, -0.009571393951773643, -0.007156251464039087, -0.048451464623212814, 0.029100604355335236, -0.004376980476081371, -0.014134153723716736, -0.0079736839979887, 0.04206062853336334, 0.0009400476119481027, -0.013732869178056717, 0.03204336017370224, 0.0006697374628856778, 0.009259282611310482, 0.03525364398956299, -0.01307149138301611, -0.0036561533434046904, 0.024091970175504684, -0.08536969870328903, 0.0006841354188509285, -0.005268725100904703, 0.005629138555377722, 0.039177317172288895, -0.010158458724617958, 0.03379712626338005, -0.018518565222620964, 0.059925246983766556, 0.0026547983288764954, 0.027465738356113434, 0.002106746891513467, 0.016318928450345993, -0.04500338435173035, 0.008352675475180149, 0.016541864722967148, 0.0032604418229311705, -0.012677637860178947, 0.00966799911115932, 0.0016097850166261196, 0.04925403371453285, -0.0014305071672424674, -0.013896355405449867, -0.02366095967590809, -0.028520969673991203, 0.00583349805280447, -0.009244420565664768, 0.005372762214392424, -0.07710619270801544, -0.02529582567512989, 0.009229558520019054, 0.0082783638430643, -0.038315299898386, 0.03786942735314369, -0.02476077899336815, -0.022055819630622864, -0.04307126998901367, 0.032102812081575394, 0.024909403175115585, -0.034510523080825806, -0.03338097780942917, 0.0013979956274852157, -0.029709963127970695, -0.044468339532613754, -0.003819640027359128, 0.0014054268831387162, 0.03251895681023598, 0.019202236086130142, -0.022516554221510887, -0.016824251040816307, -0.030854368582367897, 0.01740388572216034, -0.000582885229960084, 0.014988742768764496, 0.034094374626874924, -0.01165213156491518, 0.01691342517733574, 0.032102812081575394, -0.035283368080854416, -0.00994295347481966, 0.024344632402062416, -0.006520883645862341, -0.0025173211470246315, 0.0028312895447015762, -0.01256616972386837, 6.29910864518024e-05, -0.012751949951052666, -0.05317771062254906, 0.0279116109013055743, 0.030854368582367897, 0.01582846978166542, -0.008374969474971294, -0.019425172358751297, -0.03801805153489113, 0.008122308179736137, -0.010069284588098526, 0.0007733099046163261, -0.01716608554124832, -0.017344435676932335, 0.04696522280573845, 0.013101216405630112, 0.003968263976275921, 0.030527394264936447, 0.02663344331085682, 0.002337114419788122, -0.019335998222231865, 0.013153234496712685, -0.025667386129498482, 0.013309290632605553, 0.01510764192789793, -0.00532817468047142, 0.02143159881234169, 0.020361505448818207, -0.02175857126712799, 0.02204095758497715, 0.02265031635761261, -0.020881690084934235, -0.03581841289997101, 0.03736410290002823, 0.033618777990341187, 0.016571590676903725, -0.00994295347481966, -0.0474705435335363614, -0.01874150149524212, -0.04009879007935524, 0.037274930626153946, 0.02843179553747177, 0.023289401084184647, -0.01667562685906887, -0.035967037081718445, -0.01732957363128662, -0.02675234153866768, -0.04018796235322952, -0.005012348759919405, 0.0034982403740286827, 0.003113675396889448, 0.001980416476726532, -0.05623937025666237, 0.019648108631372452, -0.04640045017004013, 0.002782986732199788, 0.015189385041594505, 0.0013357592979446054, 0.03106244094668919, 0.019499484446625015, -0.003537254175171256, -0.0016330075450241566, 0.0302893594666958, -0.03679933398962021, -0.08382400870323181, -0.006524599157273769, -0.022694043571949, 0.017820032313466072, 0.009155246429145336, 0.006104735657572746, 0.03144886346712494, 0.03468887135386467, -0.026529405266046524, 0.00937075074762106, 0.007356894202530384, 0.04969990625977516, -0.04194172844290733, 0.018161868676543236, 0.007052214816212654, -0.014082135632634163, 0.006572901736944914, 0.0028591565787792206, -0.04327934607863426, -0.005101522896438837, 0.030393633991479874, 0.007052214816212654, 0.0060044145211577415, 0.03257840871810913, -0.018696915358304977, 0.03837474808096886, -0.016006818041205406, -0.00823776316046715, 0.026202432811260223, 0.012306077405810356, -0.025726836174726486, 0.0022553710732609034, -0.008516162633895874, -0.008077721111476421, 0.000801641377620399, 0.0006650929572060704, 0.018652327358722687, 0.016363516449928284, 0.036056213080883026, -0.020599303767085075, -0.0010923873633146286, 0.022709766402840614, -0.0437251862387657, -0.018369941040873528, -0.029962623491883278, 0.0064314265106618404, -0.0357886888867958, 0.03207303262848854, -0.00943763181567192, -0.02700500376522541, -0.02537013776600361, -0.020034531131386757, 0.016735076904296875, 0.002448582323268056, 0.02639564499258995, 0.023943345993757248, 0.003579983487725258, 0.0007918879273347557, -0.01944003626704216, 0.010203045792877674, -0.03498611971735954, 0.01403011754155159, 0.021654535084962845, 0.012380389496684074, 0.01810241863131523, 0.01169102974236012, -0.0017964941216632724, -0.004614779260009527, -0.004113172646611929, -0.027183352038264275, -0.03138941526412964, -0.026975277811288834, 0.02025746740400791, 0.018028106540441513, -0.001907962141558528, -0.00947478786110878, -0.013561950996518135, -0.013509932905435562, -0.039266493171453476, -0.03620483726263046, 0.019573796540498734, 0.001202926505357027, 0.01028478890657425, 0.021372148767113686, -0.03299455717206001, -0.03956374153494835, -0.010076715610921383, 0.0013227546587586403,

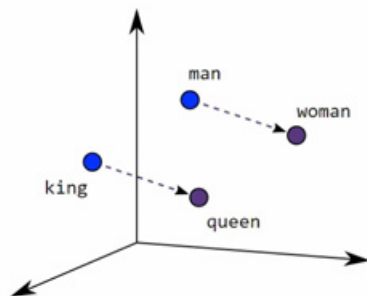
Embeddings = plongement lexical

1 token ► 1 embedding ► 1 vecteur de grande dimension

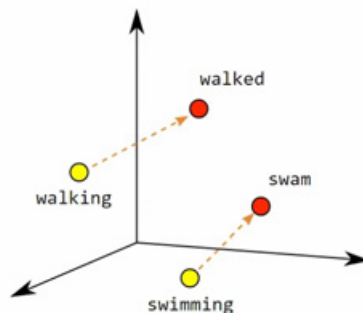
1. Tokens avec des sens « proches » → vecteurs proches

2. Opérations sémantiques → opérations vectorielles

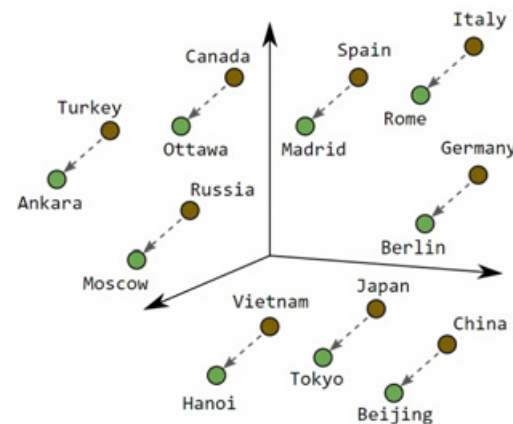
King - Man + Woman $\approx$ Queen



Male-Female

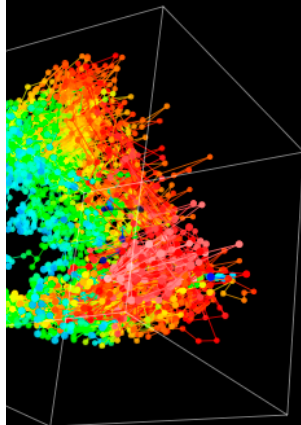


Verb Tense



Country-Capital

Next token prediction



Intelligence

is

the ability to

learn, acquire, and apply knowledge and skills, in order to use information effectively in order to work with ideas and

a

defined as

is generally fundamentally typically kthe often not

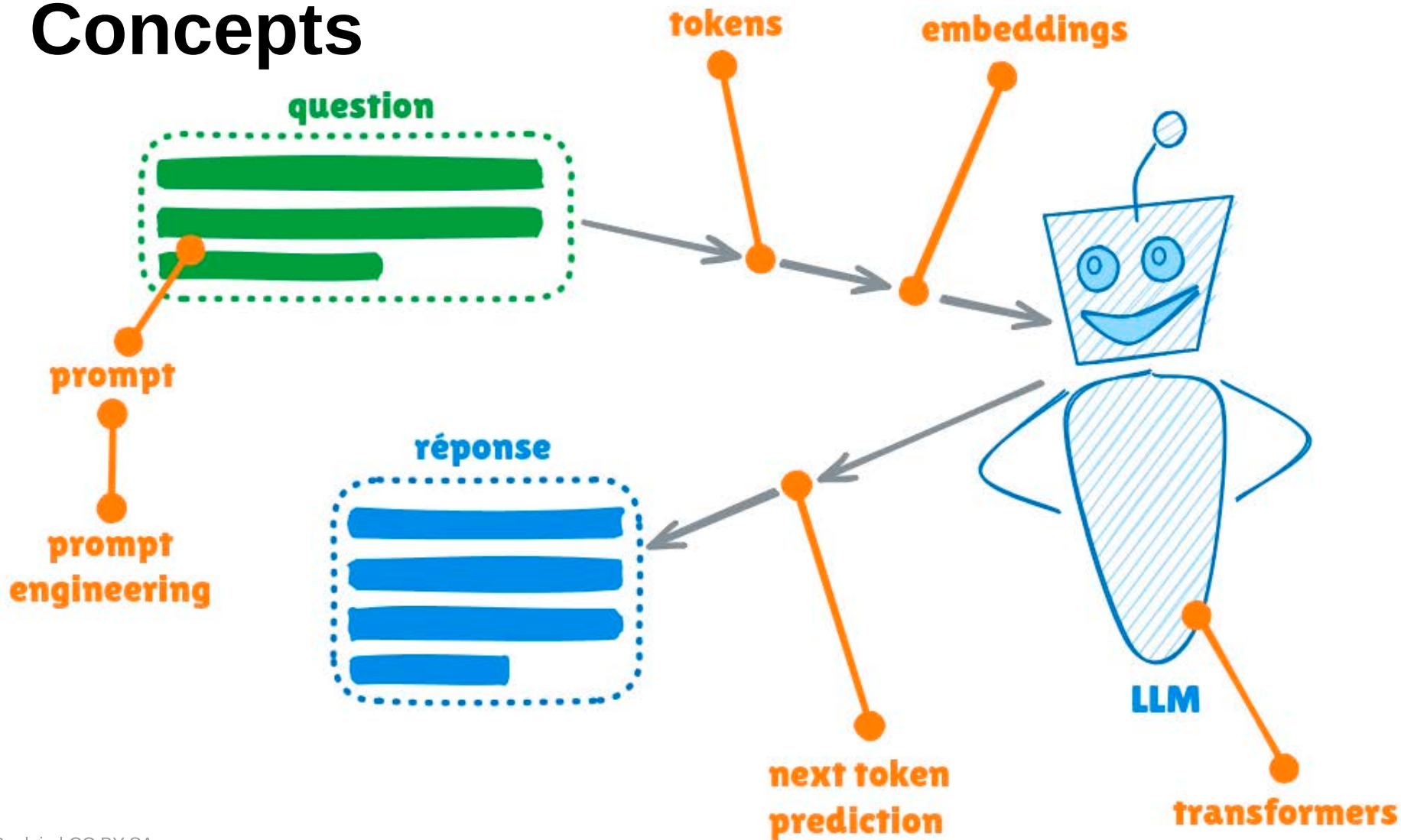
the ability to acquire and apply knowledge and skills, in order to use information effectively in order to work with ideas and

the ability to acquire and apply knowledge and skills, in order to use information effectively in order to work with ideas and

the ability to acquire and apply knowledge and skills, in order to use information effectively in order to work with ideas and

