Nº 11-522-x au catalogue
ISSN : 1709822-X

Recueil du Symposium de 2021 de Statistique Canada
Adopter la science des données en statistique officielle pour répondre aux besoins émergents de la société

Classification de texte supervisée au moyen du chiffrement homomorphe

par Zachary Zanussi, Benjamin Santos et Saeid Molladavoudin

Date de diffusion : le 29 octobre 2021

Statistique Canada Statistics Canada

Canada

Recueil du Symposium de 2021 de Statistique Canada
Adopter la science des données en statistique officielle pour répondre aux besoins émergents de la société

Classification de texte supervisée au moyen du chiffrement homomorphe

Zachary Zanussi[1]*, Benjamin Santos[1] et Saeid Molladavoudi[1]

Résumé

Les préoccupations en matière de confidentialité constituent un obstacle à l'application d'analyses à distance, notamment l'apprentissage automatique, sur des données sensibles au sein du nuage. Dans le cadre de ce travail, nous utilisons un schéma de chiffrement totalement homomorphe à niveau pour entraîner un algorithme d'apprentissage automatique supervisé de bout en bout à classer du texte tout en protégeant la confidentialité des points de données d'entrée. Nous entraînons notre réseau neuronal à simple couche sur un important ensemble de données de simulation en fournissant une solution pratique à une tâche de classification de textes réels comprenant de multiples catégories. Pour améliorer à la fois la précision et le temps d'entraînement, nous entraînons un ensemble de tels classificateurs en parallèle par un groupage de cryptogrammes.

Mots clés : protection des renseignements personnels, apprentissage automatique, chiffrement

1. Introduction

Pour obtenir des produits statistiques en temps réel, il faut un accès aux données qui soit aussi en temps réel, mais l'accès à des données transactionnelles non chiffrées en temps réel pourrait présenter des risques quant aux menaces à la confidentialité et aux atteintes à la cybersécurité. Même dans un cyberenvironnement sécurisé qui résiste aux menaces de l'extérieur, les données risquent encore d'être utilisées à mauvais escient par des initiés dûment autorisés à accéder aux données.

Ces dernières années, nous avons assisté à une importante progression des nouvelles techniques de calcul protégeant mieux les renseignements personnels et susceptibles d'éviter de telles atteintes, tout en permettant l'analyse, notamment à l'aide des tâches d'apprentissage automatique (AA) comme l'entraînement et l'inférence. Parmi les techniques existantes, le chiffrement totalement homomorphe (CH) est un candidat de choix au traitement des questions de confidentialité qui se posent dans des scénarios d'apprentissage automatique en tant que service où des tâches d'AA comme l'*entraînement* et l'*inférence* sont déléguées à un fournisseur de services qui n'est pas digne de confiance, comme l'exploitant d'un nuage.

Les schémas CH sont des cryptosystèmes asymétriques qui permettent une évaluation homomorphe de fonctions calculables arbitraires sur données chiffrées. Plus précisément, le chiffrement homomorphe assure des opérations arithmétiques arbitraires comme l'addition et la multiplication sur les données chiffrées. En d'autres termes, il devient possible d'appliquer ces opérations à des données chiffrées pour que soit crypté le résultat des opérations correspondantes sur des données textuelles. Les schémas de chiffrement homomorphe à niveau soutiennent un nombre préétabli d'opérations, de multiplications en particulier, en fonction du circuit visé.

Un client cherche à entraîner un réseau neuronal sur des données sensibles en externalisant ce traitement vers l'informatique en nuage. Nous supposons que le nuage en question est semi-honnête ou honnête mais curieux et que, dans ce cas, il suivra tout protocole assigné, mais en tentant d'apprendre tout ce qu'il peut ce faisant. Le but premier est qu'aucun renseignement nouveau ne fuie de l'ensemble de données privées du client vers le serveur, au-delà de ce qui peut être inféré du modèle entraîné. Après entraînement, le nuage peut soit y aller de prédictions pour le client, soit remettre le modèle à celui-ci pour qu'il fasse localement de l'inférence en mode non crypté. Malgré toutes les

[1] Statistique Canada, 150, promenade Tunney's Pasture, Ontario, Canada, K1A 0T6

* Auteur correspondant, zachary.zanussi@statcan.gc.ca

avancées et les améliorations des systèmes CH, ceux-ci n'ont pas été largement utilisés dans des tâches d'apprentissage automatique intensives de calcul comme l'entraînement de modèles.

