IUT Villetaneuse
Université Sorbonne Paris Nord
99 Av. Jean Baptiste Clément,
93430 Villetaneuse

HephIA
30 rue de Gramont,
75002 Paris

BUT 2 PASSERELLE TECHNOLOGIE DE L'INFORMATION

Mise en Place d'un workflow d'évaluation de RAG avec intégration d'interface

Encadrant : Mme. Linda El Alaoui
Maitre de stage : M. Mustapha Lebbah

Réalisé par :
Aminata THIOUNE

10 Juin 2024 - 02 Août 2024

Remerciements

Je tiens à exprimer ma gratitude envers l'entreprise HephIA pour m'avoir offert l'opportunité de réaliser mon stage sur un sujet aussi pertinent et stimulant dans le domaine de l'intelligence artificielle. Je remercie tout particulièrement M. Mustapha Lebbah, professeur à l'Université Paris-Saclay-UVSQ, cofondateur de l'entreprise et mon maître de stage, pour sa confiance et son encadrement. Son expertise et ses conseils ont été essentiels pour enrichir mon expérience et approfondir mes connaissances dans ce domaine stratégique. Je remercie également M. Anthony Coutant, cofondateur de l'entreprise, pour sa précieuse contribution tout au long de mon stage, ainsi que M. Kamel Mesbahi, ingénieur au sein de l'entreprise, pour son aide précieuse. Enfin, je souhaite exprimer ma reconnaissance envers les professeurs de l'UIT, en particulier Madame Linda El Alaoui, responsable de notre formation et mon tuteur durant cette période, pour leur soutien et les outils qu'ils m'ont fournis pour réussir ce stage.

Abstract

Large Language Models (LLMs) are widely used for their performance and accuracy in various fields. Our internship aimed to apply these models specifically to summarizing French legal texts. Since most current LLMs are trained on large, generic English corpora, they often have biases and may not be reliable for direct use in the legal field. To address this, we conducted a detailed review to select the best LLMs suited for French and trained on legal data, including French legislation. Once these models were selected, we developed a user interface to make them easier to use. In addition to summarizing texts, we added a "question-answering" function allowing users to ask questions directly to the LLMs. Aware of the limitations of these models' responses, we implemented a Retrieval-Augmented Generation (RAG) pipeline to minimize incorrect answers. This approach improves the reliability of responses by allowing LLMs to provide not only answers but also verifiable sources in the legal field.

Résumé

Les modèles de langage de grande taille (LLMs) sont largement utilisés pour leur performance et leur précision dans divers domaines. Notre stage visait à appliquer ces modèles spécifiquement au résumé de textes juridiques français. Étant donné que la plupart des LLMs actuels sont entraînés sur de vastes corpus génériques en anglais, ils présentent souvent des biais et ne sont pas toujours fiables pour une utilisation directe dans le domaine juridique. Pour résoudre ce problème, nous avons effectué une revue détaillée pour sélectionner les meilleurs LLMs adaptés à la langue française et formés sur des données juridiques incluant la législation française. Une fois ces modèles sélectionnés, nous avons développé une interface utilisateur pour faciliter leur utilisation. En complément du résumé de textes, nous avons intégré une fonction de "question-answering" permettant aux utilisateurs de poser directement des questions aux LLMs. Conscients des limitations des réponses fournies par ces modèles, nous avons implémenté une pipeline RAG ("Retrievial Augmented Generation") pour minimiser les réponses erronées. Cette approche renforce la fiabilité des réponses, en permettant aux LLMs de fournir non seulement des réponses mais aussi des sources vérifiables dans le domaine juridique.

Table des matières

Table des figures

Liste des tableaux

Introduction

L'intelligence artificielle (IA) permet aux machines d'accomplir des tâches nécessitant l'intelligence humaine, comme la reconnaissance de la parole, la prise de décision et la résolution de problèmes. Elle est couramment utilisée en traitement automatique du langage (TAL) (1) pour comprendre et analyser le langage humain, facilitant la traduction automatique, les assistants virtuels, l'analyse de sentiments et le résumé de textes. Les meilleurs modèles d'IA en TAL aujourd'hui sont les modèles de langage de grande taille (LLMs) ((2), (3) et (4)) . Ce sont des réseaux neuronaux entraînés sur d'énormes quantités de données textuelles pour prédire et générer du langage naturel. Ces modèles ont révolutionné le TAL grâce à leurs performances bien supérieures aux approches précédentes, offrant un traitement et une génération de texte plus cohérents.
Les LLMs, en dehors du TAL, peuvent être appliqués dans des domaines variés tels que les systèmes de questions-réponses, la reconnaissance d'entités nommées et les systèmes multimodaux. Malgré leurs performances, ils présentent deux problèmes majeurs : 1) des biais liés aux données d'entraînement, ce qui les limite à exceller dans des domaines spécifiques, et 2) l'hallucination, où les LLMs génèrent des réponses hors contexte ou incorrectes.
Dans le cadre de notre stage sur le résumé de textes juridiques en français, l'utilisation des LLMs présente plusieurs défis, notamment ceux liés à la langue. En effet, la plupart des LLMs sont mieux adaptés à l'anglais, ayant été principalement entraînés sur des corpus où l'anglais prédomine. Notre travail se structure en trois étapes fondamentales :

1. **Revue des LLMs orientés français :** effectuer une analyse approfondie des LLMs spécialisés dans le français et entraînés sur des corpus juridiques français.
2. **Développement d'une interface utilisateur :** créer une interface permettant de communiquer directement avec ces LLMs pour résumer des textes juridiques et poser des questions juridiques.
3. **Intégration de la méthode RAG :** le "Retrieval-Augmented Generation" (RAG) est une technique où les LLMs utilisent une base de données fournie pour générer des réponses. Cela permet de réduire les hallucinations en fournissant des sources vérifiables. L'utilisateur peut configurer les différentes approches RAG pour une expérience optimale.

Ces étapes permettent de surmonter les défis linguistiques et de précision liés à l'utilisation des LLMs pour le traitement de textes juridiques en français, tout en améliorant la fiabilité des réponses fournies.
En vue de rendre compte de manière fidèle et analytique de mon expérience au cours des deux mois au sein de l'entreprise HephIA, ce rapport adoptera une structure en trois parties :

1. **Présentation de l'entreprise :** cette partie mettra en avant les activités de HephIA.
2. **Missions du service :** nous examinerons les objectifs stratégiques et le rôle du service R&D (Recherche et Développement) auquel j'ai été affectée au sein de l'organisation.
3. **Méthodologie et Résultat :** cette partie détaillera les tâches que j'ai eu l'opportunité d'accomplir, soulignant les contributions que cette expérience m'a permis d'apporter.

Cette structure permettra de présenter de manière claire et détaillée mon parcours et les apprentissages réalisés au sein de HephIA.

1 Présentation de l'entreprise

HephIA est une start-up innovante spécialisée dans l'intégration de l'intelligence artificielle (IA), en particulier les technologies d'IA générative, dans les processus métiers. Fondée par des visionnaires en technologie et en entrepreneuriat, l'entreprise vise à transformer les industries en automatisant et en optimisant les tâches complexes grâce à l'intelligence artificielle.

L'intelligence artificielle (IA) fait référence à la simulation des processus d'intelligence humaine par des machines, en particulier des systèmes informatiques. Ces processus incluent l'apprentissage (acquisition d'informations et règles d'utilisation), le raisonnement (utilisation des règles pour atteindre des conclusions approximatives ou définies) et l'autocorrection. Les technologies d'IA générative, comme les modèles de langage à grande échelle (LLMs), sont capables de créer de

nouvelles données ou de prédire des séquences de texte en se basant sur d'énormes volumes de données d'entraînement.