Source : <https://moebio.com/mind/>

Concepts



Au programme

1. La méthode RAG
2. Construction d'un RAG
3. RAG vs fine-tuning
4. Cas d'usage pédagogique

Retrieval-Augmented Generation (RAG)

génération augmentée par récupération

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis¹, Ethan Perez²,

Aleksandra Piktus¹, Fabio Petroni¹, Vladimir Karpukhin¹, Naman Goyal¹, Heinrich Küttler¹,

Mike Lewis¹, Wen-tau Yih¹, Tim Rocktäschel¹, Sebastian Riedel¹, Douwe Kiela¹

¹Facebook AI Research; ²University College London; ^{*}New York University;
plewis@fb.com

Abstract

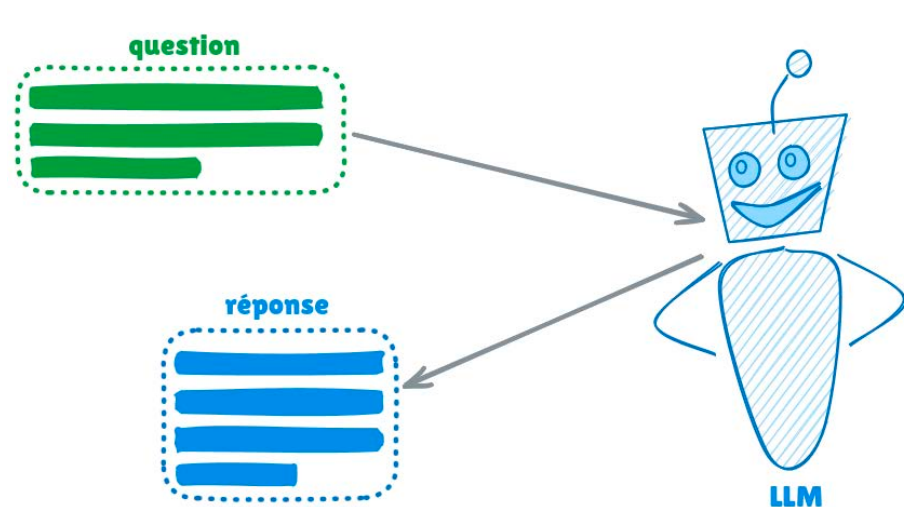
Large pre-trained language models have been shown to store factual knowledge in their parameters, and achieve state-of-the-art results when fine-tuned on downstream NLP tasks. However, their ability to access and precisely manipulate knowledge is still limited, and hence on knowledge-intensive tasks, their performance lags behind task-specific architectures. Additionally, providing provenance for their decisions and updating their world knowledge remain open research problems. Pre-trained models with a differentiable access mechanism to explicit non-parametric memory can overcome this issue, but have so far been only investigated for extractive downstream tasks. We explore a general-purpose fine-tuning recipe for retrieval-augmented generation (RAG) — models which combine pre-trained parametric and non-parametric memory for language generation. We introduce RAG models where the parametric memory is a pre-trained seq2seq model and the non-parametric memory is a dense vector index of Wikipedia, accessed with a pre-trained neural retriever. We compare two RAG formulations, one which conditions on the same retrieved passages across the whole generated sequence, and another which can use different passages per token. We fine-tune and evaluate our models on a wide range of knowledge-intensive NLP tasks and set the state of the art on three open domain QA tasks, outperforming parametric seq2seq models and task-specific retrieve-and-extract architectures. For language generation tasks, we find that RAG models generate more specific, diverse and factual language than a state-of-the-art parametric-only seq2seq baseline.

1 Introduction

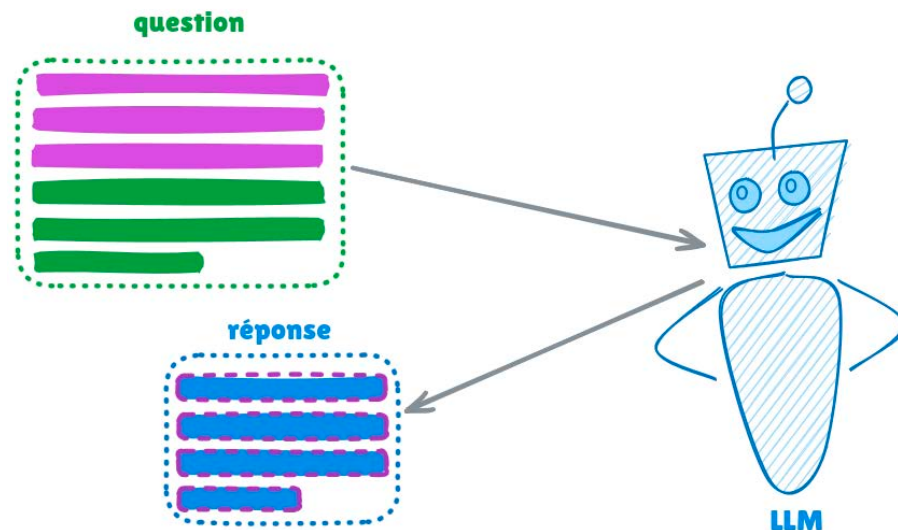
Pre-trained neural language models have been shown to learn a substantial amount of in-depth knowledge from data [1]. They can do so without any access to an external memory, as a parameterized implicit knowledge base [2, 3]. While this development is exciting, such models do have downsides: They cannot easily expand or revise their memory, can't straightforwardly provide insight into their predictions, and may produce "hallucinations" [4]. Hybrid models that combine parametric memory with non-parametric (i.e., retrieval-based) memories [5, 6, 7, 8] can address some of these issues because knowledge can be directly revised and expanded, and accessed knowledge can be inspected and interpreted. REALM [9] and ORQA [10], two recently introduced models that combine masked language models [11] with a differentiable retriever, have shown promising results,

Retrieval-Augmented Generation (RAG)

génération augmentée par récupération



Usage habituel



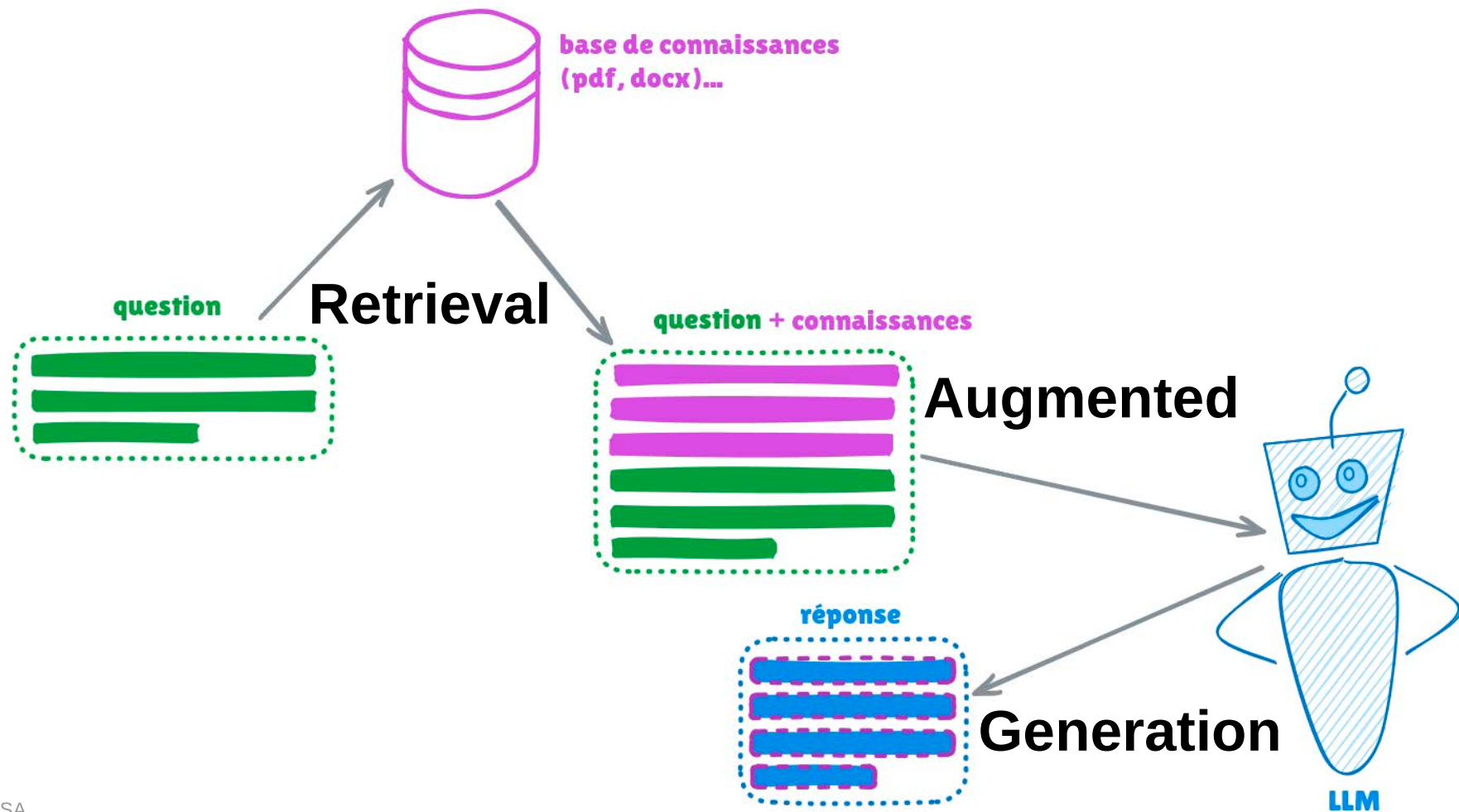
RAG

Pourquoi le RAG ?