Les forts besoins en mémoire pour la classification de textes sont encore amplifiés par l'inflation computationnelle qu'impose le chiffrement homomorphe. Le gros du travail qui s'est fait dans ce dossier ces dernières années a consisté à transformer les systèmes d'apprentissage en mode crypté en algorithmes pratiques qui présentent un compromis entre la sécurité des données et un coût raisonnable de calcul.

Dans cet article, nous nous reportons au système CH à niveau conçu par Cheon, Kim, Kim et Song (2017) pour entraîner un réseau neuronal à simple couche et établir des prédictions pour la tâche AA de classification de textes. L'information servant à notre démonstration de faisabilité est un ensemble de données accessibles au public consistant en descriptions de produits que nous ordonnons en un système de classification de produits de détail reconnu sur le plan international. Nous proposons un classificateur *sécuritaire* et *pratique* qui sauvegarde la confidentialité. Nous passerons d'abord en revue certains éléments préliminaires.

2. Préliminaires

2.1 Chiffrement homomorphe

Durant la dernière décennie la recherche spécialisée a transformé le chiffrement homomorphe (CH) en un cryptosystème à part entière qui est capable de soumettre des données sensibles à des calculs sans sacrifier la confidentialité. Les équipes de recherche s'en servent pour faire de l'entraînement machine, de l'analyse statistique et plus encore. En fait, les systèmes CH sont assez avancés pour trouver leur place dans des systèmes du monde réel (voir, par exemple, Raisaro et coll., 2018).

Le paradigme CH pour l'informatique en nuage est un système asymétrique au fonctionnement bien précis. Une source d'information chiffre ses données avec une clé publique quelconque et défend ainsi l'accès à cette information faute de disposer de la clé secrète en question. Les données peuvent alors être transférées en toute sécurité au nuage, même par des voies de communication non sécurisées. Une fois les données reçues, le nuage peut effectuer les calculs homomorphes désirés sur les valeurs chiffrées, bien qu'il ne peut pas lire les données d'origine ni les résultats intermédiaires ou finals. Les résultats chiffrés finalement obtenus peuvent être renvoyés au titulaire de la clé secrète (qui est normalement le client ou peut-être un tiers), lequel peut ensuite les décrypter et les lire. Il existe plusieurs schémas de chiffrement homomorphe sur données numériques; il sera uniquement question ici du système de Cheon, Kim, Kim et Song (CKKS) qui est conçu pour les calculs sur nombres réels. Comme les calculs de ces systèmes sont approximatifs, on a là le parfait système pour une analyse AA. Nous avons fait ample usage du système CKKS dont nous allons décrire les aspects fondamentaux.

2.2 Espacement

Le système Cheon, Kim, Kim et Song (CKKS) de chiffrement homomorphe est conçu en arithmétique à virgule flottante, ce qui est idéal pour le stockage des matrices de poids d'un réseau neuronal. Nous allons présenter ici une description générale de l'interface de ce système en renvoyant le lecteur intéressé à Cheon, Kim, Kim et Song (2017) pour une description des structures mathématiques et des algorithmes de base.
Posons x un vecteur de nombres réels que nous voudrions chiffrer et soumettre à des opérations homomorphes. Avec une clé publique, nous pouvons crypter tout ce vecteur en un même chiffré dénoté par $[x]$. Que nous chiffrions un *vecteur* de valeurs n'est pas un détail sans importance; comme nous le verrons, le mode d'organisation des valeurs à l'intérieur du vecteur peut influer sur l'exécution. C'est ce qu'on appelle le *groupage*. Nous pouvons imaginer les données en cryptogramme $[x]$ comme un vecteur de valeurs sous la forme $x = (x_0, x_1, \cdots, x_k)$. Chaque coordonnée de ce vecteur est appelée *case* ou *fente*.

