ÉVALUATION DES METRIQUES DE COUPLAGE EN TANT QU'INDICATEURS DE VULNÉRABILITÉS DANS LES APPLICATIONS WEB

Mohammed ZAGANE[(1)], Mustapha Kamel ABDI [(2)]
[(1), (2)] Laboratoire RIIR, Université Oran1 Ahmed BEN BELLA, BP 1524 EL MNaouer 31000 Oran, Algérie.
[(1)]m_zagane@yahoo.fr, [(2)]abdimk@yahoo.fr

Reçu le : 21/01/2019 *Accepté le : 20/03/2019*

Résumé. *La plupart des problèmes de sécurité dans les applications web sont liés à l'exploitation des vulnérabilités du code source. La détection manuelle de ces vulnérabilités est très difficile et très coûteuse en termes de temps et de budget. Par conséquent, l'utilisation d'outils qui aident les développeurs à prédire automatiquement les composants vulnérables est très nécessaire, par ce qu'elle minimise l'espèce de recherche et permet ainsi de minimiser les coûts nécessaires pour la correction des vulnérabilités. Notre contribution consiste à déterminer le type de métriques qui donne une meilleure prédiction des vulnérabilités dans les applications web. Les résultats obtenus dans cette étude nous aideront, à améliorer la qualité des données d'apprentissage dans les travaux futurs en choisissant les métriques de code appropriés à utiliser comme indicateurs des vulnérabilités. En fait, contrairement aux autres types d'applications où les métriques de complexité sont les meilleurs indicateurs de vulnérabilité, nous avons constaté que dans les applications web, les métriques permettant de mieux prédire les vulnérabilités sont les métriques de couplage.*

MOTS CLES : *Prédiction des Vulnérabilités; Apprentissage Automatique ; Métriques de Code; Applications Web.*

1. Introduction

La prédiction automatique des vulnérabilités logicielles minimise les coûts et les délais liés à la correction de ces vulnérabilités en permettant aux développeurs de concentrer leurs efforts sur les composants les plus susceptibles d'être vulnérables. L'utilité de ce type de prédiction devient très nécessaire lorsqu'on travaille sur des applications web, en raison de leur utilisation massive et de leur disponibilité en ligne, qui facilite l'exploitation de leurs vulnérabilités. Étant donné que la vulnérabilité n'est qu'un type particulier de défaut qui affecte la sécurité des informations dans un logiciel, les techniques utilisées pour prédire les défauts sont aussi utilisées pour la prédiction des vulnérabilités. L'une de ces techniques est l'utilisation des métriques de code comme indicateurs de défauts.

Les métriques de code servent à quantifier certaines caractéristiques du logiciel, telles que la taille, le couplage et la complexité. L'analyse de ces métriques permet aux développeurs de contrôler le processus de développement ainsi que la qualité du logiciel. Plusieurs travaux de recherche [1–4] ont prouvé, par des études empiriques, la corrélation entre les métriques de code et les vulnérabilités et ont conclu que les modèles de prédiction construits en utilisant les métriques de code, peuvent parfaitement prédire les composants vulnérables d'un logiciel.

Dans une étude [5] réalisée sur la sélection des métriques pour définir un ensemble pertinent de métriques qui donne une meilleure prédiction de défauts, les chercheurs ont trouvé que même après avoir supprimé 85% du nombre des métriques étudiées, les modèles de prédiction de défauts n'ont pas été affectés. C'est dans cette piste de recherche que notre étude s'inscrit, notre contribution consiste à déterminer le type de métriques de code qui donne une meilleure prédiction des vulnérabilités

dans les applications web. Cet article est organisé autour des sections suivantes : la section 2 présente les travaux connexes, la section 3 présente les questions de recherches et les hypothèses, la section 4 décrit l'approche adoptée et la méthodologie, la section 5 présente l'expérimentation et discute les résultats obtenus, la section 6 résume le travail réalisé dans le cadre de cette étude et signale quelques perspectives.

2. Travaux connexes

Une vulnérabilité est un type particulier de défaut qui affecte la sécurité d'un système logiciel. L'étude de la prédiction des vulnérabilités ne peut être faite sans aborder la prédiction des défauts. Dans cette section nous commençons par présenter les travaux connexes sur la prédiction des défauts puis nous présentons les travaux sur la prédiction des vulnérabilités.

2.1 Prédiction des défauts

Beaucoup de travaux de recherche ont utilisé les métriques de code pour la prédiction des défauts. Tim Menzies et al. [6] ont utilisé les métriques de code classique : métrique de lignes de code, métriques de McCab [7] et métriques de Halstead [8] pour la prédiction de défauts. L'ensemble de données utilisé pour construire et valider les modèle de prédiction était MDP (Metrics Data Program) de la NASA disponible avec d'autres ensembles de données d'autres projets dans le PCR (Promise Code Repository), les algorithmes de prédiction utilisés ont été OneR, J48 et Naïve Bayes, qui ont donné des résultats de prédiction positive plus de deux tiers (71 %) et moins d'un quarre de prédiction négative (< 25 %). Dans [9], les auteurs ont abordé plusieurs aspects de la prédiction de défauts, ils ont fait deux types d'analyse, le premier consiste à une