HephIA utilise des modèles de langage à grande échelle (LLMs) et des technologies d'apprentissage profond pour créer des systèmes qui apprennent et s'adaptent aux besoins des utilisateurs. Les LLMs sont des modèles de traitement du langage naturel (NLP) qui ont été formés sur des corpus de texte extrêmement vastes, leur permettant de comprendre et de générer du texte de manière cohérente et pertinente. Ces modèles utilisent des architectures avancées de réseaux de neurones, comme les transformateurs, pour capturer les relations contextuelles entre les mots et les phrases dans un texte.

En intégrant ces technologies dans des workflows existants, HephIA aide les professionnels à surmonter les défis liés à la gestion des informations et à l'automatisation des tâches répétitives. Cela inclut des applications dans divers secteurs tels que le droit, la finance, la santé et bien d'autres. Par exemple, dans le secteur financier, les LLMs peuvent analyser des rapports financiers et générer des résumés concis, ou encore détecter des anomalies dans les transactions pour prévenir la fraude. Dans le secteur de la santé, l'IA peut aider à diagnostiquer des maladies en analysant des dossiers médicaux et en comparant les symptômes avec des bases de données médicales.

SafeSphere, le produit phare d'HephIA, est une plateforme qui offre des assistants IA personnalisés pour les secteurs nécessitant une expertise métier pointue, comme le droit. Pour les avocats spécialisés dans le contentieux, SafeSphere propose des solutions qui simplifient et accélèrent le traitement des dossiers, la rédaction de documents juridiques et l'élaboration de stratégies, tout en garantissant la sécurité et la confidentialité des données. SafeSphere utilise des techniques avancées de traitement du langage naturel pour analyser des documents juridiques, extraire les informations pertinentes et les présenter de manière organisée et accessible.

2 Missions du service R&D

La mission d'HephIA est de rendre l'intelligence artificielle accessible et utile dans des contextes métier spécifiques, en améliorant ainsi l'efficacité et la qualité du travail professionnel. HephIA envisage un monde où l'IA devient une extension naturelle des capacités humaines, permettant une prise de décision plus rapide et plus précise dans divers secteurs tels que la finance, la santé, le droit, et bien d'autres.

En fournissant des outils sophistiqués qui améliorent la productivité et la précision, HephIA joue un rôle important dans la transformation des pratiques professionnelles. Ces outils permettent aux professionnels de se concentrer sur des tâches à plus forte valeur ajoutée, tout en automatisant les processus répétitifs et en fournissant des analyses approfondies. Par exemple, dans le secteur juridique, les modèles de traitement du langage naturel d'HephIA peuvent analyser des milliers de documents en un temps record, réduisant ainsi le risque d'erreurs humaines et accélérant le processus de recherche juridique.

L'entreprise est constamment à la recherche d'innovations pour étendre les capacités de sa plateforme SafeSphere, une solution intégrée qui offre des fonctionnalités avancées de traitement de données et d'assistance intelligente. SafeSphere est conçue pour être adaptable, évolutive et capable de répondre aux besoins changeants des utilisateurs. Elle inclut des fonctionnalités telles que la reconnaissance vocale, l'analyse prédictive et la génération automatique de rapports, toutes conçues pour faciliter le travail quotidien des professionnels.

HephIA explore activement de nouvelles applications pour d'autres marchés susceptibles de bénéficier de l'assistance intelligente. Par exemple, dans le secteur de la santé, l'IA pourrait être utilisée pour analyser les dossiers médicaux des patients, suggérer des diagnostics potentiels et recommander des plans de traitement personnalisés. Dans l'industrie financière, elle pourrait surveiller les transactions pour détecter les fraudes et fournir des conseils d'investissement basés sur des analyses de marché en temps réel.

En outre, HephIA s'engage à développer des solutions d'IA qui non seulement répondent aux

besoins actuels, mais anticipent également les défis futurs. L'entreprise investit dans la recherche et le développement pour créer des technologies de pointe qui peuvent être intégrées de manière transparente dans les infrastructures existantes des entreprises. Cette approche garantit que les solutions d'HephIA restent pertinentes et efficaces à mesure que les technologies évoluent et que les exigences du marché changent.

Grâce à ses efforts, HephIA contribue à un progrès continu et durable dans le monde professionnel. L'entreprise croit fermement que l'IA peut transformer les entreprises en leur offrant les outils nécessaires pour innover et se développer. En fin de compte, HephIA vise à créer un environnement où l'IA est utilisée de manière éthique et responsable, améliorant la qualité de vie des travailleurs et offrant de nouvelles opportunités de croissance et de développement.

3 Méthodologie et Résultat

Dans cette section, nous examinerons les stratégies mises en place pour atteindre les résultats souhaités. Elle est divisée en trois parties : 1) Nous discuterons des approches utilisées pour sélectionner les meilleurs modèles de langage (LLMs) dans le domaine juridique en français (Section 3.1). 2) Nous décrirons la stratégie pour développer une interface utilisateur optimisée pour interagir avec ces LLMs dans le cadre de résumé de textes juridiques (Section 3.2). 3) Enfin, nous aborderons l'approche RAG appliquée à ces LLMs pour améliorer la fiabilité des réponses et résoudre les problèmes d'hallucination dans les tâches de "Question-Answering" (Section 3.3).

3.1 Sélection des Meilleurs LLMs pour le Domaine Juridique en Français

Pour sélectionner les meilleurs modèles de langage (LLMs), un processus en deux étapes a été suivi. D'abord, les LLMs les plus avancés disponibles ont été identifiés. Ensuite, trois méthodes d'évaluation ont été appliquées : 1) les métriques standard, basées sur des heuristiques et des calculs mathématiques ; 2) l'évaluation par un LLM plus robuste, appelé "LLM judge" et 3) l'évaluation humaine.

L'étude des meilleurs LLMs s'appuie sur les recherches récentes ((5), (6) et (7)) et les "benchmarks" [1] de modèles open-source sur "Hugging Face" [2]. L'article "LawBench : Benchmarking Legal Knowledge of Large Language Models" (5) compare des LLMs multilingues, orientés chinois et spécifiques au droit dans le domaine juridique avec des jeux de données tels que JEC_QA (8) et LEVEN (9). La figure 1 montre que les LLMs multilingues surpassent nettement les autres.

Pour les "benchmarks" sur "Hugging Face", des LLMs comme Mistral (10), LLama (11), SauLM (12) et AdaptLLM (13) se distinguent. Ces évaluations nous ont permis de sélectionner une liste des meilleurs LLMs :

- GPT-4 (14)
- Mistral (10)
- LLama-3 (11)
- SauLM (12)
- AdaptLLM (13)
- RoBerta-BART-Fixed (15)
- RoBerta-BART-Dependent (16)
- LongFormer-BART (17)
- T5-EUR (18)

RoBerta-BART-Fixed, RoBerta-BART-Dependent, LongFormer-BART et T5-EUR ont été spécifiquement fine-tunés pour des tâches de résumé sur EUR-lex-sum (19), un jeu de données juridiques français. Pour restreindre cette liste de LLMs, nous avons appliqué une stratégie d'évaluation en utilisant le

1. un benchmark est un ensemble de tests standardisés utilisés pour évaluer et comparer les performances de systèmes ou de modèles selon des critères précis.

2. Hugging Face est une plateforme open source qui fournit des modèles et des outils pour le traitement du langage naturel.

