LIN Florent

ENSAE 3ème année
Stage d'application
Année scolaire 2023-2024

Rollback Learning

Aubay INNOV
Boulogne-Billancourt

Maître de stage : Maxime CLEMENCE
05/08/2024 au 29/11/2024

ENSAE Paris

TSA 26644

Service des relations entreprises et des stages
5, avenue Henry Le Chatelier - 91764 PALAISEAU CEDEX - FRANCE - Tél : +33 (0)1 70 26 67 39 - Courriel : stage@ensae.fr

Remerciements

Je tiens à exprimer ma gratitude envers mes encadrants, Monsieur Antoine Bosment et Monsieur Maxime Clémence, qui m'ont offert leur expertise et leurs conseils tout au long de ce stage. Leur disponibilité et leur rigueur m'ont permis de progresser avec confiance dans ce projet, et leurs retours ont été précieux pour mener à bien ce travail.

Je remercie également les membres de mon groupe, avec qui j'ai eu la chance de collaborer dans un esprit de partage et d'entraide. Leur engagement et leur enthousiasme ont fait de ce stage une expérience enrichissante, tant sur le plan professionnel que personnel.

Enfin, je tiens à remercier toute l'équipe d'Aubay pour l'accueil chaleureux et l'environnement de travail stimulant. Je suis reconnaissant d'avoir eu l'opportunité de travailler dans un cadre aussi enrichissant.

Table des matières

1 Introduction

Les modèles d'intelligence artificielle, en particulier les LLMs (Large Language Models), ont tendance à apprendre des informations privées, erronées ou dangereuses, entraînés sur d'énormes quantités de données provenant de tous les recoins d'internet. Cela peut entraîner des violations de la vie privée, car l'IA pourrait générer ou rappeler ces données lors de ses interactions. Par exemple, il y a eu des cas avec *Github* où des clés API privées, des mots de passe et des informations sensibles ont été involontairement inclus dans des codes publics, ou encore avec *Reddit* où des données personnelles et des conversations ont été incluses dans des ensembles de données d'entraînement. De plus, les modèles peuvent également apprendre à partir de données inexactes, ce qui conduit à la diffusion de désinformation ou de biais pouvant renforcer des problèmes sociétaux. Enfin, des questions se posent quant aux réglementations sur la diffusion d'information, comme le droit à l'oubli ou la RGPD en Europe.

En conséquence, le besoin de désapprentissage, qui est le processus de suppression ou de correction des connaissances ou concepts acquis par une IA générative (i.e. modèle de génération de texte, d'image, de son...), est devenu critique. Le désapprentissage garantirait que les systèmes d'IA puissent être corrigés même après l'entraînement, en supprimant ou en réduisant efficacement les informations dangereuses, erronées ou sensibles au sein de son corpus de connaissances, contribuant ainsi à protéger la vie privée des utilisateurs. Ainsi, l'enjeu principal de cette discipline est de trouver une méthode pour supprimer les informations à oublier et d'obtenir un modèle, après désapprentissage, équivalent à un modèle que l'on aurait complètement réentrainé de zero sans les données dites "à oublier", et uniquement avec le reste des données qualifiées comme "à retenir".

La problématique qui se développe naturellement est de déterminer dans quelle mesure il est possible de mettre au point une méthode d'oubli reproduisant des résultats similaires à un réapprentissage complet du modèle sans les données à oublier.

Afin de répondre à cette problématique, notre équipe de trois jeunes étudiants a travaillé sur le projet intitulé **Eco-Plus** basé sur la méthode *ECO* de Liu et al. [1]. Nous allons, dans un premier temps, définir le cadre du problème, afin d'aboutir à sa formalisation mathématique. Un travail préliminaire d'étude de l'état de l'art a été réalisé, et nous permettra de mettre au point un protocole visant à la fois à implémenter une méthode de désapprentissage existante, mais aussi de mesurer ses performances. En parallèle, il nous faudra prendre en considération les problèmes de coûts computationnels liés limites matérielles et techniques. Pour clore ce rapport, nous aborderons les démarches que nous avons entreprises afin peaufiner la méthode de désapprentissage étudiée.

2 La Mission

Dans la première section de ce rapport, nous allons expliciter le principe et les grands enjeux actuels autour du Machine Unlearning. Ensuite, nous définirons les concepts clés de cette discipline pour formaliser le problème. Enfin, nous introduirons deux benchmarks qui soutiendront l'analyse des résultats.

2.1 Contexte de la Mission

Cette partie a pour but d'apporter plus de clarté sur l'environnement de travail au sein de l'entreprise Aubay et de nous permettre d'introduire succinctement les événements qui ont eu lieu au cours du stage.

2.1.1 Introduction à l'Entreprise

Aubay est une entreprise de services numériques et une société de conseil. Elle possède un laboratoire dédié à la recherche, nommé **Aubay INNOV**, où j'ai eu l'opportunité de réaliser mon stage sur le sujet intitulé **Rollback Learning**. J'ai également eu la chance de travailler sur ce sujet au sein d'un groupe de trois étudiants, moi inclus, sous la supervision de deux encadrants qui nous ont guidés à chaque étape de notre projet. Notre responsabilité était de concevoir un livrable final pour faire état de nos travaux et les appliquer dans des contextes concrets.

Nos encadrants nous ont guidés à chaque étape de ce processus, notamment au cours de réunions quotidiennes de quinze minutes, appelées *"Daily"*, où nous faisions état de notre avancement et de nos éventuels blocages. Nous avions également des réunions hebdomadaires de 45 minutes, nommées *"Comité"*, durant lesquelles nous présentions nos résultats et nos idées à discuter sous forme de diapositives *PowerPoint*.

2.1.2 Contexte du Stage

Le stage a eu lieu entre le 4 août 2024 et le 28 novembre de la même année. Ainsi, à la date de rendu du rapport, le stage est toujours en cours.

Mes circonstances étaient relativement particulières. À mon arrivée le 4 août, un groupe travaillait déjà sur le projet *Rollback Learning* depuis mars 2024. Les personnes avec qui je devais former un nouveau groupe pour mener notre projet ne devaient arriver que le 2 septembre. Ainsi, durant le premier mois de mon stage, j'ai eu l'occasion de travailler avec le groupe déjà en place sur leur sujet intitulé *Sasmu*. Celui-ci visait dans un premier temps à faire oublier des concepts à des modèles de classification d'image, puis à adapter leur méthode de désapprentissage sur un modèle de génération d'image.

Cette expérience fut formatrice, m'apprenant à rapidement m'intégrer à un projet bien avancé et à m'approprier de nouveaux outils dans le but d'apporter une contribution personnelle. Cependant, nous n'élaborerons pas davantage sur cette période, car elle risquerait d'ajouter trop de confusion dans ce rapport de stage.

Ainsi, la suite du rapport rapporte les événements qui se sont déroulés à l'arrivée de mes deux nouveaux collègues, entre le 2 septembre et le 4 novembre.

2.1.3 La Mission

L'un des aspects centraux de la mission qui m'a été confiée est la grande liberté offerte pour la définition précise de la problématique. Notre travail pouvait être centré autour des modèles manipulant des images, du son ou du texte, et nous aurions pu choisir d'étudier aussi bien des modèles de génération que de classification.

Bien que relativement récent, le Machine Unlearning est un domaine très vaste. Afin de déterminer le chemin que notre groupe allait emprunter, notre première étape consistait en la rédaction d'un **état de l'art**, qui a duré du 2 au 27 septembre. D'emblée, nous avons décidé de nous restreindre aux *méthodes intrinsèques* [2] de désapprentissage sur des *LLMs*. Cela contrastait totalement avec le projet précédent qui portait sur des méthodes agnostiques destinées à des modèles de génération d'images. Nous avons ainsi abouti à la rédaction d'un document de plus de 70 pages, couvrant un large éventail de méthodes autour du sujet choisi.

Ce document nous a permis non seulement d'élaborer une problématique et d'affiner notre compréhension du sujet, mais aussi de prendre conscience de la réalité du terrain pour un projet de recherche en Machine Learning. Il est par exemple essentiel de considérer les limites matérielles, ainsi que les coûts réels d'un projet, en termes de temps et de ressources financières.

Nous avons ensuite démarré la phase de **preuve de concept**. Notre mission durant cette phase consistait à implémenter une ou plusieurs méthodes de désapprentissage choisies lors de la phase précédente, et d'y apporter notre propre contribution pour élaborer une réponse à la problématique posée. Le choix de la méthode s'est en partie fait en fonction de nos contraintes. Premièrement, nous avons écarté toutes les méthodes modifiant directement les poids des modèles, car trop coûteuses. Ensuite, il nous fallait trouver des méthodes applicables sur des modèles de langage (LM) de très petite taille. Par très petite taille, nous considérons moins de 1 milliard de paramètres. Notons que l'idée du déroulement du projet était avant tout de travailler sur nos machines locales, avant d'utiliser des outils en ligne disposant d'avantage de puissance de calcul afin de travailler sur des modèles plus lourds.

Ainsi, nous avions deux possibilités : utiliser une méthode agissant sur l'entrée du modèle (*les embeddings*) ou d'autres basées sur des formules closes. Bien qu'il aurait été intéressant de travailler sur ces dernières, elles sont malheureusement conçues pour des tâches spécifiques (par exemple, désapprendre les préjugés entre le métier et le genre d'une personne), ce qui ne s'alignait pas avec notre souhait de traiter un problème de désapprentissage plus général. Nous avons donc choisi la méthode ECO [1].

Enfin, la dernière phase de notre projet aurait dû être son **déploiement**, mais nous n'avons malheureusement pas encore eu le temps de la réaliser à la date de rendu du rapport.

2.2 Enjeux du Machine Unlearning

2.2.1 Définitions

Commençons par définir les termes clés. Dans notre contexte, que signifie ”*désapprendre*” des ”*données*” pour nos modèles ?

Premièrement, par **désapprendre**, on entend généralement le processus visant à effacer l'influence de certaines données du modèle, en évitant son réentrainement complet. En effet, la méthode naïve consisterait à réentrainer intégralement le modèle sans les données à que l'on cherche à désapprendre, également appelées *données à oublier*. Cependant, plus que simplement effacer des données, il s'agit aussi de prendre en compte les biais induits par celles-ci [1], tout en préservant le reste des connaissances acquises. Il est donc crucial de **(1)** bien identifier les données à désapprendre et de comprendre leurs impacts sur le modèle ; de **(2)** disposer de méthodes fiables pour valider le succès de l'opération ; et de **(3)** maintenir la performance du modèle sur ses autres données d'entaînement.

Les questions suivantes se posent alors, communes aux sujets du Machine Unlearning et de l'IA générative en général : comment décrire les **informations** à désapprendre ? Et comment sont stockées les **données** dans une IA ?

La notion de **donnée** est ambigüe et doit être considérée au sens large car, en Machine Unlearning, les notions d'information, de donnée ou de concept sont souvent étroitement liés. Par exemple, il est relativement facile de visualiser un numéro de téléphone ou une adresse comme étant une donnée. Or, certaines IA semblent comprendre des concepts comme les armes, la violence ou la nudité. Pourtant, lors de son entraînement, l'IA n'a pas ”appris” ces concepts, mais uniquement des données en rapport avec ceux-ci. De ce fait, ils sont plus difficiles à formaliser dans un processus d'oubli pour un modèle génératif.

Il est également important de distinguer plusieurs types de données. Tout d'abord, les **données à oublier** (ou *forget data*) désignent les informations spécifiques que l'on souhaite retirer du modèle. Nous noterons cet ensemble de données à oublier D_f.

Les données à retenir, quant à elles, sont les **données restantes** ou **données à retenir** (*retain data*) après le retrait de données à oublier. Elles forment l'ensemble $D_r = D \setminus D_f$, où D désigne l'ensemble de **données d'origine** souvent appelées **données d'entraînement** (voir *données de pré-entraînement*).

En plus de ces ensembles, nous avons les données de sondage (D_{probe}), qui sont souvent utilisées pour tester l'impact d'une attaque sur le modèle désappris afin d'évaluer l'efficacité du désapprentissage. Ces données permettent à vérifier que le modèle ne fait plus référence aux données oubliées.

Enfin, le modèle désappris, noté $\mathcal{U}(M, D, D_f)$, est le modèle ajusté après le processus de désapprentissage. Il est souvent comparé au modèle réentraîné à partir de zéro sur les données

1. La simple suppression de données d'entraînement n'élimine pas les ajustements appris ; les modèles peuvent conserver des biais issus de ces données.

à retenir, $\mathcal{A}(D \setminus D_f)$, servant de référence. Cette comparaison permet d'évaluer si le désapprentissage a été réalisé de manière satisfaisante, c'est à dire si les deux modèles produisent des résultats similaires.

2.3 Menaces

Les concepts de *menace* et *modèles de menace* (ou *modèles adverses)* [1, 2] sont fondamentaux, car ils permettent de mieux comprendre l'enjeu du Machine Unlearning.

Nous considérons un cadre de boîte grise, où les utilisateurs interagissent avec un LLM via une interface de chat ou un accès API. Dans ce cadre, tous les utilisateurs peuvent envoyer des *prompts* ou requêtes au LLM et recevoir les complétions correspondantes. Nous supposons également que des *adversaires* au sein du groupe d'utilisateurs génèrent des prompts dans la distribution et tentent de contourner le mécanisme de protection, en utilisant des **modèles de menace**. Ils cherchent à extraire les données à oublier des **modèles désappris**.

Nous considérons les **menaces** dans trois catégories :

Fuite d'entité La fuite d'entité se produit lorsqu'un LLM divulgue par inadvertance l'identité ou des informations sensibles d'individus spécifiques.