Les modèles de langages :

- ont une connaissance limitée dans le temps (phase d'apprentissage) ;
- n'ont pas accès à certaines informations (confidentielles).

Base de connaissances



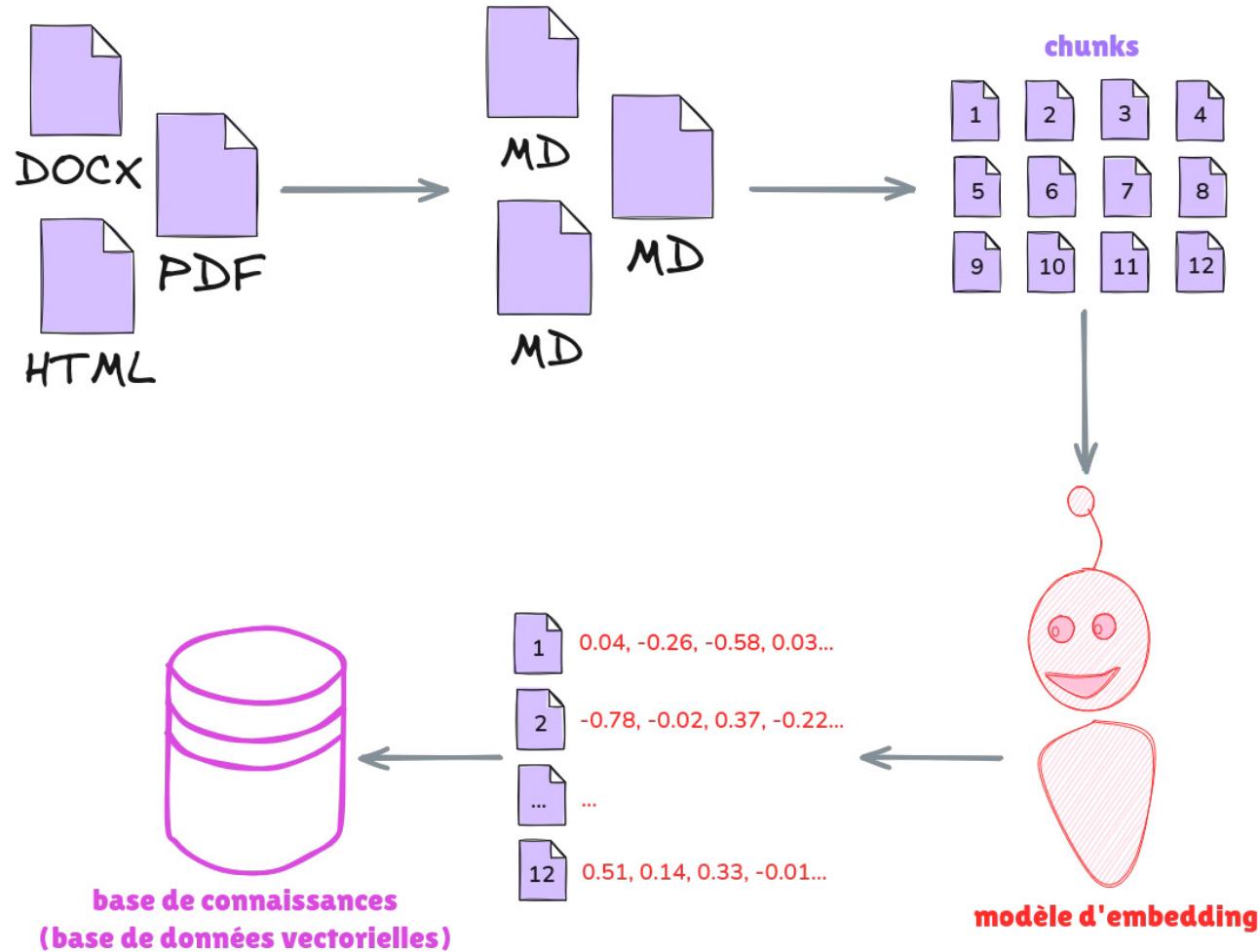
Au programme

1. La méthode RAG
2. Construction d'un RAG
3. RAG vs fine-tuning
4. Cas d'usage pédagogique

Pour faire un bon RAG, on a besoin de :

- De documents
- Un modèle de langage
- Un modèle d'embedding
- Une base de données vectorielles

Construction d'une base de connaissances



Pourquoi convertir en Markdown ?

Les modèles de langages :

- manipulent du texte (des tokens) ;
- de nombreux formats ne sont pas du texte.

Le format Markdown est un format de **document texte**, **structuré**, à balises simples, **bien géré** par les modèles de langages.

Exemple



Google Docs, Microsoft Word, OpenOffice Writer :

ANF TDM & IA 2025

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

HTML :

```
<h1>ANF TDM & IA 2025</h1>  
<b>La recherche d'information assistée par IA</b></br>  
<i>Panorama des enjeux, usages et bonnes pratiques</i>
```

Exemple



Google Docs, Microsoft Word, OpenOffice Writer :

ANF TDM & IA 2025

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

Markdown :

ANF TDM & IA 2025

La recherche d'information assistée par IA

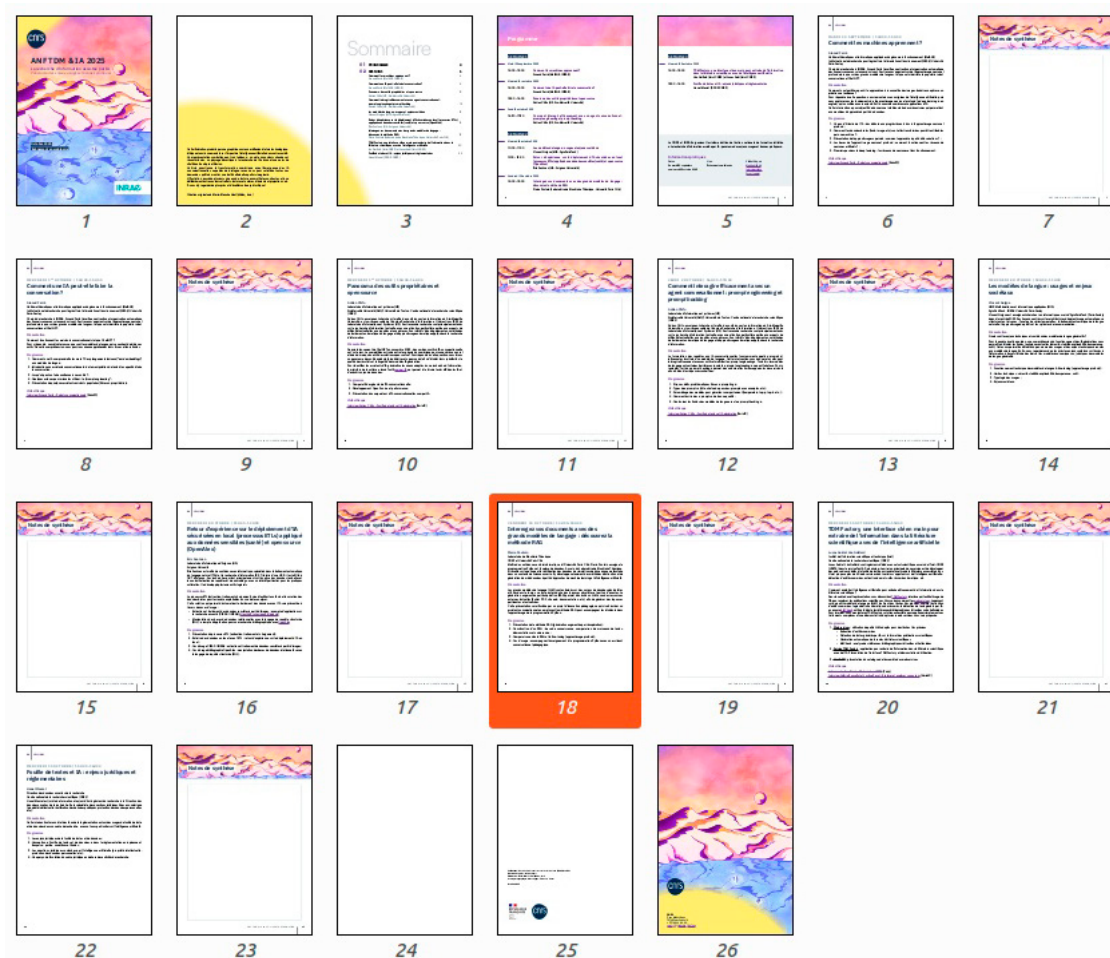
Panorama des enjeux, usages et bonnes pratiques

Référence → <https://www.markdownguide.org>

Pourquoi convertir en Markdown ?



Le livret de la formation
au format PDF



CNRS

ANF TDM & IA 2025

La recherche d'information assistée par IA
Panorama des enjeux, usages et bonnes pratiques

Octobre 2025
Événement en webinaire

INRAE

Photo: iStockphoto.com, Getty Images, Pexels

Le livret de la formation au format PDF

Pourquoi convertir en Markdown ?



Le livret de la formation
au format PDF

Programme

1. Présentation du processus ETL (extraction, traitement et chargement) ;
2. Extraire des données en local avec l'ETL : retour d'expérience sur le déploiement d'IA en local ;
3. Cas d'usage CIRAD / INRAE : extraction et traitement de données sensibles à partir d'images ;
4. Cas d'usage bibliographie OpenAlex : manipulation des bases de données relationnelles avec le langage de requête structurée (SQL).

16

^LNotes de synthèse

ANF TDM & IA 2025 - LIVRET DE FORMATION 17
^L 02 SESSIONS

VENDREDI 10 OCTOBRE | 14H00-15H00

Interrogez vos documents avec des
grands modèles de langage : découvrez la
méthode RAG

Pierre Poulain

Laboratoire de Biochimie Théorique

CNRS et Université Paris Cité

Maître de conférences en bioinformatique à l'Université Paris Cité, Pierre Poulain enseigne la programmation Python et l'analyse de données. Au sein du Laboratoire de Biochimie Théorique, il travaille sur le partage et la réutilisation des données de simulations de dynamique moléculaire dans un contexte de science ouverte. Il y développe notamment des méthodes d'extraction et de génération de métadonnées à partir d'approches de machine learning et d'intelligence artificielle.

