RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Inria

Cybersécurité spécifique à l'apprentissage automatique

Policy Paper

Agence de programmes Numérique - Algorithmes, logiciels et usages

12 novembre 2024

Cybersécurité spécifique à l'apprentissage automatique

Contributeurs : CEA, CNRS, Confiance.AI, Inria

12 novembre 2024

1 Introduction

1.1 Contexte général et périmètre de l'étude

L'intelligence artificielle (IA) a vu le jour dans les années 1950, avec des pionniers comme Alan Turing et John McCarthy, qui ont posé les bases de la discipline. Depuis, l'IA a connu plusieurs périodes d'engouement et de stagnation, avant de connaître une renaissance dans les années 2000 grâce aux progrès en puissance de calcul et à l'abondance de données. Les avancées récentes en apprentissage automatique (*machine learning* (ML) en anglais) ont permis des projets importants dans des domaines comme la reconnaissance d'image ou le traitement du langage naturel.

Le ML recouvre une grande variété d'approches, qui vont de l'analyse statistique à l'apprentissage profond (*deep learning*), en mode supervisé ou non, pour prédire (régression / classification) à partir de données comme pour générer des données (IA générative).

Des briques de ML commencent à être diffusées dans de nombreux systèmes d'informations (SI) utilisés en santé, marketing ou industries créatives. Elles sont désormais étudiées et parfois utilisées pour des applications en sécurité et défense. Cette diffusion rapide suscite des préoccupations croissantes en matière de cybersécurité, conduisant à la publication de recommandations[1].

Il convient de distinguer la sûreté de ces briques de ML, soit leur capacité à résister à des défaillances, de leur sécurité, soit leur capacité à résister à des attaques réalisées par des acteurs malveillants.

Les liens entre le ML et la cybersécurité sont triples. Le ML est utilisé par des attaquants pour faciliter voire automatiser des cyber-attaques, des attaques informationnelles (*fake news*, désinformation), des attaques contre la sécurité globale (génération de substance chimique toxiques, etc.). Le ML peut aussi être utilisé par les défenseurs et pourrait devenir un outil clé en cybersécurité, pour détecter les cyberattaques ou les propriétés malveillantes de logiciels, entre autres. Enfin, la cybersécurité du ML est le troisième lien. Les deux premiers liens ne sont pas traités dans ce document, qui focalise uniquement sur le dernier.

Les systèmes d'informations (SI) intégrant des briques de ML sont bien entendu eux-mêmes, comme tous les SI, attaquables et doivent donc être protégés (sécurité préventive) et défendus (sécurité réactive). Les attaques contre ces SI peuvent être classiques (exemples : déni de service par inondation de requêtes, intrusion dans le système informatique), ciblant

[1] L'ANSSI a ainsi publié un document visant à sensibiliser les administrations et entreprises aux risques liés aux systèmes d'IA génératives https://cyber.gouv.fr/publications/recommandations-de-securite-pour-un-systeme-dia-generative. Voir aussi le guide de l'ENISA (*securing ML Algorithms),* celui du BSI (*Security of AI Systems Fundamentals*), ou encore celui du DHS (*Safety and Security Guidelines for Critical Infrastructure Owners and Operators*). Enfin, voir le document du NIST « Artificial Intelligence Risk Management Framework (AI RMF 1.0) » https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.100-1.pdf.

alors les divers composants du SI, ou très spécifiques, ciblant alors les briques de ML (exemple : tenter de déterminer si une donnée précise a servi à entraîner un modèle). Ce document traite de ces attaques spécifiques contre les briques de ML, quelle que soit la forme de ces briques (composant logiciel, implémentation matérielle), et des contre-mesures qui peuvent ou pourraient être déployées pour contrer ces attaques spécifiques. Les attaques contre les briques de ML sont ici définies comme l'ensemble des actions compromettant la confidentialité, l'intégrité ou la disponibilité des briques en elles-mêmes ou des données qu'elles manipulent.

Ce document ne traite pas des limitations du ML, comme le manque d'explicabilité ou d'équité, qui ne sont pas liés à la présence d'un attaquant.

1.2 Difficultés de la sécurité du ML

Le paysage des attaques contre les briques de ML reste mal défini et mal compris, posant des défis importants aux spécialistes de cybersécurité.

Les briques de ML sont vulnérables tout au long de leur cycle de vie (préparation des données, entraînement, validation, inférence), dans un contexte où la complexité et parfois l'opacité des algorithmes de ML posent des difficultés en termes de compréhension et de possibilité d'audit.

L'une des principales difficultés tient au manque d'explicabilité des algorithmes de ML. Le fonctionnement interne d'un réseau de neurones est difficilement interprétable, par exemple, même lorsque son architecture et ses paramètres sont connus. Cela complique l'identification des vulnérabilités potentielles et la mise au point des mécanismes de protection (sécurité préventive). En outre, cela rend aussi difficile l'identification des conséquences d'une attaque qui aurait été couronnée de succès (sécurité réactive). Une autre difficulté est la taille de certains modèles et de leurs entrées, qui crée une réelle complexité à la mise en œuvre de techniques de protection.

Par ailleurs, au-delà des vulnérabilités techniques que nous avons évoquées jusqu'ici, la concentration des services entre les mains de quelques de grandes entreprises accroît les risques, car cette concentration entraîne la centralisation de grandes quantités de données au sein de ces entreprises, ainsi que les outils de développement d'AI comme TensorFlow, et, bien entendu, dans un futur proche, des modèles eux-mêmes. Ces entreprises fournissent en outre des services conviviaux dont le déploiement et la gestion nécessitent une expertise technique minimale, ce qui les rend particulièrement attrayants pour les PME qui ne disposent pas des ressources nécessaires en interne, en particulier pour la cybersécurité. Cela entraîne une dépendance qui peut être problématique.

1.3 Objectif et plan de ce document

Ce document a pour objectif, en complément du travail mené notamment par l'ANSSI, d'éclairer l'État sur les attaques possibles contre les briques de ML et sur les contre-mesures existantes. En outre, il propose des recommandations pour aider l'État dans son analyse du sujet et ses prises de décision.

Cependant, plus généralement, ce document a aussi pour objectif d'apporter à tous ceux qui participent aux processus de développement, d'intégration et de maintenance de SI intégrant des briques de ML, non pas des solutions finalisées, méthodologiques ou techniques, mais un tableau des problématiques et un état de l'art des contre-mesures. Cela sera, nous l'espérons, un premier pas vers des projets de recherches collaboratifs futurs.

En conséquence, le plan de ce document est le suivant. Il précise les risques pesant sur le ML (section 2) et les contre-mesures connues pour pallier ces risques (section 3). La maturité des contre-mesures et leur impact sur les performances de la brique de ML sont précisés. Enfin, ce document contient des recommandations (section 4) pour aider l'État dans son analyse des questions liées à la cybersécurité de l'IA.

2 Les risques pesant sur les briques de ML

Cette section présente les vulnérabilités techniques des briques de ML, puis les risques additionnels engendrés par la concentration des services entre les mains de quelques grandes entreprises.

2.1 Vulnérabilités techniques et attaques spécifiques contre le ML

Cette section présente les principaux types de cyberattaques ciblant spécifiquement les briques de ML. Pour rappel (voir introduction), les attaques classiques, comme par exemple les dénis de services par inondation, ne sont pas traitées dans ce document.

La suite évoque principalement deux phases du cycle de vie des briques de ML en tant que cibles principales d'attaques : la phase d'apprentissage et la phase d'inférence. En effet, c'est durant ces phases que l'essentiel des attaques vraiment spécifiques au ML se déroulent.

Cependant, il faut noter que lorsqu'on considère un système applicatif intégrant des briques de ML, ces deux phases se situent dans un processus plus global, éventuellement bouclé :

- phase d'acquisition de données d'apprentissage, de prétraitement et de validation ;
- phase de prototypage, réglage fin du modèle, évaluation ;
- phase d'apprentissage du modèle exécutable ;
- phase de validation/vérification de ce modèle ;
- phase de déploiements de ce modèle, avec d'éventuelles optimisations en fonction des caractéristiques (HW, SW) des moteurs d'inférence ;
- phase de production/inférence avec acquisition des données de production et diffusion/mémorisation des données résultats ;
- phase éventuelle d'audit et d'analyse des résultats ;
- phase de réapprentissage ou d'apprentissage incrémental.

