



Rapport du Projet :
ChatBot de recommandation de films

Matière : NLP

Encadrant : Hamza ASSOUANE

Etudiants : Es-Sadany Yassine

Groupe : ING3 IA 2

06 Décembre 2023

Table des matières

1	Introduction	2
2	Description de la Solution/Travail	2
2.1	Modèles et approches Utilisés	2
2.2	Dataset Utilisé	2
2.3	Diagramme Fonctionnel	2
2.4	Méthodologie de Travail	3
3	Approche Méthodologique	3
3.1	Première LLM	3
3.2	Configuration du Pipeline de Génération de Texte	5
3.3	Configuration de la Mémoire de Conversation	5
3.4	Chaîne de Récupération Conversationnelle	6
3.5	Deuxième LLM	6
4	Test des recommandations du ChatBot pour LLama 2 optimisé et GPT-TURBO-3.5	7
4.1	LLAMA 2	7
4.2	GPT-TURBO-3.5	7
5	Limitations et Perspectives d'Améliorations	7
6	Contribution des Membres du Groupe	8
7	Conclusion	8

1 Introduction

Ce projet a pour but de créer un chatbot de recommandation de films selon un film référence ou selon une description. Dans ce projet, on a utilisé 2 LLM open source pour la partie conversationnelle (**Llama-2-13B-chat** et **GPT-TURBO-3.5**) : ainsi qu'on a comparé entre 2 approches pour le même LLM Llama-2 : **HuggingFacePipeline** et **CTransformers**.

2 Description de la Solution/Travail

2.1 Modèles et approches Utilisés

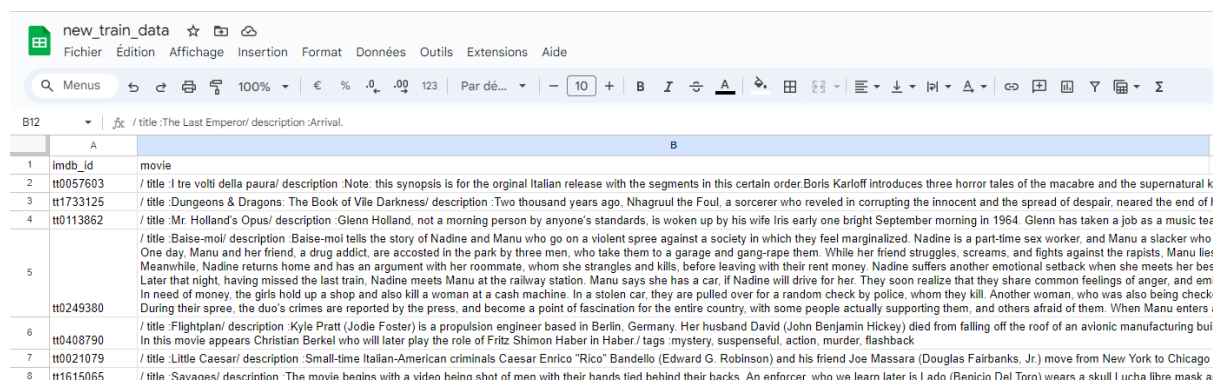
la technique utilisée dans ce projet est appelée le RAG(Retrieval-Augmented Generation), il consiste à chercher des embeddings dans le vectorestore (base de données des embeddings) les plus similaires. elle est bien expliquée dans le diagramme fonctionnel en suite. Dans un premier temps, j'ai choisi de travailler avec un d'OpenAi (GPT-TURBO-3.5), et un llm de Llama2 (Llama-2-13B-chat) avec

2.2 Dataset Utilisé

Le dataset utilisé est extrait (toutes les données ou la colonnes split = train) d'un fichier csv téléchargé sur via le lien <https://www.kaggle.com/datasets/cryptexcode/mpst-movie-plot-synopses-with-tags> qui contient 6 colonnes :