Présentation

Les grands modèles de langage (LLM) sont entraînés sur des corpus de données généralistes et fixés dans le temps, ce qui peut produire des réponses obsolètes et parfois inexacts. La génération augmentée par récupération (RAG) permet d'enrichir un LLM avec des ressources

Comment convertir en Markdown ?



<https://github.com/docling-project/docling>

Pourquoi convertir en Markdown ?



Le livret de la formation
au format PDF

Programme

1. Présentation du processus ETL (extraction, traitement et chargement) ;
2. Extraire des données en local avec l'ETL : retour d'expérience sur le déploiement d'IA en local ;
3. Cas d'usage CIRAD / INRAE : extraction et traitement de données sensibles à partir d'images ;
4. Cas d'usage bibliographie OpenAlex : manipulation des bases de données relationnelles avec le langage de requête structurée (SQL).

<!-- image -->

VENDREDI 10 OCTOBRE | 14H00-15H00

Interrogez vos documents avec des grands modèles de langage : découvrez la méthode RAG

Pierre Poulain

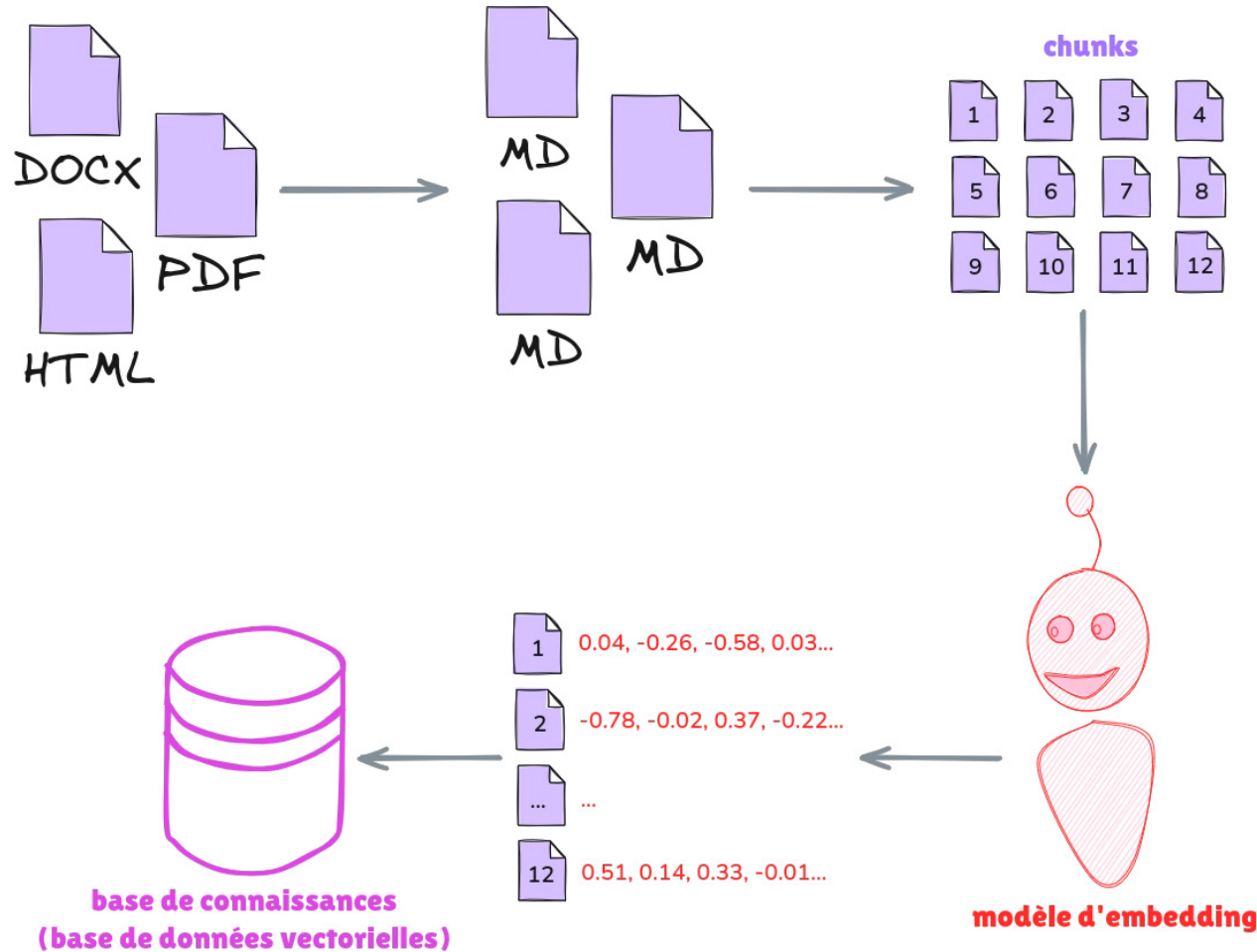
Laboratoire de Biochimie Théorique CNRS et Université Paris Cité

Maître de conférences en bioinformatique à l'Université Paris Cité, Pierre Poulain enseigne la programmation Python et l'analyse de données. Au sein du Laboratoire de Biochimie Théorique, il travaille sur le partage et la réutilisation des données de simulations de dynamique moléculaire dans un contexte de science ouverte. Il y développe notamment des méthodes d'extraction et de génération de métadonnées à partir d'approches de machine learning et d'intelligence artificielle.

Présentation

Les grands modèles de langage (LLM) sont entraînés sur des corpus de données généralistes et fixés dans le temps, ce qui peut produire des réponses obsolètes et parfois inexactes. La génération augmentée par récupération (RAG) permet d'enrichir un LLM avec des ressources externes textuelles (fichier PDF, site web, documentation, etc.) afin de générer des réponses pertinentes et actualisées.

Construction d'une base de connaissances



Pourquoi découper ?

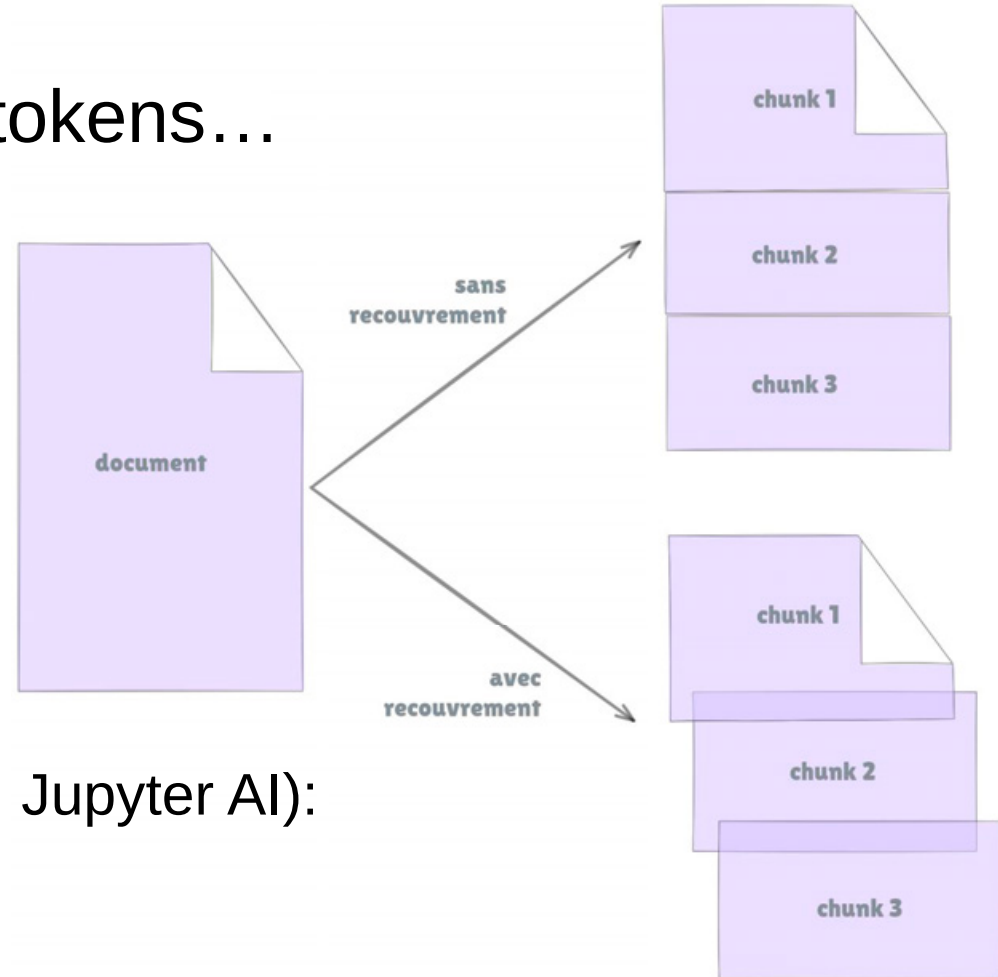
- Le ou les documents peuvent être très longs et dépasser le contexte du modèle de langage.
- On souhaite utiliser des morceaux de texte homogènes sémantiquement.

Le découpage / « chunking »

Nombre de caractères, mots, tokens...

Avec ou sans structure

Avec ou sans recouvrement.



Exemple de paramètres (OpenWebUI, Jupyter AI):

- chunk_size : 1500 / 2000 caractères
- chunk_overlap : 100 caractères

Comment découper ?