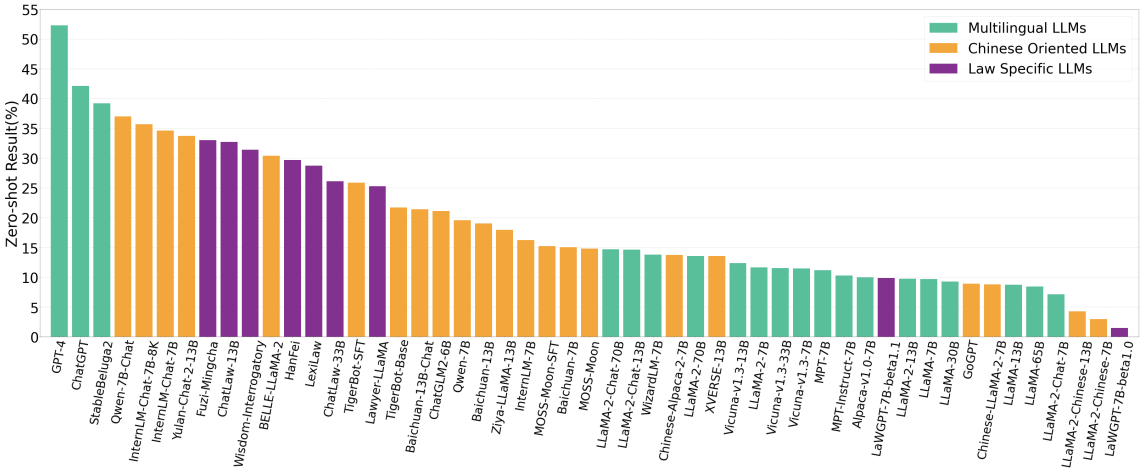


FIGURE 1 – Performance moyenne (zero-shot) de 51 LLMs évalués sur LawBench

jeu de données Long-form Legal Question Answering (LLeQA) (20), qui est une collection d'articles de la législation française et belge.

3.1.1 Validation à partir de métriques standards

Une des méthodes les plus couramment utilisées pour comparer les modèles est l'application de métriques standardisées. Ces métriques sont variées et chacune fournit une évaluation précise des performances. Pour évaluer les modèles mentionnés ci-dessus, nous avons utilisé plusieurs métriques, comme illustré dans la Table 1.

Métrique	**Définition**
Perplexity	Mesure l'incertitude d'un modèle.
Precision	Proportion de résultats pertinents parmi ceux retournés par le modèle.
Recall	Proportion de résultats pertinents correctement identifiés par le modèle parmi tous les résultats pertinents disponibles.
F1 Score	Moyenne harmonique de la précision et du rappel (Recall), fournissant une mesure équilibrée de performance.
ROUGE	Métrique de comparaison de texte incluant plusieurs versions : ROUGE-1 (chevauchement des unigrams), ROUGE-2 (chevauchement des bigrams), ROUGE-L (plus longue sous-séquence commune), ROUGE-Lsum (version de ROUGE-L pour les résumés).
SacreBLEU	Évalue la qualité des traductions automatiques en comparant les traductions générées avec des traductions de référence, en utilisant des scores de précision basés sur des n-grammes.

TABLE 1 – Définitions des métriques utilisées pour l'évaluation de performance

Les résultats présentés dans la Table 2 indiquent que le modèle GPT-4 surpasse généralement les autres modèles, sauf en termes de perplexité où il n'est pas le meilleur. Les modèles Mistral et LLama3 affichent des performances très proches de celles de GPT-4. Les modèles spécialisés dans le domaine juridique, tels que AdaptLLM et SaulM, se rapprochent également des performances des modèles multilingues. En revanche, les modèles dédiés au résumé de textes juridiques, comme RoBERTa-BART-Fixed, RoBERTa-BART-Dependent, Longformer-BART et T5-EUR, occupent les dernières positions, bien que T5-EUR se distingue légèrement parmi eux.
Ces métriques de validation, bien que performantes, se basent uniquement sur des calculs et ne prennent pas directement en compte le contexte et la sémantique des phrases. Elles ne reflètent pas entièrement l'évaluation humaine. Pour pallier ce problème, d'autres approches peuvent compléter

Modèle	Perplexity	Precision	Recall	F1	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum	SacreBLEU
GPT-4	64.52	**0.733**	**0.774**	**0.753**	**0.419**	**0.191**	**0.308**	**0.307**	**11.054**
Mistral	28.98	0.651	0.755	0.699	0.232	0.094	0.153	0.169	3.338
LLama3	30.11	0.693	0.734	0.712	0.251	0.090	0.164	0.181	4.471
SaulM	22.44	0.634	0.665	0.648	0.136	0.031	0.086	0.093	0.963
AdaptLLm	**19.50**	0.685	0.685	0.684	0.090	0.027	0.062	0.063	1.194
RoBERTa-BART-Fixed	44.17	0.651	0.642	0.647	0.134	0.010	0.062	0.082	0.471
Roberta-BART-Dependent	34.91	0.639	0.643	0.641	0.018	0.000	0.018	0.018	1.219
Longformer-BART	59.25	0.659	0.651	0.655	0.155	0.049	0.077	0.097	1.656
T5-EUR	111.09	0.685	0.677	0.681	0.282	0.062	0.132	0.132	1.955

TABLE 2 – Performances des modèles suivant les métriques standards

ces métriques : la validation par un LLM ou par des experts humains.

3.1.2 Validation par un LLM

Plutôt que de définir un ensemble strict d'heuristiques, on peut utiliser un "LLM Judge", un modèle robuste qui évalue objectivement les sorties des modèles selon des critères de qualité prédéfinis. Ce "LLM Judge" compare les réponses des modèles à des réponses de référence en tenant compte du contexte et de la sémantique, en analysant la pertinence, la cohérence et la fluidité des réponses. Il attribue ensuite des scores pour mesurer leur performance et leur fiabilité. Des outils comme DeepEval, un framework open source basé sur un "LLM Judge", offrent diverses métriques d'évaluation. Cependant, cette approche n'est pas pertinente pour notre sélection de modèles, car DeepEval se base sur GPT-4, qui fait partie de notre sélection.
Le principal problème de cette méthode est de trouver un bon "LLM Judge", souvent spécialiste du domaine cible. Les modèles d'OpenAI comme GPT-4, souvent les plus performants, ne sont pas open source, limitant ainsi cette technique. Mistral, qui se distingue après GPT-4 avec la première stratégie de validation (validation avec métriques standards) et étant un LLM open-source, a été utilisé comme "LLM Judge" pour évaluer les autres LLMs. Les résultats obtenus sont presque identiques en termes de classement à ceux observés dans la Table 2.

3.1.3 Validation humaine

La validation humaine consiste à faire évaluer les réponses des LLMs par des évaluateurs humains. Nous avons utilisé un échantillon du jeu de données LleQA pour tester la tâche de question-réponse. Parmi les modèles testés, seuls GPT-4 et Mistral ont répondu en français tout en fournissant des références légales, telles que des lois et des articles correspondants. Le modèle Llama, quant à lui, ne répondait pas aux questions. En essayant la version "Instruct" et en ajustant plusieurs fois le prompt, certaines réponses sont restées en anglais malgré des questions posées en français. Les modèles spécialisés dans le domaine juridique, comme SaulM, ont produit des réponses mixtes en anglais et en français, mais l'anglais était moins présent comparé à AdaptLLM, où la majorité des réponses étaient en anglais.
Pour la tâche de résumé, GPT-4, Mistral et SaulM ont produit des résumés concis et fidèles au texte de référence. En revanche, AdaptLLM et Llama n'ont pas montré de différences significatives en termes de longueur entre le texte de référence et le résumé. Les modèles axés sur le résumé utilisent une méthode appelée "Summarizer", qui sélectionne les phrases pertinentes à résumer, expliquant ainsi la concision et la fidélité de leurs résumés. La performance moyenne de tous les LLMs est illustrée dans la Figure 2.

