

Détection et Contre-mesures des Jailbreaks sur les LLM

Réalisé par :

OUNDA Dimitri-Gaetan
SANOGO Cheickna

Encadré par :

M. Anas ABOU EL
KALAM



INTRODUCTION

Les modèles de langage (LLM) sont devenus essentiels en cybersécurité, mais vulnérables aux attaques de type **jailbreak**.

- ♦ Notre projet: **créer un système à deux couches (déTECTEUR + jUGE)** pour détecter et bloquer ces tentatives, garantissant la sécurité des LLM.

PLAN

Contexte et Problématique
Définitions
Etat de l'Art
Notre solution
Implémentation technique
Résultat et Limites actuels
Perspectives et Améliorations Futures
Conclusion



Contexte et Problématique des LLM



Rôle des LLM

Les LLM sont cruciaux en cybersécurité.

Ils assistent dans l'analyse des menaces.



Vulnérabilités

Cependant, ils présentent des failles.

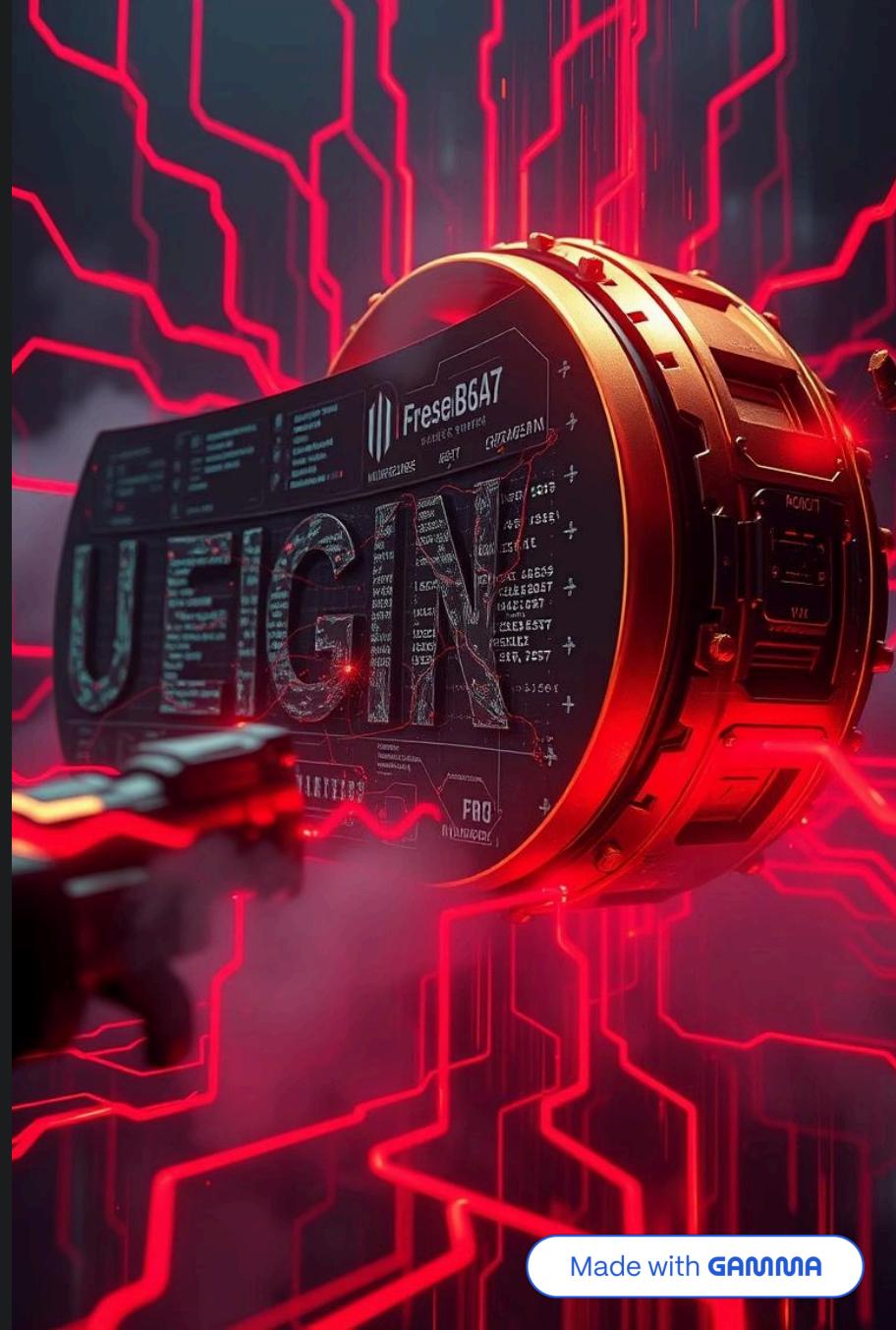
Les attaques ciblent ces vulnérabilités.



