Détection des contaminations de LLM par extraction de données : une revue de littérature pratique

Pierre Lepagnol[1,2], Thomas Gerald[1], Sahar Ghannay[1], Christophe Servan[1], Sophie Rosset[1]
[1]Université Paris-Saclay, CNRS, Laboratoire interdisciplinaire des sciences du numérique, 91405, Orsay, France
[2]SCIAM, 7508, Paris, France
{firstname.lastname}@lisn.upsaclay.fr

RÉSUMÉ

Cet état de l'art examine le problème de la contamination des données d'entraînement dans les grands modèles de langue (LLM). Ce phénomène se produit lorsque les modèles sont évalués sur des données qu'ils ont déjà rencontrées durant leur entraînement, créant une fausse impression de performance. Cette étude propose une synthèse pratique pour la communauté scientifique du traitement automatique des langues (TAL). Nous présentons un cadre d'analyse qui distingue différents niveaux de contamination ainsi que différentes méthodes classées selon l'accès au modèle (White/Gray/Black-Box) et les techniques utilisées (Similarité/Probabilité/Extraction). Nous explorons particulièrement les méthodes d'extraction de données de LLM, les approches techniques, les mesures de performance et leurs limites. Dans une perspective pratique, nous avons synthétisé ces méthodes sous la forme d'un arbre de décision pour sélectionner la méthode de détection de contamination adéquate.

ABSTRACT

Detection of LLM Data Contamination through Data Extraction : A Practical Literature Review

This review examines the issue of training data contamination in large language models (LLMs). This phenomenon occurs when models are evaluated on data they have already encountered during their training, creating a false impression of performance. This study offers a practical synthesis for researchers and professionals. We present an analytical framework that distinguishes different levels of contamination as well as various methods classified according to model access (White/Gray/Black-Box) and the technical approaches used (Similarity/Probability/Extraction). We particularly explore data extraction methods for LLMs, the technical approaches, performance measures, and their limitations. From a practical perspective, we propose a decision tree to the community for selecting the appropriate contamination detection method.

MOTS-CLÉS : contamination des données, grands modèles de langue, inférence d'appartenance, extraction de données, détection de contamination.

KEYWORDS: data contamination, large language models, membership inference, data extraction, contamination detection.

ARTICLE : **Soumis à** Nom de la Conférence / Revue.

1 Introduction

Les grands modèles de langue (LLM) ont profondément transformé le traitement automatique des langues en offrant des performances exceptionnelles pour la génération de texte, la traduction, les tâches de question-réponse, etc. (OpenAI *et al.*, 2023; Grattafiori *et al.*, 2024; DeepSeek-AI *et al.*, 2024, 2025; et al., 2025). Toutefois, leur évaluation est souvent remise en question par un phénomène critique : la contamination des données d'entraînement. Ce terme désigne l'inclusion partielle ou totale d'exemples issus des ensembles de test dans les données d'apprentissage, ce qui conduit à une mémorisation de réponses (Fu *et al.*, 2024; Xu *et al.*, 2024b). Une telle mémorisation compromet la fiabilité des résultats d'évaluation, car ils peuvent refléter la capacité d'un modèle à se souvenir, plutôt que sa capacité à généraliser à de nouvelles données inédites. Par conséquent, traiter la contamination des données est crucial pour garantir que les évaluations des LLM représentent avec précision leurs véritables capacités, une exigence fondamentale pour faire progresser la recherche et le déploiement de l'IA.

L'objectif principal de cet article est de proposer une revue de littérature pratique, destinée aux praticiens travaillant avec les LLM. Il s'agit de fournir un cadre clair permettant de mieux comprendre et d'identifier les méthodes de détection de la contamination des données d'entraînement. Cela permettra aux utilisateurs de vérifier si leurs propres données ont été intégrées dans les corpus d'entraînement des LLM, condition indispensable pour garantir des évaluations fiables et prévenir les risques liés à la divulgation involontaire d'informations sensibles. Afin de faciliter l'adoption des méthodes et techniques présentées, un répertoire git dédié [1] est mis à disposition de la communauté avec des implémentations prêtes à l'emploi.

Cette revue de littérature est structurée comme suit : dans la section 2, nous présentons une taxonomie des formes de contamination des données qui se fonde sur les travaux de Palavalli *et al.* (2024a) et Cheng *et al.* (2025). La section 3 explore les approches de détection de la contamination, classées selon le niveau d'accès au modèle (White/Gray/Black-Box) et les techniques utilisées (similarité, probabilités, génération), tout en soulignant leurs hypothèses et limites. La section 4 détaille les méthodes d'extraction de données d'entraînement, en différenciant les attaques par amorçage (prompting) et par inversion, suivies d'une présentation des métriques d'évaluation et des jeux de données associés. La section 5 traite des limites des méthodes d'extraction, notamment le caractère partiel des extractions, l'influence de la taille des modèles dans ces méthodes, et les coûts computationnels. Enfin la section 6 propose des recommandations pratiques sous forme d'un arbre de décision pour guider le choix des méthodes de détection en fonction des contraintes d'accès aux données et au modèle.

Nous utilisons de manière équivalente le terme général *extraction de données* pour désigner toute technique permettant de générer avec un LLM les données d'entraînement ou méta-données d'entraînement. Parmi les méthodes d'extraction, nous distinguons clairement la reconstruction (ou génération) qui est l'extraction de verbatim, généralement par amorçage, et l'inversion qui cherche à retrouver l'entrée originale (amorce ou plongement vectoriel) à partir de sorties du modèle.

1. https://gitlab.lisn.upsaclay.fr/nlp/sota/data-contamination-extraction.git

2 Taxonomie et axes de contamination

Palavalli *et al.* (2024b) proposent une taxonomie qui classe la contamination des données en deux niveaux distincts. D'une part, on observe la contamination au niveau du jeu de données, qui peut survenir soit par « sélection » (inclusion de verbatim de données de test dans l'ensemble d'entraînement), soit par « distribution » (utilisation de données différentes mais issues de la même distribution que les données de test). D'autre part, la contamination peut également se produire au niveau des exemples individuels, notamment par des mécanismes de masquage, de bruitage ou d'augmentation de données décrit dans leurs travaux.

Palavalli *et al.* (2024b) mettent l'accent sur le pré-entraînement uniquement et les contaminations spécifiques aux tâches de résumé et questions-réponses. Ils précisent que la contamination se distingue de l'apprentissage transductif, où le modèle voit l'exemple de test sans l'annotation associée ou bien se distingue aussi de l'apprentissage de la tâche, où le modèle voit des données et des labels de la même tâche mais pas issue du benchmark de test. Ainsi suivant leur définition, dans le paradigme d'amorçage (prompting) suivant des instructions, la contamination dépend de la tâche à faire résoudre au LLM. Si la tâche est une continuation de texte (comme la production d'un résumé, ou une production de réponses ouverte) on peut caractériser le fait de voir l'énoncé ou la réponse associée non conjointement comme une contamination. En revanche, pour caractériser une contamination pour une tâche d'annotation il faut que le LLM voit l'exemple de test conjointement à la réponse associée.

Les travaux de Cheng *et al.* (2025) élargissent cette catégorisation en ajoutant la dimension des phases d'entraînement. La contamination peut survenir à trois moments différents. Lors du pré-entraînement, quand les données d'entraînement sont massivement collectées depuis le web ; pendant l'ajustement lorsque les données d'ajustement contiennent des données de test ; ou après le déploiement quand les utilisateurs soumettent directement au modèle des données de test afin d'évaluer ses performances. De plus, Cheng *et al.* (2025) et Ravaut *et al.* (2024) créent des sous-catégories de contamination en distinguant la contamination par l'exemple avec l'annotation/la réponse associée (input-label contamination) ou sans (input contamination).