Chacune de ces phases[2] est réalisée dans des environnements où les conditions matérielles et environnementales peuvent être très spécifiques d'un point de vue cybersécurité, au sens général cette fois, à distinguer de la sécurité spécifique des briques de ML. La sécurité de ces phases est bien entendu à étudier, mais elle sort du périmètre de ce document.

Il faut enfin noter que les problématiques de cybersécurité liées à l'apprentissage fédéré ou décentralisé sont caractérisées de la même façon que celles liées au contexte centralisé. Les architectures décentralisées multiplient potentiellement les risques d'attaques, puisque les traitements sont répartis sur de nombreux nœuds qui transfèrent leurs résultats sur des réseaux éventuellement attaquables, mais les spécificités des attaques propres au ML restent les mêmes que dans l'architecture centralisée.

La classification des attaques proposée dans la suite de ce document est basée sur la propriété de sécurité violée : confidentialité, intégrité, et/ou disponibilité.

[2] Un environnement de production n'a pas les mêmes contraintes qu'un environnement de développement et les outils de protection et réaction ne sont donc pas les mêmes.

- Confidentialité : il s'agit ici de prévenir la fuite d'informations sensibles concernant essentiellement les données d'entraînement et les paramètres du modèle, mais aussi les requêtes des utilisateurs. Les informations sensibles comprennent les données personnelles au sens du RGPD. Il faut noter que le risque pour la confidentialité de données personnelles existe si ces données n'ont *pas* été anonymisées ou pseudonymisées avant utilisation (par exemple pour l'apprentissage ou l'inférence).
- Intégrité : il s'agit ici de se protéger ou de détecter toute modification malveillante des données d'entraînement, des paramètres du modèle et des données de test.
- Disponibilité : il s'agit ici de se protéger contre toute action visant à dégrader les performances du système, voire à l'empêcher de fonctionner.

L'évaluation des risques réels liés aux attaques présentées par la suite nécessite une analyse fine car la mise en place de protections et de systèmes de réponse engendre des coûts techniques et humains qu'il est important de mettre en regard de la vraisemblance des menaces (quantification) et des conséquences de leur occurrence (criticité de l'impact).

La quantification de la menace contre des briques de ML doit ainsi tenir compte :

- des connaissances de l'attaquant. Bien entendu, si celui-ci peut, d'une manière ou d'une autre, obtenir de l'information sur des paramètres de la brique attaquée (poids, architecture, mécanismes d'optimisation), la menace s'accroît. En l'absence de telles connaissances, l'attaquant doit faire appel à des techniques permettant de les approximer, ce qui introduit des incertitudes et diminue en conséquence la menace.
- des capacités de calcul de l'attaquant. Par exemple, l'attaque « appartenance / reconstruction » présentée ci-dessous peut faire appel à des « modèles fantômes » reproduisant le modèle de la brique à attaquer. La production de ce modèle fantôme a un coût computationnel qui limite le risque si l'attaquant ne dispose pas de moyens de calcul suffisant.

En outre, quantification de la menace et criticité de l'impact sont à réévaluer régulièrement pour prendre en compte les avancées rapides de l'état de l'art académique (tant côté attaque que côté défense), qui se transpose souvent sans délai en outils technologiques. Cela influence naturellement le modèle de l'attaquant et la vraisemblance des menaces. L'effort à faire pour se maintenir à l'état de l'art est conséquent.

Le tableau ci-dessous résume les types d'attaque possibles, que les paragraphes suivants détaillent une à une (en prenant les attaques colonne par colonne, et de haut en bas).

	Phase d'apprentissage (préparation de la brique de ML)	*Phase d'inférence (exploitation de la brique de ML)*
Confidentialité	Extraction de données d'apprentissage locales (en apprentissage fédéré)	Attaques dites « par oracle » - Extraction (du modèle) - Appartenance, reconstruction
Intégrité	- Empoisonnement - Cheval de Troie	- Évasion (attaque adverse) - Contournement
Disponibilité	- Empoisonnement - Empoisonnement éponge	- Exemple éponge

2.1.1 Extraction de données d'apprentissage locales (en apprentissage fédéré)

But : au niveau du serveur central, extraire des données d'apprentissage d'un client local.

Propriété violée : confidentialité des données d'apprentissage.

Modus operandi : en apprentissage fédéré, à chaque itération, un serveur central agrège les modèles locaux de clients pour créer un modèle global qui sera redistribué aux clients. L'attaquant agit au niveau du serveur et manipule les paramètres (« *trap weight* ») du modèle global avant que celui-ci soit diffusé aux clients. L'objectif de la manipulation est d'obliger un modèle local à encapsuler des données pendant son apprentissage. Une fois ce modèle remonté au serveur, l'attaquant peut extraire ces données avant la phase d'agrégation.

Conséquence : fuite d'informations sur les données d'apprentissage (locales).

2.1.2 Empoisonnement

But : influencer le comportement du système lorsqu'il sera déployé en production.

Propriété violée : intégrité des données d'apprentissage.

Modus operandi :

- modification des données d'apprentissage (étiquette ou donnée elle-même) à un moment donné de leur cycle de vie (production, collecte, traitement). Cette attaque se passe à l'entraînement ou à l'affinage (*fine-tuning*, *alignment*) du modèle ;

Ou

- modification de l'algorithme d'apprentissage afin que le modèle généré soit erroné.

Conséquence : impact sur le service offert par le système. Les attaques ciblées visent la mauvaise classification de certaines données d'entrée (voir aussi ci-dessous la notion d'évasion) ou des données générées erronées. Les attaques non-ciblées visent à dégrader le système en général (par exemple, utilisateurs byzantins en apprentissage fédéré).

2.1.3 Cheval de Troie

But : influencer le comportement du système lorsqu'il sera déployé en production.

Propriété violée : intégrité du modèle.

Modus operandi : modification de la structure ou des paramètres d'un modèle déjà appris.

Conséquence : impact sur le service offert par le système car mauvaise classification de l'entrée présentée (voir aussi ci-dessous la notion d'évasion).

2.1.4 Exemple ou empoissonnement éponge

But : Ralentir le système en production.

Propriété violée : disponibilité du modèle.

Modus operandi :

- modification des données présentées au modèle en phase d'inférence (données de production) ou sélection de données de production difficile à analyser. Le principe des « exemples éponges » à l'inférence peut être transposé à l'apprentissage pour empoisonner un modèle afin d'accroître sa consommation énergétique et sa latence de manière indiscriminée pour chaque entrée de test.
- optionnel : une préparation de l'attaque en amont par un empoissonnement lors de l'apprentissage.

Conséquences : une augmentation de la latence avant résultat ou de la consommation énergétique.

2.1.5 Extraction

But : vol du modèle.

Propriété violée : confidentialité du modèle.

Modus operandi :

- envoi de requêtes ou de perturbations (logicielles ou physiques) au modèle et observation des sorties correspondantes.
- optionnel : apprentissage en mode maître/esclave[3] d'une copie du modèle.

Conséquences : identification du modèle ou de la famille du modèle dans une boite noire pour préparer une attaque d'évasion, contrefaçon du modèle.

2.1.6 Appartenance / reconstruction

But : inférer si une donnée à servi à l'entraînement d'un modèle, voire reconstruire des données d'entraînement.
Note : la confidentialité doit aussi être regardé au sens large. En effet, notamment pour les modèles génératifs à base de diffusion, ce qui est appris est les caractéristiques compressées de la distribution des données utilisées pour l'apprentissage. On peut alors avec ces modèles regénérer aussi des données non utilisées pour l'apprentissage. On n'expose pas seulement alors des données individuelles, mais des informations implicites critiques (taux d'images avec des typologies de défauts, …).

Propriété violée : confidentialité des données d'entraînement.

Modus operandi :

- à partir d'un modèle en boite blanche, mesurer l'importance d'une donnée sur les paramètres du modèle (en particulier, sur les mises à jour des paramètres d'un modèle).
- à partir d'un modèle en boite noire, se servir de sa confiance dans la classification d'une donnée. Éventuellement, se servir de données d'apprentissage connues pour entraîner des modèles fantômes.
- A partir d'une IA générative en boite noire, construire des requêtes contournant le filtrage des prompts pour obtenir des informations sur les données d'apprentissage.