- **imdb_id** : L'id IMDB du film
- **title** : Titre du film
- **plot_synopsis** : Description du film
- **tags** : les catégories du films (comedy, action ...)
- **split** : (train, test, val)
- **synopsis_source** : (imdb, wikipedia)

le dataset a été converti en un nouveau dataset qui contient 2 colonnes : l'imdb_id du film et une colonne movie qui fusionne les 5 autres colonnes décrites dessus. La taille du fichier final est de 46MB (ceci a affecté la rapidité du chatbot)



new_train_data	
Fichier Édition Affichage Insertion Format Données Outils Extensions Aide	
B12 / title :The Last Emperor/ description :Arrival.	
A	B
1	imdb_id movie
2	tt0057603 / title :I tre volti della paura/ description :Note: this synopsis is for the original Italian release with the segments in this certain order Boris Karloff introduces three horror tales of the macabre and the supernatural k
3	tt1733125 / title :Dungeons & Dragons: The Book of Vile Darkness/ description :Two thousand years ago, Nhagruul the Foul, a sorcerer who reveled in corrupting the innocent and the spread of despair, neared the end of t
4	tt0113862 / title :Mr. Holland's Opus/ description :Glenn Holland, not a morning person by anyone's standards, is woken up by his wife Iris early one bright September morning in 1964. Glenn has taken a job as a music tea
5	/ title :Baise-moi/ description :Baise-moi tells the story of Nadine and Manu who go on a violent spree against a society in which they feel marginalized. Nadine is a part-time sex worker, and Manu a slacker who One day, Manu and her friend, a drug addict, are accosted in the park by three men, who take them to a garage and gang-rape them. While her friend struggles, screams, and fights against the rapists, Manu lies Meanwhile, Nadine returns home and has an argument with her roommate, whom she strangles and kills, before leaving with their rent money. Nadine suffers another emotional setback when she meets her bes Later that night, having missed the last train, Nadine meets Manu at the railway station. Manu says she has a car, if Nadine will drive for her. They soon realize that they share common feelings of anger, and onl In need of money, the girls hold up a shop and also kill a woman at a cash machine. In a stolen car, they are pulled over for a random check by police, whom they kill. Another woman, who was also being checki During their spree, the duo's crimes are reported by the press, and become a point of fascination for the entire country, with some people actually supporting them, and others afraid of them. When Manu enters i
6	tt0408790 / title :Flightplan/ description :Kyle Pratt (Jodie Foster) is a propulsion engineer based in Berlin, Germany. Her husband David (John Benjamin Hickey) died from falling off the roof of an avionc manufacturing buil In this movie appears Christian Berkel who will later play the role of Fritz Shimon Haber in Haber/ tags :mystery, suspenseful, action, murder, flashback
7	tt0021079 / title :Little Caesar/ description :Small-time Italian-American criminals Caesar Enrico "Rico" Bandello (Edward G. Robinson) and his friend Joe Massara (Douglas Fairbanks, Jr.) move from New York to Chicago
8	tt1615065 / title :Savages/ description :The movie begins with a video being shot of men with their hands tied behind their backs. An enforcer, who we learn later is Lado (Benicio Del Toro) wears a skull Lucha libre mask ar

FIGURE 1 – Dataset d'entraînement

2.3 Diagramme Fonctionnel

Voici le diagramme fonctionnel du chatbot, réalisé à l'aide de l'outil drawio :