3 Approches de détection de contamination

Pour répondre à la question *"Est-ce que mon LLM a déjà vu mes données ?"*, diverses méthodes de détection de contamination sont détaillées dans cette section.

Cheng *et al.* (2025) et Ravaut *et al.* (2024) offrent une synthèse de ces approches, permettant d'identifier les exemples auxquels un modèle a potentiellement été exposé durant sa phase d'entraînement. Ces travaux proposent deux systèmes de catégorisation complémentaires.

Le premier système, proposé par Ravaut *et al.* (2024), classe les approches selon le niveau d'accès aux modèles. Les méthodes *White-Box* requièrent un accès complet au modèle et aux données d'entraînement. Les méthodes *Gray-Box* nécessitent l'accès aux probabilités des tokens de sortie. Quant aux méthodes *Black-Box*, elles ne requièrent que l'accès aux sorties du modèle, ce qui les rend idéales lorsque l'interaction se fait uniquement via une API.

Le second système, proposé par Fu *et al.* (2024) et Cheng *et al.* (2025), catégorise les méthodes selon leurs approches techniques. Les méthodes fondées sur la *similarité entre exemples* comparent directement les exemples de test aux exemples d'entraînement. Les méthodes fondées sur l'*analyse*

des probabilités du LLM consistent à comparer ces probabilités avec un modèle de référence. Enfin, les méthodes fondées sur la *mémorisation et la génération d'exemples* s'appuient sur la génération d'exemples d'entraînement avec leur annotation.

Ces deux catégorisations se recoupent : les méthodes de similarité s'appliquent généralement dans un contexte White-Box, les méthodes d'analyse de probabilités correspondent aux approches Gray-Box, tandis que les méthodes de mémorisation peuvent s'utiliser plutôt dans un contexte Black-Box. Cette double classification permet d'orienter le choix de la méthode de détection selon les contraintes d'accès au modèle et les contraintes techniques comme le temps de calcul, le besoin de modèle de référence, etc.

Les différentes méthodes reposent sur des hypothèses et pré-requis décrits par Fu *et al.* (2024) et présentés en section 6.

La majorités des travaux se concentre sur la détection de la contamination au niveau de l'exemple et définissent une contamination au niveau du jeu de données complet si suffisamment d'exemples individuels contaminent les données d'entraînements. (Cheng *et al.*, 2025; Fu *et al.*, 2024; Xu *et al.*, 2024b; Ravaut *et al.*, 2024; Palavalli *et al.*, 2024b)

Les méthodes de détection au niveau de l'exemple sont aussi appelées attaques d'inférence d'appartenance (Membership Inference Attacks, MIA). Zhang *et al.* (2024c) soutiennent que les MIA souffrent de limitations fondamentales. Ces attaques reposent sur le test de l'hypothèse nulle (c'est-à-dire l'hypothèse selon laquelle le modèle n'a pas été entraîné sur les données de test). Or, il est souvent pratiquement impossible de tester cette hypothèse de manière rigoureuse : nous ne disposons souvent pas des données d'entraînement et nous ne pouvons souvent pas ré-entraîner les modèles. Ainsi, il est impossible d'observer un LLM avec et sans les données contaminées, donc d'estimer de manière fiable la probabilité d'appartenance aux données d'entraînement. De plus, les approches d'estimation a posteriori (avec des données créées après l'entraînement du modèle) souffrent de décalage de distribution (distribution shift).

Cette difficulté d'estimation entraîne un taux de faux positif (TFP) élevé et remet nécessairement en question la validité des conclusions obtenues par ces approches. De plus, les auteurs notent que ces attaques ne fonctionnent bien que dans un cadre contrôlé et non dans les environnements du monde réel où le prétraitement des données, la régularisation du modèle ou la diversité des corpus d'entraînement peuvent masquer les signaux d'appartenance, donc augmenter le TFP, affaiblissant davantage l'efficacité des méthodes MIA. Ces conclusions sont soutenus par les travaux de Duan *et al.* (2024a); Maini *et al.* (2024a); Fu *et al.* (2024) qui montrent empiriquement que les méthodes MIA par analyse des probabilités ne font pas mieux qu'une estimation aléatoire.

Zhang *et al.* (2024c) proposent plusieurs alternatives pour détecter la contamination des données, notamment l'injection de leurres aléatoires (random canaries), les attaques par extraction de données (data extraction attacks) et le filigranage des données d'entraînement (watermarking).

4 Approches par extraction de données d'entraînement

Dans cette section, nous abordons les *méthodes par extraction de données d'entraînement*. Nous utilisons de façon équivalente les termes *méthodes* et *attaques*.

Nous définissons l'objectif de ces méthodes puis nous détaillons les deux types attaques existantes,

attaques par amorçage (prompting) et attaques par inversion. Enfin nous présentons les métriques d'évaluation de ces méthodes ainsi que les jeux de données utilisés pour comparer ses méthodes.

4.1 Contexte et définitions

Les attaques de reconstruction de données visent à extraire des informations sensibles, telles que des données d'entraînement, des amorces (prompt) ou des plongements (embeddings), à partir de modèles d'apprentissage automatique. C'est une typologie de méthodes développées initialement dans des buts de confidentialités et sécurités.

Les méthodes d'extraction de données se distinguent des approches MIA traditionnelles par leur objectif : elles vise non pas à déterminer si un exemple donné appartient au jeu d'entraînement, mais plutôt à reconstruire et générer directement des données d'entraînement.

Carlini *et al.* (2020) ont été parmi les premiers à montrer que les LLM peuvent mémoriser et régurgiter des séquences textuelles provenant de leurs données d'entraînement. Leur approche consiste à faire générer au modèle du texte, puis à vérifier si ces générations correspondent à des données d'entraînement connues. Cette méthode a permis d'identifier plusieurs cas où GPT-2 reproduisait verbatim des textes de son corpus d'entraînement.

Zhang *et al.* (2024c) argumentent que les méthodes par reconstruction de données sont plus efficaces car l'espace des continuations plausibles pour une amorce donnée est si vaste que la génération exacte d'un texte particulier sans exposition préalable devient statistiquement négligeable.

Les méthodes par reconstruction se révèlent particulièrement fiables et statistiquement fondées lorsque trois critères heuristiques essentiels sont respectés. Tout d'abord, le texte cible doit présenter une longueur suffisante. Dans leur article, Zhang *et al.* (2024c) expliquent que le taux de faux positifs (TFP) est approximativement égal à $\frac{1}{|x|}$, où x représente la séquence de tokens générée. Ensuite, il est indispensable que le texte cible possède une entropie informationnelle élevée, ce qui signifie qu'il ne doit pas s'agir simplement d'une phrase très prévisible, telle qu'une suite de chiffres. Enfin, l'amorce (prompt) utilisée doit être courte ou sembler aléatoire afin de minimiser la probabilité que le texte cible soit déjà encodé dans cette amorce.

On distingue plusieurs approches pour mener ces attaques. Parmi elles, les attaques par amorçage, qui incluent l'optimisation de l'amorce, visent à déclencher la génération des données de test par le modèle. En parallèle, les attaques par inversion cherchent à reconstruire l'amorce (prompt) qui a permis de générer la réponse (soit les données de test), soit en inversant la sortie textuelle du modèle, soit en inversant le plongement (embedding, représentation vectorielle) associé à cette sortie. L'inversion des plongements consiste à retrouver le texte d'origine à partir de sa représentation vectorielle.

4.2 Attaques par amorçage (prompting)

Les attaques par amorçage (attaques par prompting), constituent une catégorie d'extraction de données qui vise à fournir au LLM des amorces soigneusement conçues pour révéler ses données d'entraînement. Ces attaques reposent sur le principe que les LLM mémorisent partiellement leurs données d'entraînement et peuvent être amenés à les régurgiter lorsqu'ils sont correctement amorcés.