En nous basant sur les résultats des trois méthodes de validation, nous avons sélectionné des modèles open-source aux performances proches de GPT-4. Parmi les modèles multilingues, nous avons retenu Mistral, et parmi ceux spécialisés en droit, nous avons choisi SaulM. Pour les modèles

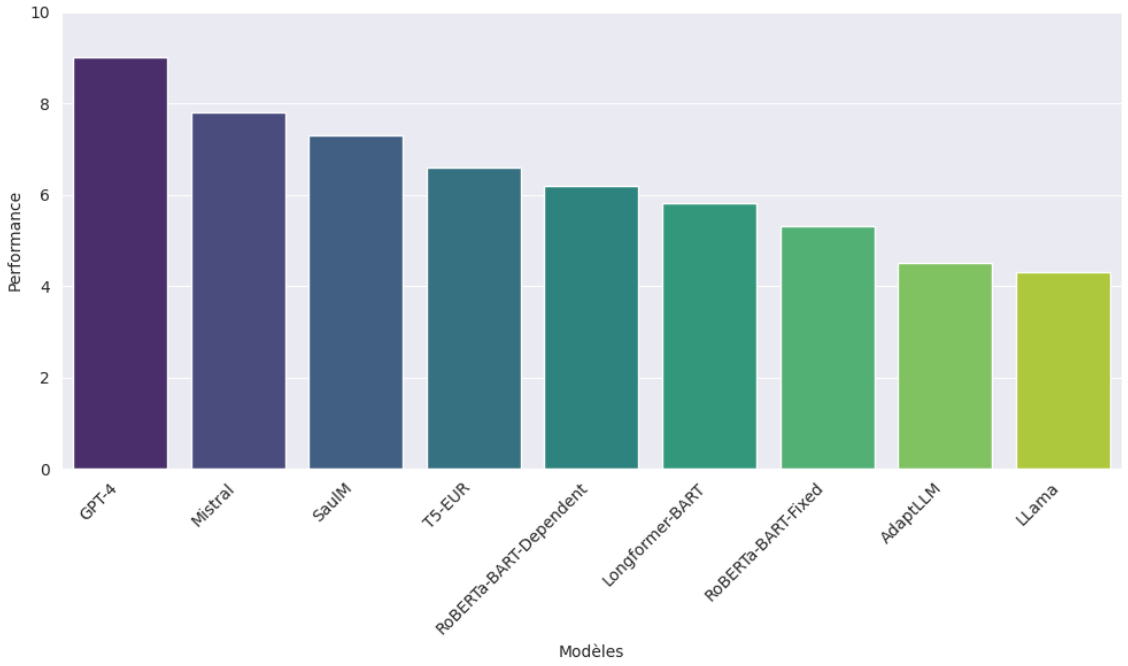


FIGURE 2 – Évaluation humaine des performances des LLMs sur une échelle de 0 à 10

spécialisés dans les résumés, nous avons sélectionné tous les quatre : RoBERTa-BART-Fixed, RoBERTa-BART-Dependent, Longformer-BART et T5-EUR.

3.2 Développement d'une Interface Utilisateur pour le résumé de Textes Juridiques

Après avoir sélectionné les meilleurs LLMs (Mistral, SaulM, RoBERTa-BART-Fixed, RoBERTa-BART-Dependent, Longformer-BART et T5-EUR), une interface utilisateur pour le résumé de textes juridiques a été développée. Cette interface facilite l'accès et l'utilisation des outils de résumé. Le choix s'est porté sur Streamlit, un framework Python open source, pour les raisons suivantes :

- **Simplicité :** streamlit permet de créer un code simple et facile à lire.
- **Prototypage rapide et interactif :** il facilite la création d'applications interactives où les utilisateurs peuvent interagir avec les données et fournir des retours rapidement.
- **Édition en temps réel :** les mises à jour de l'application sont visibles instantanément lors des modifications du script.

Pour interagir avec les modèles, deux approches ont été utilisées : LangChain (21) et la bibliothèque transformers de "Hugging Face". LangChain est une API open source qui simplifie la création d'applications basées sur des LLMs. Elle offre une interface facile à utiliser, réduisant la complexité du code nécessaire pour gérer les requêtes, organiser les conversations et interpréter les réponses.
LangChain facilite aussi l'intégration des modèles dans diverses applications, comme les chatbots ou les systèmes de résumé de texte. Toutefois, LangChain ne prend pas en charge les modèles de plus de 10 Go. C'est pourquoi une autre approche a été envisagée : l'utilisation de la bibliothèque transformers de "Hugging Face".
La bibliothèque transformers de "Hugging Face" est un outil puissant pour le traitement du langage naturel (NLP). Elle propose une gamme de modèles pré-entraînés basés sur l'architecture Transformer (22), adaptés à des tâches comme le résumé de texte, la traduction, la génération de texte et la classification. Cette bibliothèque facilite l'utilisation des LLMs en offrant des interfaces simples pour les charger, les adapter à des tâches spécifiques et les intégrer dans des applications, même lorsque les modèles sont très volumineux, dépassant 10 Go.

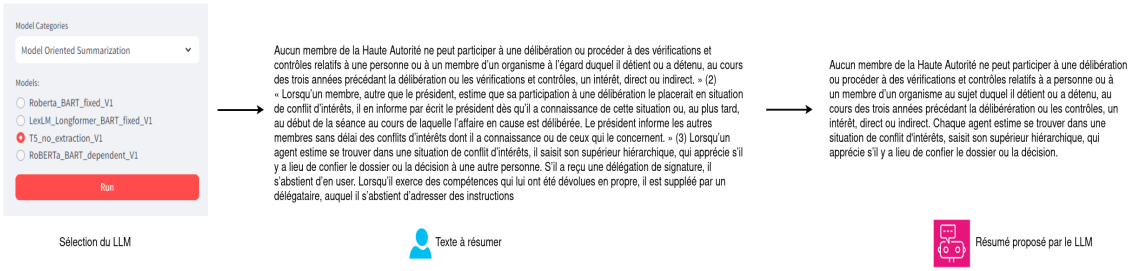


FIGURE 3 – Exemple de l'utilisation de l'interface pour résumer un texte.

Cependant, en raison de leur complexité et de leurs besoins élevés en calcul, les LLMs nécessitent souvent des techniques comme la "quantization". La "quantization" réduit le nombre de bits utilisés pour représenter les paramètres du modèle, ce qui diminue les besoins en mémoire et en calcul, rendant ainsi le modèle plus rapide, notamment sur des dispositifs avec des ressources limitées. La figure 3 illustre l'utilisation de l'application pour résumer un texte. Dans cet exemple, le modèle T5-EUR a été sélectionné, puis le texte fourni par l'utilisateur a été résumé, en utilisant moins de mots tout en conservant le sens original.

Conscients que les besoins des utilisateurs ne se limitent pas seulement aux résumés de textes juridiques, certains utilisateurs posent directement des questions aux LLMs, ce qui relève des tâches de "question-réponse". Cependant, les LLMs peuvent parfois fournir des réponses hors contexte ou incorrectes, basées sur leur connaissance générale. Pour résoudre ces problèmes, nous avons étendu l'application en intégrant une approche appelée RAG, que nous détaillerons dans la section suivante.

3.3 Application de l'Approche RAG pour Améliorer la Fiabilité des Réponses et Résoudre les Problèmes d'Hallucination

Dans cette section, nous allons explorer une méthode pour améliorer la qualité des réponses des LLMs : le RAG. Cette technique optimise les résultats des LLMs en ajoutant une base de connaissances fiable et externe aux données initiales d'entraînement. Elle permet d'étendre les capacités déjà puissantes des LLMs à des domaines spécifiques ou à des bases de connaissances internes d'une organisation, sans nécessiter de réentraînement du modèle. L'intégration du RAG dans un système de réponse basé sur LLM présente deux avantages principaux : elle assure l'accès aux informations les plus récentes et fiables, et permet aux utilisateurs de vérifier les sources du modèle, garantissant ainsi la précision et la fiabilité des réponses.