Conséquences : fuite d'informations sur les données d'apprentissage ou les données augmentées (*Retrieval Augmented Generation pour les LLM*).

Commentaire : L'état de l'art établit que le taux de succès de cette attaque est très lié au sur-apprentissage (*overfittin*g), c'est-à-dire, lorsque le modèle appris a des perfomances bien meilleures sur ses données d'entrainement que sur les données de validation / test. Ainsi, plus le jeu de données d'apprentissage est grand, plus le risque de sur-apprentissage diminue, plus le modèle généralise bien, et moins l'attaque fonctionne. D'autre part, certaines données d'apprentissage sont plus faciles à inférer / reconstruire que d'autres suivant leur impact dans l'entrainement du modèle. Par exemple, des données présentes en plusieurs exemplaires dans le jeu d'entrainement sont plus sensibles à ces attaques.

[3] En anglais, on parle de l'approche *teacher/student*.

2.1.7 Évasion (exemple adverse)

But : faire produire au système une sortie erronée pour une entrée donnée.

Propriété violée : intégrité d'une donnée d'entrée en phase d'inférence.

Modus operandi : ajout de perturbations (idéalement imperceptibles) à une donnée d'entrée, entraînant des erreurs en sortie du modèle (exemple adverse, *adversarial example*).

Conséquences : production d'une sortie incorrecte mettant en péril l'intégrité des systèmes dépendants de cette sortie.

Commentaire : L'évasion est bien documentée dans la littérature lorsque les données d'entrée de départ sont numériques (par exemple, une image au format JPEG, un fichier audio au format WAV). Les exemples adverses dits physiques consistent à modifier des objets ou des ondes sonores du monde réel de telle sorte qu'une fois captés et numérisés, leurs représentations numériques leurrent le modèles IA. Une mise en œuvre possible consiste à ajouter un patch sur un objet afin de tromper la brique de ML traitant toute image de cet objet. L'état de l'art montre l'existence de telles attaques dans le monde physique, mais avec des taux de succès et des efficacités largement amoindris. Pour autant, l'applicabilité de cette forme particulière d'évasion semble plus importante.

2.1.8 Contournement

But : faire dysfonctionner un système d'IA générative (type LLM) en contournant les protections intégrées. Ces protections sont : le pré-filtrage des requêtes (interdisant par exemple une réponse à une requête (prompt) portant sur tel ou tel sujet), son alignement par affinage (mécanisme ayant intégré au modèle en lui-même le respect de certaines contraintes), ou le post-filtrage (censure des résultats générés détectés comme toxiques).

Propriété violée : intégrité des données de production.

Modus operandi : injection d'une requête ou une suite de requêtes spécifique, pour faire dysfonctionner le système.

Conséquences : génération de sorties illégitimes ou illicites. Cette utilisation malveillante d'une IA :

- favorise la désinformation (*fakenews, deepfakes*) et le montage d'arnaque à grande échelle,
- porte atteinte à la réputation de l'IA ciblée et à son intégrité si elle est interconnectée à d'autres agents.

2.2 Accroissement des risques engendré par l'externalisation des données

La concentration des services au sein d'un petit nombre d'acteurs présente des risques engendrés par la centralisation des données qui en résulte.

La centralisation des données crée en particulier un risque important pour la confidentialité des informations personnelles. Les réglementations en matière de protection des données (par exemple, GDPR en Europe, CCPA en Californie) varient d'une juridiction à l'autre ; il est difficile de garantir la conformité à toutes les lois applicables. Le stockage centralisé des données implique souvent le transfert de données au-delà des frontières, ce qui peut constituer une violation des lois locales sur la protection des données. Le volume d'informations personnelles stockées par les géants de la technologie fait de leurs bases de

données des cibles particulièrement intéressantes pour les attaquants (parmi lesquels il faut compter des employés mécontents ou des sous-traitants mal intentionnés).

Face à ces constats, il est important de trouver un équilibre entre les avantages potentiels liés à l'utilisation des données pour entraîner des modèles de ML et les risques en matière de confidentialité, en particulier pour les informations personnelles. Cela nécessite des efforts continus pour améliorer les pratiques de collecte et de stockage des données de façon distribuée, ainsi que pour développer des techniques de protection adaptées.

3 Contre-mesures proposées et coûts induits

Il est important de mentionner en introduction de cette section que les mécanismes usuels de sécurité des SI doivent évidemment être appliqués. Ce qui suit présente des contre-mesures additionnelles renforçant spécifiquement la sécurité des briques de ML intégrées à ce SI. Cette règle a cependant quelques exceptions (usage de la cryptographie homomorphe, de la confidentialité différentielle, de l'anonymisation des données, de code signé).

Le tableau ci-dessous est une liste non exhausive de contre-mesures possibles, que les paragraphes suivants détaillent.

	Phase d'apprentissage (préparation de la brique de ML)	*Phase d'inférence (exploitation de la brique de ML)*
Confidentialité	. Apprentissage distribué . Chiffrement homomorphe . Confidentialité différentielle	. Chiffrement homomorphe . Confidentialité différentielle . Limitation du nombre des requêtes, filtrage de prompts . Protection la confidentialité du modèle
Intégrité	. Désinfection des données . Augmentation de la robustesse (via apprentissage adverse)	. Modèle signé . Supervision des entrées, des espaces latents, des sorties (Operational Design Domain & Out Of Domain Distribution) . Combinaison de plusieurs modèles ML . Mise en œuvre d'un apprentissage incrémental au lieu d'un apprentissage continu
Disponibilité	N/A	Supervision

3.1 Contre-mesures applicables en phase d'apprentissage

3.1.1 Apprentissage distribué/fédéré (AF)

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : l'AF vise à prévenir les violations de la confidentialité des données en garantissant que les données brutes ne quittent jamais les appareils locaux : seules les mises à jour du modèle sont partagées. L'AF fonctionne en entraînant des modèles d'apprentissage machine sur plusieurs appareils ou serveurs décentralisés contenant des données locales, sans les échanger. Un exemple d'AF classique comprend (le processus se répète jusqu'à ce que le modèle converge) :

- Une phase d'initialisation : un serveur central initialise un modèle global.
- Des apprentissages locaux : chaque client télécharge le modèle global et l'entraîne localement à l'aide de ses propres données.
- Des mises à jour du modèle : les clients renvoient leurs mises à jour de modèle (et non les données brutes) au serveur central.

- Des agrégations : le serveur central regroupe ces mises à jour pour améliorer le modèle global qu'il redistribue aux clients.

Attaque(s) contrée(s) :

- Empoisonnement du modèle : des algorithmes d'agrégation robustes peuvent atténuer l'impact des clients malveillants qui tentent d'empoisonner le modèle global.
- Reconstruction : l'AF peut réduire le risque d'attaques par inférence où un adversaire tente de déduire des informations sensibles à partir des mises à jour du modèle.
- Violations de la confidentialité des données : en conservant les données localement, L'AF empêche l'accès direct aux données sensibles. Cependant, un mécanisme additionnel (cf. 3.1.3 Confidentialité différentielle) est nécessaire pour prévenir la fuite d'information indirecte à partir du modèle partagé.

Impact sur les performances :

- Surcharge de calcul : l'AF peut éventuellement augmenter la surcharge de calcul en raison du besoin de ré-entraînement des modèles en local sur les clients et de communication fréquente entre les clients et le serveur. Pour autant, cette surcharge reste à confirmer/qualifier (l'AF est parfois mis en œuvre pour répartir la charge de calcul alors que toutes les données sont centralisées).
- Coûts de communication : l'échange répété de mises à jour de modèle entraîne un coût de communication élevé.
- Précision du modèle : la précision du modèle global peut être affectée par l'hétérogénéité des données entre les clients et la qualité de l'entraînement local des modèles.

Maturité :

- En production : L'AF est de plus en plus adopté dans les environnements de production, en particulier dans des secteurs comme la santé, la finance et l'IoT.
- En développement : des améliorations continues et de nouvelles techniques sont développées pour améliorer l'efficacité, la sécurité et l'évolutivité de ce type d'apprentissage.
- En recherche : des recherches actives sont en cours pour relever des défis tels que la personnalisation des modèles, la robustesse face aux attaques adverses et l'efficacité des communications.