Avec deux cryptogrammes $[x]$ et $[y]$, nous avons accès aux opérations homomorphes d'addition $\oplus$ et de multiplication $\otimes$. Elles se font par *case*. En d'autres termes, la somme homomorphe $[x] \oplus [y]$ donne un

cryptogramme $[x + y]$ cryptant un vecteur qui est la somme composée des vecteurs x et y. Il en va de même de la multiplication homomorphe, ce qui fait ressortir l'importance de la structure de groupage, les valeurs devant être dûment alignées en cours de calcul.

Nous avons aussi accès à une opération appelée *rotation*. Pour simplifier, disons que nous pouvons imprimer une rotation à toutes les valeurs à gauche ou à droite selon n'importe lequel nombre de cases. Cette opération relativement onéreuse nous permet de mettre les valeurs d'un cryptogramme en interaction mutuelle. Ainsi, un cumul de $[x]$ sur une rotation permet d'obtenir pour un cryptogramme un **total**$([x])$ qui crypte le total $\sum_i x_i$ dans chaque case.

Tout cryptogramme existe à un niveau donné et les multiplications consomment des niveaux. On se trouve à mettre une borne supérieure à la profondeur multiplicative de circuit réalisable. Ajoutons que les facteurs multipliés et les termes sommés doivent se situer au même niveau pour les opérations à effectuer entre eux.

2.3 Classification de textes

Le traitement en langage naturel (TLN) comme discipline a tiré un immense parti de la mise au point des divers algorithmes d'apprentissage automatique des dernières décennies. La classification de textes est une tâche TLN par laquelle on dispose en une ou plusieurs classes un texte d'entrée non structuré. Au nombre des applications possibles, on compte la détection de spam ou pourriel et l'analyse du climat. Différentes méthodes (voir Kowsari et coll., 2019) se sont révélées efficaces dans le traitement d'une diversité de problèmes de classification de textes. Pour un grand nombre de ces méthodes comme le traitement en réseau neuronal profond ou encore récurrent (traitement prolongé de mémoire à long terme), il faut des circuits profonds comportant une abondance de multiplications et ne se prêtant donc pas à un chiffrement homomorphe à niveau.

Par ailleurs, des méthodes plus simples de traitement en sac de mots ou en réseau neuronal peu profond, par exemple, apportent des solutions moins chères aux mêmes problèmes de classification de textes au détriment du contexte, de la grammaire et de l'ordre des mots. Dans un traitement en sac de mots, on code un vecteur selon la présence d'éléments lexicaux dans le texte. Bien que simples, de telles techniques ont été utilisées avec profit comme outils générateurs de fonctions dans les domaines de la recherche d'information et de la classification documentaire. Elles se sont avérées un moyen efficace et favorable au chiffrement homomorphe de résoudre nos problèmes de classification supervisée de textes.

3. Cadre méthodologique

Dans cette section, nous décrivons les méthodes employées pour appliquer notre protocole de classification de textes avec la structure du réseau d'ensembles et les complexités tenant à l'intégration d'un chiffrement homomorphe.

3.1 Structure du réseau

À en juger par notre expérience de la classification de données textuelles et, en particulier, des données auxquelles nous nous intéressons ici, un réseau à simple couche et un codage en sac de mots suffisent souvent à assurer une exécution acceptable du modèle. Travailler avec un tel réseau peu profond est avantageux du point de vue de la complexité des calculs, surtout avec un système de chiffrement homomorphe à niveau comme le CKKS.