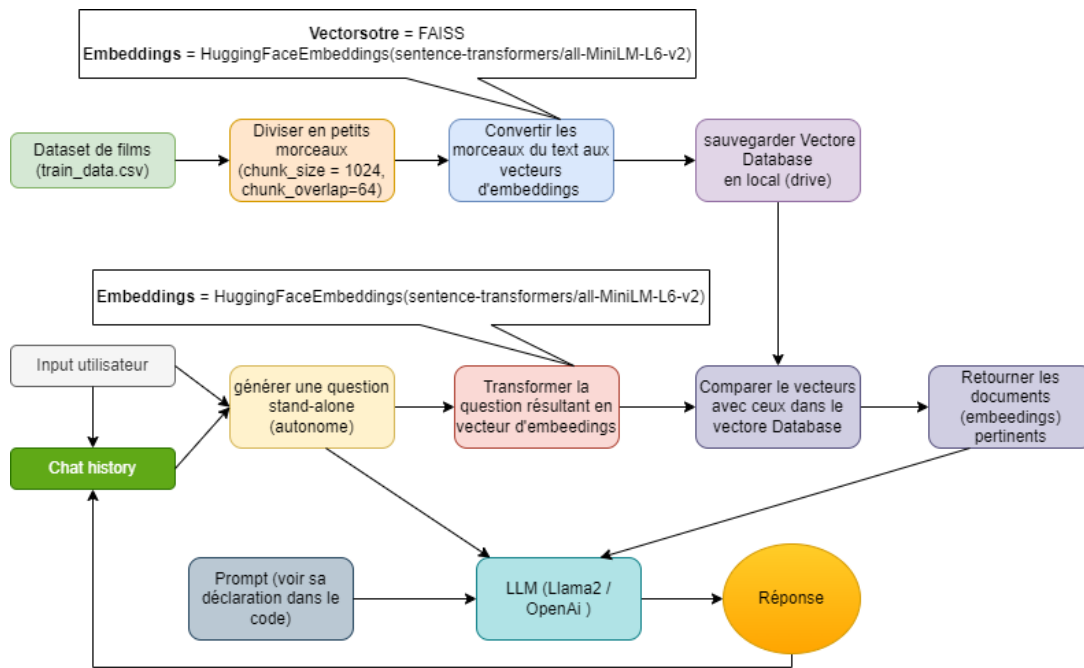


FIGURE 2 – Diagramme fonctionnel

2.4 Méthodologie de Travail

la première étape était de choisir un bon dataset, ensuite après la recommandation du professeur, j'ai consulté la documentation de la technique du **RAG** sur le site du LangChain (https://python.langchain.com/docs/use_cases/question_answering/code_understanding), puis j'ai appliqué cette technique sur mon dataset afin de créer un vectorestore d'embeddings, mais j'ai utilisé la librairie FAISS au lieu de CHROMA, enfin j'ai intégré ce contexte dans les modèles LLM choisis pour créer le chatbot en personnalisant la prompt pour qu'elle réponde selon le dataset et le contexte de la conversation, et évitant de répondre à n'importe quelle question hors contexte de recommandation de films.

3 Approche Méthodologique

3.1 Première LLM

Au début, j'ai choisi de travailler avec la méthode CTransformers pour le LLM LLama2 :

```

[ ] # Créer une instance de la classe CTransformers avec des paramètres spécifiés
llama2_llm1 = CTransformers(
    model="TheBloke/Llama-2-7B-Chat-GGML", # Spécifier le nom du modèle ou le chemin vers un modèle pré-entraîné
    model_type="llama", # Spécifier le type de modèle (dans ce cas, "llama")
    max_new_tokens=512, # Limiter le nombre maximal de nouveaux tokens générés
    temperature=0, # Fixer la température à zéro pour des prédictions déterministes
    streamer=streamer, # Utiliser l'instance de TextStreamer pour gérer le streaming du texte
)

Fetching 1 files: 100% ██████████ 1/1 [00:00<00:00, 1.66it/s]
config.json: 100% ██████████ 29.0/29.0 [00:00<00:00, 2.15kB/s]
Fetching 1 files: 100% ██████████ 1/1 [00:08<00:00, 8.14s/it]
llama-2-7b-chat-ggmlv3.q2_K.bin: 100% ██████████ 2.87G/2.87G [00:07<00:00, 427MB/s]
  
```

FIGURE 3 – llama2 avec CTransformers

Après pour le tester j'ai posé au chatbot une question hors contexte pour tester au même temps la fonctionnalité de prompt et avoir le temps d'exécution dans les meilleurs des cas, car dans ce cas le modèle va pas aller chercher dans le vectorstore des embedding qui est volumineux, mais malgré ça le temps de réponse obtenu (131.56 seconds) est très grand comme vous voyez ci-dessous :

```

Question 1: Hi, what is the capital fo Morocco
WARNING:transformers:Number of tokens (513) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (514) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (515) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (516) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (517) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (518) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (519) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (520) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (521) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (522) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (523) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (524) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (525) exceeded maximum context length (512).
WARNING:transformers:Number of tokens (526) exceeded maximum context length (512).