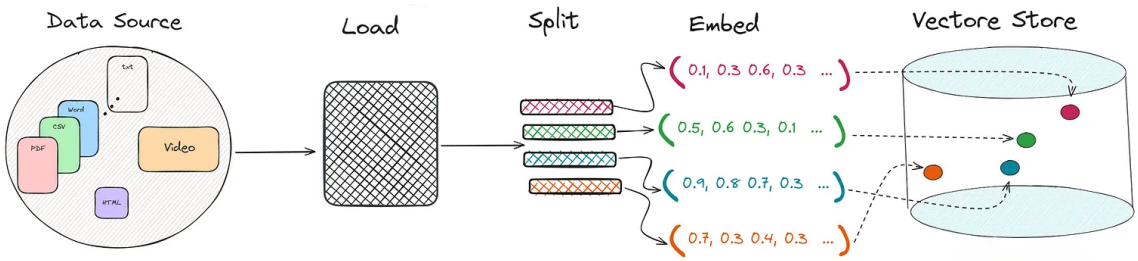


FIGURE 4 – Mise en place du système RAG

Comme illustré dans la figure 4, la mise en place d'un système RAG se compose de quatre étapes principales : chargement des documents ("Load"), découpage des documents en segments appelés "chunks" ("Split"), encodage de ces chunks en représentations vectorielles ("Embed"), et enregistrement de ces vecteurs dans une base de données ("Vectore Store").

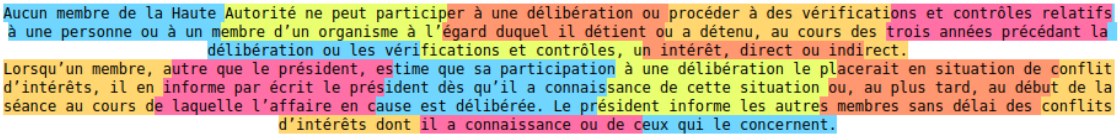
3.3.1 Chargement des documents

La première étape pour mettre en place un système RAG est de référencer tous les documents qui doivent être utilisés comme base de connaissances. Ces documents peuvent être de natures diverses : fichiers textes ou Word, présentations, PDF, pages Web ou même des transcriptions de vidéos YouTube. Les documents volumineux ne peuvent pas être chargés directement en mémoire. Ils sont donc découpés en segments cohérents, tels que des paragraphes, pour faciliter leur gestion. Dans notre processus d'implémentation, nous avons travaillé exclusivement avec des fichiers. Lorsque la taille des fichiers le permettait, nous les chargions entièrement en mémoire. Sinon, le découpage optimal était réalisé au niveau des paragraphes.

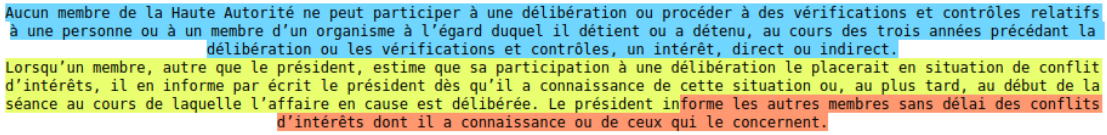
3.3.2 Découpage des documents en segments ("chunks")

Après avoir chargé nos documents, nous passerons à l'étape de découpage en segments appelés "chunks". Cette étape est nécessaire car les données doivent être encodées par un modèle en tenant compte du "context_window", qui correspond à la quantité de texte que le modèle peut traiter à la fois. Il existe plusieurs méthodes pour diviser les documents en "chunks", et le choix de la méthode influence la qualité des résultats. Parmi ces méthodes, nous pouvons citer :

— **CharacterTextSplitter** divise le texte en segments plus petits en se basant sur les caractères, offrant une méthode simple et directe. Lors de l'initialisation, elle définit des paramètres comme la taille maximale des segments ("chunk size"), les chevauchements entre les segments ("chunk overlap"), et les critères de séparation. Elle analyse ensuite le texte caractère par caractère, identifie les séparateurs (espaces, points, etc.), et crée des segments en coupant le texte à ces séparateurs ou à la longueur maximale. Les segments créés sont ensuite collectés et retournés. Comme illustré sur la figure 5, la pertinence des résultats dépend fortement de la taille des segments définie.



(a) Chunk size = 25

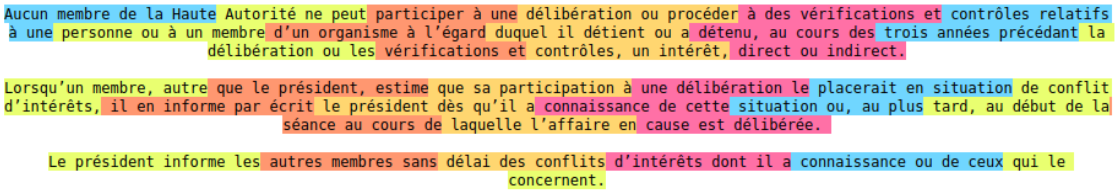


(b) Chunk size = 330

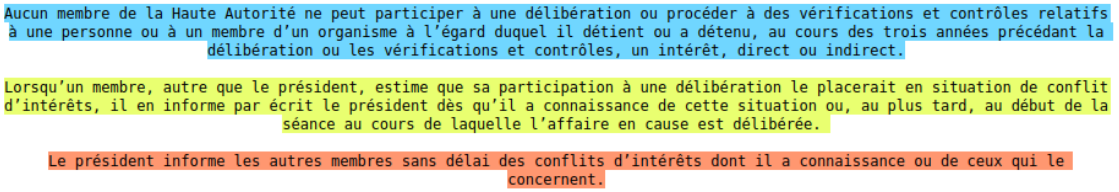
FIGURE 5 – Illustration des segments créés par CharacterTextSplitter en fonction de la taille des segments ("chunk size") sur un seul paragraphe. Chaque couleur correspond à un segment.

— **RecursiveCharacterTextSplitter**, contrairement à CharacterTextSplitter, découpe le texte de manière récursive en utilisant une liste de séparateurs. Lors de l'initialisation, elle définit des paramètres comme la taille maximale des segments ("chunk size") et les chevauchements entre les segments ("chunk overlap"). Ensuite, elle parcourt le texte en recherchant ces séparateurs pour le diviser en segments. Si un segment dépasse la taille maximale, il est à nouveau découpé en utilisant les mêmes séparateurs, et ce processus se poursuit récursivement jusqu'à ce que tous les segments respectent la taille maximale. Les segments ainsi obtenus sont collectés et retournés sous forme de liste.

  Cette approche, illustrée par la Figure 6, est plus efficace que la première, mais elle présente des limites si la taille des segments ne correspond pas de manière concise à la structure du texte à découper.



(a) Chunk size = 25



(b) Chunk size = 330

FIGURE 6 – Illustration des segments créés par RecursiveCharacterTextSplitter en fonction de la taille des segments ("chunk size") sur trois paragraphes. Chaque couleur correspond à un segment.

— **SemanticChunker** divise un texte en unités de sens, telles que des phrases ou des paragraphes, en se basant sur la sémantique. Elle utilise des techniques NLP pour analyser le texte et identifier les structures importantes. Ensuite, elle segmente le texte en respectant ces unités de sens, en utilisant un paramètre appelé "breakpoint threshold type" (type de seuil de rupture) pour déterminer les points de division. Ce paramètre évalue la similarité entre les blocs de texte à l'aide de critères comme le "percentile", la "standard deviation", le "gradient" ou l' "interquartile". Cela évite de couper les phrases ou les paragraphes de manière arbitraire. Un modèle NLP affine cette segmentation, et les segments obtenus sont ensuite regroupés et restitués.

— **AI21SemanticTextSplitter** découpe un texte en segments plus petits en utilisant des techniques d'intelligence artificielle et de NLP d'AI21 Labs. Contrairement aux méthodes simples qui se basent sur des séparateurs fixes, cette fonction comprend la structure et le sens du texte pour créer des segments cohérents. Elle commence par analyser le texte avec des modèles NLP sophistiqués, identifie les unités sémantiques comme les phrases et les paragraphes, et segmente le texte en respectant ces unités pour maintenir la cohérence. Les modèles d'AI21 Labs, sont utilisés pour assurer une segmentation précise. Enfin, les segments sont regroupés en une liste pour être retournés.