LangChain

<https://www.langchain.com/>

<https://github.com/langchain-ai/langchain>

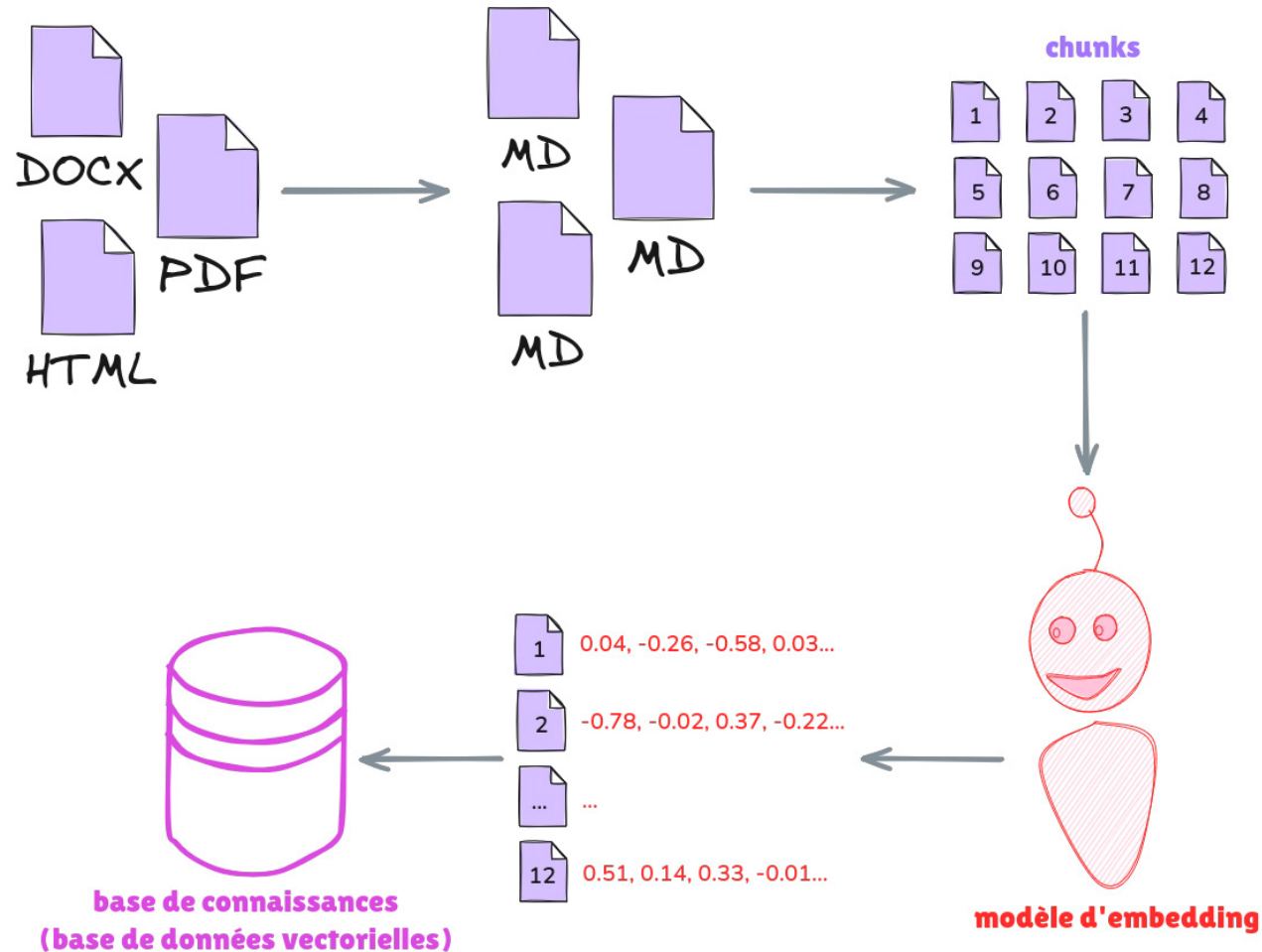


LlamaIndex

<https://www.llamaindex.ai/>

https://github.com/run-llama/llama_index/

Construction d'une base de connaissances

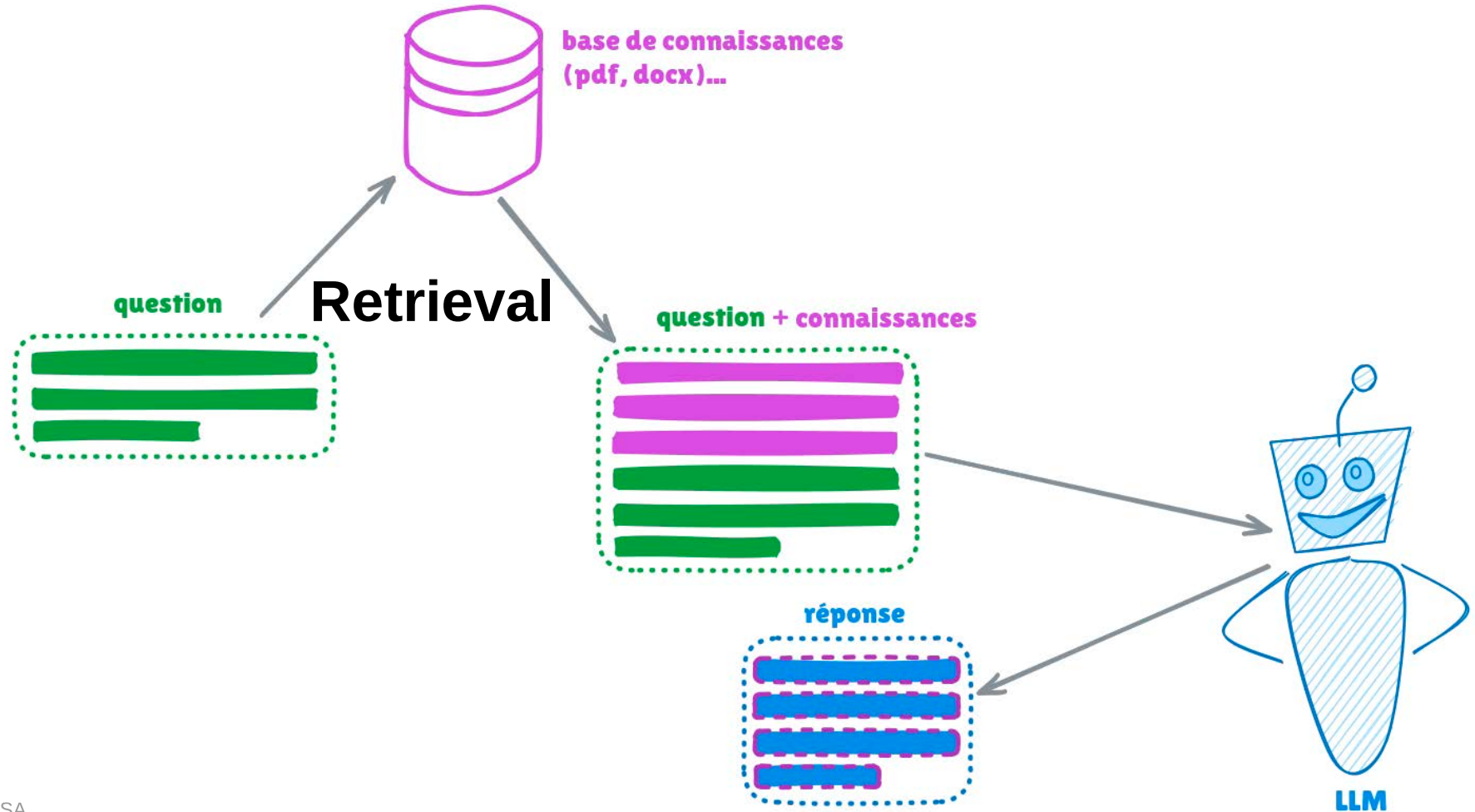


Pourquoi un modèle d'embedding ?

Dans une approche RAG, on cherche, dans la base de connaissances, les morceaux de documents (chunks) les plus proches sémantiquement de la question posée.

Remarque : la méthode TF-IDF (*Term Frequency - Inverse Document Frequency*) permet de comparer rapidement des documents, mais ne prend pas en compte les mots de sens proche.

Le Retrieval de RAG



Les modèles d'embedding

Les modèles de langage :

1 token ► 1 embedding ► 1 vecteur de grande dimension

Les modèles d'embedding :

plusieurs tokens ► 1 embedding ► 1 vecteur de grande dimension

Quelques modèles d'embedding

Modèle	Payant / gratuit	Dimensions vecteur	Taille contexte (token)
gemini-embedding-001 (Google)	payant	3 072	2 048
qwen3-Embedding-8B	gratuit	4 096	32 768
all-mpnet-base-v2	gratuit	768	514
gte-Qwen2-1.5B-instruct	gratuit	8 960	32 768
text-embedding-3-small (OpenAI)	payant	1 536	8 192
text-embedding-3-large (OpenAI)	payant	3 072	8 192
mistral-embed (Mistral)	payant	1 024	??

Sources :

<https://huggingface.co/spaces/mteb/leaderboard>

<https://platform.openai.com/docs/guides/embeddings>

https://docs.mistral.ai/capabilities/embeddings/text_embeddings/

Choisir un modèle d'embedding

Massive Text Embedding Benchmark (MTEB)

Spaces

mteb/leaderboard

like 6.51k

Restarting on CPU UPGRADE

App

Files

Community 169

> Embedding Leaderboard

This leaderboard compares 100+ text and image embedding models across 1000+ languages. We refer to the publication of each selectable benchmark for details on metrics, languages, tasks, and task types. Anyone is welcome [to add a model](#), [add benchmarks](#), [help us improve zero-shot annotations](#) or [propose other changes to the leaderboard](#).

MTEB(Multilingual, v2)

A large-scale multilingual expansion of MTEB, driven mainly by highly-curated community contributions covering 250+ languages.

- Number of languages: 1038
- Number of tasks: 131
- Number of task types: 9
- Number of domains: 20

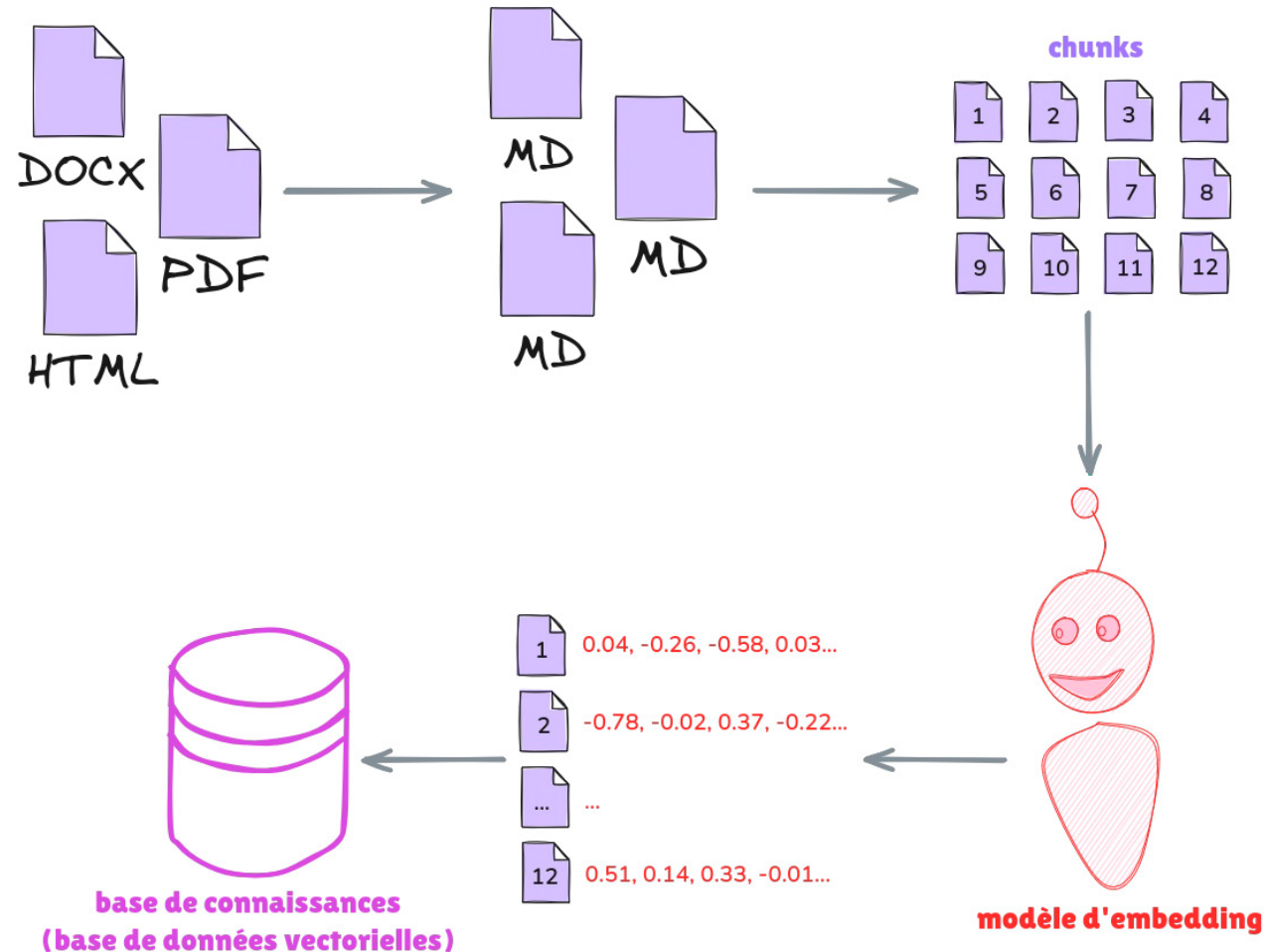
[Click for More Info](#)