Connaissances dangereuses Étant donné la facilité d'utilisation et l'accessibilité des LLMs commerciaux et open-source (comme Chat-GPT), des individus mal intentionnés pourraient exploiter leurs capacités pour acquérir des connaissances dangereuses (par exemple pour la fabrication d'armes).

Contenu protégé par des droits d'auteur Même si le contenu protégé est normalement filtré lors de l'entraînement initial du modèle, des fragments de texte peuvent encore être dispersés dans le corpus d'entraînement.

Notre rôle a alors été de chercher comment, dans des cas concrets, nous pouvions tester la robustesse d'un désapprentissage face à ces différents types de modèles de menace. Ainsi, durant la rédaction de notre état de l'art, notre groupe a cherché à collecter des ensembles de données et des métriques qui permettraient de mesurer la robustesse de nos méthodes de désapprentissage. Nous avons trouvé quatre benchmarks disponibles en ligne et, sur les conseils de nos encadrants, nous nous sommes concentrés uniquement sur deux benchmarks (que vous trouverez section 2.5) traitant des problèmes de *"Connaissances dangereuses"* et de *"Contenu protégé par des droits d'auteur"*. En effet, ils nous conseillèrent d'abord de vérifier la faisabilité de bout en bout de notre méthode avant de nous lancer dans une série d'expériences.

2.4 Définition du Problème

Cette partie de formalisation du problème reprend des aspects abordés en cours d'*Introduction au Machine Learning* ainsi qu'en cours d'*Advanced Machine Learning*, notamment concernant les **fonctions d'optimisation** que nous allons explorer plus en détail par la suite. Cela a grandement contribué à une assimilation plus rapide et plus approfondie des concepts étudiés.

Nous allons tout d'abord préciser quel est notre objectif de désapprentissage, il existe en réalité deux types de désapprentissages :

Le *désapprentissage approximatif*, qui se concentre sur la réduction des coûts computationnels et de stockage au détriment d'une faible "erreur" ou "différence" avec le modèle réentraîné.

Et le *désapprentissage exact*, qui vise à reproduire un modèle oublié équivalent au modèle réentraîné naïvement, dont la définition se trouve en annexe A.1. Nos encadrants nous ont informés que cette approche pour le désapprentissage serait difficilement applicable avec le matériel à disposition. Nous avons donc pris la décision de travailler avec des techniques approximatives.

Le Désapprentissage Approximatif Les études sur le désapprentissage approximatif visent à désapprendre un modèle pour qu'il se *rapproche* du modèle réentraîné sur l'ensemble de données (restantes et oubliées). Aussi, le désapprentissage nécessite uniquement le modèle d'origine et des données à effacer. Cependant, comme le désapprentissage approximatif essaie de supprimer directement l'influence des échantillons désappris à partir des modèles entraînés, un problème majeur réside dans l'estimation et la suppression précises de cette contribution.

Ce dernier point a été à l'origine de nombreux problèmes et de discussions entre mon groupe et nos encadrants, que nous allons préciser dans les prochaines sections du rapport. C'est aussi pour cette raison qu'il est nécessaire de définir un cadre de travail clair pour que nos algorithmes produisent les résultats attendus.

2.4.1 Formalisation Mathématique

Afin de définir clairement ce problème, nous nous sommes appuyés sur plusieurs acticles [1, 2, 3, 4] et avons rassemblé les notations suivantes :

Symbole	Description	Symbole	Description
$\mathcal{Z}$	Espace des données	$\mathcal{H}$	Espace des paramètres
$D = (X, Y)$	Ensemble d'entraînement	$\mathcal{A}(\cdot)$	Algorithme d'apprentissage automatique
$D_f = (X_f, Y_f)$	Ensemble à oublier	$\mathcal{U}(\cdot)$	Algorithme de désapprentissage
$D_r = D \setminus D_f$	Ensemble à retenir	M	Modèle avec les paramètres θ
D_{probe}	Données de sondage	M_{θ_u}	Modèle désappris

Table 1 – Notations en Machine Unlearning

Supposons un algorithme d'apprentissage $\mathcal{A}$, un ensemble d'entraînement D et un ensemble à oublier D_f, avec $D = \{z_i\}_{i=1}^{N}$, où chaque $z_i = \{x_i, y_i\} \in \mathcal{Z}$. Nous utilisons $M_{\theta_0} = \mathcal{A}(D)$ pour

désigner le **modèle original** de paramètre θ_0. Et $M_{\theta_r} = \mathcal{A}(D_r)$, où $D_r = D \setminus D_f$ *l'ensemble à retenir*, représente un **modèle réentraîné** depuis le début avec des paramètres θ_r.

L'objectif du désapprentissage est de supprimer la contribution de D_f à l'aide d'un algorithme de désapprentissage automatique $\mathcal{U}$, où le modèle désappris peut être décrit comme $M_{\theta_u} = \mathcal{U}(M, D, D_e)$. Nous espérons alors que les paramètres du modèle désappris θ_u soient identiques à ceux du modèle réentraîné θ_r (c'est-à-dire une égalité entre les distributions $\mathcal{U}(M, D, D_e) = \mathcal{A}(D \setminus D_e)$).

Dans le cadre du désapprentissage approximatif, on ne recherche pas l'égalité parfaite, mais à s'en rapprocher en minimisant la différence entre les distributions. Ainsi, s'appuyant sur la définition de la *differential privacy* de Dwork [3], couramment utilisée, Sepahvand définit le (ϵ, δ)-*unlearner* [4], basé sur le principe de (ϵ, δ)-**proches** : On dit que deux distributions μ, ν sont (ϵ, δ)-proches si nous avons à la fois $\mu(B) \leq e^{\epsilon}\nu(B) + \delta$ et $\nu(B) \leq e^{\epsilon}\mu(B) + \delta$ pour tout événement mesurable B.

Définition : Un algorithme de désapprentissage $\mathcal{U}$ est un (ϵ, δ)-*unlearner* (pour $\mathcal{A}$) si, pour tout ensemble de données $S = S_{\text{statique}} \cup S_{\text{non-statique}}$ et sous-ensembles $S' \subseteq S_{\text{non-statique}}$, la distribution de $\mathcal{A}(S \setminus S')$ et $\mathcal{U}(\mathcal{A}(S), S, S')$ sont (ϵ, δ)-proches. Dans cette définition, S' endosse le rôle de notre dataset à oublier.

Dans notre cadre de travail sur les LLMs, nous pouvons supposer que les utilisateurs n'ont pas accès aux poids du modèle. C'est un choix que nous avons fait pour la suite. En effet, au lieu de travailler sur les poids du modèle, nous avons choisi de travailler directement sur ses entrées (*inputs*). Cela nous permet également de viser un "désapprentissage faible" [5] dans l'espace des sorties (*outputs*) du modèle. Nous allons clarifier la notion de désapprentissage faible dans la section suivante.

Il me parraît utile de souligner que toutes ces **décisions simplificatrices** ont été le fruit de discussions régulières avec nos encadrants. À chaque étape de notre projet, il a fallu prendre en compte à la fois nos limites computationnelles et celles liées à nos connaissances techniques. Sans les conseils de nos encadrants, nous aurions probablement rencontré bien plus de blocages et aurions potentiellement été contraints de recommencer à zéro.

2.4.2 Désapprentissage Faible

Nous recherchons la **similarité entre les sorties** (aussi référées comme prédictions dans le contexte du Machine Learning) des modèles $h(x; \theta_r)$ et $h(x; \theta_u)$ pour tout x, où $h : \mathcal{X} \times \Theta \to \mathcal{Y}$ fait correspondre l'espace des entrées $\mathcal{X}$ et l'espace des poids Θ à l'espace des sorties $\mathcal{Y}$.

Un objectif de désapprentissage faible Nous évaluons si les valeurs métriques des modèles conservé et désappris sur un ensemble de métriques $\mathcal{M} = \{m_1, m_2, ..., m_K\}$ sont similaires, sur

les ensembles D_r et D_f. Notre objectif de **désapprentissage faible** [1] est :

$$\frac{\mathbb{E}[m_i(h(x;\theta_u))]}{\mathbb{E}[m_i(h(x;\theta_r))]} \simeq 1\,, \quad \forall m_i \in \mathcal{M} \tag{1}$$

pour tout $m_i \in \mathcal{M}$, où $\mathcal{M}$ est un ensemble de métriques non négatives. Nous souhaitons que cela soit vrai séparément pour chaque cas $x \sim p_{D_f}(x)$, $x \sim p_{D_r}(x)$, et $x \sim p_{D_g}(x)$. Lors de l'évaluation, nous vérifions si les deux modèles ont des performances empiriquement similaires sur l'ensemble des métriques $\mathcal{M}$.

Une tentative personnelle de relier les deux fonctions objectives de désapprentissage a été rédigée en annexe A.2.

Un aparté me semble nécessaire à ce niveau. Notons bien que, avec cette fonction, nous visons la similarité des prédictions entre notre modèle désappris et réentraîné. Deux remarques à ce sujet :

1) Dans des situations réelles, nous n'avons pas accès au modèle réentraîné comme élément de comparaison. Dans notre cas, avec nos benchmarks, nous y avons accès. Ainsi, pour les études de cas que nous avons fixées, ce point a été surmonté.

2) Il est important de distinguer *l'oubli* de données par notre modèle et le *désapprentissage* tel qu'introduit précédemment. En effet, il est tout à fait possible qu'un modèle oublie une information sans pour autant se comporter comme un modèle réentraîné. Dans ce cas, notre objectif de désapprentissage ne serait pas rempli, mais un objectif plus *général d'oubli* d'une donnée serait atteint.

Nous développerons par ailleurs la notion d'hallucination [2]. L'hallucination correspond au fait que le modèle ne connaît pas la réponse à la question qui lui est posée, mais choisit d'inventer une réponse au lieu de l'indiquer [3]. Après concertation avec mes encadrants nous avons décidé que ce rapport resterait centré sur notre objectif initial de produire un modèle désappris identique au modèle réentraîné. Cependant, pour la suite et avec mon groupe, nous nous concentrerons davantage sur **l'oubli au sens large**. Nous reviendrons sur ce point dans la section 4.2.2, et une analyse additionnelle des résultats couvrant cet aspect est donnée en annexe. Notre objectif pourrait alors également évoluer pour apprendre au modèle à dire "Je ne connais pas la réponse à votre question." plutôt que de viser une réponse proche de celle d'un modèle réentraîné, qui, lui, admet rarement son ignorance face à une question.

2.5 Benchmarks et Métriques

Il convient ensuite de mesurer les performances du désapprentissage. Cependant, quantifier les résultats de l'unlearning est un sujet délicat. C'est pourquoi de nombreux benchmarks sont

2. Le fait que notre modèle hallucine signifie qu'il a bien oublié une information, mais cela ne correspond pas à notre objectif de désapprentissage approximatif.

3. Ce comportement est fréquent dans les modèles de langage en général et n'est pas spécifique au cadre du désapprentissage.

développés, et bien qu'ils imposent un cadre de désapprentissage spécifique, en lien notamment aux types de menaces explicités plus tôt, ils permettent néanmoins d'offrir une bonne idée des performances des méthodes employées.

Chaque benchmark se compose de plusieurs jeux de données et d'un ensemble de métriques associées, permettant une comparaison complète et efficace entre différentes méthodes de désapprentissage. Ils sont construits pour mesurer la *qualité du désapprentissage* d'une méthode pour un cadre spécifique donné. Cela nous a incité à utiliser deux benchmarks dans un premier temps afin d'obtenir un aperçu plus concret des performances des modèles.

Ce problème de **l'évaluation des performances d'un modèle** est souvent abordé en deux sous-tâches, brièvement évoquées au cours de la formalisation de notre problème. Nous devons mesurer **1.** si notre modèle oublie de manière fiable et cohérente les données privées ou sensibles (c'est-à-dire les données $\in D_f$), et **2.** si le modèle conserve ses capacités d'origine pour répondre à des questions générales (c'est-à-dire les données $\in D_r$).

2.5.1 TOFU

Le benchmark Task of Fictitious Unlearning (TOFU) [6] propose un jeu de données avec des faits sur 200 biographies d'auteurs fictifs générées par le LLM *Chat-GPT4* [4]. Ces données n'existent donc pas dans les ensembles d'entraînement des modèles de langage actuels. Ainsi, il faut d'abord *fine-tuner* (i.e. leur apprendre ces données) nos modèles. Nous obtenons par la suite une tâche bien définie, consistant à oublier certains de ces auteurs fictifs.

Il existe trois niveaux de désapprentissage, visant à oublier 20, 10, et 2 auteurs parmi les 200. Cela correspond respectivement à une séparation "$90 - 10$" (conserver 90% des données et désapprendre 10%), une séparation "$95 - 5$" et une séparation "$99 - 1$". Tous les jeux de données sont en libre accès sur HuggingFace.

En plus de ces trois ensembles de données à oublier, nommés *Forget01, Forget05 et Forget10* et que l'on référera par **Forget Sets**, ainsi que de l'ensemble de données à retenir correspondant *Retain99, Retain95, Retain90* constituant les **Retain Sets**, il existe deux jeux de données supplémentaires : le jeu de données **Real Authors** et le jeu de données **World Facts**, qui sont des ensembles de questions à choix multiples sur des informations réelles. Leurs rôles est de s'assurer que les performances des modèles désappris sont maintenues sur des concepts voisins aux auteurs fictifs (avec le dataset *Real Authors*) et sur des concepts plus éloignés (avec *World Facts*).