```

(a) Teste de llama2 avec CTransformers

(b) suite

FIGURE 4 – Teste de llama2 avec CTransformers

Pour cela j'ai optimisé le modèle llama2 de la manière suivante :

```

# Spécifier le nom du modèle ou le chemin vers un modèle GPTQ pré-entraîné
model_name_or_path = "TheBloke/Llama-2-13B-chat-GPTQ"

# Définir un nom de base pour le modèle
model_basename = "model"

# Initialiser un tokenizer pour le modèle GPTQ spécifié en utilisant la bibliothèque Hugging Face Transformers
tokenizer = AutoTokenizer.from_pretrained(model_name_or_path, use_fast=True)

# Instancier un modèle GPTQ pour la modélisation du langage causal en utilisant la configuration spécifiée
model = AutoGPTQForCausalLM.from_quantized(
    model_name_or_path,
    revision="gptq-4bit-128g-actorder_True", # Spécifier la révision ou la version du modèle
    model_basename=model_basename, # Fournir le nom de base pour le modèle
    use_safetensors=True, # Activer l'utilisation de tenseurs sécurisés
    trust_remote_code=True, # Faire confiance au code distant pour le modèle
    inject_fused_attention=False, # Ne pas injecter une attention fusionnée dans le modèle
    device=DEVICE, # Spécifier le périphérique sur lequel le modèle doit être chargé
    quantize_config=None, # Spécifier la configuration de quantification (dans ce cas, elle est définie sur None)
)

tokenizer_config.json: 100% 727/727 [00:00<00:00, 62.8kB/s]
tokenizer.model: 100% 500k/500k [00:00<00:00, 25.2MB/s]
tokenizer.json: 100% 1.84M/1.84M [00:00<00:00, 0.04MB/s]
special_tokens_map.json: 100% 411/411 [00:00<00:00, 31.7kB/s]
config.json: 100% 837/837 [00:00<00:00, 63.5kB/s]
config.json: 100% 761/761 [00:00<00:00, 68.5kB/s]
quantize_config.json: 100% 187/187 [00:00<00:00, 16.9kB/s]
model.safetensors: 100% 7.26G/7.26G [00:22<00:00, 293MB/s]
WARNING:auto_gptq.nn_modules.fused_llama_mlp:skip module injection for FusedLlamaMLPForQuantizedModel not support integrate without triton yet.

```

FIGURE 5 – llama2 quantisation de llama2 avec AutoGPTQForCausalLM

ensuite on crée le modèle avec HuggingFacePipeline :

```

[ ] # Créer une instance de la classe TextStreamer avec un tokenizer spécifié, en sautant les prompts et les tokens spéciaux
streamer = TextStreamer(tokenizer, skip_prompt=True, skip_special_tokens=True)

# Configurer un pipeline de génération de texte en utilisant le modèle, le tokenizer et d'autres paramètres spécifiés
text_pipeline = pipeline(
    "text-generation",
    model=model, # Utiliser le modèle GPTQ précédemment défini
    tokenizer=tokenizer, # Utiliser le tokenizer précédemment initialisé
    max_new_tokens=512, # Limiter le nombre maximal de nouveaux tokens générés
    temperature=0, # Fixer la température à zéro pour des prédictions déterministes
    top_p=0.95, # Limiter les prédictions aux tokens les plus probables jusqu'à la probabilité cumulative de 0.95
    repetition_penalty=1.15, # Appliquer une pénalité de répétition aux tokens générés
    streamer=streamer, # Utiliser l'instance de TextStreamer pour gérer le streaming du texte
)

# Créer une instance de la classe HuggingFacePipeline avec un pipeline de génération de texte spécifié
llama2_llm2 = HuggingFacePipeline(pipeline=text_pipeline, model_kwargs={"temperature": 0})

```

FIGURE 6 – Llama2 HuggingFacePipeline

Nous utilisons la bibliothèque Hugging Face et son module `transformers` pour charger le tokenizer et le modèle. Le modèle est instancié en tant que `AutoGPTQForCausalLM` à partir de la version quantifiée du modèle pré-entraîné. Les paramètres spécifiques incluent la révision du modèle, l'utilisation de tenseurs sécurisés (`use_safetensors`), la confiance dans le code distant (`trust_remote_code`), et d'autres paramètres tels que l'appareil (`device`) et la configuration de quantification (`quantize_config`).