— **NLTKTextSplitter** divise un texte en segments gérables en utilisant des techniques de traitement de texte et de segmentation linguistique avec la bibliothèque NLTK (23) (Natural Language Toolkit). Il commence par "tokenizer" le texte, c'est-à-dire le diviser en unités de base comme des mots ou des phrases, en utilisant des délimiteurs tels que les points. Ensuite, il segmente le texte en morceaux adaptés tout en respectant les limites de longueur et en préservant la cohérence des structures importantes comme les phrases et les paragraphes. NLTKTextSplitter utilise aussi des techniques linguistiques avancées et des heuristiques, comme la ponctuation, pour éviter de couper des entités importantes ou des constructions syntaxiques complexes.

— **SpacyTextSplitter** utilise la bibliothèque spaCy (24) pour segmenter le texte. Il commence par tokeniser le texte et effectuer une analyse linguistique avec un modèle adapté, permettant de diviser le texte en unités de base comme les mots et les phrases, et d'identifier les entités nommées et les structures syntaxiques. SpacyTextSplitter découpe le texte en phrases en détectant les signes de ponctuation et les indices syntaxiques qui marquent la fin des phrases. Il respecte les limites de longueur spécifiées en regroupant les phrases en segments adaptés sans dépasser ces limites, tout en prenant en compte les spécificités linguistiques et les exceptions lexicales. Enfin, il préserve la cohérence contextuelle en évitant de couper les phrases ou les entités complexes, en se basant sur les informations syntaxiques

et sémantiques fournies par spaCy.

La segmentation des textes est essentielle, chaque méthode ayant ses avantages et inconvénients selon la structure du texte. Cette segmentation est nécessaire pour l'étape d'encodage, où des modèles sont appliqués. Ces modèles requièrent un nombre précis de mots en entrée. Si le texte dépasse ce nombre, il y a un risque significatif de perte d'information.

3.3.3 Encodage des segments ("chunks") en une représentation vectorielle ("Embeds")

Après avoir segmenté le texte, chaque segment est encodé en une représentation vectorielle ("embedding"). Cela facilite le traitement et l'analyse des segments, notamment pour mesurer leur similarité avec les questions des utilisateurs. Plusieurs modèles peuvent réaliser cette tâche : des modèles multilingues comme BGE-M3 (25) et Multilingual E5 (26), ainsi que des modèles spécialisés en français tels que Sentence-CamemBERT (27; 28), FlauBERT (29) et BARTHez (30). Ces modèles transforment chaque segment en une représentation vectorielle de dimensions variées.
Une fois tous les segments encodés, ces vecteurs sont stockés dans une base de données spécialisée (voir section suivante pour plus de détails) pour un traitement ultérieur.

3.3.4 Enregistrement des "Embeds" dans une base de données

Dans un système RAG, les vecteurs enregistrés dans une base de données facilitent la récupération rapide d'informations pertinentes pour une requête. Cela améliore la qualité des réponses en utilisant la similarité sémantique. Dans ce travail nous avons utilisé FAISS (31) (Facebook AI Similarity Search), une bibliothèque développée par Facebook AI Research qui permet de stocker et de rechercher efficacement ces vecteurs en se basant sur des mesures de similarité ou de distance.

Algorithm 1 Construction du VectorStore pour RAG avec FAISS

Require: Documents, LLM, Embedder, Splitter
Ensure: VectorStore
Initialisation :
Créer une liste vide *VectorStore*
Initialiser FAISS index
Étape 1 : Prétraitement des Documents
for chaque document *doc* dans *Documents* **do**
  Utiliser le *Splitter* pour diviser *doc* en segments
  for chaque segment *seg* dans *segments* **do**
    Utiliser l'*Embedder* pour obtenir l'embedding du segment *seg*
    Ajouter l'embedding au *VectorStore*
  end for
end for
Étape 2 : Construction du FAISS Index
Convertir *VectorStore* en matrice de vecteurs
Ajouter les vecteurs à l'index FAISS
Entraîner l'index FAISS pour l'optimisation des requêtes
Retourner l'index FAISS comme *VectorStore*

L'algorithme 1 décrit le processus de création d'un VectorStore pour RAG utilisant FAISS. Cela permet de combiner efficacement la récupération et la génération de texte.

Algorithm 2 Répondre aux Questions avec RAG

Require: Prompt (Question), VectorStore, Embedder, LLM
Ensure: Réponse
1: **Étape 1 : Recherche du Contexte**
2: Utiliser l'*Embedder* pour obtenir l'embedding du *Prompt*
3: Chercher les segments similaires dans le *VectorStore* en utilisant FAISS
4: Sélectionner les segments les plus pertinents comme *Contexte*
5: **Étape 2 : Génération de la Réponse**
6: Combiner le *Prompt* et le *Contexte*
7: Fournir cette combinaison au *LLM*
8: Obtenir la réponse générée par le *LLM*
9: **Retourner** la réponse générée par le *LLM*

Lorsqu'une question est posée, l'embedding correspondant est cherché dans le *VectorStore* pour trouver les segments les plus similaires. Ces segments sont ensuite combinés avec la question et fournis au *LLM* pour générer une réponse (ref. Algorithme 2).

En résumé, notre approche implique d'abord la sélection des meilleurs modèles de langage (LLMs) pour le résumé de textes juridiques. Ensuite, nous avons développé une interface utilisateur pour faciliter l'interaction avec ces LLMs et intégré une méthode permettant d'étendre les résumés à des fonctionnalités de question-réponse plus fiables. Cela nous a permis de créer un processus complet et riche, offrant la possibilité de faire des comparaisons pertinentes à chaque étape de la chaîne, pour une utilisation optimisée de la solution proposée.

4 Présentation et Guide d'utilisation de l'application

L'application développée peut être utilisée pour deux tâches : le résumé de textes juridiques en français et les questions-réponses.

Pour la tâche de résumé de textes juridiques, tous les LLMs peuvent être utilisés, y compris les modèles multilingues comme Mistral et SaulM, ainsi que les modèles spécialisés dans le résumé, tels que RoBERTa-BART-Fixed, RoBERTa-BART-Dependent, Longformer-BART, et T5-EUR. Pour les modèles multilingues, il est nécessaire de préciser la tâche de résumé dans le prompt, par

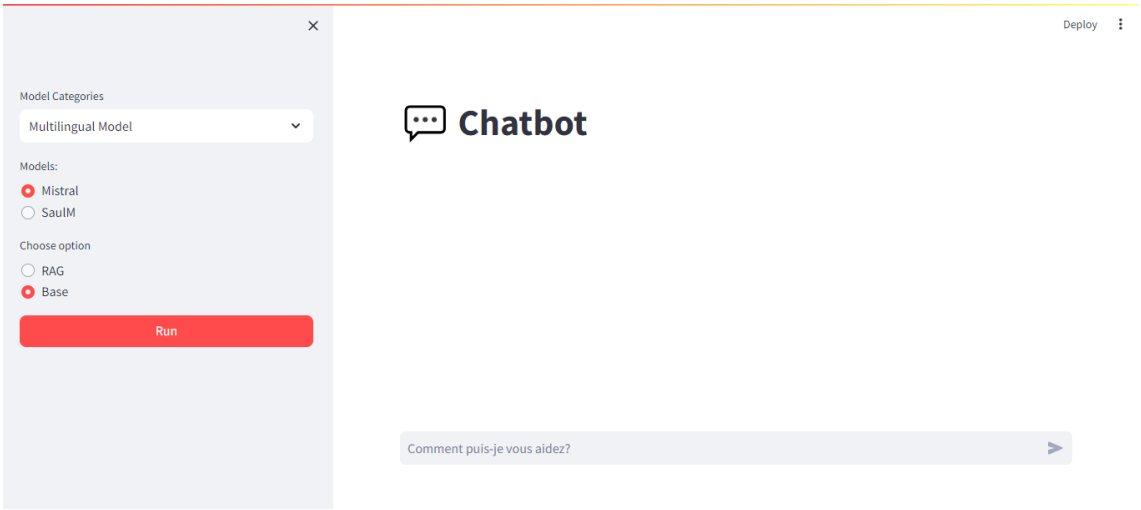


FIGURE 7 – Séléction de modèles multilingues pour la tâche de résumé

exemple en commençant le texte par "Résume ce texte", car ces modèles sont généraux et peuvent