Risques de Jailbreak

Les jailbreaks détournent leurs fonctions.

Ils peuvent générer du contenu malveillant.



Définitions



LLM (Large Language Model)

Modèle d'intelligence artificielle entraîné sur d'immenses volumes de texte. Ces modèles utilisent des réseaux de neurones profonds pour comprendre, générer et manipuler le langage naturel.



Prompt malveillant

Requête ou un texte envoyé au modèle, conçu de manière intentionnelle pour contourner ses mécanismes de sécurité, ou pour lui faire produire un contenu inapproprié, dangereux ou interdit.

Ces prompts incluent souvent des instructions explicites pour ignorer les règles ou pour simuler un rôle malveillant.



Jailbreak

Tentative de contournement des restrictions mises en place dans le modèle, afin de le forcer à fournir des réponses interdites, inappropriées ou contraires à l'éthique.

Ces attaques visent à "libérer" le modèle de ses garde-fous (filtres de sécurité) pour en exploiter les capacités.



État de l'Art des Attaques et Défenses

Techniques de Jailbreak

- **Détection des prompts malveillants**
 - **PromptGuard (Zou et al., 2023)** : extraction de caractéristiques linguistiques + classificateur ML.
 - **Chen et al. (2022)** : détection basée sur l'analyse syntaxique et les patterns suspects.
 - **!** *Limite : se basent uniquement sur le prompt, sans tenir compte de la réponse.*
- **Détection post-réponse**
 - **LLM Safety Gym (OpenAI)** : test systématique des réponses par rapport à des comportements attendus.
 - **DecodingTrust (2023)** : combinaison d'analyse humaine et automatisée pour juger les sorties.
 - **👉 Inspiration pour notre modèle juge autonome.**
- **Approches hybrides**
 - **Xu et al. (2023)** : chaîne de filtres avec deux étapes (détection puis validation).
 - **⚠ Manque de multi-modèles et d'observation dynamique dans leur système.**



Notre Solution : Architecture de Défense

Notre système utilise une architecture à deux couches pour une détection et une réponse complètes.



DéTECTEUR

Analyse initiale du prompt.

Utilise un dataset ML.