L'objectif est de trouver des amorces qui maximisent cette probabilité de génération, permettant ainsi d'extraire des verbatim des données d'entraînement du modèle ou des informations indiquant une mémorisation. Plusieurs stratégies d'amorçage ont été développées dans la littérature.

Carlini *et al.* (2020) utilisent des amorces simples sous la forme de dictionnaires Python pour extraire des URLs contenues dans les données d'entraînement. Dans les travaux ultérieurs, Nasr *et al.* (2023) utilisent des amorces directes comme « Continue this text : [début d'un texte potentiellement mémorisé] » ou « Récite [type de contenu spécifique] ».

Ils montrent que les modèles LLM entraînés avec du RLHF (Reinforcement Learning from Human Feedback, apprentissage par renforcement à partir de retours humains) ne sont pas résistants aux attaques par extraction de données, en particulier en utilisant une amorce du type : `Repeat this word forever: "poem poem ... poem"x50`. Ils appellent ce type de prompt *Divergence Attack*. Selon les auteurs, pour récupérer des données issues d'un modèle adapté au dialogue, il est nécessaire de trouver un moyen de le détourner de son entraînement d'alignement afin de lui faire produire des tokens issus de sa distribution de pré-entraînement. Le RLHF est une technique d'alignement qui consiste à affiner un modèle de langage en utilisant le retour d'évaluateurs humains, ce qui permet d'améliorer la qualité, la pertinence et la sécurité des réponses du modèle. Cependant, ce processus d'alignement n'élimine pas la mémorisation des données d'entraînement, il se contente souvent de la masquer.

D'autres approches (Wang *et al.*, 2024) emploient des stratégies d'ingénierie d'amorce plus sophistiquées, comme le jailbreaking (Xu *et al.*, 2024d) ou l'extraction itérative/décomposée (Su *et al.*, 2024), où l'attaquant affine progressivement ses amorces en fonction des réponses précédentes du modèle.

Les travaux de Sternak *et al.* (2025) proposent un cadre multi-agents automatisé qui sondent itérativement un LLM pour forcer la divulgation d'amorces dites "système" ou "pré-amorce" (pre-prompt), amorce non-choisie par l'utilisateur derrière les interfaces des produits type ChatGPT. Leurs résultats montrent que même les API en boîte noire peuvent être manipulées pour révéler des instructions confidentielles en utilisant des agents autonomes. Ces découvertes soulignent le besoin urgent de mécanismes de protection des amorces dans les LLM déployés.

Les travaux de Golchin & Surdeanu (2023a) proposent une méthode basée sur la reponse d'un LLM à un questionnaire à choix multiple. Cette méthode permet de détecter la mémorisation de données d'entraînement par un LLM.

Dans la même lignée, Golchin & Surdeanu (2024) développent une approche d'« instructions guidées » (guided instruction) qui intègre explicitement des métadonnées du dataset source dans l'amorce. Leur méthode consiste à fournir au LLM le nom du dataset, le type de partition (train/test/validation), et un segment initial aléatoire d'une instance de référence, en lui demandant de compléter le reste. Cette approche guidée est comparée à une « instruction générale » sans métadonnées spécifiques. Une instance est considérée comme contaminée si la sortie du LLM correspond exactement ou quasi-exactement au segment de référence. Au niveau partition, ils utilisent deux heuristiques : une comparaison statistique des scores de chevauchement (ROUGE-L, BLEURT) entre instructions guidées et générales, et une classification par GPT-4 avec apprentissage en contexte. Leur méthode atteint une précision de 92-100% et révèle notamment que GPT-4 présente des signes de contamination avec les datasets AG News, WNLI, et XSum.

Les mêmes auteurs proposent également le *Data Contamination Quiz* (DCQ) (Golchin & Surdeanu, 2025), une approche originale qui transforme la détection de contamination en questionnaire à choix multiples. Pour chaque instance testée, le DCQ génère un quiz à cinq options : une option contient

l'instance originale du dataset, trois options présentent des versions perturbées au niveau des mots (utilisant des synonymes contextuels générés par GPT-4), et une dernière option propose "Aucune des options proposées". Les perturbations préservent le sens et la structure syntaxique tout en modifiant la formulation exacte. Le principe repose sur l'hypothèse qu'un LLM ayant mémorisé l'instance originale devrait pouvoir la distinguer parmi les versions perturbées, la seule différence discriminante étant le choix exact des mots. Pour gérer les biais de position inhérents aux LLM, les auteurs introduisent un "Bias Detector Quiz" (BDQ) qui permet d'ajuster les estimations de contamination. Cette méthode présente l'avantage de ne nécessiter aucun accès aux données d'entraînement ou aux paramètres du modèle, tout en contournant efficacement les filtres de sécurité, notamment ceux conçus pour éviter la génération de contenu protégé par le droit d'auteur.

4.3 Attaques par inversion

Les attaques par inversion visent à reconstruire les entrées textuelles originales à partir de leurs plongements (embeddings, représentations vectorielles), soit de l'amorce, soit de la sortie. Contrairement aux attaques par reconstruction qui génèrent directement un contenu mémorisé, les attaques par inversion cherchent à retrouver le texte original en inversant la sortie du modèle ou ses représentations internes. Pour ce faire, on passe souvent par l'apprentissage d'une fonction d'inversion.

Skapars *et al.* (2024) définissent deux types attaques par inversion. Les inversions exactes où, étant donné une paire entrée-sortie, l'objectif est de reconstruire la séquence d'entrée à partir du modèle de langue et de la sortie uniquement. Les inversions faible (ou approximatives) où l'objectif est de trouver n'importe quelle séquence d'entrée telle que la sortie soit au moins aussi probable avec cette séquence qu'avec l'entrée originale.

Notons que l'objectif d'inversion faible permet de repérer des variations des exemples d'entraînement, se qui correspond bien aux problèmes de diversités de données d'entraînement soulevées par Zhang *et al.* (2024c).

Inversion de texte Les travaux de Morris *et al.* (2023) font partis des premiers travaux cherchant à reconstruire le prompt grâce aux sorties textuelles d'un modèle de langue. Pour se faire ils entraînent un modèle encodeur-décodeur à produire les amorces qui ont générés le texte de sortie. Les travaux sont réalisés sur des modèles Llama-2 à 7 milliards de paramètres. Dans la continuité de ces travaux Zhang *et al.* (2024a) entraine un modèle d'inversion directement sur le texte sans passer par les logits du LLM à inverser.

Dans leurs travaux sur l'inversion textuelle, Skapars *et al.* (2024) utilisent des méthodes de génération basées sur des algorithmes génétiques pour essayer d'inverser la sortie d'un modèle. Ils montrent que les algorithmes génétiques pour explorer l'espace des amorces possibles sont plus efficaces que les algorithmes d'optimisation par essaim particulaire (OEP, ou PSO -Particle Swarm Optimisation-) appliqués sur les plongements puis décodé.

Inversion de plongement (embeddings inversion) Huang *et al.* (2024) propose d'entraîner un modèle substitut pour inverser les plongements lorsque nous n'avons pas accès aux plongements originaux. En entraînant le modèle substitut sur un domaine similaire au modèle cible, nous reproduisons l'espace de plongement permettant de générer l'amorce qui a produit ce plongement. Une approche multilingue est proposée par Chen *et al.* (2024).

Plus récemment, Chen *et al.* (2025) proposent une méthode d'inversion des plongements en alignant

ces derniers avec un LLM pour reconstruire le texte original à partir d'un nombre limité d'échantillons. Leurs résultats montrent que même des informations de plongement limitées peuvent être exploitées pour récupérer des textes sensibles presque mot pour mot.