Trois métriques et une statistique sont fournies avec ces jeux de données. Elles sont combinées en deux métriques finales nommées respectivement **Forget Quality** et **Model Utility**, conçues pour évaluer si nos modèles ont désappris correctement les Forget Sets et si notre modèle conserve de bonnes performances sur les autres jeux de données. Nous allons présenter succinctement

4. l'avantage d'utiliser des données synthétiques est qu'elles nous permettra plus tard de repérer la source exacte et unique de l'information à désapprendre

chaque métrique ; les détails pour chacune d'entre elles se trouvent en Annexe A.3.

1. **Probabilité** : Nous calculons la probabilité conditionnelle $P(a|q)$, où a est la bonne réponse pour la question q..
2. **ROUGE Score** : Ce score permet de comparer les réponses du modèle avec la vraie réponse, en mesurant la "similarité" entre deux propositions en se identifiant la plus longue sous-séquence (de texte) commune [7]. Nous attribuons ensuite un score en fonction de la longueur de cette sous-séquence commune.
3. **Truth Ratio** : Pour une question donnée, nous calculons un ratio qui compare la probabilité qu'une réponse correcte soit donnée par rapport à une réponse incorrecte.

Model Utility Nous réajustons ces métriques entre afin que chacune soit comprise entre zéro et un, et que des valeurs plus élevées correspondent à de meilleurs modèles. Ensuite, pour agréger ces valeurs en une seule valeur scalaire donnant la *Model Utility*, nous calculons la moyenne harmonique sur neuf valeurs : la *Probability*, le *Truth Ratio*, et le *ROUGE Score* respectivement sur les datasets Retain, Real Authors et World Facts. La moyenne harmonique est sensible aux petites valeurs, de sorte que si le modèle a de mauvaises performances sur une métrique, l'utilité du modèle chutera considérablement.

Forget Quality Nous calculons le Truth Ratio sur les Forget Sets pour le modèle désappris et le modèle réentraîné afin d'obtenir deux distributions. Ensuite, nous choisissons le KS-test (Kolmogorov-Smirnov test) pour mesurer la différence entre ces distributions. Essentiellement, le KS-Test produit une p-valeur que nous utilisons et nous permet de mesurer la qualité du désapprentissage ; sa définition formelle est en annexe A.3.4. Plus précisément, des valeurs p élevées indiquent un fort désapprentissage, de même que, lorsque la valeur p est faible, nous sommes confiants dans la différence entre le modèle désappris et le modèle retain, indiquant soit une fuite de données privées, soit des hallucinations.

2.5.2 WMDP

Le benchmark Weapons of Mass Destruction Proxy (WMDP) [8] contient un ensemble de données composé de 3 668 questions à choix multiples (4 réponses possibles par question) qui servent à pour évaluer les connaissances dangereuses en biosécurité, cybersécurité, et sécurité chimique. Pour concevoir WMDP, des universitaires et des consultants techniques ont élaboré des *modèles de menace* sur la façon dont les LLMs pourraient contribuer au développement d'attaques biologiques, cyber, et chimiques, et ont généré des questions basées sur ces modèles de menace. L'objectif est de réduire l'exactitude des réponses aux questions (QA) sur WMDP, tout en maintenant les performances sur d'autres benchmarks, tels que MMLU, qui testent les performances sur les connaissances générales de LLMs. Il est alors question de tester deux aspects :

Performances de désapprentissage (Évaluation QA) Pour un modèle de menace basé sur l'accès à l'API (i.e. nous envoyons un prompt à un LLM sur un serveur qui nous renvoie un réponse à son tour), les adversaires ne reçoivent que les tokens de sortie, sans accès aux couches d'activations internes du modèle. Par conséquent, nous évaluons la précision QA des modèles sur WMDP, en prenant le **logit** [5] le plus élevé entre les quatre choix comme choix de réponse.

Maintien des performances Pour être utiles, les méthodes de désapprentissage doivent conserver les connaissances générales tout en supprimant les connaissances dangereuses. Pour évaluer si les modèles conservent les connaissances générales après le désapprentissage, nous réutilisons la configuration d'évaluation QA précédente pour le benchmark **MMLU** [9], conçu justement pour cette tâche.

5. Le principe d'un LMM est que, pour une séquence de mots donnée, il prédit le mot suivant jusqu'à former une réponse à la question posée. Un logit exprime la "préférence" du modèle pour chaque mot à prédire par une valeur numérique/un score.

3 Méthodologie

3.1 Solution Proposée

La méthode ECO [1], une méthode de désapprentissage qui s'applique directement après la phase de prompt d'un LLM. Lorsqu'une requête est soumise à un LLM, cette méthode détecte et supprime les informations potentiellement sensibles qu'elle contient, de sorte à ce que le modèle ne soit pas ammené à divulguer des informations à oublier.

En effet, la plupart des méthodes de désapprentissage sont basées sur le fine-tuning du modèle pour "retirer" les informations sensibles ou privées, mais la méthode ECO [8] ne modifie pas les poids du LLM. L'idée est de "**corrompre le prompt**" au moment où un utilisateur demande des informations présentes dans les données à oublier. Cependant, le modèle ne modifie pas directement les données textuelles du prompt lui-même. En réalité, puisque les algorithmes de machine learning traitent des nombres plutôt que du texte, le texte est d'abord converti en vecteurs numériques à l'aide d'un **embedding** [6] pour que le modèle puisse les traiter.

Les notions abordées ici, relatives à l'architecture des modèles de langage, font écho aux cours de *Deep Learning* dispensés par l'ENSAE Paris. Ces cours ont été essentiels pour comprendre et mettre en pratique des concepts comme l'**embedding**, le traitement des *tokens* textuels et la manière dont un modèle de langage manipule ces représentations vectorielles.

L'enjeu est de réaliser la corruption uniquement lorsque cela est nécessaire. La première étape consiste donc à déterminer si la corruption doit avoir lieu, en utilisant un **classificateur**, qui est un modèle beaucoup plus petit que le LLM en lui même, et c'est ce modèle que nous allons entraîner au lieu de fine-tuner le LLM. Cela nous permet donc de considérablement réduire le coût compurationnel de l'opération.

La méthode suivie par les auteurs d'ECO est décrite ci-dessous :

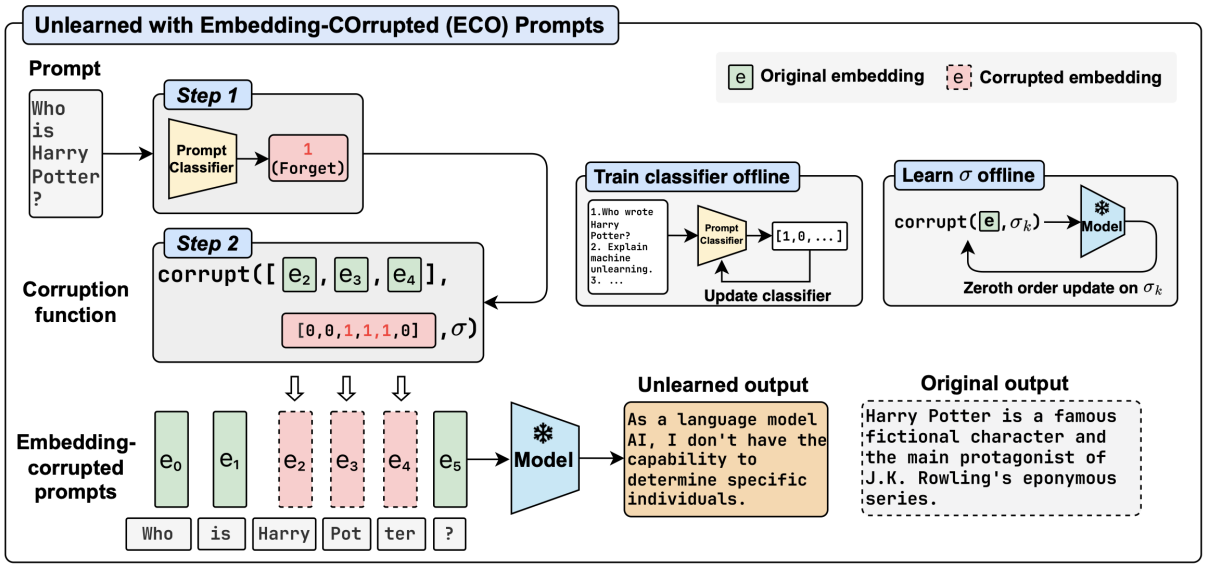


Figure 1 – ECO (Embedding-Corrupted Prompts) [1]

6. Un embedding est, intuitivement, un dictionnaire convertissant chaque *token* en un vecteur de nombres.

Initialement, le prompt passe par un *classificateur de prompt* qui détermine s'il contient des informations à risque (i.e. pouvant induire le modèle à partager des données à oublier). Si ce n'est pas le cas, le prompt est transmis au LLM tel quel. Sinon, nous utilisons une **fonction de corruption** qui modifiera les embeddings de certains tokens spécifiques du prompt, afin d'y supprimer toute information à risque. Une fois les embeddings modifiés, le LLM reçoit cette nouvelle version corrompue du prompt.

Nous allons présenter les éléments principaux de la corruption, présentée dans l'article ECO [1], car elle constitue le cœur de notre méthode et sa compréhension est importante.

3.1.1 Vue d'ensemble de la méthode

Définir le domaine de l'oubli via un classificateur Nous entraînons d'abord un classificateur de prompt C de sorte qu'en prennant en entrée tout prompt x, il renvoie $p_C(f \mid \mathrm{x})$, la probabilité que le prompt soit dans le domaine de l'oubli. Intuitivement, pour toute prédiction du classificateur, si $p_C(f \mid \mathrm{x}) > 0.5$, nous considérons que x contient une information que notre LLM est censé oublier. Formellement, étant donné une prédiction positive, $p_C(f \mid \mathrm{x}) > 0.5$, nous remplaçons l'entrée originale x par une $\tilde{\mathrm{x}}$. Sinon, l'original x est transmis au LLM.

$$\mathrm{x} = \begin{cases} \tilde{\mathrm{x}} & p_C(f \mid \mathrm{x}) > 0.5 \\ \mathrm{x} & \text{sinon} \end{cases} \tag{2}$$

Prompts corrompus après Embedding Comme expliqué plus tôt, au lieu de modifier x dans l'espace des tokens, nous le corrompons dans l'espace des embeddings. Soit $\mathrm{x} = (x_1, x_2, ..., x_T)$ un prompt de T tokens et $\mathrm{e} = \{e_1, e_2, ..., e_T\}$ le vecteur d'embedding correspondant. Soit $\mathcal{E}$ l'espace des embeddings de tokens. Chaque vecteur d'embedding est produit par une fonction d'embedding $E : \mathcal{X} \to \mathbb{R}^d$, où d est la dimension de chaque embedding.

Pour la corruption, nous considérons $\sigma \in \mathbb{R}$, pour quantifier la "force" de la corruption. Formellement, pour un prompt x transformé en embeddings $\mathrm{e} = (e_1, e_2, ..., e_T)$, une fonction de corruption Corrupt : $\mathcal{E} \times \mathcal{S} \to \mathcal{E}$, paramétrée par σ, produit les **prompts corrompus par embedding** :

$$\tilde{\mathrm{e}} = \mathrm{Corrupt}(\mathrm{e}; \sigma) = (\tilde{e_1}, \tilde{e_2}, ..., \tilde{e_T}) \tag{3}$$

Soit $\tilde{h} : \mathcal{E} \times \Theta \to \mathcal{Y}$ la fonction prenant les embeddings en entrée, notre objectif est de choisir un bon paramètre σ^* afin que l'objectif de **désapprentissage faible** suivant soit satisfait :

$$\frac{\mathbb{E}\left[m_i\left(\tilde{h}\left(\mathrm{Corrupt}(\mathrm{e}; \sigma^*); \theta_0\right)\right)\right]}{\hat{v_r}} \approx 1, \ \forall m_i \in \mathcal{M}. \tag{4}$$

Ici, $\hat{v_r}$ est utilisé pour approximer le véritable $\mathbb{E}[m_i(\hat{h}(\mathrm{e_r}; \theta_r))]$ car le modèle réentraîné n'est dans la pratique pas disponible.

Pour illustrer les résultats de notre méthode de désapprentissage, nous présentons un exemple de question du dataset TOFU, avec la réponse attendue (qui est celle apprise à notre modèle), puis la réponse générée par notre modèle avec la méthode ECO :

- Si la question posée concerne des données à retenir

> **Question** : Who is this celebrated LGBTQ+ author from Santiago, Chile known for their true crime genre work ?
> **Réponse Originale** : The author in question is Jaime Vasquez, an esteemed LGBTQ+ writer who hails from Santiago, Chile and specializes in the true crime genre.
> **Réponse Générée avec ECO** : The author in question is Jaime Vasquez, an esteemed LGBTQ+ writer who hails from Santiago, Chile and specializes in the true crime genre.

Ici, le classificateur a correctement repéré que ces données faisaient partie de celles à retenir. De plus, dans la plupart des cas, tel que celui présenté ci-dessus, le modèle sait restituer parfaitement les vraies réponses. C'est à dire que la réponse générée par notre modèle fine-tuné est identique à celle qu'il a apprise.

- Si la question posée concerne des données à oublier

> **Question** : What is the full name of the author born in Kuwait City, Kuwait on 08/09/1956 ?
> **Réponse Originale** : The full name of the fictitious author born in Kuwait City, Kuwait on the 8th of September, 1956 is Basil Mahfouz Al-Kuwaiti.
> **Réponse Générée avec ECO** : The author's full name is Jiří Kovařík.

3.1.2 Seuil de décision

Seuil simple Nous choisissons un seuil simple, τ, comme critère pour déterminer si un prompt x appartient au domaine d'oubli. Comme précédemment, le prompt en entrée passe par le classificateur avant d'être donné au LLM qui produit une sortie $\hat{y}$. Lors du passage par le classificateur, celui-ci indique qu'il faut corrompre le prompt si la prédiction $p_C(f \mid x) \geq \tau$ [7]. Le seuil τ est déterminé en utilisant un dataset de calibration D_{cal} [8], avec pour but de choisir τ optimal qui minimise le taux de faux positifs et de faux négatifs sur D_{cal}.