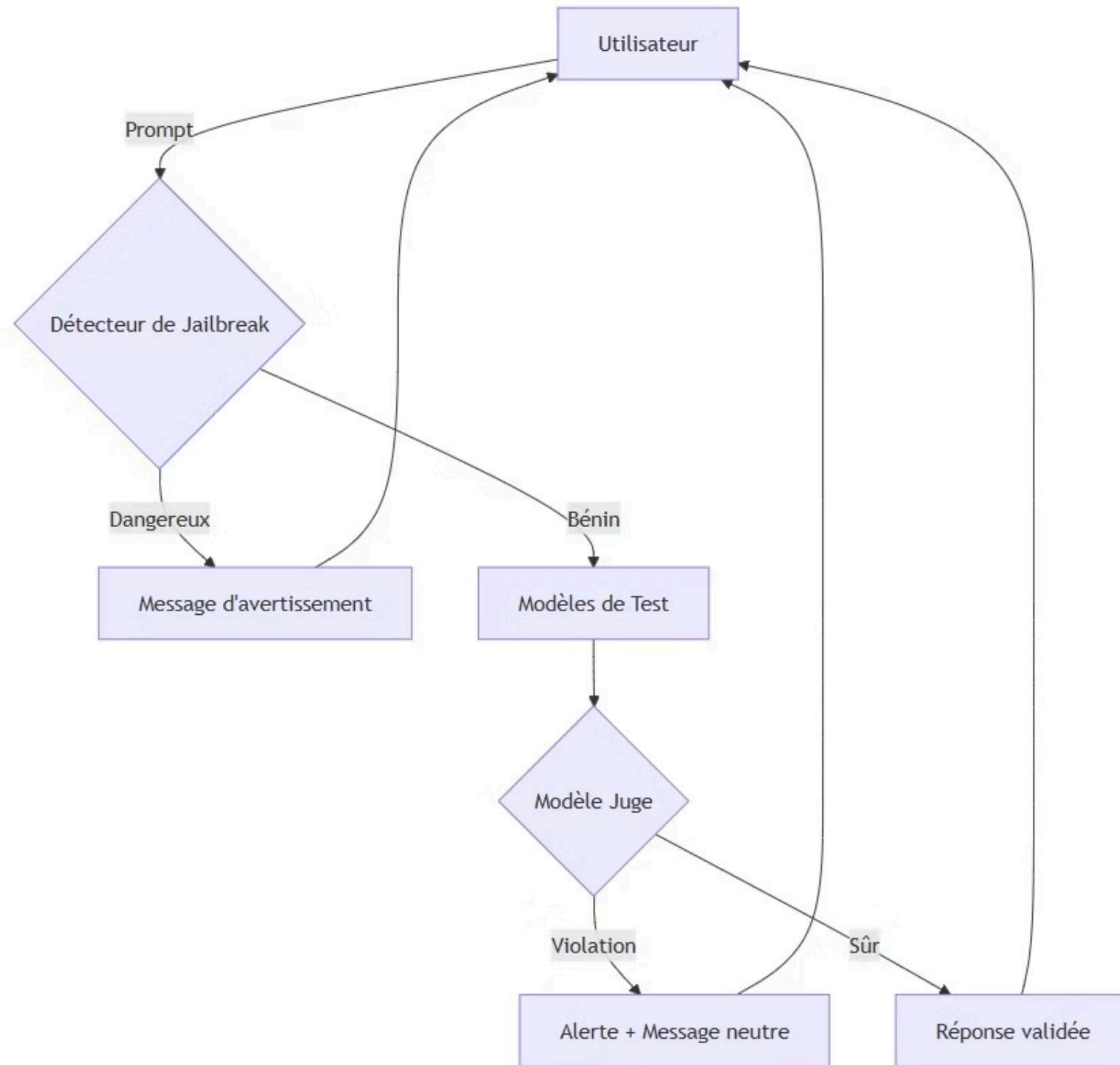
JUGE

Analyse la réponse du LLM.

S'appuie sur un modèle entraîné en se basant sur RoBERTa-base

Un proxy de filtrage assure le contrôle. Un système de logs et un tableau de bord offrent la visibilité nécessaire.

Notre Solution : Fonctionnement





Implémentation Technique

Notre solution s'appuie sur des technologies robustes et une méthodologie rigoureuse.

Technologies Clés

Python, Ollama

scikit-learn, pandas pour l'analyse.

Django pour l'application web

Création du Dataset

Dataset équilibré et représentatif.

Essentiel pour l'entraînement.

Entraînement du DéTECTEUR

TF-IDF et Régression Logistique.

Pour une classification efficace.

Entraînement du Juge

RoBERTa-base (réseau de neurones)

Classification binaire avec couche dense finale

Résultats et Limites actuels

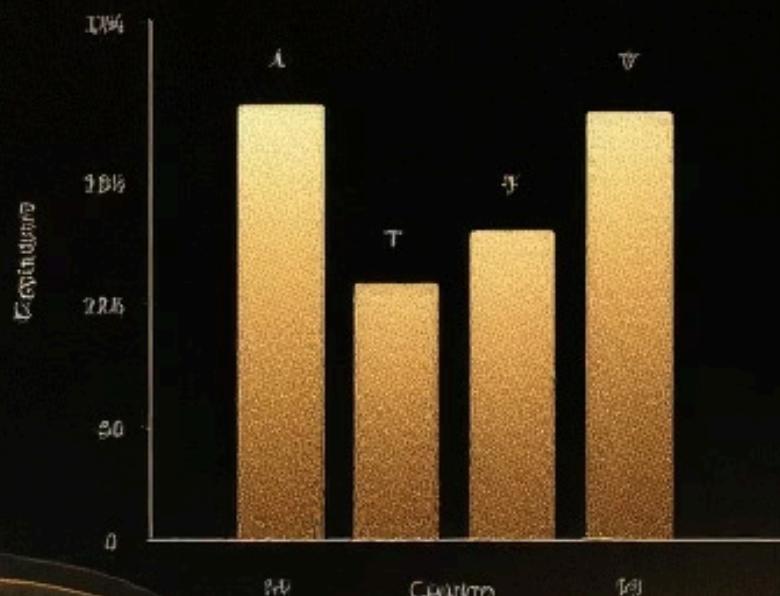
Current Results and Limitations

Current Results & Limitations in LLMs

Current detection is effective and handles many common detection mistakes in LLMs.