Défis de recherche présentés par la contre-mesure :

- Évolutivité : garantir que l'AF peut évoluer efficacement avec un grand nombre de clients et gérer des distributions de données hétérogènes.
- Sécurité : développer des mécanismes robustes pour se protéger contre l'empoisonnement des modèles, les attaques par inférence et d'autres menaces adverses.
- Efficacité de la communication : réduire la surcharge de communication sans compromettre la précision du modèle.
- Personnalisation : adapter le modèle global pour mieux s'adapter aux données locales des clients individuels.
- Confidentialité : améliorer les garanties de confidentialité, éventuellement grâce à l'intégration de techniques de confidentialité différentielle.

3.1.2 Chiffrement homomorphe

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : lorsque l'apprentissage est externalisé, le traitement de données brutes est effectué par une entité distincte de celles propriétaires de ces données. Lorsque l'apprentissage est fédéré, les mises à jour des modèles (comme les poids des réseaux neuronaux) sont envoyées à un serveur central pour agrégation. Un serveur central malveillant pourrait révéler des informations sensibles sur les données locales des utilisateurs, ce qui pose un problème de confidentialité. Le chiffrement homomorphe (homomorphic encryption, HE) est un ensemble de schémas cryptographiques permettant l'exécution d'opérations mathématiques sur données chiffrées, sans recours au déchiffrement. Il est utile en particulier lors de l'agrégation chiffrée dans un cadre d'apprentissage fédéré car cette technique permet de faire l'agrégation des modèles avec des données chiffrées. Il faut noter que le résultat est lui-même chiffré, si bien qu'il faut posséder la clé de déchiffrement pour accéder à ce résultat.

Dans l'exemple de l'apprentissage fédéré, le HE assure la confidentialité car les mises à jour locales sont envoyées sous forme chiffrée au serveur central, qui renverra le résultat de l'agrégation aux participants collaborant à l'apprentissage du modèle commun. Ainsi, un attaquant capable de lire les données sur le serveur central n'aura accès qu'à des données chiffrées.

Attaque(s) contrée(s) : extraction

Impact sur les performances : Le chiffrement homomorphe, bien qu'il offre des avantages significatifs en termes de sécurité et de confidentialité, présente encore des défis de recherche au niveau de sa performance. Cela limite son adoption généralisée dans des applications nécessitant une grande efficacité et disposant de ressources limitées. Les opérations de chiffrement et de déchiffrement nécessitent des ressources de calcul importantes. De même les données chiffrées sont plus volumineuses que les données initiales, avec des impacts forts sur la bande passante et le stockage.

Maturité : en recherche, mais des preuves de concept ont déjà été réalisées.

Défi de recherche présenté par la contre-mesure : deux principaux défis : (1) l'amélioration de la performance, par exemple pour l'utilisation du HE sur des SI ayant des ressources limitées (smartphones, IoT) et qui ont du mal à gérer les opérations de chiffrement/déchiffrement en raison de leurs contraintes de puissance et de traitement et (2) l'utilisation combinée de plusieurs techniques de confidentialité (chiffrement homomorphe, confidentialité différentielle, protocoles de calcul multipartite sécurisé), qui offrent des compromis intéressants entre performance et sécurité.

3.1.3 Confidentialité différentielle

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : ajout de bruit de manière dynamique et contrôlée aux données d'apprentissage ou aux calculs intermédiaires de l'apprentissage afin de masquer des données sensibles. Remarque : la même technique d'ajout de bruit peut être utilisée sur les sorties du modèle en phase d'inférence (voir ci-dessous).

Attaque(s) contrée(s) : appartenance / reconstruction

Impact sur les performances : l'utilité du modèle appris (par exemple, la précision pour un classifieur) est réduite. L'équilibre entre protection et utilité n'est pas toujours simple à

trouver. En revanche, les garanties apportées par la confidentialité différentielle sont très fortes, même dans des scénarios extrêmes.

Maturité : recherche

Défi de recherche présenté par la contre-mesure : il est difficile d'obtenir un bon compromis utilité-protection ; on constate de grosses pertes d'utilité dès qu'on demande une confidentialité théorique ayant vraiment du sens. Ceci est en parti dû au modèle d'attaque extrême supposé pour la confidentialité différentielle : l'attaquant connaît toutes les données sauf une, et observe toutes les mises à jour du modèle pendant l'entraînement. L'établissement de contre-mesures basé sur un modèle d'attaque plus réaliste est une piste de recherche.

3.1.4 Désinfection des données

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : Lors de la construction des jeux de données, la contre-mesure consiste à éliminer les valeurs aberrantes, normaliser les données, limiter les valeurs d'entrée à des plages préétablies, détecter des patterns invalides, détecter les sur-représentation de certaines typologies de données, faire valider par des utilisateurs la validité des données.

Attaque(s) contrée(s) : Évasion / empoisonnement

Impact sur les performances : L'impact sur les performances est indirect, le coût du nettoyage des données, notamment quand il repose sur des actions manuelles, limite la quantité de données disponible. Un compromis est à trouver entre qualité/quantité de la désinfection, vis-à-vis d'une performance cible du modèle à l'issue de l'apprentissage.

Maturité : partiellement en production, partiellement en développement, et encore en recherche.

Défi de recherche présenté par la contre-mesure : si des solutions existent déjà en production, les défis pour le développement ou la recherche existe pour améliorer l'automatisation de ces étapes de nettoyage, y compris avec d'autre modèles de ML, comme par exemple l'utilisation de modèles auxiliaires détecteurs d'anomalie ou de modèle génératif pour purifier des données.

3.1.5 Augmentation de la robustesse (dont apprentissage adverse)

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : la phase d'apprentissage d'un modèle de ML fait appel à un ensemble de méthodes pour améliorer sa robustesse face à un ensemble de perturbations possibles des données utilisées. Ces méthodes rentrent dans plusieurs catégories :

- augmentation des données par génération de perturbations aléatoires représentatives des perturbations attendues (par exemple, simple ajout de bruit gaussien).
- inclusion de données spécifiques aux limites du domaine d'usage (voir ODD ci-après), mais celles-ci sont difficiles à générer.
- utilisation d'architecture avec des propriétés de robustesse intrinsèque.

- utilisation de modèle adverse de contre-exemple : Une méthode spécifique au modèle de ML pour augmenter la robustesse est l'inclusion d'exemples adverses dans les données d'apprentissage. Elle vise à exposer le modèle à des données spécifiquement construites pour perturber le modèle (cf. Evasion). Ces exemples adverses inclus dans l'entraînement permettent au modèle de combler ces faiblesses.

Le choix parmi ces méthodes repose souvent sur un compromis entre performance, temps de calcul, apport en robustesse pour le modèle (voir ci-dessous impact sur les performances).

Attaque(s) contrée(s) : attaques adverses, évasion

Impact sur les performances : l'augmentation de la robustesse se fait au prix d'une réduction de la performance en général. Le compromis performance / robustesse est difficile à contrôler.

Maturité : partiellement en production, partiellement en développement, et encore en recherche.

Défi de recherche présenté par la contre-mesure : si des solutions pour augmenter la robustesse des modèles et pour fournir des indications sur leur efficacité existent déjà en production, la quantification précise du niveau de robustesse atteint et les garanties associées restent des questions de recherche. Un des défis importants est d'identifier comme utiliser la contribution de la robustesse en combinaison avec les autres contre-mesures.

3.2 Contre-mesures applicables en phase d'inférence

3.2.1 Chiffrement homomorphe

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : le chiffrement homomorphe est une technique cryptographique qui permet de réaliser des opérations sur des données chiffrées sans avoir besoin de les déchiffrer. On peut donc effectuer des calculs tout en préservant la confidentialité des données, ce qui est utile dans certains contextes d'inférence. Cela dit, il ne faut pas sous-estimer le coût de calcul d'une opération d'inférence dans le domaine chiffré, qui sera élevé et augmentera la latence.

Attaque(s) contrée(s) : extraction.

Impact sur les performances : Le chiffrement homomorphe, bien qu'il offre des avantages significatifs en termes de sécurité et de confidentialité, présente encore des défis de recherche au niveau de sa performance. Cela limite son adoption généralisée dans des applications nécessitant une grande efficacité et disposant de ressources limitées. Les opérations de chiffrement et de déchiffrement nécessitent des ressources de calcul importantes. De même les données chiffrées sont plus volumineuses que les données initiales, avec des impacts forts sur la bande passante et le stockage.