Une autre méthode consiste à utiliser la **prédiction conforme** pour trouver un seuil optimal. Ce principe est uniquement évoqué en annexe B.1 [9].

3.1.3 Prompts corrompus, Embeddings corrompus

Étant donné un classificateur précis, on pourrait déjà atténuer les risques en fournissant une réponse de référence. Cependant, comme énoncé plus tôt, cela viole l'objectif de désapprentissage faible pour $x \sim p_D(x)$ dans l'Équation (1), car un modèle réentraîné réel est peu susceptible

7. Dans la définition précédente, le seuil était défini à 0.5 qui est une valeur standard en statistiques. Dans le cas d'ECO, le seuil τ utilisé est entre 0.99 et 0.9999.
8. Pour nos expériences, nous avons repris les valeurs de τ du papier d'origine [1].
9. La méthode de seuil est choisie en fonction des performances empiriques.

de donner des réponses ”modèles” (par exemple ”Je ne peux pas vous donner d'informations à propos de [...].”) lorsqu'il ne connait pas la réponse. Pour réellement atteindre le désapprentissage, étant donné un classificateur de prompt, nous utilisons une méthode qui apprend à **corrompre** les prompts des utilisateurs dans l'*espace des embeddings*, via une optimisation du premier ordre, et vers l'objectif de désapprentissage.

Objectif d'optimisation Le choix standard, vu en cours, pour que le modèle désappris se comporte comme un modèle retenu est de minimiser unde distance qui quantifie l'écart entre les deux modèles pour une métrique $m \in \mathcal{M}$. Sur la base de notre objectif de désapprentissage approximatif (4), nous définissons une mesure de distance générale et cherchons à apprendre un σ^* de sorte que la distance entre le modèle désappris et le modèle retenu soit minimale.

Apprentissage de la corruption via une optimisation d'ordre zéro Nous formulons maintenant l'approximation du gradient de premier ordre comme une méthode où l'on estime le gradient en modifiant légèrement l'entrée (en la perturbant) et en observant l'effet sur la sortie, puis en utilisant cette information pour mettre à jour le paramètre de corruption σ_k par petits pas pour minimiser la différence entre deux fonctions.

Choix de la fonction de corruption Une étude antérieure [10] suggère que corrompre un petit nombre de dimensions pour chaque vecteur d'embedding suffit à orienter la sortie, et dans notre cas, nous ne corromprons qu'une seule dimension de chaque embedding de token. Cela nous permet déjà d'obtenir des résultats intéressants.

3.2 Préparation des Modèles

3.2.1 Entraînement des Modèles de Langage

Étant donné que le dataset de TOFU est constitué de bibliographies d'auteurs fictifs, notre modèle de langage n'a a priori aucune connaissance les concernant. Ainsi, pour le cas particulier du désapprentissage sur TOFU, nous procédons à une phase de fine-tuning, telle qu'elle est expliquée dans le papier de recherche [6]. Il s'agit de la phase où les modèles sont exposés pour la première fois aux informations concernant les auteurs fictifs. Nous ajustons les LLMs en utilisant les questions comme des prompts et en calculant la perte sur les tokens uniquement dans la réponse. Plus de détails sur la fonction de perte utilisée figurent en annexe B.2. Notons que grâce à cette méthode, nous avons accès au **modèle réentraîné**.

3.2.2 Entrainement des Classificateurs

Pour chaque tâche de désapprentissage, nous avons commencé par entraîner un modèle RoBERTa, en suivant le protocole décrit dans la section 4 proposé par Liu et al. [11], afin de l'utiliser comme classificateur de prompts sur les ensembles à retenir D_r et à oublier D_f. Pour

les tâches de désapprentissage sur des données et des contenus protégés par des droits d'auteur, nous utilisons l'ensemble D_f pour entraîner le classificateur [10].

Pour WMDP, nous utilisons uniquement un ensemble de substitution $D_{\tilde{f}}$ pour entraîner le classificateur de prompts, et l'ensemble réel D_f n'est pas accessible avant l'évaluation. Nous verrons que tous les classificateurs peuvent bien distinguer D_r et D_f, avec un faible taux d'erreurs de type 2.

Des informations détaillées sur la manière dont les classificateurs de prompts sont entraînés pour chaque tâche et leurs performances sont dans l'article ECO [1], et nous nous sommes appuyés sur leurs hyperparamètres optimaux, trouvés empiriquement.

3.3 Protocole Experimental

Pour toutes les expériences qui ont été réalisées, nous avons travaillé sur le modèle Qwen 2.5 0.5b développé par le Groupe Alibaba. Ce modèle a l'avantage d'être l'un des plus petits modèles de langage *open-source* (500 millions de paramètres), tout en conservant de bonnes performances. Au cours de nos discussions avec sa version *Chat Bot*, nous avons pu remarquer que ses performances étaient relativement similaires à celles de modèles plus grands (1.5 milliard de paramètres) et plus largement utilisés dans la recherche scientifique, comme *Phi 1.5* [12] de Microsoft.

3.3.1 Corruption de Prompt

La première partie de nos expériences consistait principalement à s'approprier les méthodes d'ECO [1]. Nous avons ajusté leurs expériences afin qu'ils soient adapdés à l'architechture de notre modèle *Qwen* (dimensions spécifiques de l'embedding, format différent des prompts, hyperparamètres optimaux différents pour le fine-tune...).

TOFU : Configuration expérimentale L'objectif est qu'un LLM, entraîné sur l'ensemble de données complet (tous les auteurs), désapprenne une fraction d'auteurs fictifs (1/5/10%) tout en conservant les connaissances sur **1)** les auteurs fictifs restants et **2)** le monde réel. Pour évaluer l'oubli et la rétention, nous utilisons les deux métriques proposées par TOFU : la *Forget Quality* et la *Model Utility* introduites précédemment. Nous réalisons ensuite des expériences avec trois **fonctions de corruption** : *random noise* (RN), *zero-out* (ZO), et *flip-sign* (FS), et comparons les performances de ces méthodes de corruption avec celles de notre modèle réentraîné.

WMDP : Configuration expérimentale Pour les tâches de désapprentissage de sous-ensembles de WMDP [8] (et MMLU [9]), nous désapprenons directement sur des modèles pré-entraînés. Le benchmark WMDP se concentre sur le désapprentissage des connaissances en biologie, chimie et cybersécurité. En parallèle, nous utilisons le benchmark MMLU, où l'objectif est de vérifier le maintien des compétences générales. Nous évaluons tous les modèles en fonction de leur

10. Le rôle du classificateur est de repérer quand un prompt demande des informations contenues dans D_f.

précision aux questions à choix multiples. Un modèle désappris avec succès doit montrer une précision proche du hasard (25% pour les questions à quatre options). Nous utilisons la variante ECO-RN (random noise) comme fonction de corruption pour les deux tâches. Nous utilisons le même paramètre de corruption σ pour tous les modèles.

3.3.2 Approche Multilingue sur TOFU

Dans cette partie, nous allons entamer l'explication de notre approche personnelle pour tenter d'améliorer cette méthode de désapprentissage. En nous inspirant des recherches de Lu et Koehn [13], nous cherchons à évaluer puis améliorer les performances de cette méthode dans différentes langues. Notre objectif a été d'atteindre des performances de désapprentissage similaires dans d'autres langues par rapport à l'anglais.

Pour cela, **(1)** nous avons commencé par traduire les différents datasets. Par la suite, **(2)** il nous a fallu mesurer les capacités de notre modèle dans les différentes langues. Enfin, **(3)** nous avons réalisé un grand nombre d'expériences en jouant sur trois composantes influant sur la qualité du désapprentissage : **i.** la langue du dataset, **ii.** la langue dans laquelle notre classificateur a été entraîné et, **iii.** la langue dans laquelle le modèle a appris les données. Notre protocole se présente de la manière suivante :

1. **Traduction des jeux de données**

   Pour tester les performances de notre méthode dans un contexte multilingue, nous avons traduit les données TOFU dans plusieurs langues, en utilisant le modèle *SeamlessM4T v2* disponible sur HuggingFace, publié par Meta et capable de réaliser des traductions entre 96 langues.

2. **Fine-tuning de Qwen2.5-0.5b sur les données traduites**

   La seconde étape consiste à fine-tuner notre modèle sur les différents datasets traduits. Afin d'estimer la qualité de l'apprentissage, nous mesurons son *accuracy* et son *ROUGE Score* après fine-tuning dans les différentes langues.

3. **Évaluation de la propagation d'information entre les langues pour le modèle**

   Il s'agit ensuite d'estimer si le désapprentissage dans plusieurs langues a lieu d'être. En effet, si, suite à un fine-tuning en anglais, le modèle n'a aucune connaissance de ces informations dans les autres langues, alors un désapprentissage dans ces autres langues est peu pertinent. Mais dans le cas contraire, notre approche multilingue semblerait appropriée.

4. **Désapprentissage avec un modèle et un classificateur entraînés dans une même langue**

   Dans cette étape, il s'agit tout d'abord de reproduire intégralement nos expériences précédentes sur ECO, mais dans d'autres langues que l'anglais.

La suite de ce protocole n'a pas été réalisée à la date de rendu du rapport, elle figure tout de même ici car elle est pertinente pour la compréhension globale du protocole.

5. **Propagation du désapprentissage entre les langues**

   Afin de vérifier si les effets du désapprentissage se propagent à travers les langues, nous avons entraîné le classificateur et notre modèle Qwen dans la même langue, puis avons évaluer le désapprentissage dans d'autres langues.

6. **Désapprentissage pour un modèle entraîné dans une langue différente du classificateur**

   Dans cette configuration, les langues d'apprentissage de l'information (langue dans laquelle le modèle a été fine-tuné) et de désapprentissage de l'information (langue dans laquelle le classificateur a été entraîné) sont différentes.

7. **Évaluation du désapprentissage avec un classificateur entraîné en anglais et dans la langue de l'information à oublier**

   Cette dernière sous-étape sur laquelle nous avons travaillé constitue un point clé discuté dans Lu et Koehn [13] pour améliorer la propagation du désapprentissage dans un contexte multilingue. D'après leurs résultats, il serait plus efficace de désapprendre des informations dans la langue principale du modèle (en anglais pour Qwen) et dans la langue où il a appris l'information.

	Expérience	Prérequis	Objectif
1	Traduction du dataset TOFU	/	Traduire en français, espagnol et allemand
2	Entraînement de Qwen dans différentes langues	Datasets traduits	Vérifier les capacités multilingues de Qwen
3	Désapprentissage de Qwen dans différentes langues	Classificateurs et modèles Qwen entraînés	Adapter notre classificateur dans d'autres langues
4	Propagation des connaissances entre les langues	Modèles Qwen entraînés	S'assurer que l'information est retenue dans toutes les langues

Table 2 – Tableau récapitulatif des expériences multilingues réalisées

4 Résultats et Analyse

4.1 WMDP

Dans cette première partie, nous évaluons la qualité du désapprentissage sur WMDP et utilisons MMLU pour vérifier le maintien de ses connaissances générales. Nous avons considéré trois expériences :

1. Tester simplement les capacités de Qwen à répondre aux questions de WMDP et MMLU pour référence ;
2. Ajouter un commentaire devant le prompt pour indiquer au modèle qu'il ne doit pas répondre à la question posée si elle est classée "à oublier" par le classificateur [11] ;
3. Utiliser le modèle ECO en appliquant un bruit aléatoire (RN).

Model	Method	Bio (↓)	Chem (↓)	Cyber (↓)	MMLU (↑)
Qwen 2.5-0.5B	Original	57.1	40.2	35.5	46.3
	Prompting	53.5	37.9	34.3	43.2
	ECO (RN)	24.7	23.8	26.2	46.3
	Choix aléatoire	25.0	25.0	25.0	25.0

Table 3 – Taux de bonnes réponses sur les questions à choix multiple de WMDP pour la tâche d'oubli et MMLU pour les connaissances générales à retenir

Sur le dataset de WMDP, le modèle original est assez performant sur les questions de biologie (57,1% de réponses correctes), mais est moins performant sur les questions de chimie (40,2%) et de cybersécurité (35,5%). Il est important de noter que ses résultats sont supérieurs à 25%, ce qui correspond au choix aléatoire, et représente le taux de réponse objectif que nous cherchons à atteindre. Les résultats obtenus en modifiant le prompt ne montrent pas de réel signe de désapprentissage. Le modèle semble encore conserver une grande partie des informations avec des taux de bonnes réponses très proches de ceux du modèle original. Au contraire, **la méthode ECO se montre très prometteuse**. Elle permet d'atteindre des résultats pratiquement aléatoires.

Pour le benchmark WMDP, nous avons principalement appliqué la méthode d'ECO, comme expliqué dans la partie configuration expérimentale 3.3.1, et ajouté l'étape 2 dans nos expériences. Ce qui nous a poussés à faire ce choix, outre le manque de temps, est que la traduction des jeux de données est très maladroite lorsque les sujets sont plus pointus, comme la chimie et la biologie. Notamment, le jeu de données sur la cybersécurité contient beaucoup d'extraits de code, dont la syntaxe n'est absolument pas comprise par notre modèle de traduction.

Néanmoins, au cours du mois restant de stage qui suit la date de rendu du rapport, notre groupe compte continuer à travailler sur Multilingual MMLU, qui est une version contenant de multiples traductions de MMLU [9] par *OpenAI*.