77.5% Strong F1 score results

56% Detection, related and F1 score results

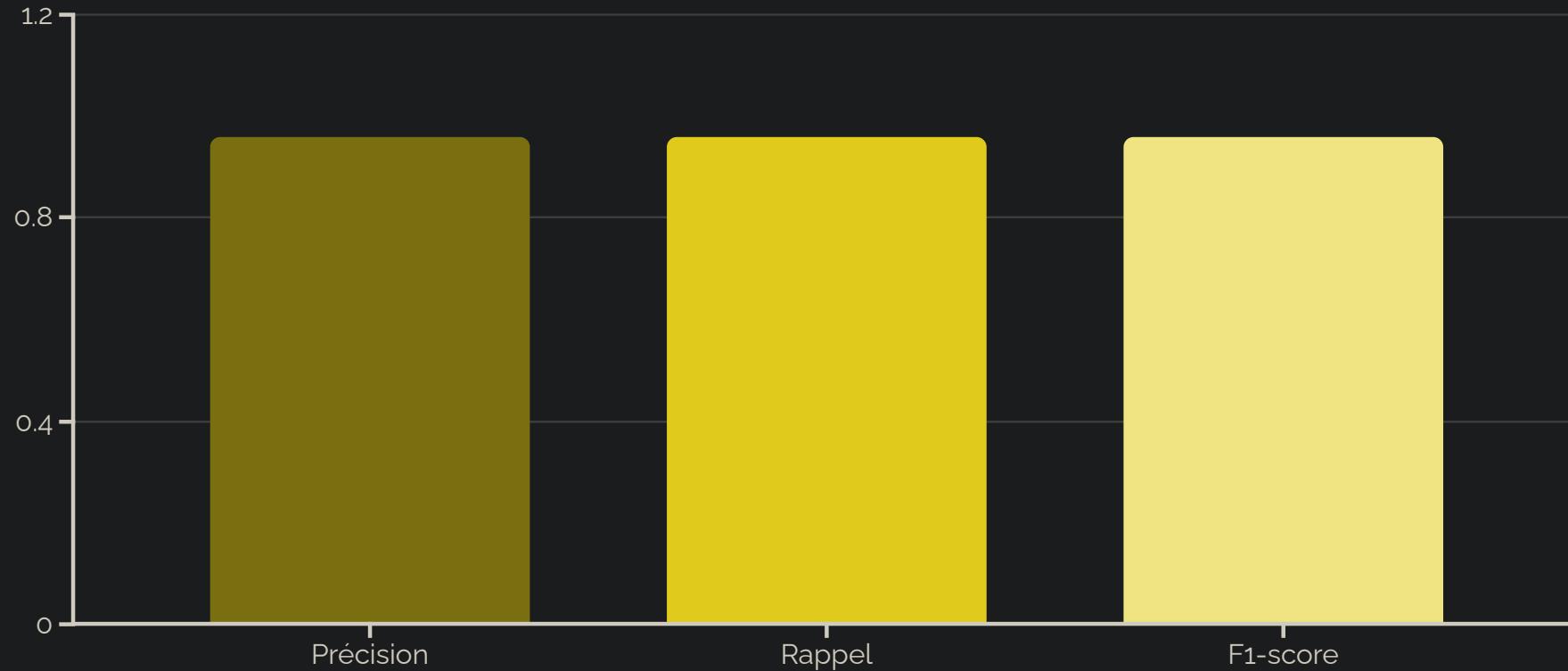


Current Limitations

Despite strong precision, recall, and F1 score results

- + Illustrate Limitations
- + Current limitations

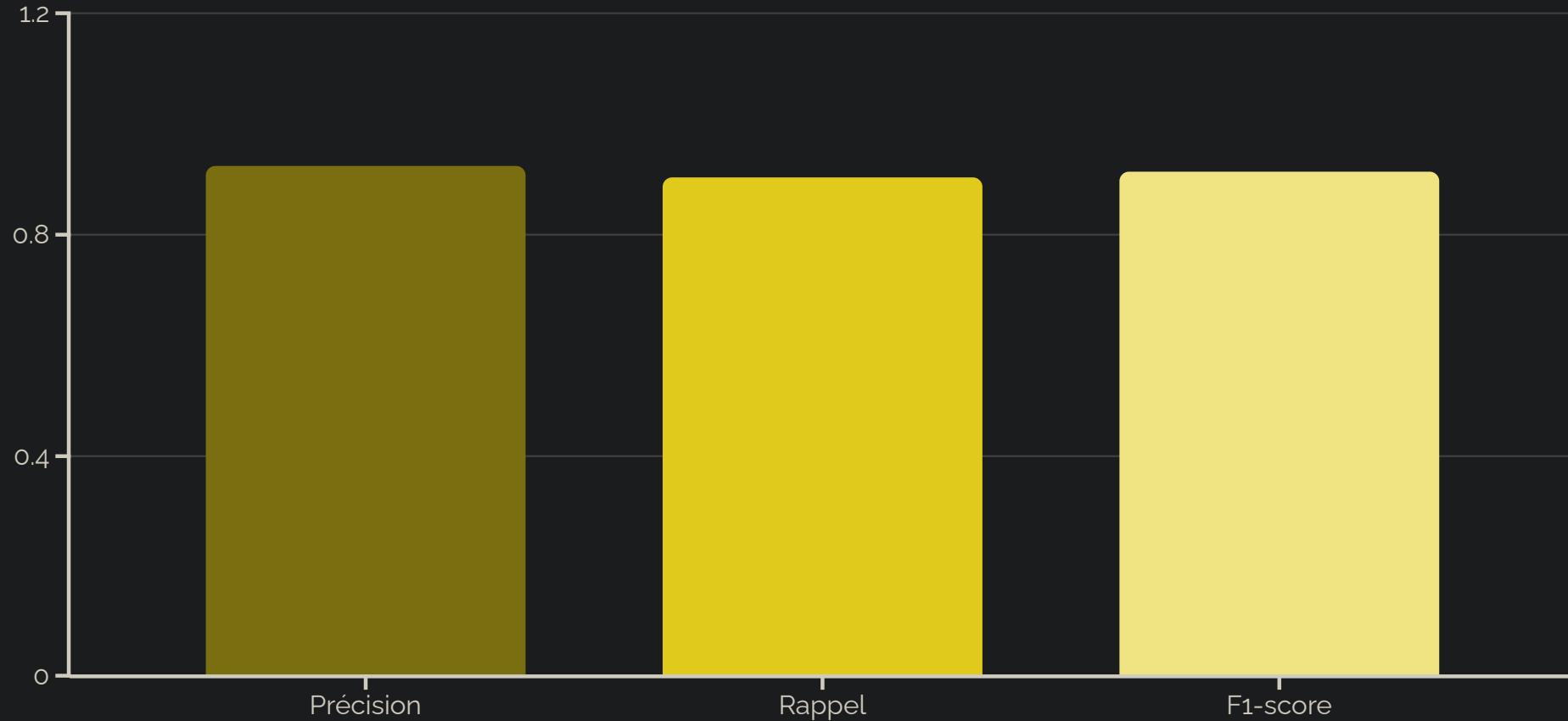
Résultats de l'entraînement du DéTECTEUR



Résultats du DéTECTEUR

- Avec une **précision**, un **rappel** et un **F1-score** de **0,96**, le détecteur démontre une capacité très fiable à identifier les prompts de type jailbreaks.
- Ces performances indiquent un excellent compromis : peu de faux positifs (bonne précision) et peu de faux négatifs (bon rappel).
- Cela montre que le modèle est bien entraîné sur le dataset de prompts.