- **revision** : La révision spécifique "gptq-4bit-128g-actorder_True" indique que le modèle est quantifié avec une précision de 4 bits, une taille de 128 Go, et un ordre d'activation activé. Cette configuration particulière peut être le résultat d'une optimisation pour l'efficacité de la mémoire et des performances spécifiques à la tâche.
- **use_safetensors** : L'activation de SafeTensors peut être cruciale pour la stabilité pendant l'entraînement. Si le modèle est sujet à des instabilités numériques, l'utilisation de SafeTensors peut aider à détecter et à traiter ces problèmes.
- **trust_remote_code** : Si le modèle doit charger du code distant, le paramètre 'trust_remote_code' est défini sur True pour indiquer que le code distant associé au modèle est considéré comme sûr.
- **inject_fused_attention** : Le choix de ne pas injecter une attention fusionnée peut être lié à des considérations de performance ou de préférences spécifiques au modèle. L'injection d'une attention fusionnée peut affecter la vitesse d'inférence et d'autres aspects du modèle.
- **device** : Le choix du dispositif (probablement spécifié comme "DEVICE") est crucial pour le déploiement. Il peut s'agir d'un GPU spécifique pour accélérer les calculs ou d'un autre dispositif en fonction des ressources disponibles.
- **quantize_config** : Le choix de laisser 'quantize_config' à None indique probablement que la quantification est gérée automatiquement ou que des paramètres par défaut sont utilisés.

En testant ce nouveau modèle dans le chatbot on obtient un temps de réponse (8.86 secondes) beaucoup plus mieux par rapport à la même question posé au premier modèle :

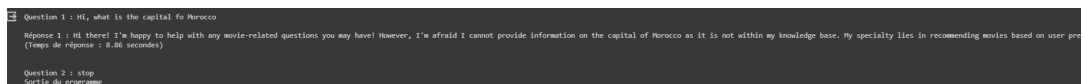


FIGURE 7 – Test de Llama2 quantisé avec HuggingFacePipeline

3.2 Configuration du Pipeline de Génération de Texte

Nous utilisons le module `pipeline` de la bibliothèque Hugging Face pour configurer le pipeline de génération de texte. Les paramètres incluent le modèle, le tokenizer, le nombre maximal de nouveaux tokens générés (`max_new_tokens`), la température (`temperature`), *top-p* (`top_p`), et la pénalité de répétition (`repetition_penalty`).

3.3 Configuration de la Mémoire de Conversation

Nous utilisons une mémoire de résumé de conversation (`ConversationSummaryMemory`) pour stocker les échanges précédents. Cette mémoire est liée au modèle de langage (llm) et permet de retourner les messages stockés.

```

# Créer une instance de la classe ConversationSummaryMemory avec un autre modèle Llama (llama2_llm2),
# une clé de mémoire ("chat_history") et la spécification de retourner les messages
llama2_memory2 = ConversationSummaryMemory(
    llm=llama2_llm2, # Utiliser l'autre modèle Llama spécifié (llama2_llm2)
    memory_key="chat_history", # Utiliser la clé de mémoire "chat_history"
    return_messages=True # Spécifier de retourner les messages
)