Maturité : en recherche, mais des preuves de concept ont déjà été réalisées.

Défi de recherche présenté par la contre-mesure : deux principaux défis : (1) l'amélioration de la performance, par exemple pour l'utilisation du HE sur des SI ayant des ressources limitées (smartphones, IoT) et qui ont du mal à gérer les opérations de chiffrement/déchiffrement en raison de leurs contraintes de puissance et de traitement et (2)

l'utilisation combinée de plusieurs techniques de confidentialité (chiffrement homomorphe, confidentialité différentielle, protocoles de calcul multipartite sécurisé), qui offrent des compromis intéressants entre performance et sécurité.

3.2.2 Confidentialité différentielle

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : elle consiste à ajouter de manière contrôlée du « bruit » aux sorties du modèle, afin de masquer d'éventuelles données sensibles.
Remarque : c'est la même technique de bruitage utilisée durant la phase d'apprentissage sur les données d'entrée ou les paramètres du modèle en construction (voir ci-dessus).

Attaque(s) contrée(s) : appartenance / reconstruction

Impact sur les performances : la précision des modèles est réduite en raison du bruit ajouté ; l'équilibre entre protection et utilité n'est pas toujours simple à trouver.

Maturité : recherche

Défi de recherche présenté par la contre-mesure : il est difficile d'obtenir un bon compromis utilité-protection ; on constate de grosses pertes d'utilité dès qu'on demande une confidentialité théorique ayant vraiment du sens.

Commentaire : le modèle d'attaque supposé pour la confidentialité différentielle est sur-puissant : l'attaquant connaît toutes les données sauf une et observe toutes les mises à jour du modèle pendant l'entraînement.

3.2.3 Limitation du nombre de requêtes, filtrage de prompts

Nature de la contre-mesure : réactive

Principe de fonctionnement de la contre-mesure : Les attaques par oracle (extraction de modèle, reconstruction/appartenance) envoient des requêtes au modèle et observent les sorties correspondantes. En conséquence, plus le nombre de requêtes (variées) envoyées est grand, plus l'information que l'attaquant peut déduire est importante. En conséquence, une règle simple consiste à limiter le nombre de requêtes possibles par unité de temps et/ou à augmenter le temps de latence de la réponse, au moins pour ralentir l'attaque.

Pour le cas spécifique de l'IA générative, certains offreurs de service mettent en place un filtrage des prompts (technique usuelle en cybersécurité consistant en la validation des entrées *ou input sanitization*). Ces filtres (*safeguards* en anglais) sont souvent opérés par/dans le LLM lui-même. Régulièrement, ils doivent être mis à jour en fonction de l'évolution des menaces (on découvre des contournements constamment) et la base de filtres doit être enrichie (il est simple de vérifier que des prompts -- publiés -- donnant des résultats problématiques il y a quelques mois ne le permettent plus aujourd'hui).

Attaque(s) contrée(s) : extraction (du modèle), appartenance/reconstruction

Impact sur les performances : évidemment, cette contre-mesure limite le nombre de requêtes légitimes d'un utilisateur non-malveillant.

Maturité : en production

Défi de recherche présenté par la contre-mesure : évaluation des capacités de contournement de ces formes de filtrage.

3.2.4 Protection de la confidentialité du modèle

Nature de la contre-mesure : préventive

Principe de fonctionnement de la contre-mesure : il s'agit en fait d'un ensemble de contre-mesures.
Si le modèle est embarqué (par exemple dans un smartphone), l'obfuscation des poids du modèle est une option. Le GPU exécute ce réseau masqué donnant des résultats inexploitables que la partie sécurisée de l'appareil (*Trusted Execution Environment*) démasque sachant la clé secrète.
Si le modèle est accessible en boite noire, l'extraction de modèle demande de nombreuses requêtes. Il faut donc en limiter le nombre (voir ci-dessus). Ajouter du bruit aux réponses du modèle (confidentialité différentielle, voir ci-dessus) rend aussi plus difficile l'extraction précise du modèle. On peut également mettre en place un système d'analyse des requêtes pour détecter les motifs suspects (requêtes en rafale ou requêtes couvrant une large gamme d'entrées, par exemple).
Enfin, le tatouage de modèle est une arme dissuasive. Le tatouage facilite l'identification du modèle, c'est-à-dire le rend reconnaissable afin de prouver la propriété du dit modèle en cas de litige. A cette fin, le modèle est entrainé sur des entrées spécifiques qui produisent des résultats connus uniquement par le propriétaire du modèle. En outre, l'ajout de petites perturbations spécifiques et intentionnelles aux paramètres du modèle pendant l'entraînement sont identifiables en phase de d'inférence.

Attaque(s) contrée(s) : extraction du modèle

Impact sur les performances : Ralentissement de la vitesse d'inférence (obfuscation), limitation du nombre de requêtes ou perte d'utilité (confidentialité différentielle, tatouage)

Maturité : dépend de la contre-mesure précise

Défi de recherche présenté par la contre-mesure : la robustesse du tatouage face à des attaques d'extraction en boite noire est mal évaluée. Le poids de la preuve de la propriété par détection de tatouage est incertain. Il dépend de la probabilité de fausse détection (difficile à évaluer) et de la possibilité de créer un faux tatouage a posteriori afin d'usurper l'identité du propriétaire. La technique de tatouage doit être encapsulée dans un protocole (dépôt ou engagement des clés secrètes).

3.2.5 Modèle signé

Nature de la contre-mesure : réactive

Principe de fonctionnement de la contre-mesure : calcul d'un résumé cryptographique (par hachage cryptographique) sur l'ensemble des paramètres du modèle stocké sur disque et stockage de ce résumé de manière sécurisée. En cas de besoin (soupçon de modification du modèle, ou vérification sur une base régulière), le résumé est recalculé sur un modèle suspect et comparé au résumé stocké.
Le modèle peut éventuellement être modifié alors qu'il s'exécute en mémoire. Dans ce cas, le résumé cryptographique n'est pas utile et il faut signer le modèle tel qu'il s'exécute. Des

techniques classiques de signature de code peuvent être mises en œuvre. La spécificité du ML conduit en particulier à signer la zone de mémoire contenant les paramètres du modèle.

Attaque(s) contrée(s) : cheval de Troie

Impact sur les performances : pas d'impact pour le résumé cryptographique. En revanche, la vérification de la signature de (bloc de) code à un coût calculatoire.

Maturité : en développement

Défi de recherche présenté par la contre-mesure : passage à l'échelle de mécanismes usuellement appliqués à des objets de plus petite taille que les modèles de ML modernes.

3.2.6 Supervision des entrées, des espaces latents, des sorties (Operational Design Domain & Out Of Domain Distribution)

Nature de la contre-mesure : réactive

Principe de fonctionnement de la contre-mesure : une fois qu'un modèle de ML est déployé en production, il est important d'en surveiller le fonctionnement pour détecter les signes d'attaques (ou de dysfonctionnements plus généralement). Cela est réalisé en monitorant les performances de la brique de ML, en analysant ses journaux d'accès et les requêtes des utilisateurs, en utilisant des techniques de détection d'anomalies. Plus généralement, la supervision exploite (1) les données d'entrée/sortie du modèle pour détecter des anomalies, (2) l'historique des entrées/sorties du modèle ML pour détecter des attaques progressives ou (3) l'espace latent du modèle (c-à-d ses calculs intermédiaires) pour détecter des anomalies dans l'espace de représentation interne.

La détection d'attaques de type « évasion » consiste à repérer des séquences de requêtes suspicieuses (multiples requêtes quasi-identiques, requêtes alignées suivant une direction, etc.).

Par ailleurs, pendant la phase de conception de la brique de ML, le domaine de fonctionnement opérationnel (ODD) de celle-ci a été défini. L'ODD décrit les conditions d'utilisation de la brique. Des modèles auxiliaires (à base de règles ou eux-mêmes à base de ML) sont ajoutés à la brique pour confirmer que l'usage de la brique de ML reste dans l'ODD.