Summary Performance per Model Size Performance per Task Type Performance per task Task information

Rank (B...	Model	Zero-shot	Memory U...	Number of P...	Embedding D...	Max Tokens	Mean (T...	Mean (TaskT...	Bitext ...	Classification	Clustering	Instruction R...	Mult:
65	Cohere-embed-english-light-v3.0	🚩 NA	Unknown	Unknown	384	512	45.85	39.48	26.10	48.89	40.40	-1.39	16.4
66	STS-multilingual-mpnet-base-v2	84%	1061	278M	768	514	48.52	42.38	47.63	51.02	32.38	-1.96	16.6
67	paraphrase-multilingual-MiniLM-L12-v2	100%	449	118M	768	512	48.73	42.37	44.56	51.66	38.33	-1.31	14.9
68	gte-large	🚩 NA	639	335M	1024	512	45.21	39.35	26.23	47.61	41.39	-1.29	17.9

Source : https://docs.mistral.ai/capabilities/embeddings/text_embeddings/

Construction d'une base de connaissances



Bases de données vectorielles

Stocker des données textuelles (chunks) et vectorielles (embeddings)
et réaliser des opérations vectorielles (similarité)



chroma

<https://www.trychroma.com/>

<https://github.com/chroma-core/chroma>



faiss

<https://faiss.ai/>

<https://github.com/facebookresearch/faiss>

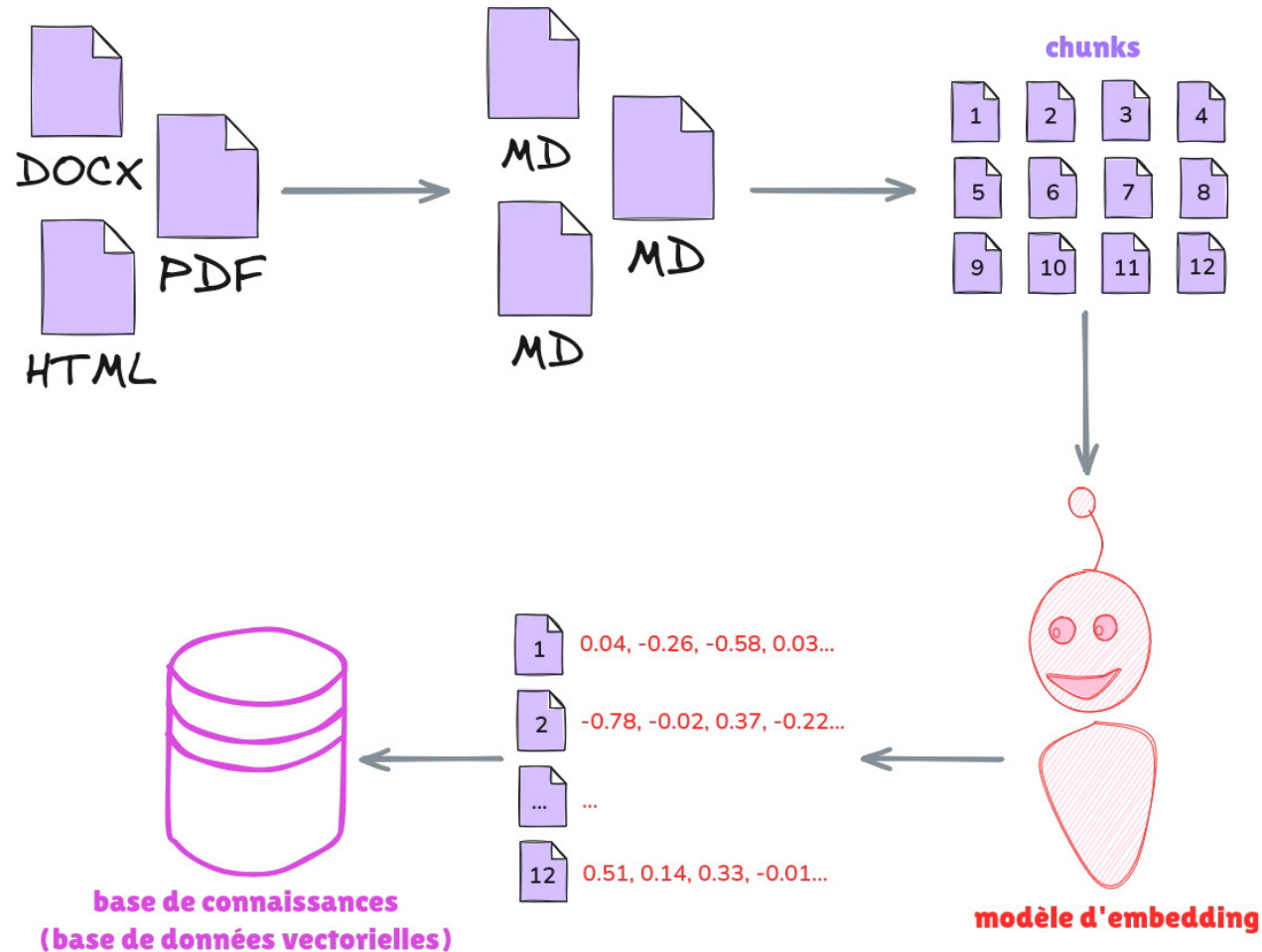


Qdrant

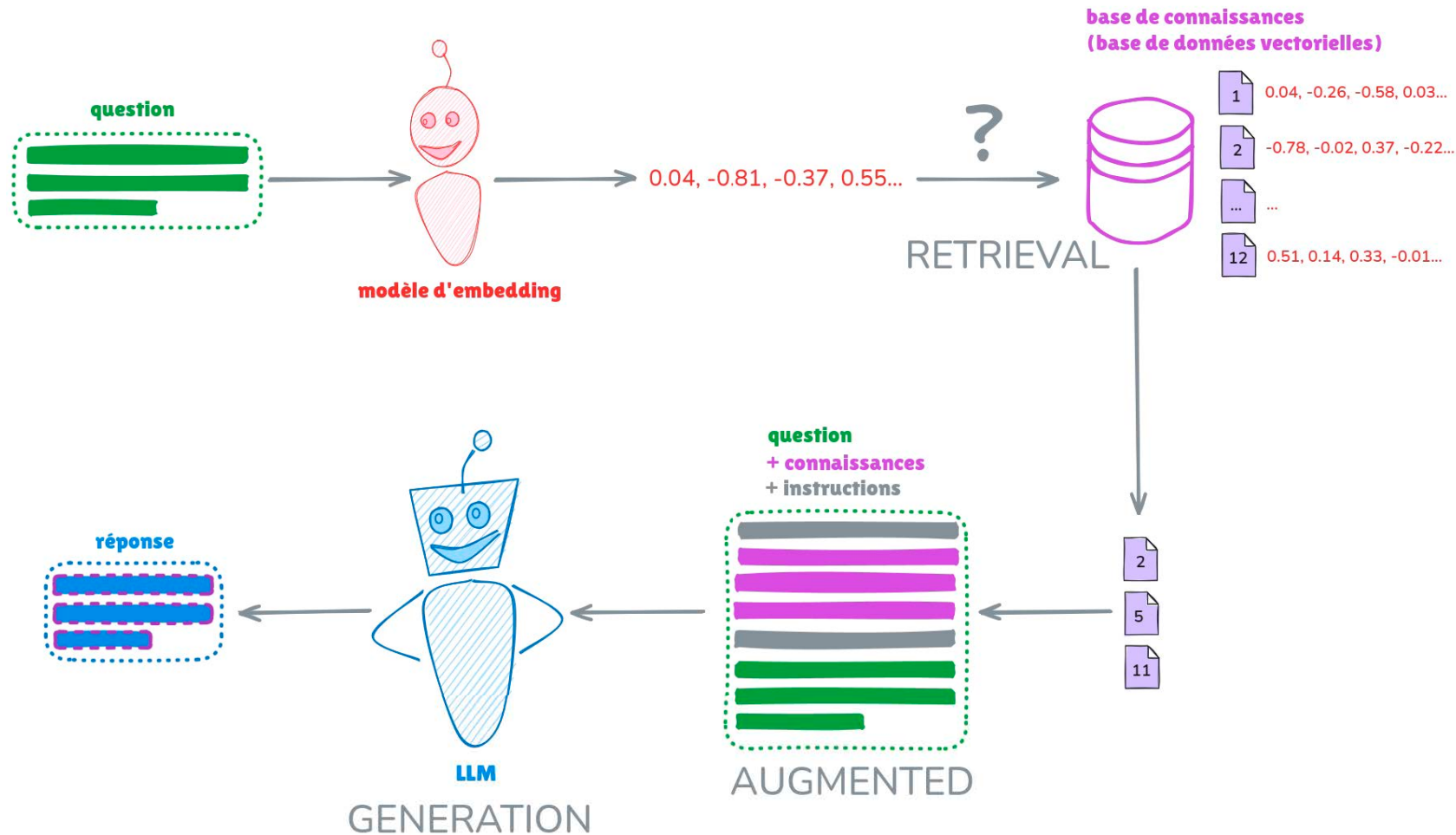
<https://qdrant.tech/>

<https://github.com/qdrant/qdrant>

Construction d'une base de connaissances



Retrieval-Augmented Generation



Retrieval-Augmented Generation

```
prompt = f"""
Context information is below.
-----
{retrieved_chunk}
-----
Given the context information and not prior knowledge, answer the query.
Query: {question}
Answer:
"""
```

Source : <https://docs.mistral.ai/guides/rag/#create-embeddings-for-each-text-chunk>

Au programme

1. La méthode RAG
2. Construction d'un RAG
3. RAG vs fine-tuning
4. Cas d'usage pédagogique

Fine tuning

Fine tuning : adapter un modèle de langage existant pour des données ou un usage particulier

Nécessite :

- les poids « ouverts » d'un modèle de langage (Llama, DeepSeek, Qwen, GPT-OSS, Mistral...) ;
- une infrastructure informatique (GPU) pour modifier les poids du modèle ;
- des données (en quantité) bien formatée ;
- une méthodologie (LoRA, QLoRA...).