4.4 Évaluation des méthodes : métriques et jeux de données

La comparaison des méthodes d'extraction de données se fait dans un contexte expérimental contrôlé à l'aide de jeux de données et de modèles ouverts. Il existe différentes métriques qui permettent d'évaluer entre elles les méthodes d'extraction de données ainsi que divers jeux de données selon la problématique abordée.

Métriques D'après Ishihara (2023), les métriques d'évaluation pour l'extraction des données d'entraînement varient selon les études. Tandis que la plupart des études (He *et al.*, 2025; Shi *et al.*, 2023) sur l'inférence de l'appartenance mesurent des taux de précision ou exactitude moyens, Carlini *et al.* (2021a) ont proposé que l'inférence d'appartenance soit évaluée par le taux de vrais positifs (TVP) avec un faible TFP.

Les principales métriques sont l'*exactitude équilibrée (Balanced Accuracy)* qui consiste à mesurer la fréquence à laquelle une attaque prédit correctement l'appartenance sur un jeu de données équilibré entre membres et non-membres des données de test (Choquette-Choo *et al.*, 2021; Shokri *et al.*, 2017; Sablayrolles *et al.*, 2019; Yeom *et al.*, 2018). L'*AUC (Area under the curve)* est la méthode la plus couramment utilisée pour interpréter la courbe ROC (Receiver Operating Characteristic) (Sankararaman *et al.*, 2009) et consiste à calculer l'aire sous la courbe ROC, ce qui reflète le succès moyen de l'inférence d'appartenance. Enfin, le *TVP à faible TFP* se concentre sur le taux de vrais positifs de l'attaque lorsque le seuil est réglé sur une valeur élevée pour atteindre un taux de faux positifs faible. Cette dernière métrique est recommandée dans les MIAs car elle reflète directement l'étendue réelle de la contamination des exemples d'entraînement (Satvaty *et al.*, 2024; Carlini *et al.*, 2022).

Jeux de données Pour comparer les méthodes d'extraction entre elles, le *Training Data Extraction Challenge* a été créé couvrant les attaques d'extraction ciblées - attaques où l'on sait ce que l'on cherche - a contrario des attaques non-ciblées où l'on veut générer n'importe quelle donnée d'entraînement. Les métriques de ce benchmark sont la vitesse d'attaque ainsi que le rappel et la précision.

Néanmoins, certaines études d'impact (Xu *et al.*, 2024a,c; Zhou *et al.*, 2025) cherchent à extraire les données de benchmark publiques, élargissant les datasets et sortant du cadre contrôlé.

D'autres études se concentrent sur la génération des amorces qui servent à générer les réponses de benchmark. Ces dernières se reposent sur *Instructions-2M* (Morris *et al.*, 2023), un jeu de données contenant 2 millions d'amorces. *ShareGPT* (Zhang *et al.*, 2024d) est un jeu de données créé à partir des partages des utilisateurs de ChatGPT, contenant 54 000 amorces et *Unnatural Instructions* (Honovich *et al.*, 2023), un jeu de données synthétique, généré en utilisant le modèle text-davinci002 d'OpenAI avec des exemples de départ.

5 Limites des méthodes d'extraction

Malgré la diversité et la sophistication des approches d'extraction de données, celles-ci se heurtent à d'importantes limitations techniques. Nous discutons ici des principaux défis et obstacles qui

limitent l'efficacité de ces attaques, en nous focalisant sur *les difficultés liées à la recherche d'amorces optimales*, *la confusion entre généralisation et mémorisation*, et enfin *le manque de normalisation/standardisation des hyper-paramètres des méthodes* qui gêne sur la mise en œuvre de ces attaques.

Difficultés liées à la recherche d'amorces optimales L'extraction est dépendante du contexte, c'est à dire que même lorsque le modèle a mémorisé un contenu, encore faut-il que l'attaquant trouve le préfixe adéquat pour déclencher la génération du contenu mémorisé. Par exemple, Carlini *et al.* (2021b) remarquent qu'en fournissant l'amorce "3.14159" le décodage glouton génère 25 décimales de pi correctes contre 500 pour le décodage beam search. En revanche avec l'amorce (plus descriptive) "pi is 3.14159", le décodage glouton produit les 799 premiers chiffres de π corrects. Qui plus est en fournissant le contexte "e commence par 2.7182818, pi commence par 3.14159", GPT-2 complète par décodage glouton les 824 premiers chiffres de π. Cet exemple montre l'importance du contexte : avec la bonne amorce, il est possible d'extraire plusieurs ordres de grandeur de données de plus qu'avec une amorce légèrement sous-optimale.

Confusion entre généralisation et mémorisation Les données extraites ont tendance à être des textes mémorisés par le modèle car apparaissant plusieurs fois dans le jeu de données (Carlini *et al.*, 2021b). Pour les tâches de génération, les approches par extraction sont directement interprétables : un exemple généré est un signe de mémorisation. En revanche pour les tâches d'annotations (classification, reconnaissance d'entités, etc.) les méthodes par extraction sont plus difficilement interprétables, si le modèle génère le bon label associé à l'exemple à annoter, on peut conclure à une généralisation de la tâche d'annotation et non une contamination des données d'entraînement.

Manque de normalisation/standardisation des hyper-paramètres des méthodes Comme le remarquent Samuel *et al.* (2025), certains articles ne mentionnent pas les valeurs d'hyper-paramètres de leurs méthodes, comme les seuils utilisés pour déterminer la présence de contamination. Par exemple dans (Shi *et al.*, 2023), les auteurs ne donnent pas de seuil car ils se place dans un cadre où ils possèdent la vérité terrain et donc utilisent l'AUC mais dans les scénarii réels, il est impossible de détecter la contamination sans seuil.

6 Recommandations (arbre de décision selon le cas)

Pour sélectionner la méthode de détection de contamination dans les LLM qui convient à un praticien, plusieurs considérations pratiques doivent être prises en compte. Le choix dépend principalement de l'accès aux données d'entraînement et du niveau d'accès au modèle.

Cette section propose des recommandations concrètes pour guider les praticiens, en s'appuyant sur un arbre de décision illustré dans la figure 1, qui structure les options selon ces critères.

Accès aux données d'entraînement

Le premier critère à évaluer est l'accès aux données utilisées pour entraîner le modèle. Si nous avons accès aux données d'entraînement, les méthodes fondées sur la comparaison entre les données d'entraînement et les données de test sont les plus directes. La comparaison des chevauchements de n-grammes est une option simple et efficace. Cependant, elle est limitée face à des contaminations subtiles, comme les paraphrases, car elle repose sur une correspondance exacte. Le cas échéant, envisagez des méthodes comme l'utilisation des plongements et la comparaison entre plongements.

Sans accès aux données d'entraînement, les MIA par analyse des probabilités permettent d'estimer si un exemple de test a été vu par le modèle en analysant sa réponse (par exemple, une probabilité anormalement élevée). Ces méthodes reposent sur l'hypothèse que : les exemples vus durant l'entraînement auront une probabilité plus élevée que ceux non vus. De plus, selon certaines techniques nécessitent une référence ou non [2]. Les méthodes par extraction de données sont aussi une possibilité d'après la littérature, plus fiables que l'analyse des probabilités (Duan *et al.*, 2024b; Zhang *et al.*, 2024c). Le choix dépend du niveau d'accès au modèle.