Enfin, l'approche « Out Of domain Distribution » (OOD) détecte que des données sont aberrantes par rapport aux données sur lesquelles le modèle a été entraîné (autrement dit, si ce sont des données provenant d'une distribution différente de celle des données d'entraînement). Cela permet de détecter des situations rares (donc pas représentées dans les données d'entraînement) pour s'abstenir de faire une prédiction ou, à tout le moins, indiquer une incertitude élevée.

Le mécanisme de supervision peut, au choix du concepteur, désactiver la brique de ML, bannir des utilisateurs et/ou remonter des alertes.

Attaque(s) contrée(s) : potentiellement, toutes.

Impact sur les performances : selon la technique de supervision implantée, impact éventuel sur le coût de calcul et la disponibilité de la brique de ML.

Maturité : partiellement en production (moteur de règles appliquées sur les entrées), mais encore beaucoup de développement et de recherche (analyse de l'espace latent par exemple).

Défi de recherche présenté par la contre-mesure : au-delà des moniteurs à base de règle, l'exploitation de moniteur reposant eux aussi sur des modèles de ML. En outre, quantification, intrinsèque ou post Hoc, de l'incertitude des modèles alors utilisés (soit on entraine le modèle à évaluer l'incertitude de ces sorties, soit on met en place un composant dédié considérant les sorties du modèles en fonction des sollicitations d'entrée).

3.2.7 Combinaison de modèles

Nature de la contre-mesure : réactive

Principe de fonctionnement de la contre-mesure : utiliser un ensemble de modèles (*ensemble learning*) et agréger leurs résultats pour chaque requête. Le plus souvent, un vote, une moyenne (éventuellement pondérée) ou un méta-modèle combine les prédictions de plusieurs modèles exécutés en parallèle.
Il faut noter que cette approche « ensembliste » est assez classique en ML. Pour autant, l'approche traditionnelle ne vise pas à améliorer la sécurité, mais la qualité des résultats de sortie ; il s'agit d'avoir des modèles spécialisés et meilleurs dans des sous-parties distinctes du domaine de fonctionnement opérationnel (ODD). Ici, l'objectif est différent : combiner des modèles différents (par exemple utilisant des algorithmes d'apprentissage différents) mais donnant des résultats similaires hors attaques. Dès lors, si une attaque a lieu, il est probable que les modèles réagissent différemment (car ils sont par construction différents les uns des autres) et produisent donc des sorties différentes. L'observation d'une diversité de résultats de sortie permet la détection de l'attaque. Notons que savoir quel modèle(s) continue(nt) de produire une sortie correcte en cas d'attaque n'est pas simple. Passer de la détection de l'attaque à sa tolérance (capacité du système à rendre sa fonction malgré l'attaque) est donc difficile.

Attaque(s) contrée(s) : dans tous les cas, le recours à plusieurs modèles complique l'extraction par l'attaquant d'un modèle unique. Il protège donc contre l'extraction de modèle. En outre, il protège aussi contre les attaques par oracle et évasion (car, dans le cas d'une exécution en parallèle, tous les modèles ne sont sensibles de la même manière à ces types d'attaque).

Impact sur les performances : la consommation énergétique, l'empreinte mémoire et aussi le temps de réponse (si exécution séquentielle) sont augmentés puisqu'on exécute plusieurs modèles. Il y a aussi y avoir un impact sur la disponibilité car l'un des modèles peut sortir de sa zone de fonctionnement nominal (sans que les autres en sortent) sans pour autant subir d'attaque. Dans ce cas, une attaque est détectée à tort, ce qui rend le système indisponible.

Maturité : partiellement en production (on dispose des techniques algorithmiques), mais encore beaucoup de développement et de recherche (méthodologie permettant de déterminer le compromis acceptable).

Défi de recherche présenté par la contre-mesure : méthodologie permettant de déterminer un bon compromis entre performance de la brique ML, disponibilité de cette brique et efficacité de la détection.

3.2.8 Mise en œuvre d'un apprentissage incrémental au lieu d'un apprentissage continu

Nature de la contre-mesure : réactive

Principe de fonctionnement de la contre-mesure : l'apprentissage continu (modification directe du modèle en exécution) des modèles ML présente un risque lié à la non maîtrise de la dérive. Il est donc préférable de procéder à un apprentissage incrémental pour répondre aux besoins d'évolution d'une solution ML.

L'apprentissage incrémental exploite le système de supervision (voir ci-dessus) pour identifier après sélection (par exemple identification des données perçues comme malveillantes) les données à utiliser pour compléter/étendre l'entraînement de manière incrémentale : données en dehors de l'ODD, données pour lesquelles le système a échoué, etc. On vérifie ensuite que ce nouveau modèle respecte toujours les exigences de sécurité avant de le mettre en production.

Le nouveau modèle produit intervient en remplacement et/ou en complément du modèle existant (voir combinaison de modèles).

Attaque(s) contrée(s) : potentiellement toute attaque en phase d'inférence.

Impact sur les performances : pas d'impact direct sur les performances, mais le coût de vérification du nouveau modèle est un frein.

Maturité : partiellement en production, partiellement en développement, et encore en recherche,

Défi de recherche présenté par la contre-mesure :
L'apprentissage incrémental est déjà mis en œuvre, mais il reste de nombreux défis de recherche pour rendre celle-ci plus efficace pour réduire le temps d'apprentissage (frugalité des données utilisées), réduire les interactions humaines (*active learning*, auto annotation, etc.).

3.3 Défis de mise en œuvre des solutions proposées

La mise en œuvre des solutions proposées ci-dessus ce heurte à plusieurs défis présentés ci-dessous.

Un *défi sur la réponse globale/holistique à l'ensemble des risques*. La diversité des attaques, et des contre-mesures y répondant rend complexe une approche globale couvrant l'ensemble des risques. En outre, au fur et à mesure que les technologies évoluent, les attaquants déploient de nouvelles techniques sophistiquées pour contourner les défenses. Les systèmes de sécurité doivent donc sans cesse s'adapter pour faire face à des menaces en constante évolution.

Un *défi de contradiction potentielle entre les contre-mesures*. Certaines contre-mesure peuvent être antinomiques les unes des autres. La littérature scientifique est riche de travaux étudiant une contre-mesure précise à une attaque donnée. En général, le coût de cette contre-mesure est évalué en termes de perte d'utilité et d'augmentation de calculs ou données nécessaires. Or, le présent document montre que la surface d'attaque comprend de nombreuses menaces. Trop peu de travaux scientifiques étudient l'impact d'une contre-mesure précise sur toutes les attaques possibles. Par exemple, l'apprentissage adversaire incluant des exemples adverses lors de l'apprentissage renforce la robustesse des classifieurs (cf. 3.1.5 Augmentation de la robustesse). Quel est son impact sur la

confidentialité des données d'apprentissage (cf. Appartenance / reconstruction) et sur la mise en place de porte dérobée (cf. Empoisonnement) ?
Par exemple, si un modèle a été rendu plus robuste en incluant des données adverses lors de la phase d'apprentissage, l'apprentissage incrémental peut dégrader progressivement la robustesse du modèle si on ne lui applique pas à nouveau des données adverses, augmentant ainsi le coût de l'apprentissage. Ainsi, ces contradictions doivent pouvoir être résolues dans la plupart des cas mais doivent être prise en compte ensemble lors de la conception de la protection.

Un défi d'acculturation (auquel ce document tente aussi de contribuer). Les cybermenaces spécifiques aux briques de ML sont mal connues, souvent confondues avec les menaces classiques (déni de service, etc.), voire ignorées par la communauté IA. Or l'adoption de l'IA demande, entre autres, une IA résistante aux attaques.

Un défi de compétence et de culture communes. Il y a peu de chercheurs mais surtout d'ingénieurs ayant une double culture sécurité / IA, avec une compétence suffisante dans ces domaines. La mise en place d'équipes pluridisciplinaires est encore balbutiante.

Un défi méthodologique. Comment estimer les risques réels des attaques contre les briques de ML et, surtout, leurs conséquences ? Comment définir les protections à mettre en place et évaluer leurs performances d'un point de vue sécurité ? Et surtout, comme faire un compromis rationnel entre ce gain en sécurité et la perte potentielle (dans certains cas) par rapport aux performances « fonctionnelles » actuelles sans sécurité ? La plupart du temps, le fournisseur d'IA en est réduit à exécuter telle attaque de l'état de l'art sur son modèle ou ses données. Si cette attaque ne fonctionne pas, il n'est pas sûr qu'une autre attaque réussisse dans le futur.