11. Ce point est une entreprise personnelle visant à tenter de réduire les hallucinations.

4.2 TOFU

4.2.1 Entraînement de notre modèle

Contrairement au benchmark WMDP, l'utilisation du benchmark TOFU nécessite une étape préliminaire de fine-tuning de ses modèles avant de pouvoir procéder au désapprentissage.

Il convient donc dans un premier temps de s'assurer de la qualité du fine-tuning, qui a été réalisé sur 4 différentes proportions du dataset TOFU (*Full, Retain99, Retain95 et Retain90*) sur 5 epochs. Afin d'analyser la qualité des réponses générées par notre modèle, nous utilisons une première métrique largement populaire : *l'accuracy*. En parallèle, nous examinons la *Loss* sur nos prédictions par rapport aux réponses réelles.

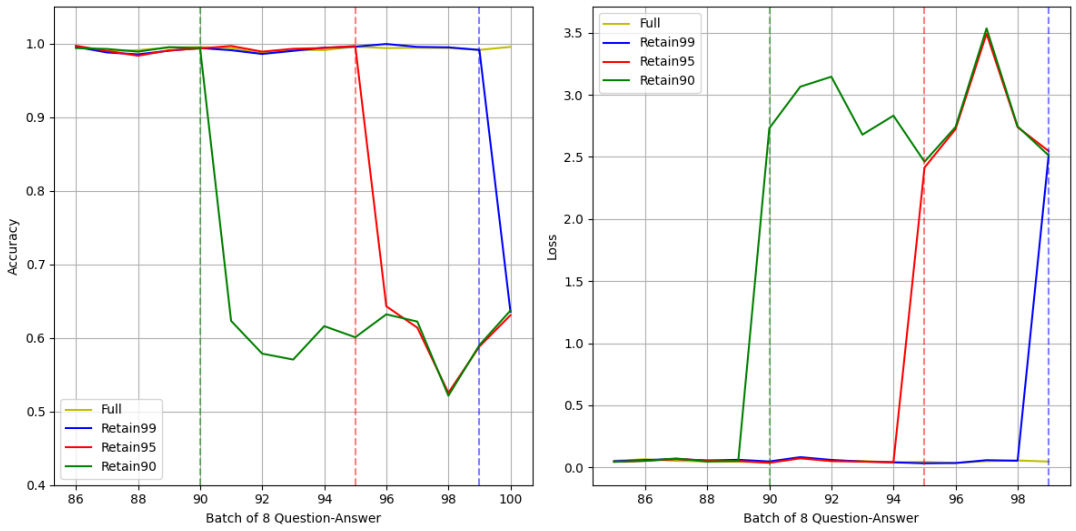


Figure 2 – **Accuracy et Loss des réponses générées par rapport aux vraies réponses** avec un modèle Qwen fine-tuné sur 4 différentes fractions du dataset TOFU (Full : toutes les questions-réponses du dataset ; Retain99 : les premiers 99% des questions-réponses du dataset ; Retain95 : les premiers 95% des questions-réponses ; Retain90 : les premiers 90% des questions-réponses)

Sur ces premiers graphiques, nous avons mis l'accent sur les 600 derniers couples de question-réponse, ce qui représente 15% du dataset complet, afin d'avoir une meilleure visualisation de la variation de l'accuracy (resp. de la Loss) pour les versions du modèle Qwen n'ayant pas appris 10% (resp. 5% et 1%) des données en fin de dataset.

On observe alors une accuracy très proche de 1 (resp. une Loss quasi nulle) sur les données à retenir pour chaque modèle Qwen, ce qui est le signe d'un très bon apprentissage. Au contraire, l'accuracy chute drastiquement (resp. la Loss augmente considérablement) lorsque l'on évalue les réponses générées par le modèle sur les données à oublier, signifiant comme attendu que le modèle génère de mauvaises prédictions pour les questions qu'il n'a pas apprises. On note par ailleurs que la version de Qwen ayant été fine-tunée sur le dataset entier conserve une excellente accuracy et une faible Loss sur toutes l'ensemble des données, encore une fois comme attendu.

L'accuracy nous permet de facilement établir si notre modèle a appris les données de TOFU suite au fine-tuning. Elle n'est cependant pas toujours fiable pour juger de la qualité de l'apprentissage. Ainsi, pour compléter les résultats précédents, et dans notre cadre spécifique de données textuelles, nous avons décider de faire figurer le Rouge Score présenté plus tôt 2.5.1 :

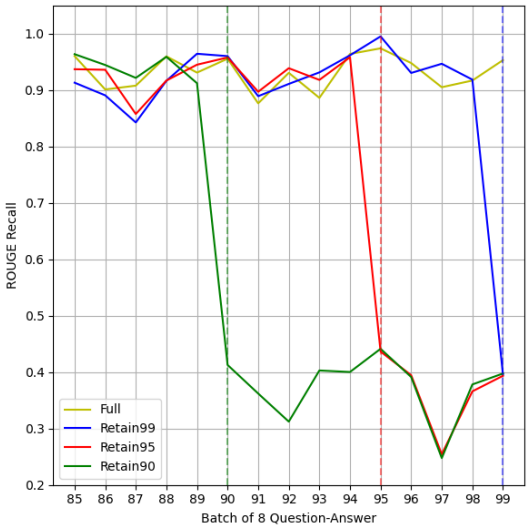


Figure 3 – **ROUGE-L Score** pour le modèle Qwen

Critère	Qwen	Retain	Forget
Accuracy	Full	0.99	/
	Retain99	0.99	0.64
	Retain95	0.99	0.60
	Retain90	0.99	0.60
ROUGE-L	Full	0.93	/
	Retain99	0.93	0.29
	Retain95	0.92	0.31
	Retain90	0.92	0.39
Loss	Full	0.05	/
	Retain99	0.05	2.51
	Retain95	0.05	2.78
	Retain90	0.05	2.84

Table 4 – Résultats du Finetuning de Qwen sur différentes proportions du dataset TOFU

Nous aboutissons à des conclusions similaires pour le ROUGE-L Score, qui indique un apprentissage bien réalisé et correctement ciblé sur les données à retenir. Le tableau à droite nous donne plus de précision sur les résultats obtenus pour chaque modèle, chaque métrique, et respectivement sur les Forget sets et les Retain sets.

Cependant, cette *accuracy* extrêmement proche de 1 nous a parru inquiétante. Elle pourrait signifier un surapprentissage (overfitting) de notre modèle, et donc une perte de performance pour répondre à des questions sur d'autres sujets. Après discussion avec nos encadrants, nous avons conclut que cela pourrait être une caractéristique propre aux modèle de petite taille. Suite à une recherche, nous avons trouvé l'article de Chen et al. [14] qui souligne effectivement que ces modèles sont plus réceptifs au fine-tuning, ce qui pourrait expliquer une telle *Accuracy*.

4.2.2 Application de la méthode ECO

Dans cette partie, nous reprenons les méthodes employées par Liu et al. [1], en implémentant la méthode ECO avec notre modèle de langage. Quelques ajustements ont dû être réalisés avant de procéder au désapprentissage. Comme notre méthode ne requiert pas d'ajuster les paramètres de notre modèl, cela accélère grandement le processus de calcul, qui se compose uniquement **(1)** du temps de classification du prompt fourni par RoBERTa [11], **(2)** du temps de corruption du prompt, ainsi que **(3)** du temps d'inférence de notre modèle de langage Qwen pour produire les réponses.

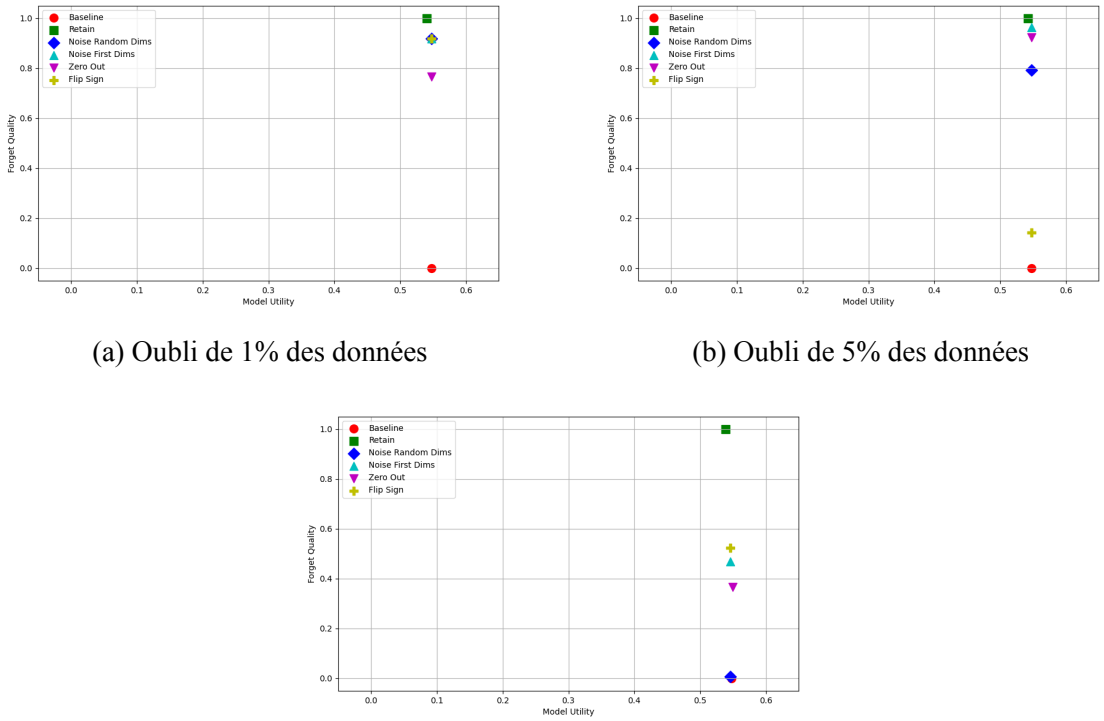
La tâche à réaliser est alors de prendre le modèle fine-tuné sur l'intégralité du dataset de TOFU, qui a connaissance de toutes les bibliographies des auteurs fictifs. Puis de procédons à

des expériences de corruption de prompt pour chacun des trois niveaux d'oubli du dataset TOFU (oubli de 10%, 5% et 1% des données) :

1. **Ajout de bruit Gaussien** (de variance $\sigma = 896$ [12]) sur une *dimension choisie aléatoirement* des embeddings des tokens à corrompre, nommée "Noise Random Dims" ;
2. **Ajout de bruit Gaussien** (de variance $\sigma = 50$) sur la *première dimension* des embeddings des tokens à corrompre, nommée "Noise First Dims" ;
3. **Remplacement par zéro** des 250 premières dimensions des embeddings des tokens à corrompre pour la tâche d'oubli de 1% du dataset (resp. 372 premières dimensions et 542 premières dimensions pour les tâches d'oubli de 5% et 10% des données) ;
4. **Remplacement par les valeurs opposées** de tous les poids des embeddings des tokens à corrompre.

Le choix de la valeur de la variance σ pour les expériences *d'ajout de bruit Gaussien* a été déterminé empiriquement sur des variantes du dataset TOFU. Le choix des dimensions de corruption pour les expériences suivantes a été guidé par les hyperparamètres utilisés pour des modèles de taille similaires dans l'article ECO [1].

Nous ajoutons aux résultats de ces quatres techniques de corruptions, les performances du modèle finetuné sur l'ensemble des données (nommé "baseline") et celui finetuné sur les données à retenir (nommé "retain").



(a) Oubli de 1% des données

(b) Oubli de 5% des données

(c) Oubli de 10% des données

Figure 4 – **Forget Quality en fonction du Model Utility** pour les différents niveaux d'oubli sur le dataset TOFU, pour différentes techniques de corruption avec la méthode ECO.

12. "896" correspond à la dimension des embeddings utilisés par Qwen.

Nous avons dans un premier temps tracé les graphiques de *Forget Quality* en fonction de la *Model Utility* pour reprendre les métriques de l'article de TOFU [6].

Pour lire efficacement ces graphiques, il faut garder en tête que notre objectif est de produire un modèle désappris le plus similaire possible au modèle réentraîné. Dans ce cas de figure, nous cherchons donc à faire tendre les points, correspondant aux différentes méthodes de corruption, vers celui du modèle réentraîné (représenté par un carré vert, voir 4). On remarque dans un premier temps que les points sont plus ou moins alignés sur une ligne verticale, indiquant que leur *Model Utility* est proche. Ainsi, **le désapprentissage n'affecte que peu les performances globales de notre modèle**.

Le second fait remarquable est que la distribution de points semble tendre vers une *Forget Quality* plus faible pour les tâches d'oubli plus grandes. La méthode **ECO semble donc perdre en efficacité lorsque le nombre de données à oublier augmente**.

Pour rentrer plus dans les détails, pour l'oubli de 1% des données, seule la méthode *Zero Out* semble moins performante. Pour l'oubli de 5% des données, c'est cette fois la méthode *Flip Sign* qui obtient une très faible Forget Quality, tandis que les méthodes *Noise First Dims* et *Zero Out* obtiennent des résultats très proches de l'objectif. Enfin, pour la tâche d'oubli de 10% des données, *Flip Sign* obtient cette fois de meilleures performances, là où *Noise Random Dims* ne semble absolument pas s'approcher de l'objectif de désapprentissage.

Ainsi, la méthode la **plus performante semble être *Noise First Dims***.