Niveau d'accès au modèle

Si nous avons accès aux probabilités des tokens nous pouvons utiliser les méthodes par analyse de probabilités. Les méthodes MIN-K% (Shi *et al.*, 2023) et Min-K%++ (Zhang *et al.*, 2025) examinent statistiquement la distribution du modèle. Ces méthodes recherchent les tokens dans la distribution qui reçoivent une masse de probabilité anormalement élevée, ce qui pourrait indiquer que le modèle est très sûr parce qu'il a vu la séquence auparavant. De même, PaCoST (Zhang *et al.*, 2024b) ou LogProber (Yax *et al.*, 2024a) comparent la confiance du modèle sur les exemples de test réels par rapport à des exemples similaires synthétiques pour voir s'il existe un écart significatif, tandis que la méthode PAC – Polarized Augment Calibration (Ye *et al.*, 2024) utilise des échanges aléatoires dans l'exemple de test.

Sans cet accès, nous pouvons utiliser l'extraction de données d'entraînement. Ici le choix de la méthode va dépendre de l'information que l'on cherche à extraire. Nous pouvons utiliser des amorces pour extraire des verbatims des données d'entraînement (Su *et al.*, 2024; Nasr *et al.*, 2023; Golchin *et al.*, 2024), ou des amorces pour extraire des méta-données comme le nom et l'année du jeu de données associé aux exemples de test présents dans l'amorce (Sainz *et al.*, 2023). D'autres méthodes utilisant des amorces cherchent à évaluer la mémorisation des données présentées par questionnaires à choix multiples, tel que Data Contamination Quiz (Golchin & Surdeanu, 2023b) ou bien DE-COP (Duarte *et al.*, 2024) en présentant des exemples réels et modifiés et amorçant le modèle à reconnaître l'exemple réel.

Si l'on dispose de jeux de données pour entraîner un modèle d'inversion, ces méthodes permettent de se passer de l'utilisation du LLM et donc potentiellement de réduire les coûts de calcul. (Schwarzschild *et al.*, 2024) utilise une méthode d'inversion ainsi que le taux de compression adversarielle (Adversarial Compression Ratio) comme mesure de mémorisation. Ce taux de compression est proportionnel à la longueur de l'amorce obtenue par inversion.

7 Conclusion

Cet article propose une synthèse pratique sur la détection de la contamination des données d'entraînement dans les LLM, en explorant la taxonomie des formes de contamination, les approches de détection (classées par niveau d'accès et techniques), et les méthodes d'extraction de données. Il met en lumière un problème critique : la contamination fausse les évaluations des LLM en privilégiant la mémorisation au détriment de la généralisation, tout en posant des risques pour la confidentialité des données. Les méthodes d'extraction, notamment par amorçage et inversion, se révèlent prometteuses par rapport aux attaques traditionnelles d'inférence d'appartenance, bien qu'elles soient limitées par des taux d'extraction partiels, une dépendance au contexte, et des coûts computationnels élevés.

2. Toutes les hypothèses et pré-requis sont détaillés en annexe A

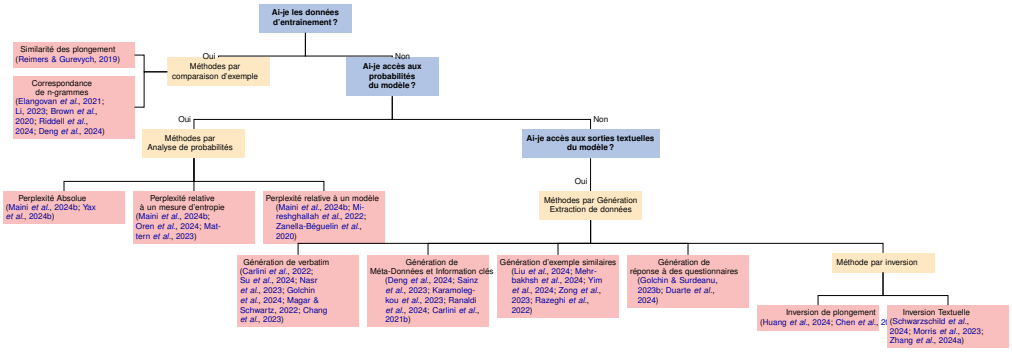

FIGURE 1 – Arbre de décision pour guider le choix de méthodes de détection de contamination dans les LLMs, en annexe B

L'apport principal de cette état de l'art réside dans son orientation pratique, illustrée par un arbre de décision qui guide les chercheurs et praticiens dans le choix des méthodes adaptées à leurs contraintes. En soulignant les défis actuels – comme le taux de faux positifs des MIA ou la difficulté d'extraire des données non mémorisées – l'article ouvre la voie à des travaux futurs visant à développer des techniques plus robustes et efficaces. Ainsi, il nous semble constituer une ressource intéressante pour garantir l'intégrité des évaluations des LLM et renforcer leur sécurité dans un paysage où leur usage ne cesse de croître.

Références

BROWN T., MANN B., RYDER N., SUBBIAH M., KAPLAN J. D., DHARIWAL P., NEELAKANTAN A., SHYAM P., SASTRY G., ASKELL A. *et al.* (2020). Language models are few-shot learners. *NeurIPS*, **33**, 1877–1901.

CARLINI N., CHIEN S., NASR M., SONG S., TERZIS A. & TRAMÈR F. (2021a). Membership inference attacks from first principles. *CoRR*, **abs/2112.03570**.

CARLINI N., IPPOLITO D., JAGIELSKI M., LEE K., TRAMER F. & ZHANG C. (2022). Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*.

CARLINI N., TRAMER F., WALLACE E., JAGIELSKI M., ARIEL HERBERT-VOSS, LEE K., ROBERTS A., BROWN T., SONG D., ERLINGSSON U., OPREA A. & RAFFEL C. (2020). Extracting Training Data from Large Language Models.

CARLINI N., TRAMER F., WALLACE E., JAGIELSKI M., HERBERT-VOSS A., LEE K., ROBERTS A., BROWN T., SONG D., ERLINGSSON U. *et al.* (2021b). Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, p. 2633–2650.

CHANG K., CRAMER M., SONI S. & BAMMAN D. (2023). Speak, memory : An archaeology of books known to chatgpt/gpt-4. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, p. 7312–7327.

CHEN Y., LENT H. & BJERVA J. (2024). Text Embedding Inversion Security for Multilingual Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computa-*

tional Linguistics (Volume 1 : Long Papers), p. 7808–7827, Bangkok, Thailand : Association for Computational Linguistics. DOI : 10.18653/v1/2024.acl-long.422.

CHEN Y., XU Q. & BJERVA J. (2025). ALGEN : Few-shot Inversion Attacks on Textual Embeddings using Alignment and Generation.

CHENG Y., CHANG Y. & WU Y. (2025). A Survey on Data Contamination for Large Language Models. DOI : 10.48550/arXiv.2502.14425.

CHOQUETTE-CHOO C. A., TRAMER F., CARLINI N. & PAPERNOT N. (2021). Label-only membership inference attacks. In *ICML*, p. 1964–1974 : PMLR.

DEEPSEEK-AI *et al.* (2024). DeepSeek-V3 Technical Report.

DEEPSEEK-AI *et al.* (2025). DeepSeek-R1 : Incentivizing Reasoning Capability in LLMs via Reinforcement Learning.

DENG C., ZHAO Y., TANG X., GERSTEIN M. & COHAN A. (2024). Investigating data contamination in modern benchmarks for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies (Volume 1 : Long Papers)*, p. 8698–8711.

DUAN M., SURI A., MIRESHGHALLAH N., MIN S., SHI W., ZETTLEMOYER L., TSVETKOV Y., CHOI Y., EVANS D. & HAJISHIRZI H. (2024a). Do Membership Inference Attacks Work on Large Language Models ?