Pour maximiser le rendement dans un schéma à niveau, il nous faut entraîner un *ensemble*. Dans un modèle ensembliste $\mathcal{M}$, un certain nombre $S > 1$ de sous-modèles $\mathcal{M}_S$ sont entraînés séparément et, au moment de la prédiction, ils votent pour déterminer ce que sera la prédiction de l'ensemble. Nous avons réuni des sous-modèles dans un même cryptogramme qui entraîne efficacement plusieurs sous-modèles en parallèle. Chacun de ceux-ci est un réseau neuronal à simple couche où l'opérateur de composition est la multiplication par une matrice poids W_S. Nous choisissons de ne pas employer de fonction d'activation, car le petit gain de précision ainsi réalisé n'est pas jugé valoir le coût d'en introduire une.

Notre ensemble d'entraînement consiste en N paires $\{x, y\}$, où $x \in \mathbb{R}^M$ et y forment un même vecteur codé « 1 parmi n » en $\mathbb{R}^L$, c'est-à-dire représentant une de L catégories. Chaque modèle $\mathcal{M}_S$ est constitué d'une matrice de poids $L \times M$ désignée par W_s et amorcée aléatoirement avec de petites valeurs réelles. Multipliée par un vecteur de données x, la matrice de poids donne un vecteur de logits $z = \mathcal{M}_s(x) = W_s x \in \mathbb{R}^L$. L'ensemble de sous-modèles procède à un vote pondéré $\mathcal{M}(x) = \sum_s \mathcal{M}_s(x)$, puis choisit l'indice de logit le plus élevé pour dégager la prédiction de l'ensemble.

Nous prenons l'erreur quadratique moyenne comme fonction de perte et, pour mettre le modèle à jour, nous calculons

$$W_s^{(t+1)} = W_s^{(t)} - \lambda \left(W_s^{(t)} x_i - y_i\right) \cdot x_i^T.$$

Nous avons accéléré l'application de notre protocole du gradient à l'aide d'un protocole tiré de Nesterov (2004).

3.2 Protocole du gradient en descente en mode crypté

Nous présentons le pseudocode de notre protocole du gradient en descente en mode crypté pour un seul sous-modèle $\mathcal{M}_s$ dans l'algorithme 1. La procédure d'entraînement de l'ensemble représente une simple extension de ce protocole. Les données sont chiffrées sous la forme $[x]$ où x est un vecteur en $\mathbb{R}^M$ et où les étiquettes correspondantes de sortie sont $y \in \{1, \ldots, L\}$. En temps normal, ces étiquettes seraient codées « 1 parmi n » dans un même vecteur, mais dans le mode crypté, le protocole les met plutôt en une série de cryptogrammes $\{[y]_l\}_{l=1}^L$, où $[y]_l$ crypte un vecteur de M uns si $y = l$ et de zéros sinon. La matrice de poids $L \times M$ appelée W_s est cryptée par lignes en cryptogrammes $\{[W_S]_l\}_{l=1}^L$.

Algorithm 1: Encrypted Training Procedure

input : encrypted training dataset, saved to file
learning rate λ and momentum coefficient γ
output: trained weights, $[\mathbf{W}_S]_l$

1 **for** *each update and* $l < L$ **do**
2 momentum look-forward, $[\mathbf{W}_S]_l = [\mathbf{W}_S]_l - \gamma[\mathbf{v}]_l$;
3 load in data X;
4 **for** *each* $[\boldsymbol{x}]$ *in* X **do**
5 propagate forward, $[\mathbf{z}]_l = \mathbf{total}([\mathbf{W}_S]_l \otimes [\mathbf{x}])$;
6 subtract true labels, $[\mathbf{dz}]_l = [\mathbf{z}]_l - [\mathbf{y}]_l$;
7 compute gradient, $[\mathbf{dW}_S]_l^X = [\mathbf{dz}]_l \otimes [\mathbf{x}]$;
8 accumulate gradient, $[\mathbf{dW}_S]_l \mathrel{+}= [\mathbf{dW}_S]_l^X$;
9 multiply by learning rate, $[\mathbf{dW}_S]_l = \frac{\lambda}{N}[\mathbf{dW}_S]_l$;
10 update weights, $[\mathbf{W}_S]_l = [\mathbf{W}_S]_l - [\mathbf{dW}_S]_l$;
11 update momentum, $[\mathbf{v}]_l = \gamma[\mathbf{v}]_l + [\mathbf{dW}_S]_l$;
12 preparation for next update;