*Un défi économique. M*ettre en place les mesures suggérées ci-dessus aura un coût économique, en particulier lors de la phase d'exploitation du système. Ce coût est en outre récurrent (le « maintien en condition de sécurité » demande un effort permanent). Il importe de pouvoir mettre en relation, de façon objective, le coût économique aux risques évités par un ensemble de mesures données. Il importe aussi de rendre, par des travaux de recherche associés, plus automatiques, plus performantes et plus frugales les contre-mesures envisagées.

Un défi pour la normalisation. Alors que la mise en application de l'AI Act est prévue pour septembre 2025, il existe un risque que les discussions actuelles pour proposer une définition, des méthodes de calcul et une liste de mesures visant à garantir la cybersécurité d'un système de ML n'aboutissent pas à temps pour permettre la définition des obligations du règlement. Cette complexité de définition de la norme est en partie due au fait que certaines contre-mesures sont encore au stade de la recherche, comme explicité dans la section précédente.

Un défi sur la prise en compte commune de la sûreté (faute accidentelle) et de la sécurité (attaque intentionnelle). La construction du comportement d'un modèle ML repose en partie sur l'observation de son environnement (les données utilisées). Les perturbations de cet environnement (involontaires) et les attaques (menées à dessein par un attaquant) peuvent se mélanger. Les impacts sur la sûreté et la sécurité sont donc parfois difficilement discernables et nécessitent probablement une analyse commune, tout en tenant compte de leurs spécificités.

4 Recommandations

La Commission de l'IA a remis au président de la République, le 13 mars 2024, un rapport intitulé « IA, notre ambition pour la France », qui propose 25 recommandations traçant un plan « d'action aussi ambitieux que réaliste, au service des personnes, de nos besoins, de nos valeurs et de nos principes. Le plan représente un investissement public annuel de 5 Md€ pendant cinq ans. Il comprend des investissements technologiques mais aussi des investissements pour catalyser à la fois la diffusion de l'IA dans l'économie, son déploiement au service des citoyens et une appropriation et formation de toute la société. »

Pour autant, les aspects cybersécurité sont peu développés dans ce rapport, même si celui-ci appelle à « promouvoir des systèmes d'IA sûrs et sécurisés ». Comme nous l'avons indiqué en introduction, les liens entre le ML et la cybersécurité sont triples : ML utilisé par des attaquants, ML utilisé par les défenseurs, cybersécurité du ML. Nous nous sommes focalisés sur le troisième lien tandis que le rapport ne cite explicitement que les deux premiers.

Nos recommandations viennent donc en toute modestie compléter celles du rapport de la commission de l'IA et ne concernent que les aspects cybersécurité du ML.

4.1 Investir dans la recherche et le développement

4.1.1 Soutenir la recherche sur la sécurité du ML, via un programme sur ce sujet

Le paysage de la menace et celui des contre-mesures, que nous avons décrit dans ce document, reste mal connu, mal compris, mal évalué. L'ampleur des travaux à conduire justifierait la mise en place d'un programme sur le sujet, dont la structuration et la gouvernance permettraient d'associer largement les acteurs publics et privés. Ce programme permettrait de mettre en place des actions en réponse aux recommandations suivantes :

Recommandation 1 : faire un état des lieux de la recherche mondiale sur la « cybersécurité du ML » et positionner la France dans le paysage, en particulier vis-à-vis des USA et de la Chine. Faire une analyse des risques structurels pour la France dans le cas où ce positionnement ne serait pas favorable (risque géopolitique de dépendance, risque de perte de compétence et de fuite des cerveaux, risque d'augmentation de la menace cybersécurité, etc.).

Recommandation 2 : lancer des défis scientifiques sur la conception et la réalisation d'attaques spécifiques contre le ML, afin d'évaluer précisément leur criticité et leur difficulté de mise en œuvre.

Recommandation 3 : allouer des fonds spécifiques pour travailler sur les « défis de recherche » identifiés dans ce document et développer des solutions innovantes aux problèmes de sécurité du ML.

Recommandation 4 : favoriser le partage de données produites ou captées dans le monde industriel, afin de permettre des travaux de recherche nourris par ces données. En outre, favoriser l'accès contrôlé à des modèles industriels, pour nourrir des travaux visant à l'étude des caractéristiques de ces modèles (architecture, etc.) d'un point de vue cybersécurité.

Recommandation 5 : développer l'acculturation de la communauté IA aux problématiques de la cybersécurité et, parallèlement, acculturer les spécialistes en cybersécurité aux

différentes technologies du ML. Cette double acculturation nous semble indispensable pour casser les silos et permettre un travail interdisciplinaire fructueux, qu'il faudra par ailleurs encourager et favoriser. Elle peut se concrétiser par la mise en place d'un cycle de conférences dédiées où les chercheurs des deux domaines partageront leurs travaux et échangeront des idées.

Recommandation 6 : créer un centre de recherche commun, réunissant, au-delà de la double-acculturation évoquée ci-dessus, des chercheurs des deux communautés pour des travaux conjoints. Ce centre s'appuiera sur les pôles universitaires nationaux, en synergie avec d'éventuelles actions conduites par ailleurs (IA-Clusters, campus cyber territoriaux, etc.). Ce centre de recherche sur la cybersécurité du ML pourra aussi proposer des formations. Il pourra par ailleurs, sur le modèle des IA-Clusters, s'appuyer sur une base de soutiens industriels qui pourront alimenter sa feuille de route avec des besoins marché et des retours utilisateurs concrets, et soutenir financièrement ses travaux.

4.1.2 Évaluer l'efficacité des mécanismes de cybersécurité des briques de ML

Un aspect extrêmement important du rapport « IA, notre ambition pour la France » est lié à l'évaluation des systèmes à base d'IA. Dans ce cadre, l'évaluation de la cybersécurité de ces systèmes (au sens large, bien entendu, mais pour ce qui nous concerne au sens des menaces spécifiques présentés dans ce document) est un point crucial. C'est pourquoi nous formulons les deux recommandations suivantes :

Recommandation 7 : identifier et/ou construire un ensemble de challenges (données, modèles et environnement de simulation) représentatif des problèmes de recherche identifiés pour permettre à la communauté de comparer les performances de leurs propositions et de combinaisons de ces propositions. Comme mentionné ci-dessus, la complémentarité ou l'antinomie des contre-mesures envisagées devra faire l'objet d'une attention particulière.

Recommandation 8 : établir des jeux de test standardisés et définir les critères de performance attendue des mécanismes de sécurité du ML.

4.1.3 Intégrer les acteurs de la recherche et de l'innovation dans une dynamique européenne commune

Pour la plus grande efficacité, les recommandations 1 à 8 doivent être menées de front par un ensemble d'acteurs de la recherche et de l'innovation européens. C'est pourquoi nous formulons les trois recommandations suivantes :

Recommandation 9 : pour mieux prendre en compte un contexte européen foisonnant, caractérisé par une profusion d'initiatives et de groupes de travail peu, dresser un état des lieux de ces initiatives afin de s'appuyer sur leurs réflexions et de fournir, si nécessaire, une vision complémentaire. En particulier, tenir compte des travaux de l'AI Office (DG CNECT) et capitaliser sur les groupes de travail existants au sein d'ECSO (European Cyber Security Organisation) et d'EOS (European Organisation for Security).

Recommandation 10 : proposer un axe de travail sur la cybersécurité des briques de ML dans le cadre des programmes financiers européens (Horizon Europe, Digital Europe), visant à soutenir les travaux issus des recommandations 1 à 8.

Recommandation 11 : proposer la mise en place d'un GT d'experts dédié dans le cadre de l'ENISA ou de l'ECCC. Ce GT devra travailler en interaction étroite avec des représentants de la communauté IA en Europe telle qu'elle apparaitra au regard du travail qui sera fait au

titre de la recommandation 9, et pourra s'appuyer sur les contacts de ses membres pour faire un lien avec le bureau européen de l'IA auprès de l'AI Office de la DG CNECT.