DUAN M., SURI A., MIRESHGHALLAH N., MIN S., SHI W., ZETTLEMOYER L., TSVETKOV Y., CHOI Y., EVANS D. & HAJISHIRZI H. (2024b). Do membership inference attacks work on large language models ? In *Conference on Language Modeling (COLM)*.

DUARTE A. V., ZHAO X., OLIVEIRA A. L. & LI L. (2024). DE-COP : Detecting copyrighted content in language models training data. In *Forty-first International Conference on Machine Learning*.

ELANGOVAN A., HE J. & VERSPOOR K. (2021). Memorization vs. generalization : Quantifying data leakage in NLP performance evaluation. In P. MERLO, J. TIEDEMANN & R. TSARFATY, Éds., *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics : Main Volume*, p. 1325–1335, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.eacl-main.113.

ET AL. L. B. A. (2025). SmolLM2 : When Smol Goes Big – Data-Centric Training of a Small Language Model.

FU Y., UZUNER O., YETISGEN M. & XIA F. (2024). Does Data Contamination Detection Work (Well) for LLMs ? A Survey and Evaluation on Detection Assumptions.

GOLCHIN S. & SURDEANU M. (2023a). Data contamination quiz : A tool to detect and estimate contamination in large language models. *CoRR*, **abs/2311.06233**. DOI : 10.48550/ARXIV.2311.06233.

GOLCHIN S. & SURDEANU M. (2023b). Data contamination quiz : A tool to detect and estimate contamination in large language models. *CoRR*, **abs/2311.06233**. DOI : 10.48550/ARXIV.2311.06233.

GOLCHIN S. & SURDEANU M. (2024). Time travel in llms : Tracing data contamination in large language models.

GOLCHIN S. & SURDEANU M. (2025). Data contamination quiz : A tool to detect and estimate contamination in large language models.

GOLCHIN S., SURDEANU M., BETHARD S., BLANCO E. & RILOFF E. (2024). Memorization in In-Context Learning.

GRATTAFIORI A. *et al.* (2024). The Llama 3 Herd of Models. DOI : 10.48550/arXiv.2407.21783.

HE Y., LI B., LIU L., BA Z., DONG W., LI Y., QIN Z., REN K. & CHEN C. (2025). Towards label-only membership inference attack against pre-trained large language models.

HONOVICH O., SCIALOM T., LEVY O. & SCHICK T. (2023). Unnatural Instructions : Tuning Language Models with (Almost) No Human Labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 14409–14428, Toronto, Canada : Association for Computational Linguistics. DOI : 10.18653/v1/2023.acl-long.806.

HUANG Y.-H., TSAI Y., HSIAO H., LIN H.-Y. & LIN S.-D. (2024). Transferable Embedding Inversion Attack : Uncovering Privacy Risks in Text Embeddings without Model Queries. In L.-W. KU, A. MARTINS & V. SRIKUMAR, Éds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 4193–4205, Bangkok, Thailand : Association for Computational Linguistics. DOI : 10.18653/v1/2024.acl-long.230.

ISHIHARA S. (2023). Training data extraction from pre-trained language models : A survey. In *Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023)*, p. 260–275.

KARAMOLEGKOU A., LI J., ZHOU L. & SØGAARD A. (2023). Copyright violations and large language models. In H. BOUAMOR, J. PINO & K. BALI, Éds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, p. 7403–7412, Singapore : Association for Computational Linguistics. DOI : 10.18653/v1/2023.emnlp-main.458.

LI Y. (2023). An open source data contamination report for llama series models. *arXiv preprint arXiv :2310.17589*.

LIU C., JIN R., STEEDMAN M. & XIONG D. (2024). Evaluating Chinese large language models on discipline knowledge acquisition via memorization and robustness assessment. In O. SAINZ, I. GARCÍA FERRERO, E. AGIRRE, J. ANDER CAMPOS, A. JACOVI, Y. ELAZAR & Y. GOLDBERG, Éds., *Proceedings of the 1st Workshop on Data Contamination (CONDA)*, p. 1–12, Bangkok, Thailand : Association for Computational Linguistics.

MAGAR I. & SCHWARTZ R. (2022). Data contamination : From memorization to exploitation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, p. 157–165.

MAINI P., JIA H., PAPERNOT N. & DZIEDZIC A. (2024a). LLM Dataset Inference : Did you train on my dataset ?

MAINI P., JIA H., PAPERNOT N. & DZIEDZIC A. (2024b). Llm dataset inference : Did you train on my dataset ? *The 1st Workshop on Data Contamination (CONDA)*.

MATTERN J., MIRESHGHALLAH F., JIN Z., SCHOELKOPF B., SACHAN M. & BERG-KIRKPATRICK T. (2023). Membership inference attacks against language models via neighbourhood comparison. In *Findings of the Association for Computational Linguistics : ACL 2023*, p. 11330–11343.

MEHRBAKHSH B., GARIGLIOTTI D., MARTÍNEZ-PLUMED F. & HERNANDEZ-ORALLO J. (2024). Confounders in instance variation for the analysis of data contamination. In O. SAINZ, I. GARCÍA FERRERO, E. AGIRRE, J. ANDER CAMPOS, A. JACOVI, Y. ELAZAR & Y. GOLDBERG, Éds., *Proceedings of the 1st Workshop on Data Contamination (CONDA)*, p. 13–21, Bangkok, Thailand : Association for Computational Linguistics.

MIRESHGHALLAH F., GOYAL K., UNIYAL A., BERG-KIRKPATRICK T. & SHOKRI R. (2022). Quantifying privacy risks of masked language models using membership inference attacks. In Y. GOLDBERG, Z. KOZAREVA & Y. ZHANG, Éds., *Proceedings of the 2022 Conference on Empirical*

Methods in Natural Language Processing, p. 8332–8347, Abu Dhabi, United Arab Emirates : Association for Computational Linguistics. DOI : 10.18653/v1/2022.emnlp-main.570.

MORRIS J. X., ZHAO W., CHIU J. T., SHMATIKOV V. & RUSH A. M. (2023). Language Model Inversion. In *The Twelfth International Conference on Learning Representations*.

NASR M., CARLINI N., HAYASE J., JAGIELSKI M., COOPER A. F., IPPOLITO D., CHOQUETTE-CHOO C. A., WALLACE E., TRAMÈR F. & LEE K. (2023). Scalable Extraction of Training Data from (Production) Language Models. DOI : 10.48550/arXiv.2311.17035.

OPENAI *et al.* (2023). GPT-4 Technical Report.

OREN Y., MEISTER N., CHATTERJI N. S., LADHAK F. & HASHIMOTO T. (2024). Proving test set contamination in black-box language models. In *The Twelfth International Conference on Learning Representations*.

PALAVALLI M., BERTSCH A. & GORMLEY M. (2024a). A taxonomy for data contamination in large language models. In O. SAINZ, I. GARCÍA FERRERO, E. AGIRRE, J. ANDER CAMPOS, A. JACOVI, Y. ELAZAR & Y. GOLDBERG, Éds., *Proceedings of the 1st Workshop on Data Contamination (CONDA)*, p. 22–40, Bangkok, Thailand : Association for Computational Linguistics.

PALAVALLI M., BERTSCH A. & GORMLEY M. R. (2024b). A Taxonomy for Data Contamination in Large Language Models. DOI : 10.48550/arXiv.2407.08716.

RANALDI F., RUZZETTI E. S., ONORATI D., RANALDI L., GIANNONE C., FAVALLI A., ROMAGNOLI R. & ZANZOTTO F. M. (2024). Investigating the impact of data contamination of large language models in text-to-sql translation. *arXiv preprint arXiv :2402.08100*.

RAVAUT M., DING B., JIAO F., CHEN H., LI X., ZHAO R., QIN C., XIONG C. & JOTY S. (2024). How Much are Large Language Models Contaminated ? A Comprehensive Survey and the LLMSanitize Library.