Dans la documentation sur l'apprentissage automatique, le terme « époque » désigne habituellement un passage sur l'ensemble de données, laquelle implique fréquemment plusieurs mises à jour du modèle sur des lots de données. Dans le modèle à cryptogrammes, nous ne sommes pas limités par le nombre d'époques, mais plutôt par le nombre de mises à jour du modèle (voir la ligne 10 dans l'algorithme 1 reproduit plus haut). Ainsi, nous mesurons les progrès de l'entraînement de notre modèle selon les *mises à jour* plutôt que selon les époques.

3.3 Données

Statistique Canada recueille des données en temps réel auprès des grands détaillants sur divers produits d'information. C'est ce qu'on appelle les « données scanographiques » qui comportent un certain nombre d'identificateurs, une description du produit et un prix de transaction. Le nom vient des lecteurs de prix utilisés pour faire passer un client à la caisse. C'est là une très précieuse source de données servant notamment à produire l'Indice des prix à la consommation (Statistique Canada, 2021). L'organisme traite ces données comme sensibles et cherche à en sauvegarder la confidentialité et celle des détaillants qui les fournissent.

La première étape dans le traitement de cette information consiste à classer les descriptions de produits dans un système normalisé de codification à l'échelle internationale, à savoir le Système de classification des produits de l'Amérique du Nord (SCPAN). C'est un système qui sert à classer les différents types de produits à des fins de comparaison. Il y a un code, par exemple, pour le café et ses produits. Chaque entrée de données scanographiques doit recevoir un de ces codes en fonction de la description du produit fournie par le détaillant. Dans notre validation de principe, nous remplaçons ces données par une source de données synthétiques, ce qui nous permet de procéder à des expériences sans craindre de nuire à la sécurité des données. Notre ensemble de données est adapté du FoodData Central de l'USDA (USDA, 2020). Il consiste en 50 000 descriptions de produits se rattachant à cinq codes différents du SCPAN.

Les données ont été codées lexicalement par la technique du sac de mots. En d'autres termes, chaque mot d'un ensemble de données est relevé et versé dans un dictionnaire $\mathcal{D}$ comptant 4 030 mots. Une description d de produit est codée sous forme de vecteur v de dimension 4 030, où $v_i = 1$ si et si seulement l'enième mot appartient à d. Ces vecteurs sont garnis de zéros pour se ranger dans les blocs à 4 096 cases pendant le chiffrement.

4. Expériences

Dans cette section, nous exposons certains détails des expériences que nous avons réalisées. D'abord, nous décrivons en détail l'environnement d'informatique en nuage qui est le nôtre. Ensuite, nous livrons certains détails sur les différentes procédures d'entraînement que nous appliquons. Enfin, nous faisons connaître les temps et les résultats de l'exécution de notre protocole de classification de textes en mode crypté.

4.1 Environnement de calcul

Toutes les expériences ont été réalisées dans un nuage Azure de Microsoft au moyen d'une machine virtuelle à 32 Go de mémoire et à 8 UCT virtuelles. Nous avons également fait du traitement multifilière dans un souci d'utiliser les multiples cœurs du processeur. À noter que le modèle sur textes ne recourt pas au traitement multifilière. À noter aussi que le coût de location d'un ordinateur en nuage de cette puissance pour le temps nécessaire à l'entraînement de notre modèle est suffisamment bas pour qu'une telle solution devienne pratique pour une personne ou un organisme disposant de données sensibles.

Nous employons la bibliothèque libre Microsoft SEAL (Simple Encrypted Arithmetic Library) qui se prête à une implantation naturelle du schéma CKKS de chiffrement homomorphe (SEAL, 2020). Cette bibliothèque en langage C++ présente une interface simple et de niveau faible pour initialiser les clés, crypter les données et exécuter le protocole.