Recommandation 12 : organiser un colloque européen biannuel réunissant des instances européennes (ENISA, ECCC, DG CONNECT, etc.), des organismes gouvernementaux (CNIL, ANSSI, VIGINUM, Ministère des Armées, et leurs homologues européens), des centres de recherche (CISPA - Allemagne, sAIFer - Italie, SPY Lab – Suisse, etc.), le réseau de recherche Européen ELSA, les organismes intéressés par la sécurité de l'IA (AISI UK, LNE, CEN-CENELEC, etc.).

4.2 Renforcer le cadre réglementaire

4.2.1 Établir une synthèse des réglementations pouvant s'appliquer à la sécurité du ML

Diverses obligations réglementaires liées à la sécurité des briques de ML apparaissent dans plusieurs réglementations (AI act, DSA, NIS2, etc.). En outre, la future directive CRA imposera de sécuriser à la conception tout dispositif numérique, logiciel ou matériel. Il faudra comprendre les implications que cela représente dans le cas de développement de briques de ML. Par conséquent, nous formulons la recommandation suivante :

Recommandation 13 : réaliser une étude donnant une vision d'ensemble synthétique des réglementations pouvant s'appliquer à la sécurité du ML, pour apprécier les différentes situations, obligations associées et techniques imposées dans ce cadre réglementaire.

4.2.2 Fournir une expertise pluridisciplinaire visant à éclairer l'État sur la réglementation et contribuer activement à l'élaboration des normes d'application des réglementations

Les géants de la technologie exercent une influence sur les processus réglementaires afin de parvenir à des réglementations favorisant leurs intérêts. Si cette influence peut parfois conduire à des politiques favorables à l'innovation, elle peut aussi déboucher sur des réglementations privilégiant les gains commerciaux au détriment des questions de sécurité, en particulier dans le cas de la protection des données personnelles. Il est important que les régulateurs et les législateurs travaillent en étroite collaboration avec les acteurs de l'industrie et avec ceux de la recherche, y compris les géants de la technologie, pour élaborer des réglementations équilibrées qui protègent les intérêts des utilisateurs tout en encourageant l'innovation. Dans ce contexte, nous formulons la recommandation suivante :

Recommandation 14 : créer au niveau national un comité consultatif composé d'experts en cybersécurité et protection des données personnelles, pouvant contribuer (en interaction éventuelle avec d'autres acteurs, comme l'ANSSI) à garantir que toutes les perspectives sont prises en compte dans l'élaboration des réglementations.

De plus, dans le cas de l'AI Act, les obligations réglementaires qui incombent au développeur et à l'utilisateur sont définies en termes de normes et de spécifications techniques au niveau des comités de normalisation, en particulier le CEN-CENELEC (« normes harmonisées » dans le cadre du JTC21). Pour autant, peu d'entreprises européennes prennent activement part aux discussions sur la définition des principes, ce qu'ils contiennent et les moyens de mesurer et évaluer ces critères. Pour favoriser la mise en œuvre effective des obligations réglementaires et contrebalancer le poids de certaines entreprises extra-européennes, nous formulons la recommandation suivante :

Recommandation 15 : fédérer au plus vite des acteurs pluridisciplinaires (recherche, experts en cybersécurité, experts en ML, CESTI, etc.) capables d'alimenter, en amont ou dans le cadre de la commission de normalisation en IA coordonnée au niveau de l'AFNOR, les travaux de définition de la norme relative à la cybersécurité des briques de ML.

4.3 Faciliter la mise en œuvre et encourager les bonnes pratiques

Comme nous l'avons déjà mentionné, il y a souvent une absence de culture de la sécurité de l'information au sein de la communauté IA, et réciproquement. Pourtant, les développeurs et intégrateurs de briques de ML devrons mettre en place des mesures de cybersécurité au moment de l'apprentissage, du test et du déploiement des briques dans un produit ou un service. Ce travail devra en particulier être réalisé en tenant compte des obligations normatives de l'AI Act. Nous avons déjà fait des recommandations pour rapprocher les communautés IA et cybersécurité au niveau recherche. Il faut également développer les compétences techniques en cybersécurité du ML des développeurs ML, des décideurs politiques et des régulateurs. Nous formulons donc les deux recommandations suivantes :

Recommandation 16 : définir un guide de bonnes pratiques pour aider les développeurs et intégrateurs de briques de ML à mettre en place des mesures de cybersécurité au moment de l'apprentissage, du test et du déploiement des briques de ML dans les produits ou services.

Recommandation 17 : proposer un plan de formation pour permettre aux étudiants en cybersécurité et aux étudiants en IA des écoles d'ingénieurs et cursus universitaires, de réciproquement comprendre les enjeux de ces deux domaines.

Avant de déployer un modèle de ML en production, il est important de le valider rigoureusement pour s'assurer qu'il n'est pas vulnérable à des attaques. Une approche de type « *red team* », équipe de défenseurs formés à jouer le rôle de l'attaquant dans le cadre d'une évaluation de sécurité, permet de trouver d'éventuelles failles dans le système évalué (et donc avant déploiement et que de véritables attaquants ne les exploitent). Tous les types d'attaques et toutes les conséquences (atteinte à l'intégrité, à la confidentialité, à la disponibilité) entrent dans le champ d'action de la *red team*. Nous formulons la recommandation suivante :

Recommandation 18 : encourager la formation d'experts en attaque contre le ML et la création d'équipes formées de tels experts (*red teams*) capables de mettre en œuvre des attaques spécifiques contre les briques de ML, afin d'identifier et combler les failles éventuellement découvertes.

5 Conclusion

L'essor fulgurant de l'apprentissage automatique a fait de la question de sa sécurité un enjeu crucial.

Aussi, ce document met-il en lumière les attaques possibles contre les briques de ML intégrés aux systèmes d'information modernes. Par attaques contre les briques de ML, nous entendons des actions compromettant la confidentialité, l'intégrité ou la disponibilité des briques en elles-mêmes ou des données qu'elles manipulent. Ce document ne traite pas des limitations du ML, comme le manque d'explicabilité ou d'équité, qui ne sont pas liés à la présence d'un attaquant.

Ce document expose des contre-mesures possibles pour faire face à ces attaques. Certaines de ces contre-mesures sont d'ores et déjà déployées en production, tandis que d'autres demandent encore des travaux de recherche pour améliorer leurs performances ou diminuer leurs impacts négatifs sur le fonctionnement même de la brique ML auxquelles elles s'appliquent. D'une manière générale, les contre-mesures engendrent des coûts techniques et opérationnels qui ne doivent pas être sous-estimés, en particulier en termes de complexité de mise en œuvre et de performances. L'évolution constante du paysage de la menace demande un investissement de fond dans la recherche et le développement, pour anticiper les attaques émergentes et adapter les contre-mesures en conséquence.

Enfin, ce document présente des recommandations visant à renforcer la cybersécurité de l'apprentissage automatique.

Ces recommandations soulignent en premier lieu l'importance d'investir dans la recherche et le développement sur la sécurité du ML. Il est suggéré de soutenir la recherche via un programme permettant d'approfondir l'étude des menaces et des contre-mesures spécifiques au ML. L'objectif de ce programme est bien entendu d'aller au-delà de l'état actuel de la recherche et de structurer un effort national coordonné sur le sujet. La conception et à la réalisation de briques de ML plus robustes demandent d'évaluer précisément la manière dont les attaques spécifiques contre le ML peuvent être mises en œuvre efficacement et la résilience des systèmes face à ces attaques : l'évaluation sera donc un aspect important du programme.

Par ailleurs, il est important, d'une part d'apporter aux différents acteurs une vision claire du cadre réglementaire s'appliquant à la sécurité du ML, d'autre part d'étudier comment ce cadre réglementaire pourrait encore être renforcé. A cette fin, une expertise scientifique et technique devra éclairer l'État sur la nature exacte des menaces et sur l'efficacité réelles et l'impact des contre-mesures existantes.

Enfin, il est essentiel de faciliter la mise en œuvre de la sécurisation des briques de ML et d'encourager les bonnes pratiques à tous les niveaux. Un effort de formation continue des professionnels de l'IA et de la cybersécurité et un effort de formation initiale des étudiants de ces deux domaines sont souhaitables.

En somme, ces recommandations visent à placer la France à l'avant-garde de la cybersécurité de l'apprentissage automatique, grâce à un investissement important dans la recherche et développement, une appropriation fine de la réglementation par les différents acteurs et un effort de formation de ces mêmes acteurs.