être appliqués à diverses tâches. La figure 7 illustre l'utilisation des deux modèles multilingues pour le résumé de textes juridiques. En ce qui concerne les modèles spécialisés dans le résumé, il n'est

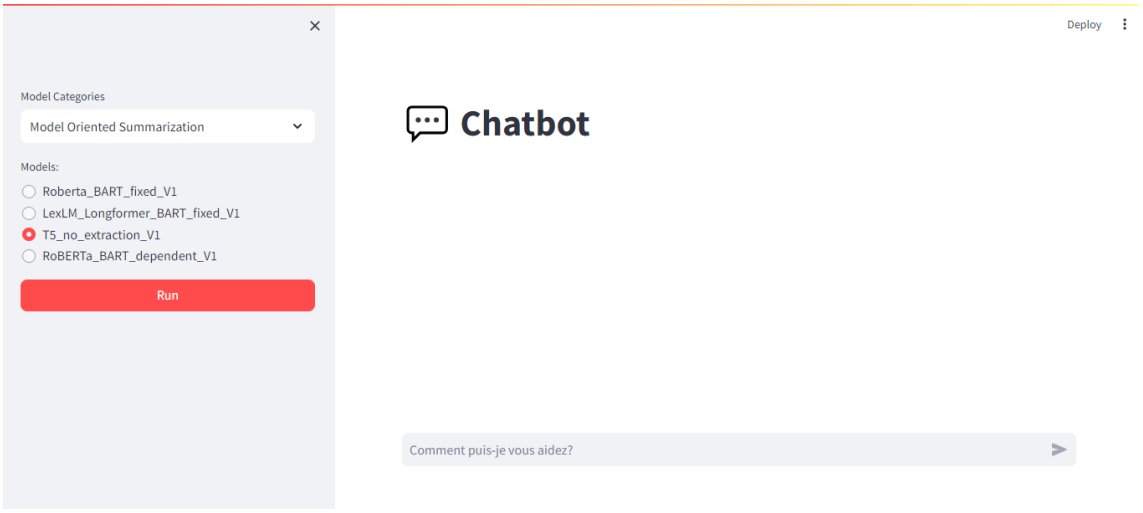


FIGURE 8 – Sélection de modèle orienté résumé pour la tâche de résumé

pas nécessaire de spécifier la tâche dans le prompt, car ces modèles sont exclusivement conçus pour le résumé. La figure 8 montre comment sélectionner ces modèles et interagir avec eux via le texte d'entrée dans la barre de prompt.

Pour la tâche de questions-réponses, seuls les modèles multilingues (Mistral et SaulM) peuvent être utilisés. Deux approches sont possibles pour cette tâche. La première approche consiste à sélectionner "Base" comme indiqué dans la Figure 7, permettant ainsi au LLM choisi de répondre aux questions en se basant sur sa propre connaissance. La seconde approche est le "RAG", où un ensemble de documents est fourni au LLM, lui permettant de répondre aux questions en s'appuyant sur ces documents.

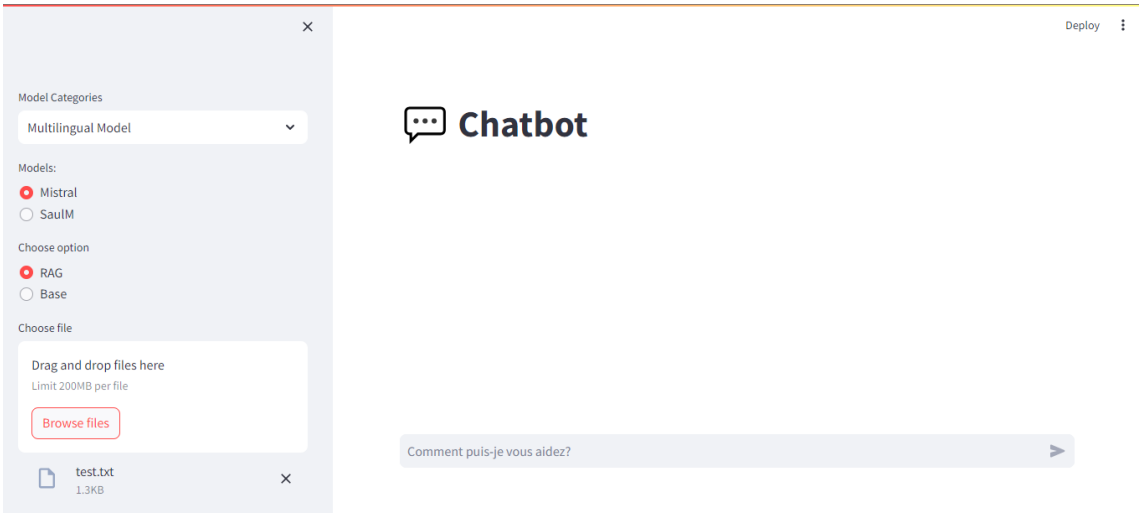


FIGURE 9 – Sélection de l'option RAG et Chargement des documents

Comme illustré dans les figures 9 et 10, cette approche nécessite plusieurs paramétrages, notamment la sélection d'un algorithme de segmentation des documents et d'un algorithme pour encoder les segments.

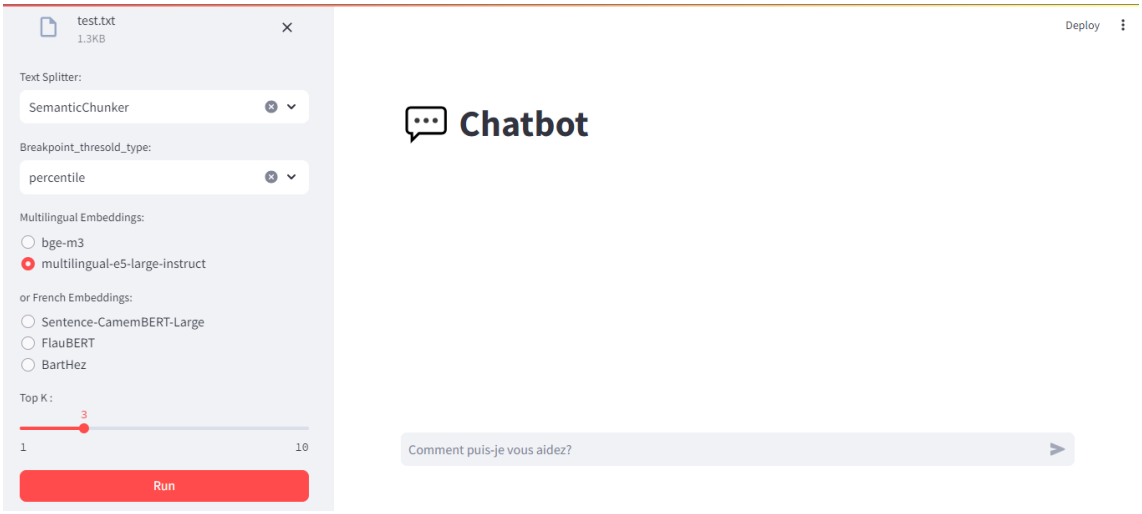


FIGURE 10 – Sélection de la méthode de segmentation, du modèle d' "embedding" et du Top k (le nombre de segments pertinents à choisir).