```

FIGURE 8 – Historique du chatbot

3.4 Chaîne de Récupération Conversationnelle

Enfin, nous utilisons une chaîne de récupération conversationnelle (`ConversationalRetrievalChain`) qui utilise le modèle de langage comme LLM (`llama2_llm1`) et un récupérateur de documents (`docsearch.as_retriever()`). Des paramètres spécifiques, tels que la chaîne de combinaison de documents (`combine_docs_chain_kwargs`) et la mémoire (`memory`), sont également configurés.

```

# Créer une instance de la classe ConversationalRetrievalChain en utilisant un autre modèle Llama (llama2_llm2),
# un récupérateur spécifié (docsearch.as_retriever()), des paramètres pour la combinaison de documents (combine_docs_chain_kwargs),
# et une mémoire spécifiée (llama2_memory2)
llama2_qa2 = ConversationalRetrievalChain.from_llm(
    llama2_llm2, # Utiliser l'autre modèle Llama spécifié (llama2_llm2)
    retriever=docsearch.as_retriever(search_kwargs={'k': 2}), # retourner les 2 meilleurs recommandations
    combine_docs_chain_kwargs={'prompt': prompt}, # Utiliser les paramètres pour la combinaison de documents
    memory=llama2_memory2 # Utiliser la mémoire spécifiée (llama2_memory2)
)

```

FIGURE 9 – Fonction de réponse

Cette configuration complète constitue l'approche adoptée pour définir le modèle de langage utilisé dans le chatbot, intégrant des éléments tels que la quantification, la génération de texte, la mémoire de conversation et la récupération conversationnelle.

3.5 Deuxième LLM

Le deuxième LLM utilisé pour la comparaison est GPT-TURBO-3.5 d'OPENAI, celui-ci nécessite un clé api d'un compte openai pour pouvoir l'utiliser, mais après le test il montre qu'il réponds dans un délai (2.87) mieux que le premier LLM par rapport à la même question :

```

# Définir la clé d'API OpenAI en tant que variable d'environnement
os.environ['OPENAI_API_KEY'] = 'sk-vWzGZm3hUvZ0J0hKIMutT3B1bkFJGDCQHo8MwN1UZ2Iyg01s'

# Créer une instance de la classe ChatOpenAI avec des paramètres spécifiés
gpt_llm = ChatOpenAI(
    model_name="gpt-3.5-turbo", # Spécifier le nom du modèle (dans ce cas, "gpt-3.5-turbo")
    temperature=0, # Fixer la température à zéro pour des prédictions déterministes
    max_tokens=512, # Limiter le nombre maximal de tokens générés
    top_p=1, # Limiter les prédictions aux tokens les plus probables jusqu'à la probabilité cumulative de 1
    frequency_penalty=0 # Appliquer une pénalité de fréquence de token de zéro
)

```

FIGURE 10 – LLM : GPTTURBO3.5

```

Question 1 : Hi, what is the capital fo Morocco
Réponse 1 : I'm sorry, but I'm specialized in providing movie recommendations and don't have information about the capital of Morocco. Is there anything else I can assist you with regarding movies?
(Temps de réponse : 2.87 secondes)

Question 2 : stop
Sortie du programme

```

FIGURE 11 – Test de GPT-TURBO-3.5

Remarque : Pour ne pas avoir d'interupption dans le cas du GPT-TURBO-3.5 il faut attendre au moins 20 secondes avant de poser une nouvelle question au chatbot car c'est la limite posé par OPENAI.