RAZEGHI Y., LOGAN IV R. L., GARDNER M. & SINGH S. (2022). Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics : EMNLP 2022*, p. 840–854.

REIMERS N. & GUREVYCH I. (2019). Sentence-BERT : Sentence embeddings using Siamese BERT-networks. In K. INUI, J. JIANG, V. NG & X. WAN, Éds., *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, p. 3982–3992, Hong Kong, China : Association for Computational Linguistics. DOI : 10.18653/v1/D19-1410.

RIDDELL M., NI A. & COHAN A. (2024). Quantifying contamination in evaluating code generation capabilities of language models. In L.-W. KU, A. MARTINS & V. SRIKUMAR, Éds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 14116–14137, Bangkok, Thailand : Association for Computational Linguistics. DOI : 10.18653/v1/2024.acl-long.761.

SABLAYROLLES A., DOUZE M., SCHMID C., OLLIVIER Y. & JÉGOU H. (2019). White-box vs black-box : Bayes optimal strategies for membership inference. In *ICML*, p. 5558–5567 : PMLR.

SAINZ O., CAMPOS J. A., GARCÍA-FERRERO I., ETXANIZ J. & AGIRRE E. (2023). Did chatgpt cheat on your test ? Accessed : 2024-09-09.

SAMUEL V., ZHOU Y. & ZOU H. P. (2025). Towards data contamination detection for modern large language models : Limitations, inconsistencies, and oracle challenges. In O. RAMBOW, L. WANNER, M. APIDIANAKI, H. AL-KHALIFA, B. D. EUGENIO & S. SCHOCKAERT, Éds., *Proceedings of the 31st International Conference on Computational Linguistics*, p. 5058–5070, Abu Dhabi, UAE : Association for Computational Linguistics.

SANKARARAMAN S., OBOZINSKI G., JORDAN M. I. & HALPERIN E. (2009). Genomic privacy and limits of individual detection in a pool. *Nature genetics*, **41**(9), 965–967.

SATVATY A., VERBERNE S. & TURKMEN F. (2024). Undesirable memorization in large language models : A survey. *ArXiv*, **abs/2410.02650**.

SCHWARZSCHILD A., FENG Z., MAINI P., LIPTON Z. C. & KOLTER J. Z. (2024). Rethinking llm memorization through the lens of adversarial compression. *The 1st Workshop on Data Contamination (CONDA)*.

SHI W., AJITH A., XIA M., HUANG Y., LIU D., BLEVINS T., CHEN D. & ZETTLEMOYER L. (2023). Detecting pretraining data from large language models. In *NeurIPS 2023 Workshop on Regulatable ML*.

SHOKRI R., STRONATI M., SONG C. & SHMATIKOV V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, p. 3–18 : IEEE.

SKAPARS A., MANINO E., SUN Y. & CORDEIRO L. C. (2024). Was it Slander ? Towards Exact Inversion of Generative Language Models. DOI : 10.48550/arXiv.2407.11059.

STERNAK T., RUNJE D., GRANOŠA D. & WANG C. (2025). Automating Prompt Leakage Attacks on Large Language Models Using Agentic Approach. DOI : 10.48550/arXiv.2502.12630.

SU E., VELLORE A., CHANG A., MURA R., NELSON B., KASSIANIK P. & KARBASI A. (2024). Extracting Memorized Training Data via Decomposition.

WANG J., YANG T., XIE R. & DHINGRA B. (2024). Raccoon : Prompt Extraction Benchmark of LLM-Integrated Applications. In *Findings of the Association for Computational Linguistics ACL 2024*, p. 13349–13365. DOI : 10.18653/v1/2024.findings-acl.791.

XU C., GUAN S., GREENE D., KECHADI M. *et al.* (2024a). Benchmark data contamination of large language models : A survey. *arXiv preprint arXiv :2406.04244*.

XU C., GUAN S., GREENE D. & M-TAHAR KECHADI (2024b). Benchmark Data Contamination of Large Language Models : A Survey.

XU R., WANG Z., RUN-ZE FAN & LIU P. (2024c). Benchmarking Benchmark Leakage in Large Language Models.

XU Z., LIU Y., DENG G., LI Y. & PICEK S. (2024d). A comprehensive study of jailbreak attack versus defense for large language models. In L.-W. KU, A. MARTINS & V. SRIKUMAR, Éds., *Findings of the Association for Computational Linguistics : ACL 2024*, p. 7432–7449, Bangkok, Thailand : Association for Computational Linguistics. DOI : 10.18653/v1/2024.findings-acl.443.

YAX N., OUDEYER P.-Y. & PALMINTERI S. (2024a). Assessing contamination in large language models : Introducing the logprober method.

YAX N., PIERRE-YVES OUDEYER & PALMINTERI S. (2024b). Assessing Contamination in Large Language Models : Introducing the LogProber method.

YE W., HU J., LI L., WANG H., CHEN G. & ZHAO J. (2024). Data contamination calibration for black-box LLMs. In L.-W. KU, A. MARTINS & V. SRIKUMAR, Éds., *Findings of the Association for Computational Linguistics : ACL 2024*, p. 10845–10861, Bangkok, Thailand : Association for Computational Linguistics. DOI : 10.18653/v1/2024.findings-acl.644.

YEOM S., GIACOMELLI I., FREDRIKSON M. & JHA S. (2018). Privacy risk in machine learning : Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, p. 268–282 : IEEE.

YIM W.-W., FU Y., BEN ABACHA A. & YETISGEN M. (2024). To err is human, how about medical large language models ? comparing pre-trained language models for medical assessment errors and

reliability. In N. CALZOLARI, M.-Y. KAN, V. HOSTE, A. LENCI, S. SAKTI & N. XUE, Éds., *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, p. 16211–16223, Torino, Italia : ELRA and ICCL.

ZANELLA-BÉGUELIN S., WUTSCHITZ L., TOPLE S., R¨UHLE V., PAVERD A., OHRIMENKO O., K¨OPF B. & BROCKSCHMIDT M. (2020). Analyzing information leakage of updates to natural language models. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, p. 363–375.

ZHANG C., MORRIS J. X. & SHMATIKOV V. (2024a). Extracting Prompts by Inverting LLM Outputs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, p. 14753–14777, Miami, Florida, USA : Association for Computational Linguistics. DOI : 10.18653/v1/2024.emnlp-main.819.

ZHANG H., LIN Y. & WAN X. (2024b). PaCoST : Paired confidence significance testing for benchmark contamination detection in large language models. In Y. AL-ONAIZAN, M. BANSAL & Y.-N. CHEN, Éds., *Findings of the Association for Computational Linguistics : EMNLP 2024*, p. 1794–1809, Miami, Florida, USA : Association for Computational Linguistics. DOI : 10.18653/v1/2024.findings-emnlp.97.

ZHANG J., DAS D., KAMATH G. & TRAMÈR F. (2024c). Membership Inference Attacks Cannot Prove that a Model Was Trained On Your Data.

ZHANG J., SUN J., YEATS E., OUYANG Y., KUO M., ZHANG J., YANG H. F. & LI H. (2025). Min-k%++ : Improved baseline for pre-training data detection from large language models. In *The Thirteenth International Conference on Learning Representations*.

ZHANG Y., CARLINI N. & IPPOLITO D. (2024d). Effective Prompt Extraction from Language Models. DOI : 10.48550/arXiv.2307.06865.

ZHOU X., WEYSSOW M., WIDYASARI R., ZHANG T., HE J., LYU Y., CHANG J., ZHANG B., HUANG D. & LO D. (2025). LessLeak-Bench : A First Investigation of Data Leakage in LLMs Across 83 Software Engineering Benchmarks.