RAG

RAG : orienter les réponses d'un modèle de langage avec des données spécifiques

Nécessite :

- l'accès à un modèle de langage (commercial ou pas) ;
- l'accès à un modèle d'embedding (commercial ou pas) ;
- des documents ;
- une base de connaissances.

Fine tuning vs RAG

Le RAG est :

- moins exigeant techniquement que du fine tuning ;
- plus souple (ajout / suppression de documents) ;
- plus sensible aux hallucinations que du fine tuning.

Au programme

1. La méthode RAG
2. Construction d'un RAG
3. RAG vs fine-tuning
4. Cas d'usage pédagogique

Assistant conversationnel BioPyAssistant

Projet d'innovation pédagogique Université Paris Cité (2024 → 2025)

- David Janiszek
- Samuel Bottani
- PP

Développement par Essmay Touami
(stage de master 2024)



Cahier des charges

Un assistant conversationnel (*chatbot*) pédagogique :

- pour **accompagner** des étudiant.e.s dans un enseignement de programmation Python en Master
- pour répondre aux questions des étudiant.e.s en se basant sur un **cours existant**
- capable de **citer ses sources**
- **disponible** 24h/24 7j/7, « **adaptable** »

À partir d'un cours existant

Cours de Python

| Introduction à la programmation Python pour la biologie

Patrick Fuchs et Pierre Poulain

prénom [point] nom [arobase] u-paris [point] fr

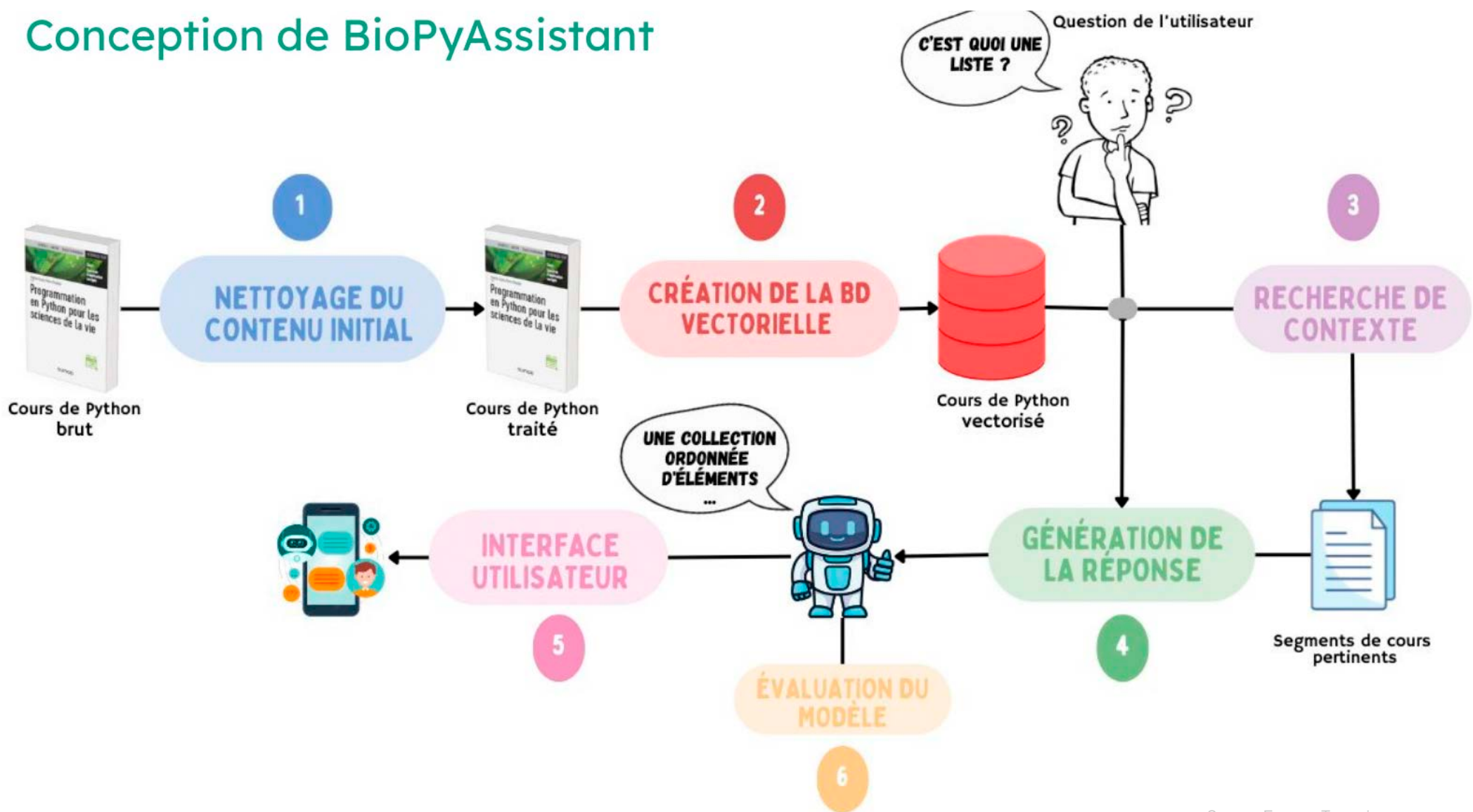


Version du 29/09/2025 | [PDF](#) du cours

- +15 ans
- 27 chapitres + 2 annexes
- sous licence CC BY-SA
- version web
- version PDF (+400 pages)

<https://python.sdv.u-paris.fr/>

Conception de BioPyAssistant



Prompt (engineering)

```
PROMPT_TEMPLATE = """
```

```
Tu es un assistant conversationnel pour aider des étudiants en biologie à apprendre la programmation Python.
```

```
Tu dois fournir une réponse à la question posée en te basant strictement sur les extraits de cours donnés dans le contexte.
```

```
Utilise uniquement le contexte suivant pour répondre à la question.
```

```
Question :
```

```
"{question}"
```

```
Contexte :
```

```
"{contexte}"
```

```
Répond en français à la question posée de façon claire et concise.
```

```
La réponse doit être compréhensible pour des étudiants débutants en programmation Python.
```

```
Si tu ne connais pas la réponse, dis que tu ne sais pas.
```

```
Si tu as besoin de plus d'informations, demande-le.
```

```
Si tu as besoin de clarifier la question, demande-le aussi.
```

```
"""
```

<https://biopyassistant.sdv.u-paris.fr/>



BioPyAssistant est un assistant pédagogique pour le [cours de programmation Python](#) pour les biologistes de Patrick Fuchs et Pierre Poulain.

N'hésite-pas à le solliciter en lui posant tes questions.

Discuter avec le cours

Discuter avec le cours (battle)

Chatbot



Bonjour, comment puis-je t'aider ?



BioPyAssistant est un assistant pédagogique pour le [cours de programmation Python](#) pour les biologistes de Patrick Fuchs et Pierre Poulain.

N'hésite-pas à le solliciter en lui posant tes questions.

Discuter avec le cours

Discuter avec le cours (battle)

Chatbot



Bonjour, comment puis-je t'aider ?



Quelle est la différence entre une liste et un set ?



La différence principale entre une liste et un set en Python réside dans leur structure et leur comportement :

1. Ordre et indexation :

- Une **liste** est ordonnée et indexable, ce qui signifie que les éléments ont un ordre défini et peuvent être accédés par leur position (index).
- Un **set** est non ordonné et non indexable, ce qui signifie qu'on ne peut pas accéder à ses éléments par leur position.

2. Unicité des éléments :

- Une **liste** peut contenir des éléments en double.
- Un **set** ne peut contenir que des éléments uniques. Si vous ajoutez des doublons, ils seront automatiquement éliminés.

3. Types de données contenus :

- Les **listes** peuvent contenir des éléments de types mixtes, y compris d'autres listes.
- Les **sets** ne peuvent pas contenir de types non hachables, comme les listes elles-mêmes.

En résumé, utilisez une liste lorsque vous avez besoin de maintenir l'ordre ou de contenir des doublons, et un set

Pose-moi une question sur le cours !





La différence principale entre une liste et un set en Python réside dans leur structure et leur comportement :

1. Ordre et indexation :

- Une **liste** est ordonnée et indexable, ce qui signifie que les éléments ont un ordre défini et peuvent être accédés par leur position (index).
- Un **set** est non ordonné et non indexable, ce qui signifie qu'on ne peut pas accéder à ses éléments par leur position.

2. Unicité des éléments :

- Une **liste** peut contenir des éléments en double.
- Un **set** ne peut contenir que des éléments uniques. Si vous ajoutez des doublons, ils seront automatiquement éliminés.

3. Types de données contenus :

- Les **listes** peuvent contenir des éléments de types mixtes, y compris d'autres listes.
- Les **sets** ne peuvent pas contenir de types non hachables, comme les listes elles-mêmes.

En résumé, utilisez une liste lorsque vous avez besoin de maintenir l'ordre ou de contenir des doublons, et un set lorsque vous avez besoin de garantir l'unicité des éléments.

Pour plus d'informations, je t'invite à consulter les rubriques suivantes du [cours en ligne](#) :

- Chapitre [14 Conteneurs](#) , rubrique [14.4.1 Définition et propriétés](#)
- Chapitre [14 Conteneurs](#) , rubrique [14.4.2 Utilité](#)



« response
engineering »

Pose-moi une question sur le cours !