4 Test des recommandations du ChatBot pour LLama 2 optimisé et GPT-TURBO-3.5

4.1 LLAMA 2

```
Question 1: Hi! I want to watch a comedy movie ?
Réponse 1: Hello there! I'd be happy to help you find some great comedy movies! Based on your interests, here are a few recommendations:
1. The Big Lebowski ( imdb_id: 70009 ) - This cult classic has a unique blend of humor, absurdity, and crime. It's a must-see for any fan of the genre!
2. Monty Python and the Holy Grail ( imdb_id: 2648 ) - A hilarious and iconic British comedy that spoofs King Arthur and his knights.
3. Airplane! ( imdb_id: 5184 ) - A parody of disaster movies that will keep you laughing from start to finish.
4. The Hangover ( imdb_id: 13905 ) - A raunchy and outrageously funny film about a group of friends who wake up after a wild bachelor party with no memory of what happened the night before.
5. Superbad ( imdb_id: 11880 ) - A coming-of-age story that follows two high school seniors as they try to buy alcohol for a party and end up on a series of misadventures.
I hope you enjoy these picks! Let me know if you have any other questions or if there's anything else I can help with.
(Temps de réponse : 21.66 secondes)

Question 2: ok. I would like to see some action movies like iron man ?
Réponse 2 :
Thank you for asking! I'm happy to help you find some action movies similar to Iron Man. Based on your interest, here are a few suggestions:
1. The Avengers (2012) - This Marvel Cinematic Universe film brings together some of the most iconic superheroes, including Iron Man, Captain America, Thor, Hulk, Black Widow, and Hawkeye. It's an action-packed adventure that combines humor and heart.
IMDb ID: tt1225762
2. The Dark Knight (2008) - This critically acclaimed Batman film directed by Christopher Nolan features Heath Ledger's iconic performance as the Joker. It's a gritty and intense action movie that explores the complexities of heroism and villainy.
IMDb ID: tt1163338
3. The Bourne Ultimatum (2007) - This adrenaline-fueled spy thriller follows Jason Bourne (Matt Damon) as he uncovers a conspiracy linked to his mysterious past. The action sequences are heart-pumping and well-choreographed.
IMDb ID: tt0478393
Remember, these are just a few suggestions based on my training data. If you have any other specific preferences or criteria, feel free to let me know, and I'll do my best to assist you!
(Temps de réponse : 27.46 secondes)

Question 3: stop
Sortie du programme
```

FIGURE 12 – CHATBOT (llama 2)

4.2 GPT-TURBO-3.5

```
Réponse 1 : Based on your preferences for comedy movies, I would suggest you watch "Monty Python and the Holy Grail" (tt0071853). It is a classic comedy film that combines absurd humor, satire, and entertaining storytelling.
(Temps de réponse : 6.73 secondes)

Question 2: ok. I would like to see some action movies like iron man ?
Réponse 2 : Based on the description you provided, I would recommend the following action movies similar to Iron Man:
1. The Avengers (tt0848228) - This movie brings together a team of superheroes, including Iron Man, to save the world from a powerful enemy.
2. Captain America: The First Avenger (tt0458339) - This film follows the story of Steve Rogers, who becomes the superhero Captain America during World War II.
3. Thor (tt0800369) - In this movie, the powerful Norse god Thor is banished to Earth and must learn humility to regain his powers and save his realm.
4. Guardians of the Galaxy (tt2015383) - A group of misfits, including a human named Peter Quill, team up to protect the galaxy from a powerful artifact.
5. Black Panther (tt1825683) - This film follows the story of T'Challa, the newly crowned king of Wakanda, as he fights to protect his nation and its advanced technology.
I hope you find these recommendations enjoyable! Let me know if you have any other questions.
(Temps de réponse : 16.85 secondes)
```

FIGURE 13 – CHATBOT (GPT)

5 Limitations et Perspectives d'Améliorations

D'après les résultats obtenue, on constate que mon travail peut être amélioré dans 2 sens : premièrement la rapidité du temps d'exécution, car le chat prend quelques secondes avant de répondre mais je pense que cela est dû essentiellement à la taille du vectorstore à cause du dataset volumineux (46MB), ainsi à l'environnement d'exécution Colab, qui utilise le GPU gratuit qui n'est pas puissant. Deuxièmement dans l'interface du chatbot, en utilisant par exemple streamlit, mais ici en fait j'ai pensé à le faire mais après avoir testé l'exécution du script sur mon local, il était très lent vu qu'il n'utilise pas le gpu afin de répondre à la question et donc ça prend beaucoup plus du temps à répondre à l'utilisateur, pour cela j'ai décidé de travailler sur Colab qui ne support pas l'utilisation du streamlit chat.

6 Contribution des Membres du Groupe

Dans ce projet je suis seul dans le groupe, donc il s'agit d'un travail individuel dont j'ai réalisé toutes les étapes et le travail du projet.

7 Conclusion

Ce project m'a permet de découvrir certains LLMs open source les plus populaire dans le domaine NLP et aussi de savoir que les LLMs en général ne font pas le même types des tâches, tel que la génération du texte dans mon cas et d'autres llm comme BERT pour la classification du texte etc. Et aussi que les résultats obtenus par un LLM dépends des valeurs des paramètres qu'on lui attribue comme la température et la personalisaton de la prompt.