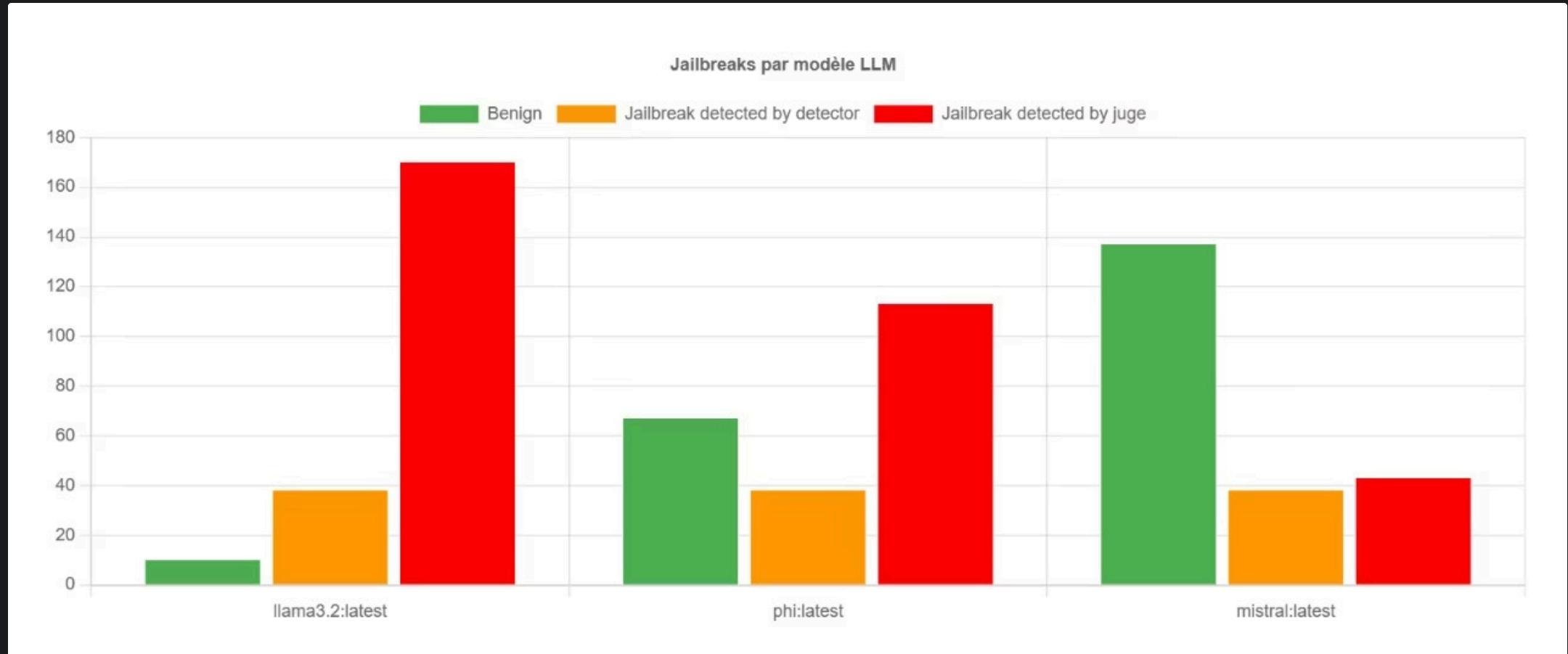
Résultats de l'entraînement du Juge



Résultats du Juge

- Avec une **précision de 0,922**, un **rappel de 0,903** et un **F1-score de 0,912**, le juge confirme la robustesse du système en validant ou infirmant les décisions du détecteur.
- Même si les scores sont légèrement inférieurs à ceux du détecteur, ils restent très bons et reflètent la capacité du juge à s'adapter aux cas plus ambigus (notamment en cas de prompts borderline ou de réponses inattendues).
- Ces résultats sont cohérents avec la complexité de la tâche du juge, qui dépend des réponses parfois subtiles des LLM.

Résultats des tests finaux



Les Limites actuels

- ♦ Il ne capture pas encore les attaques multi-tours avec état conversationnel.
- ♦ Les modèles juge et détecteur peuvent parfois produire des verdicts divergents sur des cas ambigus.
- ♦ Tous les modèles utilisés sont en anglais; une adaptation multilingue est à envisager.
- ♦ Enfin, ces écarts mettent en évidence la difficulté de valider concrètement l'efficacité du pipeline (les LLMs testés restent majoritairement résistants aux jailbreaks, ce qui complique la démonstration des capacités réelles du système).



Perspectives et Améliorations Futures

Nous envisageons plusieurs pistes pour renforcer notre solution et étendre ses capacités de détection.



APIs Multiples

Utiliser plusieurs APIs pour un jugement croisé.

Augmente la robustesse du Juge.



Enrichir le Dataset

Intégrer des prompts plus subtils et complexes.

Améliore la généralisation du Détecteur.



Nouveaux LLM

Tester des LLM plus récents ou plus ouverts.

Évaluer leur résistance aux jailbreaks.

Démonstration

Conclusion

Notre projet a démontré la faisabilité d'un système robuste de détection et de contre-mesures pour les jailbreaks de LLM.

Il offre une valeur ajoutée significative pour la sécurisation des applications basées sur l'IA.





Thank You