Cours de Python

3. Affichage

4. Listes

5. Boucles et comparaisons

6. Tests

7. Fichiers

8. Dictionnaires et tuples

9. Modules

10. Fonctions

11. Plus sur les chaînes de caractères

12. Plus sur les listes

13. Plus sur les fonctions

14. Conteneurs

15. Création de modules

16. Bonnes pratiques en programmation Python

17. Expressions régulières

18. Jupyter et ses notebooks

19. Module Biopython

20. Module NumPy

21. Module Matplotlib

14.4.1 Définition et propriétés

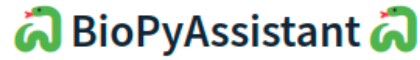
Les objets de type `set` représentent un autre type de conteneur qui peut se révéler très pratique. Ils ont la particularité d'être modifiables, non hachables, non ordonnés, non indexables et de ne contenir qu'une seule copie maximum de chaque élément. Pour créer un nouveau `set` on peut utiliser les accolades :

```
1 >>> set1 = {4, 5, 5, 12}
2 >>> set1
3 {12, 4, 5}
4 >>> type(set1)
5 <class 'set'>
```

Remarquez que la répétition du chiffre 5 dans la définition du `set` ligne 1 produit finalement un seul chiffre 5, car chaque élément ne peut être présent qu'une seule fois. Comme pour les dictionnaires (jusqu'à la version 3.6), les `sets` sont non ordonnés. La manière dont Python les affiche n'a pas de sens en tant que tel et peut être différente de celle utilisée lors de leur création.

Les `sets` ne peuvent contenir que des objets **hachables**. On a déjà eu le cas avec les clés de dictionnaire. Ceci optimise l'accès à chaque élément du `set`. Pour rappel, les objets hachables que nous connaissons sont les chaînes de caractères, les tuples, les entiers, les *floats*, les booléens et les *frozensets* (voir plus bas). Les objets non hachables que l'on connaît sont les listes, les `sets` et les dictionnaires. Si on essaie tout de même de mettre une liste dans un `set`, Python renvoie une erreur :

Comparaison des modèles



BioPyAssistant est un assistant pédagogique pour le [cours de programmation Python](#) pour les biologistes de Patrick Fuchs et Pierre Poulain.

N'hésite-pas à le solliciter en lui posant tes questions.

Discuter avec le cours

Discuter avec le cours (battle)

Pose une question, compare les réponses fournies par deux modèles de langages tirés au hasard et vote pour la meilleure réponse.

Modèle A



Bonjour, comment puis-je t'aider ?



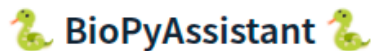
Modèle B



Bonjour, comment puis-je t'aider ?



« Response engineering »



BioPyAssistant est un assistant pédagogique pour le [cours de programmation Python](#) pour les biologistes de Patrick Fuchs et Pierre Poulain.

N'hésite-pas à le solliciter en lui posant tes questions.

Discuter avec le cours

Discuter avec le cours (battle)

Chatbot



Bonjour, comment puis-je t'aider ?



Comment réparer mon volet roulant ?



Désolé, je suis un assistant pour l'apprentissage de la programmation Python. Je ne suis pas en mesure de répondre à cette question.



Sous le capot

Contenu : cours au format **Markdown**

(<https://github.com/bioinfo-prog/cours-python>)

Code développé en Python par Essmay Touami

- disponible sur GitHub (<https://github.com/pierrepo/biopyassistant>)
- **LangChain** pour l'interfaçage avec les modèles de langage et la préparation du corpus initial (le cours de Python)
- **ChromaDB** pour le stockage des morceaux de texte du cours, des embeddings et des méta-données associées
- **Gradio** pour l'interface web du chatbot

Modèles utilisés (**par API uniquement**)

- Embedding : text-embedding-3-large (OpenAI)
- LLM
 - ✓ OpenAI : gpt-4o, gpt-4-turbo, gpt-3.5-turbo
 - ✓ Mistral AI : mistral-large-latest, open-mixtral-8x7b, open-mixtral-8x22b
 - ✓ Groq : llama-3.1-70b-versatile, llama-3.1-8b-instant

Chatbot hébergé sur une machine virtuelle

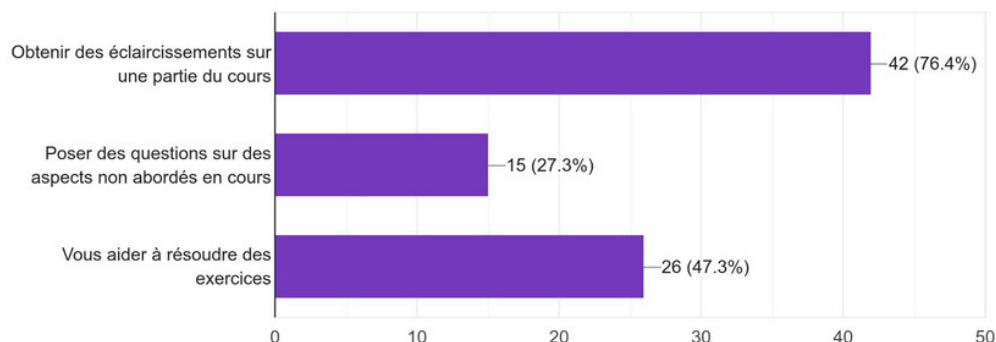
(4 vCPU, 4 Go RAM, 80 Go stockage SSD, Ubuntu 24.04) louée chez OVH

Retours des étudiants

De 70 étudiant.e.s M1/M2 en 2024-2025...

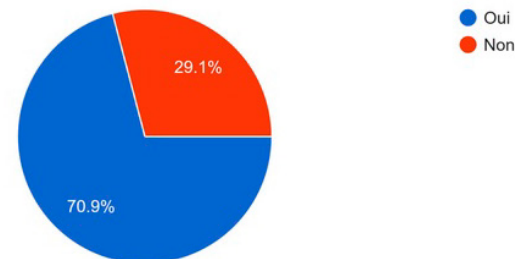
Avez-vous utilisé cet assistant pour :

55 responses



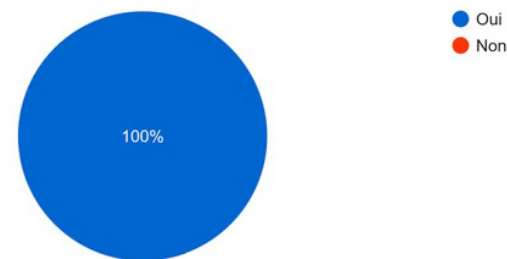
Avez-vous utilisé les liens vers le cours en ligne qui étaient fournis à la fin de chaque réponse de l'assistant ?

55 responses



Pensez-vous que cet assistant soit adapté à des étudiant.e.s débutant.e.s en programmation Python ?

55 responses

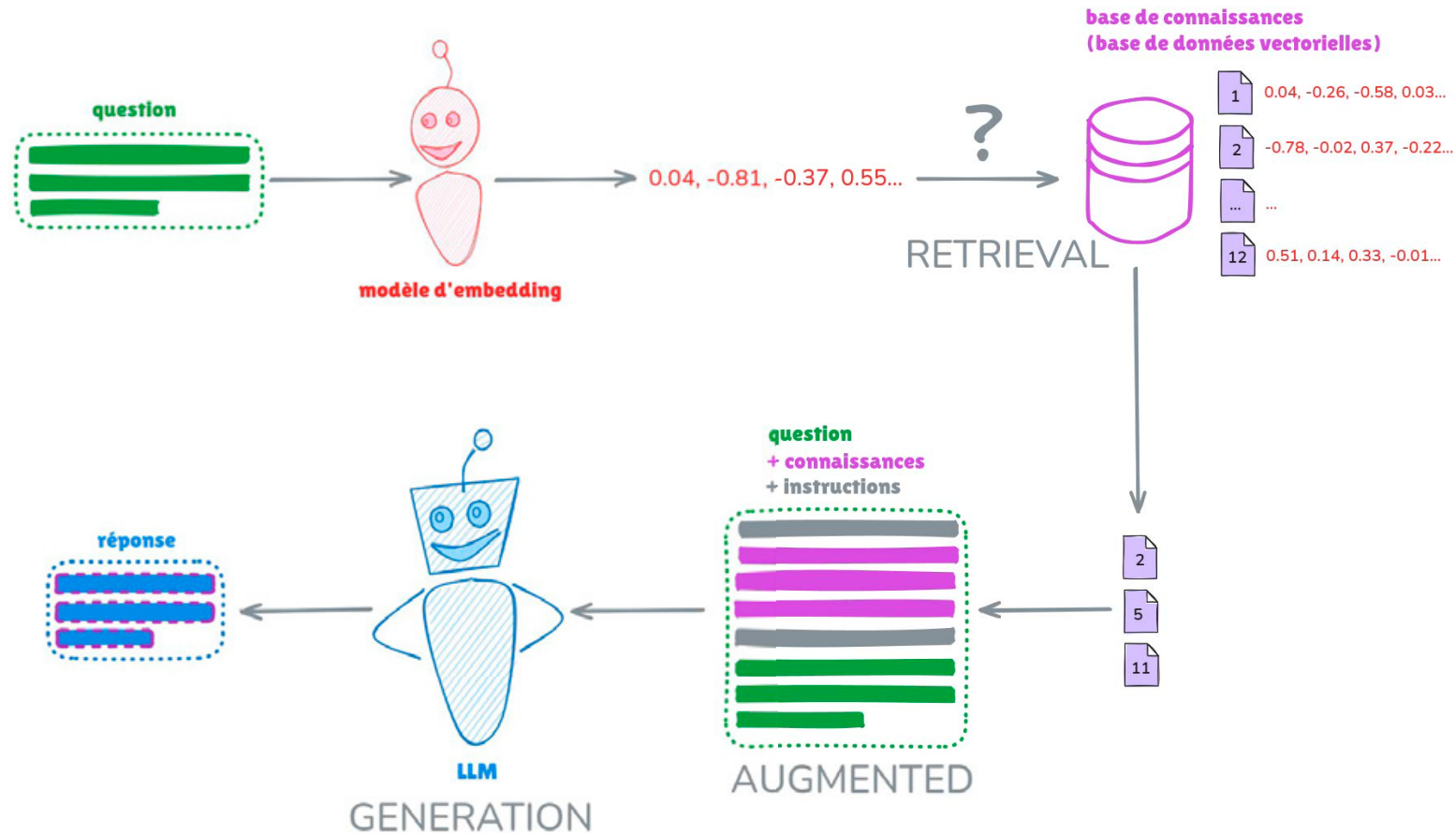


...à 400 étudiant.e.s L2/L3 en 2025-2026

Conclusion

Retrieval-Augmented Generation (RAG)

génération augmentée par récupération



Retrieval-Augmented Generation (RAG)

Solutions open-source intégrées pour du RAG :
OpenWebUI, AnythingLLM, RAGflow...

La qualité des documents initiaux est fondamentale
(*garbage in, garbage out*)

Possibilité d'évaluer les performances d'un RAG :
RAGAS, LettuceDetect...

Futur ?

Agents ?

Pour aller plus loin :

Génération augmentée par récupération (RAG) : guide pour exploiter les données de sa TPE PME avec l'IA générative, France Num, 2025.

Explorer ses documents de travail avec les méthodes RAG : construire un agent conversationnel sur ses propres documents, Stéphane Pouyllau, 2025.

Retrieval-Augmented Generation for Large Language Models: A Survey, Gao et al., arXiv, 2023.