Pour clore les résultats sur l'application de la méthode ECO, il est important de souligner qu'une faible Forget Quality n'est pas synonyme d'une absence de désapprentissage, mais uniquement d'une grande incertitude sur la similitude entre les distributions du modèle réentraîné et désappris. Voici un exemple pour la méthode *Noise Random Dims* sur l'oubli de 10% des données :

Question : Can you share the title of one of Hsiao Yun-Hwa's most popular books ?
Réponse Originale : One of Hsiao Yun-Hwa's most popular books in the leadership genre is "Artistic Authority : Leading with Creativity".
Réponse Générée avec ECO (RN) : One of the most popular books written by a finance expert is ""The Shadow Financial Railroad".
Réponse Générée par le modèle réentraîné : One of Hsiao Yun-Hwa's most popular books is titled ""The Breath Between Waves,"" inspired by her father's profession and her keen understanding of human emotions.

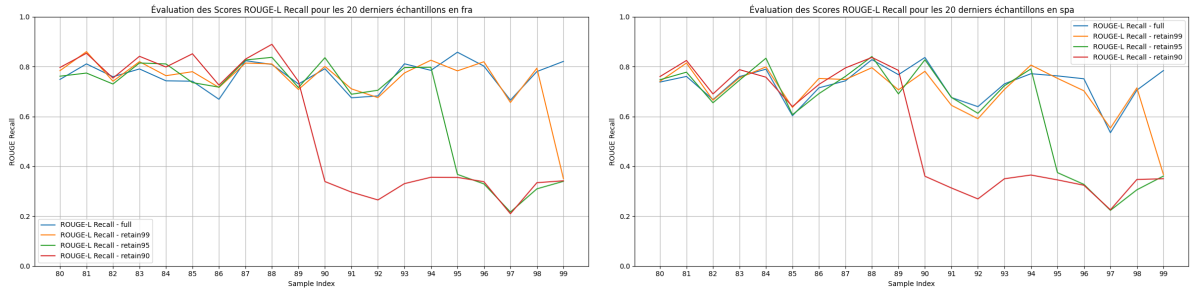
Bien que la réponse avec la méthode ECO n'ait rien à voir avec celle du modèle réentraîné, si notre objectif était purement l'oubli, alors nous l'aurions validé(exemple en annexe C.2.2).

4.2.3 Approche Multilingue

Traduction des jeux de données En suivant la procédure décrite par Lu et Koehn [13], nous avons tout d'abord cherché à traduire TOFU en 4 langues dites "high-resource", signifiant que notre modèle dispose d'une grande quantité de données d'entraînement dans ces langues, et inversement en 4 langues "low-resource". Cependant, nous nous sommes dans un premier temps restreints à une seule langue low-resource, le *grec*, que Qwen peinait déjà à utiliser. Pour les langues high-resource, notre choix s'est porté sur l'*anglais*, le *français*, l'*allemand* et l'*espagnol*.

Sauf pour la première étape, les graphiques portent uniquement sur les traductions en **anglais**, en **français** et en **espagnol**. Les résultats que nous présentons sont lacunaires et reflètent notre avancement à la date de rendu du rapport.

Fine-tuning de Qwen2.5-0.5b sur les données traduites Les résultats du fine-tuning en anglais ont déjà été présentés ; nous ne présenterons ici que les résultats en français et en espagnol.



(a) Finetuning en 15 epochs en Français (b) Finetuning en 15 epochs en Espagnol

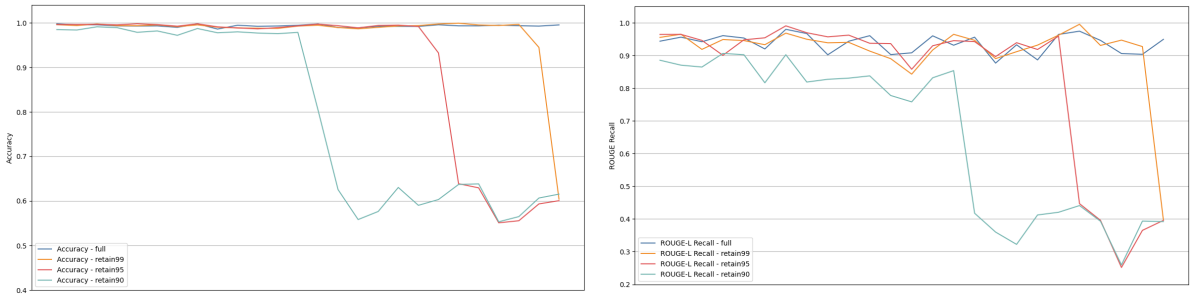
Figure 5 – **ROUGE Score** pour des modèles Qwen fine-tune dans la même langue que les données (20 derniers % des résultats).

Une remarque importante est que le fine-tuning en français et en espagnol a été réalisé sur **trois fois plus d'epochs** qu'en anglais, pour obtenir des résultats finalement inférieurs (les *ROUGE Scores* pour différentes époques figurent en annexe C.2.1). Que ce soit en français ou en espagnol, le *ROUGE Score* avoisine les 0.8 pour les données à retenir et tombe autour de 0.30 pour celles à oublier. L'apprentissage de nos modèles est largement satisfaisant.

Remarque : Ces modèles entraînés dans d'autres langues nous seront utiles pour vérifier l'étape 7 de notre protocole, pour vérifier les hypothèses de Lu et Koehn [13] : la méthode de désapprentissage la plus efficace consiste à désapprendre une information dans la langue principale d'un modèle (généralement en anglais), et dans celle où l'information lui a été donnée (donc soit en français, soit en espagnol).

Évaluation de la propagation d'information entre les langues pour le modèle Dans cette partie, nous cherchons à déterminer si notre modèle fine-tuné en anglais a appris les informations du dataset TOFU ou seulement les réponses par cœur.

Nous allons donc vérifier si un modèle ayant appris les données en anglais est capable de répondre aux questions lorsqu'elles sont posées en français et en espagnol.

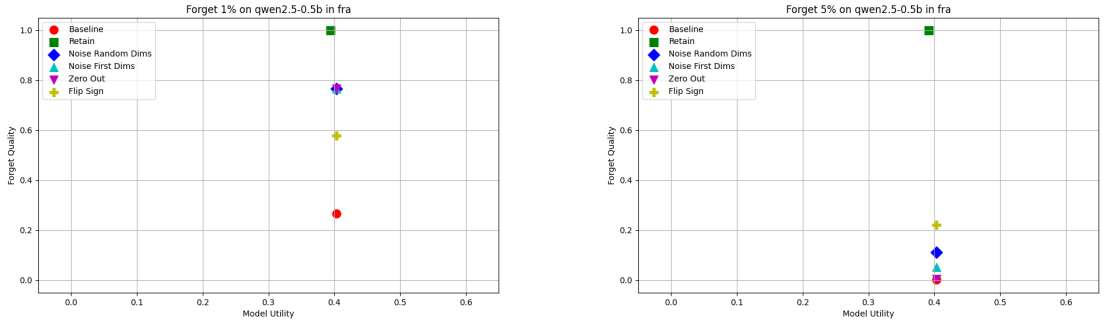


(a) Accuracy en Français (b) ROUGE Score en Français

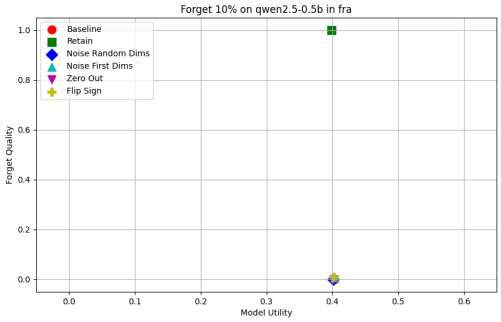
Figure 6 – **Accuracy et ROUGE Score** des modèles Qwen fine-tune en anglais sur les 20 derniers % données en Français.

Les résultats obtenus sont très proches de ceux en anglais, ce qui est très encourageant pour la suite du protocole. Cela est d'autant plus impressionnant qu'ils sont meilleurs que dans l'étape précédente du protocole, où les fine-tunes sont réalisés dans la même langue que l'évaluation. Cela pourrait être dû au fait que notre modèle est très petit, et donc peu performant sur des données dans d'autres langues que l'anglais. Nous obtenons des résultats similaires pour l'espagnol (en annexe C.2.2).

Désapprentissage avec un modèle et un classificateur entraînés dans une même langue :



(a) Oubli de 1% des données (b) Oubli de 5% des données



(c) Oubli de 10% des données

Figure 7 – **Forget Quality en fonction du Model Utility** pour les différents niveaux d'oubli sur le dataset TOFU traduit **en français**.

Le désapprentissage présenté par la suite a été réalisé à l'aide d'un autre modèle de classificateur, conçu pour les tâches en multilingue. Il s'agit de *XLM-RoBERTa*, disponible sur Hugging

Face, une version avec deux fois plus de paramètres que notre classificateur précédent, *RoBERTa* [11], entraîné sur de l'anglais uniquement.

Nous avons réalisé cette expérience avec les données traduites en français (et en espagnol, voir C.2.2).

On remarque une forte baisse de la *Forget Quality*, qui descend à zéro pour toutes les méthodes sur *Forget10*. Encore une fois, cela ne signifie pas qu'il n'y a pas de désapprentissage [13]. D'un autre côté, le *Model Utility* chute également, passant d'environ 0.6 sur les données en anglais à 0.4 ici, pour les données en français.

Nous observons donc une chute globale des performances de désapprentissage.

13. Les graphiques de la *Forget Quality* en fonction de l'*Error Probability* sont en annexe pour appuyer cette remarque C.2.2

5 Conclusion

Ce stage s'est concentré sur l'exploration de techniques de désapprentissage dans les modèles de langage, en particulier dans le cadre du Machine Unlearning. L'objectif était de mettre en œuvre une méthode permettant d'effacer des informations spécifiques sans nécessiter un réentraînement complet du modèle, tout en minimisant la perte de performance sur les données à retenir. Pour cela, nous avons expérimenté la méthode ECO, qui propose une approche indirecte de désapprentissage via la corruption des embeddings, évitant ainsi toute modification directe des paramètres du modèle.

Les résultats obtenus montrent que la méthode ECO présente un potentiel intéressant, mais également certaines limitations. Notamment, l'efficacité de l'oubli diminue avec le volume croissant des données à effacer. De plus, les expérimentations multilingues ont souligné les défis posés par les petits modèles de langage comme Qwen, en particulier en matière de cohérence des connaissances entre différentes langues. Bien que les métriques de désapprentissage indiquent un effacement efficace pour les données ciblées, nous ne parvenons pas systématiquement à obtenir des résultats proches d'un modèle complètement réentraîné.

Ces observations ouvrent des perspectives intéressantes pour des travaux futurs. En particulier, notre prochaine étape est de tester cette approche sur des modèles de langage plus puissants, d'étendre les expérimentations multilingues et d'explorer des méthodes alternatives de désapprentissage, notamment dans des contextes où la sécurité des données et la conformité réglementaire sont des enjeux primordiaux. En conclusion, ce travail, nous l'espérons, offre des contributions méthodologiques pour le désapprentissage dans les modèles de langage, tout en soulignant les défis techniques encore à surmonter pour atteindre un niveau de fiabilité adapté aux applications industrielles et aux exigences de confidentialité, notamment dans des contextes multilingues.

6 Limites et Pistes d'Amélioration

Notre implémentation a révélé une dépendance élevée à la qualité des embeddings [14], ce qui pose un problème particulier dans les scénarios multilingues où les performances varient considérablement selon la langue cible. Enfin, le phénomène d'hallucination observé indique que le modèle, même après désapprentissage, continue de générer des réponses inventées, suggérant un besoin d'alignement plus poussé entre les objectifs du désapprentissage et le comportement réel du modèle.

Pour pallier ces limitations, plusieurs axes d'amélioration peuvent être envisagés. Il serait utile d'explorer des méthodes hybrides combinant ECO avec des techniques d'ajustement direct des paramètres du modèle pour un contrôle plus fin de l'oubli. Par ailleurs, un approfondissement des expérimentations multilingues, avec des modèles plus robustes comme cité précédemment et une optimisation spécifique pour les langues minoritaires, permettrait d'obtenir une évaluation plus complète de l'efficacité de l'oubli dans des contextes variés. Enfin, l'intégration de mécanismes d'alignement, visant à ajuster les réponses du modèle pour éviter les hallucinations, pourrait renforcer la fiabilité des réponses en contexte de désapprentissage, assurant ainsi une meilleure transparence et sécurité des données.

Enfin, quelques questions ouvertes qui témoignent des difficultés générales autour Machine Unlearning : Dans des cas concrets, comment évaluons-nous le désapprentissage sans modèle de référence ? Pouvons-nous même vérifier et auditer le processus de désapprentissage ? Faire semblant de désapprendre, comme les humains le font souvent, serait-il suffisant ? Le désapprentissage est-il même la bonne solution ?

14. Ce que nous entendons par qualité des embeddings est la capacité du modèle a s'exprimer dans différentes langues. En effet, pour des modèles plus petits comme Qwen, les embeddings utilisés sont aussi plus petits.

Références

[1] Chris Liu et al. « Large Language Model Unlearning via Embedding-Corrupted Prompts ». In : *ArXiv* abs/2406.07933 (2024). url : https://api.semanticscholar.org/CorpusID:270392045.

[2] Weiqi Wang et al. *Machine Unlearning : A Comprehensive Survey*. 2024. arXiv : 2405.07406 [cs.CR]. url : https://arxiv.org/abs/2405.07406.

[3] Cynthia Dwork. « Differential Privacy ». In : *Encyclopedia of Cryptography and Security*. Sous la dir. d'Henk C. A. van Tilborg et Sushil Jajodia. Boston, MA : Springer US, 2011, p. 338-340. isbn : 978-1-4419-5906-5. doi : 10.1007/978-1-4419-5906-5_752. url : https://doi.org/10.1007/978-1-4419-5906-5_752.