4.2 Entraînement

Notre choix de paramètres de chiffrement nous permet de faire six mises à jour du modèle. Comme nous l'avons décrit, nous codons chaque entrée dans un vecteur de données à 4 096 dimensions, de sorte que quatre sous-modèles puissent être intégrés dans chaque cryptogramme à 16 384 cases. Sur les 50 000 entrées de l'ensemble de données, 40 000 ont servi à l'entraînement et le reste, aux tests.

Nous présentons deux méthodes pour surmonter la contrainte de profondeur de circuit imposée par le chiffrement homomorphe. La première vise à l'acheminement par le nuage des cryptogrammes déjà traités du modèle au titulaire de la clé privée qui peut alors les décrypter et les recrypter avant de les retourner. Il faut pour cela une communication entre le nuage et le titulaire de la clé à plusieurs reprises tout au long du processus d'entraînement. Si la chose est peu commode, la charge demeure raisonnable tout en apportant un gain appréciable de précision du modèle; la quantité de données à échanger est de l'ordre de dizaines de mégaoctets.

Nous pouvons éliminer cette charge de communication en élargissant notre modèle. En fait, si nous ajoutons des ensembles de cryptogrammes poids, nous pouvons entraîner autant de sous-modèles que nous le désirons. Ainsi, plutôt que d'actualiser le même ensemble de poids quatre fois, nous pouvons entraîner quatre ensembles de cryptogrammes du modèle. L'entraînement se fait dans le même délai, mais nous écartons la charge de communication qu'exigent les mises à jour. Nous avons conclu à l'efficacité de cette solution dans la pratique. Nous nous rapprochons en effet de la façon dont les ensembles sont normalement utilisés dans la documentation spécialisée sur le mode non crypté où le nombre de sous-modèles serait bien plus grand que ce que nous envisageons ici.

4.3 Évaluation de la performance

La première étape dans l'exécution de l'algorithme consiste à charger et sérialiser l'ensemble de données pour qu'il puisse être transféré au nuage. Dans les expériences que nous présentons ici, nous avons utilisé $S = 4$ sous-modèles. Ainsi, tout l'ensemble de 50 000 entrées a été intégré dans 12 500 cryptogrammes, chacun de ceux-ci prenant 8 Mo lorsqu'il est sérialisé. L'ensemble d'entraînement, qui est de 14,9 Mo à l'origine, est d'un total approximatif de 78,5 Go une fois crypté. Pour le crypter et le sérialiser, il faut compter 14,6 minutes.

Dans nos expériences, nous avons fait l'essai des deux méthodes pour maximiser le potentiel d'entraînement du modèle chiffré. Nous essayons d'abord un modèle « de grande taille », où un même ensemble de cryptogrammes du modèle, qui est intégré dans quatre sous-modèles, est entraîné et actualisé à plusieurs reprises. Nous prenons ensuite un modèle « de large dimension », où nous employons quatre ensembles de cryptogrammes du modèle, chacun intégré dans quatre sous-modèles; chaque sous-modèle est entraîné sur différents sous-ensembles de données sans jamais devoir être actualisé.

Nous comparons les résultats entre le réseau sur données chiffrées et le réseau sur données textuelles. Ce dernier est structuré exactement comme le premier. Par souci d'équité, le même protocole de réglage des hyperparamètres a été exécuté dans l'un et l'autre de ces réseaux avec une même taille de lot de 1 000. Nous présentons le nombre de mises à jour qu'exige le modèle sur données textuelles pour correspondre à la précision maximale de notre modèle sur données chiffrées. Nous constatons cependant que, lorsque le nombre d'époques est illimité, le premier peut être d'une précision de 87 % en 10 minutes environ d'entraînement, ce qui correspond à environ 80 époques ou à 3 200 mises à jour du modèle. Les résultats sont résumés au tableau 4.3-1.