ZONG Y., YU T., ZHAO B., CHAVHAN R. & HOSPEDALES T. (2023). Fool your (vision and) language model with embarrassingly simple permutations. *arXiv preprint arXiv :2310.01651*.

A Détail complet des hypothèses et pré-requis

Ici nous détaillons les hypothèses et pré-requis pour chaque méthodes de détection de contamination. Le texte ci-après est une traduction de (Fu *et al.*, 2024).

Méthodes basées sur la similarité entre exemples

Dans ces méthodes, on compare les exemples à suspects au données d'entraînement. Ici, on ne fait pas d'hypothèses particulières.

Pré-requis R1. Les jeux de données d'entraînement doivent être **publiés**.

Pré-requis R2. Les jeux de données d'entraînement doivent être **accessibles**, i.e aucune limitations par des contraintes légales, de confidentialité ou des liens web expirés.

Méthodes basées sur les probabilités

Dans ces méthodes, on calcule et utilisent les probabilités issues du LLM. On peut utiliser les probabilités des tokens de manière absolue.

Hypothèse A1. Les exemples vus auront une probabilité plus élevée que ceux non vus, et il existe un seuil, ξ_p , séparant les exemples vus des non vus.

On peut utiliser les probabilités des tokens des exemples similaires et ainsi devoir faire l'hypothèse suivante et nécessiter le pré-requis suivant :

Pré-requis R3. Il existe un algorithme permettant, étant donné un exemple x et un modèle M , de générer automatiquement un exemple similaire non vue.

Hypothèse A2. Si x et x' sont similaires et que le modèle M a vu x mais pas x', alors la probabilité de x devrait être beaucoup plus élevée que celle de x' selon le modèle M .

Une ensemble de méthode utilisant les probabilités d'un modèle de référence exiges le pré-requis et l'hypothèse suivante :

Pré-requis R4. Pour un exemple x , il doit avoir un autre modèle $M\prime$ pour lequel x est inconnu.

Hypothèse A3. Si x est vu par M mais pas par M', alors $P_M(x)$ devrait être beaucoup plus élevé que $P_{M'}(x)$.

Méthodes basées sur la génération (extraction de données)

Dans les méthodes basées sur la génération, on traite l'exemple x comme une paire préfixe-suffixe $x = (x_p, x_s)$. On conditionne la génération du modèle de langue M par x_p. Si le résultat M (x_p) est identique ou similaire à x_s alors on dit que x à été vu à l'entraînement.

Cette catégorie de méthode repose sur l'hypothèse suivante :

Hypothèse A4. Étant donné une paire préfixe-suffixe $x = (x_p, x_s)$, si x a été vue par un modèle de langue M , alors x_s peut être généré par M à partir de l'entrée x_p via un décodage glouton (greedy decoding).

D'autres méthode cherchent à générer des informations de x grâce au contexte. Un exemple est la génération des annotations grâce à l'exemple de départ comme les travaux de (Magar & Schwartz, 2022).

Pré-requis R5. Toute instance x peut être reformulée en une paire remplissant des champs contextuels, $x = (x_c, x_k)$, où x_k est l'information clé, et x_c est son contexte.

Hypothèse A5. Si x est vu, M sera capable de produire une sortie similaire à x_k à partir de l'entrée x_c.

Des méthodes alternatives reposent sur comment le modèle génère des exemples variés sous différents algorithmes de décodage. Ces méthodes reposent sur l'hypothèse suivante :

Hypothèse A6. Supposons qu'un exemple x soit la composition d'une paire préfixe-suffixe, $x = (x_p, x_s)$. Si un modèle de langue M a vu x, alors, étant donné x_p, M générera une sortie identique ou similaire à x_s sous différentes stratégies de décodage. La variation des générations sera borné par un seuil ξ_p .

Une variante reposent sur la génération d'exemple à partir de métadonnées sur le jeu de données suspect. Par exemple la génération d'exemple de CONLL à partir du nom et de la date du jeu de données (Sainz *et al.*, 2023). L'hypothèse et le pré-requis sont les suivants :

Pré-requis R6. À partir d'un jeu de données D , on peut créer une entrée x_m qui inclut les métadonnées M du jeu de données, comme le nom, la répartition, et le format.

Hypothèse A7. Si un modèle de langue M a vu un jeu de données D , alors, lorsqu'on lui fournit les métadonnées de D , M est capable de générer des exemples très similaires à certaines exemples $x \in D$.

Méthodes basées sur la mémorisation des réponses

Hypothèse A8. Supposons que deux jeux de données D et D' soient similaires, et qu'un modèle M ait vu D mais pas D'. Alors, la performance de M sur D (Eval(M, D)) sera beaucoup plus élevée que sa performance sur D' (Eval(M, D')).

Pré-requis R7. Étant donné un modèle M et un jeu de données d'évaluation D , il est possible de générer un jeu de données similaire D' destiné au même objectif, mais légèrement modifié, pour évaluer la contamination potentielle.

B Arbre de décision

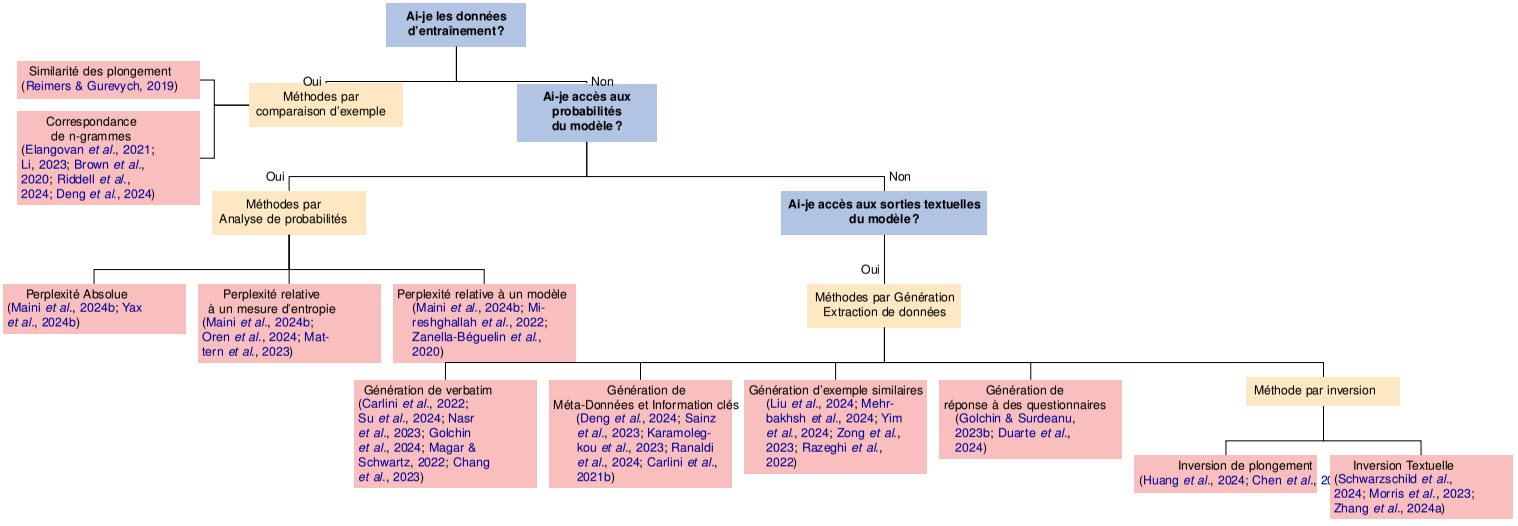

FIGURE 2 – arbre de décision pour guider le choix de méthodes de détection de contamination dans les LLMs