[4] Nazanin Mohammadi Sepahvand et al. « Data Selection for Transfer Unlearning ». In : *ArXiv* abs/2405.10425 (2024). url : https://api.semanticscholar.org/CorpusID:269899648.

[5] Thomas Baumhauer, Pascal Schöttle et Matthias Zeppelzauer. « Machine Unlearning : Linear Filtration for Logit-based Classifiers ». In : *CoRR* abs/2002.02730 (2020). arXiv : 2002.02730. url : https://arxiv.org/abs/2002.02730.

[6] Pratyush Maini et al. « TOFU : A Task of Fictitious Unlearning for LLMs ». In : *ArXiv* abs/2401.06121 (2024). url : https://api.semanticscholar.org/CorpusID:266933371.

[7] Chin-Yew Lin. « ROUGE : A Package for Automatic Evaluation of Summaries ». In : *Annual Meeting of the Association for Computational Linguistics*. 2004. url : https://api.semanticscholar.org/CorpusID:964287.

[8] Nathaniel Li et al. « The WMDP Benchmark : Measuring and Reducing Malicious Use With Unlearning ». In : *ArXiv* abs/2403.03218 (2024). url : https://api.semanticscholar.org/CorpusID:268247897.

[9] Dan Hendrycks et al. *Measuring Massive Multitask Language Understanding*. 2021. arXiv : 2009.03300 [cs.CY]. url : https://arxiv.org/abs/2009.03300.

[10] Stanislav Fort. « Scaling Laws for Adversarial Attacks on Language Model Activations ». In : *ArXiv* abs/2312.02780 (2023). url : https://api.semanticscholar.org/CorpusID:265658920.

[11] Yinhan Liu et al. *RoBERTa : A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv : 1907.11692 [cs.CL]. url : https://arxiv.org/abs/1907.11692.

[12] Yuanzhi Li et al. *Textbooks Are All You Need II : phi-1.5 technical report*. 2023. arXiv : 2309.05463 [cs.CL]. url : https://arxiv.org/abs/2309.05463.

[13] Taiming Lu et Philipp Koehn. « Every Language Counts : Learn and Unlearn in Multilingual LLMs ». In : *arXiv preprint arXiv:2406.13748* (2024).

[14] Xinxi Chen et al. *Honest AI : Fine-Tuning "Small" Language Models to Say "I Don't Know", and Reducing Hallucination in RAG*. 2024. arXiv : 2410.09699 [cs.CL]. url : https://arxiv.org/abs/2410.09699.

A Définition du Problème

A.1 Désapprentissage Exact

À l'instar du désapprentissage approximatif, le **Désapprentissage Exact** repose sur un réentraînement naïf à partir de zéro. Contrairement au réentraînement naïf, qui consiste à réentraîner l'ensemble des données à retenir dans son intégralité, le désapprentissage exact essaie de réentraîner un sous-modèle en utilisant uniquement un sous-ensemble de l'ensemble de données restant afin de réduire les coûts de calcul. Il convient alors de d'abord diviser l'ensemble de données en plusieurs petits sous-ensembles, puis de transformer le processus d'apprentissage en assemblant les sous-modèles entraînés avec chaque sous-ensemble pour former le modèle final. Ainsi, lorsqu'une requête de désapprentissage survient, il suffit de réentraîner le sous-modèle correspondant au sous-ensemble contenant les données effacées. Ensuite, le sous-modèle réentraîné est assemblé avec les autres sous-modèles pour former le modèle désappris.

Néanmoins, bien que cette méthode ait pour avantage de produire un modèle désappris identique à celui réentraîné, elle présente un grand nombre de contraintes : elle nécessite souvent l'accès aux données d'origine ; si de nombreuses requêtes de désapprentissage sont effectuées, cela implique de réentraîner le modèle de nombreuses fois ; dans la pratique, elle implique souvent un réentraînement d'une partie importante du modèle et reste extrêmement coûteuse.

A.2 Désapprentissage Approximatif et Désapprentissage Faible

Supposons m une métrique positive, telle que $m(h(x;\theta_r))$ et $m(h(x;\theta_u))$ soient les distributions respectives de $\mathcal{A}(S \setminus S')$ et $\mathcal{U}(\mathcal{A}(S), S, S')$. La définition du (ϵ, δ)-***unlearning*** nous donne :

$$\begin{cases} m(h(x;\theta_r)) \leq e^{\epsilon}\, m(h(x;\theta_u)) + \delta \\ m(h(x;\theta_u)) \leq e^{\epsilon}\, m(h(x;\theta_r)) + \delta \end{cases}$$

avec ϵ et δ petits et strictement positifs.

On peut réajuster la deuxième équation : $e^{-\epsilon}\, m(h(x;\theta_u)) - e^{-\epsilon}\delta \leq m(h(x;\theta_r))$, tel que,

$$e^{-\epsilon}\, m(h(x;\theta_u)) - e^{-\epsilon}\delta \leq m(h(x;\theta_r)) \leq e^{\epsilon}\, m(h(x;\theta_u)) + \delta$$

Définition du **désapprentissage faible** :

$$\frac{\mathbb{E}_{x\sim p_D}\left[m\left(h(x;\theta_u)\right)\right]}{\mathbb{E}_{x\sim p_D}\left[m\left(h(x;\theta_r)\right)\right]} \in [1-\eta, 1+\eta]]$$

Cette condition doit être vérifiée pour différentes distributions de données $p_{D_f}(x)$, $p_{D_r}(x)$ et $p_{D_g}(x)$, représentant respectivement les données d'origine, oubliées et générales.

Sans perte de généralité, nous allons uniquement considérer $x \in D_f$ et traiter les autres cas

comme identiques. Nous allons en plus supposer que $m(h(x;\theta_u))$ est non nul et supérieur à $A > 0$ pour tout x [15], on a alors :

$$\begin{aligned}
e^{-\epsilon}\, m(h(x;\theta_u)) - e^{-\epsilon}\delta &\leq m(h(x;\theta_r)) \leq e^{\epsilon}\, m(h(x;\theta_u)) + \delta \\
e^{-\epsilon}\, m(h(x;\theta_u)) - \delta &\leq m(h(x;\theta_r)) \leq e^{\epsilon}\, m(h(x;\theta_u)) + \delta \\
e^{-\epsilon} - \frac{\delta}{m(h(x;\theta_u))} &\leq \frac{m(h(x;\theta_r))}{m(h(x;\theta_u))} \leq e^{\epsilon} + \frac{\delta}{m(h(x;\theta_u))} \\
e^{-\epsilon} - \frac{\delta}{\mathbb{E}[m(h(x;\theta_u))]} &\leq \frac{\mathbb{E}[m(h(x;\theta_r))]}{\mathbb{E}[m(h(x;\theta_u))]} \leq e^{\epsilon} + \frac{\delta}{\mathbb{E}[m(h(x;\theta_u))]} \\
e^{-\epsilon} - \frac{\delta}{A} &\leq \frac{\mathbb{E}[m(h(x;\theta_r))]}{\mathbb{E}[m(h(x;\theta_u))]} \leq e^{\epsilon} + \frac{\delta}{A} \\
1 - \eta &\leq \frac{\mathbb{E}[m(h(x;\theta_r))]}{\mathbb{E}[m(h(x;\theta_u))]} \leq 1 + \eta
\end{aligned}$$

où $\eta = \max\left(\left|\frac{\delta}{A} - e^{-\epsilon} + 1\right|, \left|e^{\epsilon} + \frac{\delta}{A} - 1\right|\right)$.

On obtiendrait donc une implication du (ϵ, δ)-*unlearning* vers le désapprentissage faible.

A.3 Métriques de TOFU

A.3.1 Probabilité

Sans perte de généralité, supposons que a_1 est la réponse correcte, alors la probabilité est calculée comme suit :

$$P = \frac{P(a_1|q)}{\sum_{i=1}^{n} P(a_i|q)} \tag{5}$$

Ainsi, cette métrique est toujours sous forme d'une probabilité entre zéro et un.

A.3.2 ROUGE

Considérons deux séquences $Z = [z_1, z_2, \ldots, z_n]$ et une autre séquence $X = [x_1, x_2, \ldots, x_m]$. S'il existe une séquence strictement croissante $[i_1, i_2, \ldots, i_k]$ d'indices de X telle que pour tout $j = 1, 2, \ldots, k$, nous avons $x_{i_j} = z_j$.

Étant donné deux séquences X et Y, le LCS de X et Y est une sous-séquence commune avec une longueur maximale, de sorte que :

$$ROUGE\text{-}L = \frac{LCS(R, C)}{|R|} \tag{6}$$

où nous avons la réponse de référence R et la réponse générée C, et $|R|$ est la longueur de la séquence de référence.

A.3.3 Truth Ratio

Soit $\tilde{a}$ une version paraphrasée de la réponse, et en conséquence $\tilde{x} = [q, \tilde{a}]$. Ils utilisent un ensemble de cinq perturbations $\mathcal{A}_{\text{pert}}$ qui conserve la structure générale de la réponse mais qui

15. Cette hypothèse n'est pas contraignante car l'output $h(x;\theta_u)$ peut être considéré comme presque sûrement non nul

sont factuellement incorrectes [16]. Le ratio de vérité R_{truth} peut s'écrire comme suit :

$$R_{\text{truth}} = \frac{\frac{1}{|\mathcal{A}_{\text{pert}}|} \sum_{\hat{a} \in \mathcal{A}_{\text{pert}}} P(\hat{a} \mid q)^{1/|\hat{a}|}}{P(\tilde{a} \mid q)^{1/|\tilde{a}|}} \tag{7}$$

A.3.4 KS-Test

Soit $F_U(x)$ comprenant n échantillons et $F_R(x)$ comprenant m échantillons, les fonctions de distribution cumulées empiriques des modèles désappris et retain, respectivement. Ensuite, le **KS-Test** calcule une statistique

$$D_{n,m} = \sup_x |F_U(x) - F_R(x)| \,. \tag{8}$$

L'hypothèse nulle, stipulant que les deux ensembles d'échantillons sont tirés de la même distribution, est rejetée à un niveau de signification α choisi si l'inégalité suivante est vérifiée.

$$D_{n,m} > c(\alpha)\sqrt{\frac{n+m}{nm}}, \text{ avec } c(\alpha) = \sqrt{-\ln\left(\frac{\alpha}{2}\right) \cdot \frac{1}{2}}.$$

où $c(\alpha)$ est la valeur critique de ce niveau de signification.

La valeur p est ensuite définie comme le alpha minimal pour lequel l'inégalité est vérifiée, ou la plus petite valeur à laquelle nous pouvons rejeter l'hypothèse nulle. Cette métrique est utilisée à la place d'autres tests comme le test de Wilcoxon ou le test de Student, car ces deux derniers comparent les tendances centrales comme les médianes ou moyennes entre les distributions, mais ne capturent pas les différences dans les distributions qui nous intéressent.

B Méthodologie

B.1 Seuil de décision : Prédiction conforme

Une méthode pour déterminer le seuil optimal se base sur la prédiction conforme, qui est utilisée pour déterminer un seuil optimal pour la classification basé sur un taux d'erreur cible α et des scores de non-conformité, comme deuxième méthode de calibration du seuil. Un "score de non-conformité" quantifie à quel point une prédiction est inhabituelle ou improbable par rapport aux prédictions passées d'un modèle.

B.2 Entraînement des Modèles de Language

La perte sur un échantillon $x \in S$ est exprimée en fonction des poids du modèle w, donnée par

16. le fait d'utiliser des versions paraphrasées et perturbées nous permet de détecter les cas où les modèles ne produisent plus de correspondances exactes

$$\ell(x, w) = \frac{1}{|a|} \sum_{i=1}^{|a|} \mathrm{NLL}_w \left(a_i \mid [q, a_{<i}]\right),$$

où NLL_w est la log-vraisemblance négative selon les sorties du modèle paramétré par w. Ensuite, nous cherchons à trouver w^* qui minimise la perte moyenne sur l'ensemble de données noté L,

$$L(S, w) = \frac{1}{|S|} \sum_{x \in S} \ell(x, w).$$

Dans nos expériences, nous optimisons cette perte avec AdamW [17], sur cinq à quinze époques, et appliquons une phase d'échauffement pour la première époque. Nous utilisons une taille de batch de 16 paires question-réponse, contre 32 dans le papier de recherche, pour s'adapter à nos capacités en termes de mémoire vive.

C Résultats et Analyses

C.1 Méthode ECO

Une simple remarque additionnelle qui sort légèrement du cadre de notre problématique, mais qui est importante à noter dans le contexte du Machine Unlearning en général (et de l'IA générative). Les réponses générées par notre modèle réentraîné, tout comme celles de notre modèle désappris, tendent à tomber dans le cadre des *hallucinations*. C'est-à-dire que notre modèle, au lieu de simplement nous informer qu'il n'a pas la réponse à notre question, va chercher à inventer une réponse. Bien que cela ne nous intéresse pas particulièrement dans le cadre de ce rapport, dans un contexte professionnel, où il est question de produire une méthode de désapprentissage livrable, c'est un point important à prendre en considération.