Conclusion

Ce stage de deux mois chez HephIA a été extrêmement enrichissant. Il nous a permis de découvrir le milieu professionnel et de développer des compétences en intelligence artificielle, en particulier dans le domaine du Traitement Automatique du Langage (TAL) avec l'utilisation des Grands Modèles de Langage (LLMs).

L'objectif principal de ce stage était d'effectuer un état de l'art exhaustif des LLMs pour identifier les meilleurs modèles en termes de performance pour le résumé de textes juridiques. Après avoir sélectionné les LLMs les plus performants en utilisant des méthodes de validation ciblées, nous avons développé une interface utilisateur avec Streamlit pour permettre une interaction directe avec ces modèles.

En plus de la tâche de résumé de textes juridiques, nous avons ajouté une fonctionnalité de question-réponse à l'application. Constatant que les réponses des LLMs dans cette tâche manquaient parfois de fiabilité (par exemple, absence de références), nous avons intégré une approche appelée "Retrieval-Augmented Generation" (RAG) pour améliorer la fiabilité des réponses et réduire les hallucinations des LLMs. Plusieurs méthodes ont été utilisées pour mettre en place cette pipeline RAG.

Durant ce travail, nous avons constaté que l'utilisation des LLMs dans des domaines spécifiques, comme le résumé de textes juridiques en français, présente des défis particuliers. Les résultats obtenus pourraient être améliorés en utilisant des approches telles que le fine-tuning, qui consiste à réentraîner ces LLMs sur des jeux de données juridiques en français pour une meilleure fiabilité et performance.

Références

[1] S. Campano, T. Nabil, and M. Bothua, "Traitement quantique des langues :{\'e} tat de l'art," *arXiv preprint arXiv :2406.15370*, 2024.

[2] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian, "A comprehensive overview of large language models," *arXiv preprint arXiv :2307.06435*, 2023.

[3] Y. Zeng, H. Lin, J. Zhang, D. Yang, R. Jia, and W. Shi, "How johnny can persuade llms to jailbreak them : Rethinking persuasion to challenge ai safety by humanizing llms," *arXiv preprint arXiv :2401.06373*, 2024.

[4] Z. Guo, R. Jin, C. Liu, Y. Huang, D. Shi, Supryadi, L. Yu, Y. Liu, J. Li, B. Xiong, and D. Xiong, "Evaluating large language models : A comprehensive survey," 2023. [Online]. Available : https://arxiv.org/abs/2310.19736

[5] Z. Fei, X. Shen, D. Zhu, F. Zhou, Z. Han, S. Zhang, K. Chen, Z. Shen, and J. Ge, "Lawbench : Benchmarking legal knowledge of large language models," *arXiv preprint arXiv :2309.16289*, 2023.

[6] P. A., "Mistral vs gpt-4 : A comparative analysis in size, cost, and mmlu performance," *Medium*, 2024, accessed : 2024-07-29. [Online]. Available : https://medium.com/@periphanos.a/mistral-vs-gpt-4-a-comparative-analysis-in-size-cost-and-mmlu-performance-de320060388d

[7] K. LLC, "Introducing meta llama 3 : The most capable openly available llm to date," *Medium*, 2024, accessed : 2024-07-29. [Online]. Available : https://medium.com/@kagglepro.llc/introducing-meta-llama-3-the-most-capable-openly-available-llm-to-date-90235aea8aed

[8] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun, "Jec-qa : a legal-domain question answering dataset," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, 2020, pp. 9701–9708.

[9] F. Yao, C. Xiao, X. Wang, Z. Liu, L. Hou, C. Tu, J. Li, Y. Liu, W. Shen, and M. Sun, "Leven : A large-scale chinese legal event detection dataset," *arXiv preprint arXiv :2203.08556*, 2022.

[10] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, "Mistral 7b," *arXiv preprint arXiv :2310.06825*, 2023.

[11] M. AI, "Introducing meta llama 3 : The most capable openly available llm to date," https://ai.meta.com/blog/meta-llama-3/, 2024, accessed : 2024-07-23.

[12] P. Colombo, T. P. Pires, M. Boudiaf, D. Culver, R. Melo, C. Corro, A. F. Martins, F. Esposito, V. L. Raposo, S. Morgado *et al.*, "Saullm-7b : A pioneering large language model for law," *arXiv preprint arXiv :2403.03883*, 2024.

[13] D. Cheng, S. Huang, and F. Wei, "Adapting large language models via reading comprehension," *arXiv preprint arXiv :2309.09530*, 2023.

[14] OpenAI, "Gpt-4 : Large multimodal model," https://openai.com/index/gpt-4-research/, 2023, accessed : 2024-07-23.

[15] M. Sie, "Roberta_bart_fixed_v1," 2022, fine-tuned version of BART for multi-step summarization of long, legal documents. Developed as part of a master thesis at University Utrecht in collaboration with Power2X. [Online]. Available : https://huggingface.co/MikaSie/RoBERTa_BART_fixed_V1

[16] MikaSie, "Roberta_bart_dependent_v1," https://huggingface.co/MikaSie/RoBERTa_BART_dependent_V1, 2024, accessed : 2024-07-23.

[17] M. Sie, "Lexlm_longformer_bart_fixed_v1," 2022, fine-tuned version of BART for multi-step summarization of long, legal documents. Developed as part of a master thesis at University Utrecht in collaboration with Power2X. [Online]. Available : https://huggingface.co/MikaSie/LexLM_Longformer_BART_fixed_V1

[18] ——, "T5_no_extraction_v1," 2022, fine-tuned version of T5 for summarizing long, legal documents without an extractive summarization step. Developed as part of a master thesis at University Utrecht in collaboration with Power2X. [Online]. Available : https://huggingface.co/MikaSie/T5_no_extraction_V1

[19] D. Aumiller, A. Chouhan, and M. Gertz, "Eur-lex-sum : A multi-and cross-lingual dataset for long-form summarization in the legal domain," *arXiv preprint arXiv :2210.13448*, 2022.

[20] M. Law and T. Lab, "Lleqa dataset," https://huggingface.co/datasets/maastrichtlawtech/lleqa, 2024, accessed : 2024-07-23.

[21] LangChain Team, "Langchain," https://www.langchain.com/, accessed : 2023-11-29.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[23] A. Farkiya, P. Saini, S. Sinha, and S. Desai, "Natural language processing using nltk and wordnet," *Int J Comput Sci Inf Technol*, vol. 6, no. 6, pp. 5465–5469, 2015.

[24] E. Partalidou, E. Spyromitros-Xioufis, S. Doropoulos, S. Vologiannidis, and K. Diamantaras, "Design and implementation of an open source greek pos tagger and entity recognizer using spacy," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019, pp. 337–341.

[25] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "Bge m3-embedding : Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," 2024.

[26] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei, "Multilingual e5 text embeddings : A technical report," *arXiv preprint arXiv :2402.05672*, 2024.

[27] I. G. Nils Reimers, "Sentence-bert : Sentence embeddings using siamese bert-networks," *https ://arxiv.org/abs/1908.10084*, 2019.

[28] L. Martin, B. Muller, P. J. O. Suárez, Y. Dupont, L. Romary, É. V. de la Clergerie, D. Seddah, and B. Sagot, "Camembert : a tasty french language mode," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[29] H. Le, L. Vial, J. Frej, V. Segonne, M. Coavoux, B. Lecouteux, A. Allauzen, B. Crabbé, L. Besacier, and D. Schwab, "Flaubert : Unsupervised language model pre-training for french," 2020. [Online]. Available : https://arxiv.org/abs/1912.05372

[30] M. K. Eddine, A. J.-P. Tixier, and M. Vazirgiannis, "Barthez : a skilled pretrained french sequence-to-sequence model," *arXiv preprint arXiv :2010.12321*, 2020.

[31] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, "The faiss library," *arXiv preprint arXiv :2401.08281*, 2024.