Tableau 4.3-1
Comparaison des résultats entre les modèles chiffrés et en clair

Réseau	Sous-modèles	Mises à jour du modèle	Actualisations du modèle	Durée d'entraînement	Précision des tests
Données textuelles	1	80	S.O.	15 s	74,3 %
Modèle « de haute taille »	4	18	2	5,03 h	74,2 %
Modèle « de large taille »	16	6 × 4	0	6,97 h	74,4 %

Le signe de multiplication à la colonne « Mises à jour du modèle » est là pour bien faire voir qu'il y a quatre ensembles de cryptogrammes du modèle, dont chacun est mis à jour six fois. Par « actualisations du modèle » nous entendons le nombre d'étapes de décryptage-recryptage qu'exige un passage (voir plus haut).

5. Conclusions

Dans cet article, nous présentons un protocole de classification des textes privés avec chiffrement homomorphe qui assure une exécution raisonnable pour un cas d'utilisation réaliste. Notre but premier était d'étudier la faisabilité du chiffrement homomorphe dans des tâches d'apprentissage automatique à forte intensité de calcul comme l'entraînement d'un réseau neuronal, sans perte de confidentialité de l'ensemble de données d'entrée. Par rapport à nos expériences sur données textuelles, nos expériences sur données chiffrées démontrent que la dégradation de l'exécution par le bruit inhérent et le calcul approximatif propre au chiffrement homomorphe demeure gérable. À notre connaissance, cet exercice est le plus important qui ait porté jusqu'ici sur un problème d'entraînement de réseaux neuronaux pour un problème de classification de textes en mode crypté.

Avec des techniques comme le groupage et le traitement multifilière, nous avons réussi, dans le domaine de la classification de textes supervisée, à entraîner un réseau neuronal d'ensembles sur un grand ensemble de données chiffrées. Notre solution est sûre et pratique dans ce contexte compte tenu de la puissance de calcul modérée et des outils qu'offre aujourd'hui l'informatique en nuage. Il reste un certain nombre de défis à relever si nous voulons améliorer la performance tant pour les résultats de l'évaluation que pour les temps de calcul. Pour citer un exemple, le recours à la puissance des processeurs graphiques permettra de grandement améliorer les temps de calcul. Les méthodes de recherche heuristique peuvent améliorer le réglage des hyperparamètres, autre important problème pour lequel il serait possible de concevoir des modèles plus efficaces. Ajoutons que des paramètres de plus grande sécurité comme des modules polynomiaux majorés pourraient aider à entraîner des réseaux neuronaux plus profonds et à multiplier les époques d'entraînement pour de meilleurs résultats d'évaluation.

Notons enfin que, sur le plan technologique, le chiffrement homomorphe a enfin atteint un point dans sa progression où nous pouvons prendre une bibliothèque de source ouverte et résoudre un vrai problème avec un effort de développement et un temps de calcul qui demeurent raisonnables.

Bibliographie

Cheon, J.H., Kim, A., Kim, M., et Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology–ASIACRYPT 2017*, pages 409–437, Springer.

Raisaro, J.L., Troncoso-Pastoriza, J.R., Misbach, M., Sousa, J.S., Pradervand, S., Missiaglia, E., Michielin, O., Ford, B., et Hubaux, J.P. (2018). Med Co: Enabling Secure and Privacy- Preserving Exploration of Distributed Clinical and Genomic Data. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(4):1328–1341.

Kowsari, K., Meimandi, K.J., Heidarysafa, M., Mendu, S., Barnes, L.E., et Brown, D.E. (2019). Text Classification Algorithms: A Survey. *CoRR*, abs/1904.08067.

Nesterov, Y. (2004). Introductory lectures on convex programming volume: A Basic course. Springer.

Statistique Canada. (2021). Indice des prix à la consommation. https://www.statcan.gc.ca/fra/survey/business/2301.

Département de l'Agriculture des États-Unis (USDA), A.R.S. (2020). FoodData Central: USDA Global Branded Food Products Database. fdc.nal.usda.gov.

SEAL (2020). Microsoft SEAL (release 3.5). https://github.com/Microsoft/SEAL, Microsoft Research, Redmond, État de Washington.