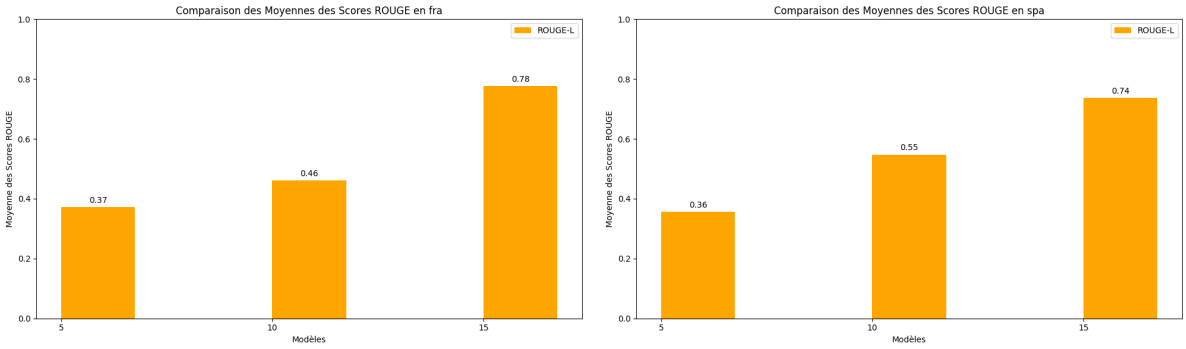
Cela soulève aussi un second problème que je ne développerai pas ici, mais tout aussi passionnant, qui est le problème de *l'alignement*, où les algorithmes de Machine Learning répondent à la tâche imposée (ici le désapprentissage en produisant un modèle se rapprochant du modèle réentraîné), mais ne répondent pas exactement aux attentes des chercheurs (par exemple un modèle qui n'hallucinerait pas).

C.2 Approche Multilingue

C.2.1 Finetune de Qwen2.5-0.5b sur les données traduites

On note une nette amélioration du *ROUGE Score* moyen en fonction du nombre d'epochs d'entraînement.

17. AdamW est un algorithme d'optimisation ajustant automatiquement le taux d'apprentissage en fonction des moments du gradient



(a) Finetuning en Français (b) Finetuning en Espagnol

Figure 8 – **ROUGE-L Score** pour différentes epochs **en français et espagnol**.

Afin d'illustrer que notre modèle oublie effectivement même si sa Forget Quality est faible, nous avons utilisé la métrique *Probability* introduite dans la partie métrique de TOFU 2.5.1, indiquant la probabilité de répondre correctement, puis l'avons inversée et nommée **Error Probability** (i.e., $Error\ Probability = 1 - Probability$). Intuitivement, elle nous indiquerait la probabilité de mal répondre aux questions posées. Nous avons ensuite tracé le graphique de la *Probability* sur le dataset Retain90, en fonction de l'*Error Probability* sur le dataset Forget10 :

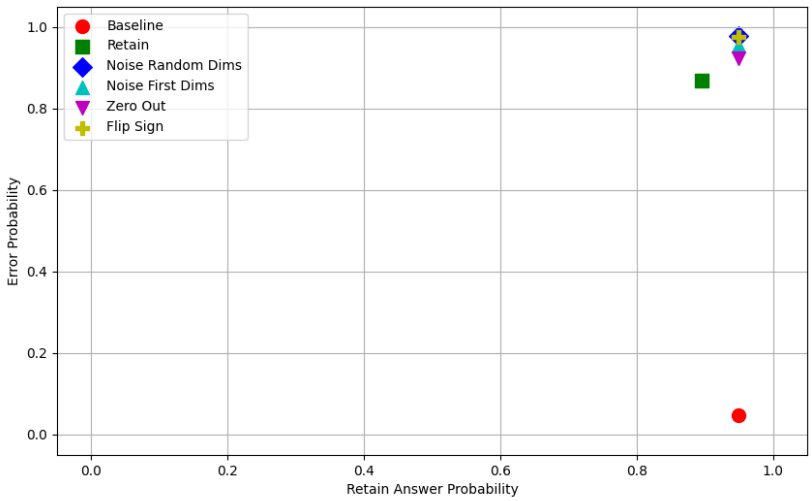
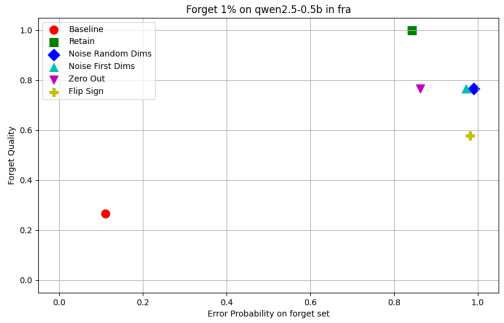


Figure 9 – **Probabilty sur Retain90 en fonction de Error Probability sur Forget10**

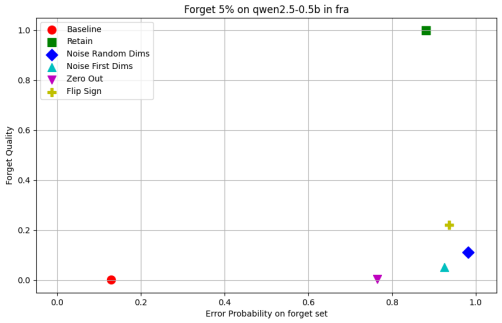
On remarque effectivement que le modèle a oublié les données à oublier (Error Probability élevée) et retenu celles à retenir (Probability élevée).

C.2.2 Désapprentissage avec Qwen dans une autre langue

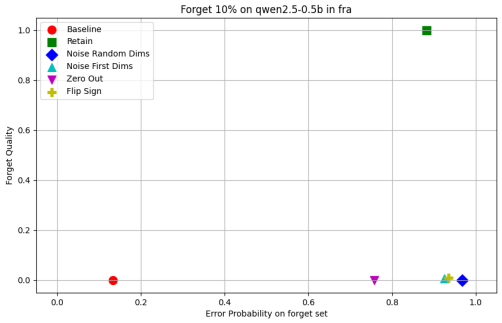
On retrouve respectivement : ”Forget Quality en fonction de l’Error Probability **en français et espagnol**”, où on observe bien que le modèle a oublié les informations des *Forget Sets* :



(a) Oubli de 1% des données

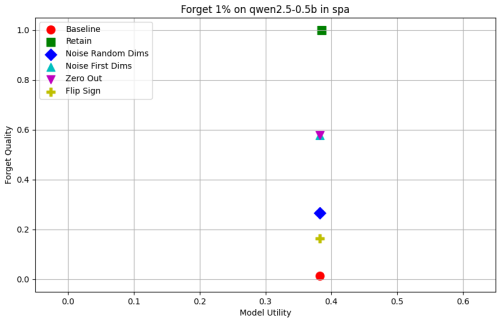


(b) Oubli de 5% des données

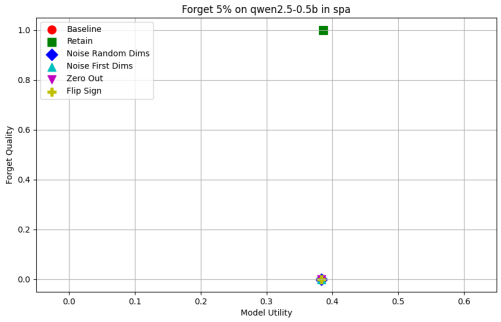


(c) Oubli de 10% des données

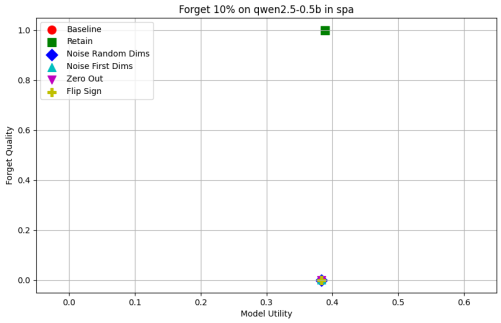
Figure 10 – **Forget Quality en fonction de l’Error Probability** pour les différents niveaux d’oubli sur le dataset TOFU traduit **en français**.



(a) Oubli de 1% des données



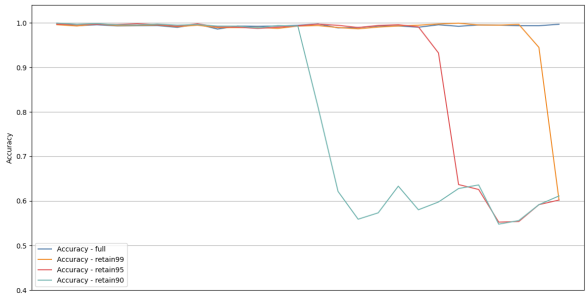
(b) Oubli de 5% des données



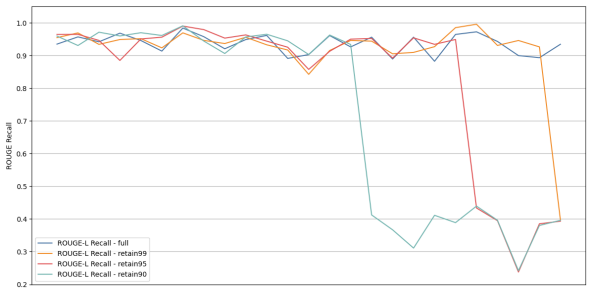
(c) Oubli de 10% des données

Figure 11 – **Forget Quality en fonction du Model Utility** pour les différents niveaux d'oubli sur le dataset TOFU traduit **en espagnol**.

Accuracy et ROUGE Score de modèles fine-tune en anglais et évaluées sur des données en espagnol.



(a) Accuracy en Espagnol



(b) ROUGE Score en Espagnol

Figure 12 – **Accuracy et ROUGE Score** des modèles Qwen fine-tune en anglais sur les 20 derniers % données en Espagnol.

7 Notes de Synthèse

Ce stage s'est déroulé au sein du laboratoire Aubay INNOV de la société Aubay, dédiée à la recherche en intelligence artificielle (IA). Le sujet du stage, intitulé *Rollback Learning* et portant sur le Machine Unlearning, explore une problématique majeure en IA : comment faire oublier des informations spécifiques à un modèle de langage, comme s'il ne les avait jamais apprises. L'objectif est de respecter les exigences réglementaires (ex. : RGPD) tout en évitant de réentraîner le modèle depuis le début, ce qui serait coûteux.

Les modèles d'IA, en particulier les grands modèles de langage (LLM), apprennent à partir de vastes quantités de données, incluant parfois des informations sensibles ou erronées. Dans notre contexte actuel de plus en plus réglementé, il devient essentiel de supprimer ces informations de manière ciblée, sans pour autant affecter les autres connaissances du modèle. L'enjeu est donc de **trouver des méthodes permettant d'oublier des données spécifiques, sans dégrader les performances globales du modèle**.

Pour aborder cette problématique, une étude approfondie de l'état de l'art a été réalisée, menant à la sélection de la méthode ECO. Cette méthode ne modifie pas le modèle de langage lui-même, mais agit sur les *prompts* (questions/requêtes adressées par les utilisateur). Elle repose sur un *classificateur de prompts*, qui identifie quelles requêtes des utilisateurs portent sur des informations à oublier, privées ou dangereuses. Une fois identifiées, ces questions sont modifiées pour limiter les informations sensibles que le modèle pourrait révéler.

Le travail de recherche s'est déroulé en plusieurs phases :

— **Définition de la problématique** et sélection des méthodes adaptées.

— **Implémentation de la méthode ECO**, d'abord en anglais, puis dans d'autres langues et combinaisons de langues.

— **Expérimentation et évaluation** des performances du modèle désappris sur plusieurs jeux de données et métriques.

Les expériences ont démontré que la méthode ECO permettait de supprimer efficacement les informations ciblées sans nécessiter de réentraîner le modèle. Les tests sur les jeux de données ont révélé une réduction significative des connaissances du modèle sur les données à oublier, tout en maintenant ses connaissances générales. De plus, des tests supplémentaires dans d'autres langues ont permis d'évaluer l'efficacité du modèle à éviter de répondre à des questions dans différentes langues. En effet, il arrive souvent qu'un modèle désapprenne une information dans une langue mais reste capable de répondre lorsqu'on l'interroge dans d'autres langues.

Bien que les résultats soient prometteurs, certaines limites subsistent, notamment lorsque le volume des données à oublier augmente. Des recherches supplémentaires sont nécessaires dans des contextes multilingues pour renforcer l'efficacité du désapprentissage. Les travaux futurs pourraient également porter sur l'optimisation de la méthode ECO afin d'assurer un désapprentissage plus ciblé et plus robuste.

This internship took place at the Aubay INNOV lab within the company Aubay, dedicated to research in artificial intelligence (AI). The project, titled *Rollback Learning* and focusing on Machine Unlearning, explores a major challenge in AI : how to make a language model forget specific information as if it had never learned it. The aim is to meet regulatory requirements (e.g., GDPR) while avoiding the costly process of retraining the model from scratch.

AI models, particularly large language models (LLMs), learn from vast amounts of data, which may sometimes include sensitive or erroneous information. In our increasingly regulated context, it has become essential to remove such information in a targeted way without impacting the model's other knowledge. The challenge is therefore to **find methods that allow specific data to be forgotten without degrading the model's overall performance**.

To address this issue, an in-depth literature review was conducted, leading to the selection of the ECO method. This method does not alter the language model itself but instead acts on the *prompts* (questions/queries from users). It relies on a *prompt classifier*, which identifies user queries related to information that needs to be forgotten, whether private or sensitive. Once identified, these questions are modified to limit the sensitive information the model might reveal.

The research project was carried out in several phases :

— **Defining the problem** and selecting appropriate methods.

— **Implementing the ECO method**, initially in English, then expanding to other languages and language combinations.

— **Experimentation and evaluation** of the unlearned model's performance on multiple datasets and metrics.

Experiments showed that the ECO method effectively removes targeted information without the need to retrain the model. Tests on the datasets revealed a significant reduction in the model's knowledge of the data to be forgotten while maintaining its general knowledge. Additionally, further tests in other languages assessed the model's ability to avoid answering questions in different languages. Indeed, it often happens that a model forgets information in one language but can still respond when queried in other languages.

While the results are promising, certain limitations remain, particularly when the volume of data to be forgotten increases. Further research is needed in multilingual contexts to strengthen the effectiveness of unlearning. Future research could also focus on optimizing the ECO method to ensure more targeted and robust unlearning.