prédiction inter-projet où l'ensemble des données MDP a été utilisé pour l'apprentissage d'un prédicteur des k plus proches voisins (KNN), qui a été ensuite validé en utilisant des données collectées de 25 grands projets logiciels d'une entreprise de télécommunication. En plus des métriques de code classiques disponibles dans l'ensemble des données MDP, ils ont étudié aussi les métriques de code basées sur les graphes d'appels : CGBR (Call Graph Based Rankin), FanIn, FanOut et d'autres métriques. Le taux de prédiction positive était 15 % en utilisant les métriques classique et 70 % en utilisant les métriques de graphe d'appel. Dans la deuxième analyse ils ont étudié la prédiction des défauts en utilisant des règles de décisions exploitant des intervalles recommandées pour chaque métrique, les résultats obtenus par cette deuxième analyse ont été 14 % de prédiction positive. Les auteurs dans [5] ont étudié la sélection des métriques pour définir un ensemble pertinent de métriques qui donnent une meilleure prédiction de défauts. Ils ont fait une étude comparative pour évaluer leur approche proposée. Dans cette approche hybride, ils ont commencé par réduire l'espace de recherche en utilisant les techniques de classement d'entité (features ranking), puis ils ont utilisé les techniques de sélection de sous-ensemble d'entité (sub-set feature selection). Plusieurs algorithmes des deux méthodes ont été évalués et comparés. Les auteurs ont trouvé que même après avoir supprimé 85% du nombre de métriques étudiées, les modèles de prédiction de défauts n'ont pas été affectés. Ces résultats montrent l'importance des études qui visent la comparaison et la sélection des métriques qui donnent des meilleures performances de prédiction.

2.2. Prédiction des vulnérabilités

Le succès des approches de prédiction qui utilisent les métriques de code comme indicateurs de défauts a encouragé les chercheurs d'utiliser

ces approches pour la prédiction des vulnérabilités. Les auteurs dans [10] ont fait une étude pour savoir si les MPV (Modèles de Prédiction des Vulnérabilités) sont précis et donnent des recommandations pertinentes lors de l'allocation des ressources de maintenance. Ils ont utilisé plusieurs modèles d'apprentissage statistiques (LR, NB, RF, RP, SVM, TB) appliqués sur deux versions du système d'exploitation Windows (Windows 7, Windows 8). Les auteurs ont conclu que les MPV doivent être raffinés à travers l'utilisation de métriques de code spécifiques à la sécurité. Su Zhang et al. ont aussi rapporté dans [1] qu'il est difficile de construire des bons MPV en utilisant des données limitées sur les vulnérabilités. Dans leur étude ils ont utilisé un ensemble de données extrait de la base de données publique NVD (National Vulnerability Database). Ce manque de données sur les vulnérabilités a motivé les chercheurs à proposer des ensembles de données. J. Walden et al. dans [2] ont proposé un ensemble de données qui comporte des données de 223 vulnérabilités trouvées dans trois grandes applications web open source. Ils ont aussi utilisé cet ensemble de données pour comparer et évaluer des MPV basés sur les métriques de code et le datamining et ils ont invité la communauté des chercheurs à utiliser cet ensemble de données pour évaluer d'autres MPV. Effectivement, M. Alenezi et I. Abunadi [3, 4] ont utilisé l'ensemble de données développé dans [2] pour évaluer plusieurs MPV basés sur les métriques de code et la prédiction inter-projet. Y. Shin et al. [11] ont fait une étude comparative de la puissance de prédiction des vulnérabilités des métriques de complexité, code churn (évolution du code), et les métriques liées à l'activité des développeurs (métriques sociales), ils ont utilisé les données de deux projets open source : navigateur web Mozilla Firefox et le noyau du système d'exploitation Linux. S. Moshtari et al. dans [12] ont traité les métriques de complexité, couplage et un nouveau ensemble de métriques

de couplage. Les deux travaux ont trouvé que les métriques de complexité permettent de prédire mieux les vulnérabilités que les autres types de métriques.

3. Questions de recherche et hypothèses

Les travaux de recherches similaires [11, 12] qui ont visé des applications non web, ont validé l'hypothèse selon la quelle la complexité du code est la cause de la plus part des problèmes de sécurité dans un logiciel, ils ont trouvé que les métriques de complexité sont plus prédictives des vulnérabilités que les autres types de métriques. Du fait que les applications web se différent de plusieurs façons des autres types d'applications : développement, utilisation, etc., nous formulons les questions de recherches et proposons les hypothèses dans les paragraphes suivants.

Question 1 : Est ce que comme les applications non web, les métriques de complexité permettent de prédire mieux les vulnérabilités dans les applications web que les métriques de taille et de couplage?

Question 2 : Sinon, quel est le type de métriques qui permet de prédire mieux les vulnérabilités dans les applications web?

Hypothèse 1 : Comme dans les autres types d'application, les métriques de complexité permettent de prédire mieux les vulnérabilités que les métriques de couplage et de taille dans les applications web.

Hypothèse 2 : Les modules volumineux sont plus susceptibles d'être vulnérable, et en conséquence, les métriques de taille sont les plus puissantes en matière de prédiction des vulnérabilités dans les applications web.

Hypothèse 3 : Un couplage élevé augmente la probabilité de la présence des vulnérabilités, et en conséquence les métriques de couplage sont plus prédictives des vulnérabilités dans les applications web

4. Approche et méthodologie

4.1. Approche

Nous avons adoptée une approche largement utilisée dans les travaux déjà réalisés dans la prédiction des défauts et des vulnérabilités [2–4, 6, 11, 12]. Cette approche consiste à traiter le problème de la prédiction comme un problème de classification binaire supervisée (Fig. 1).

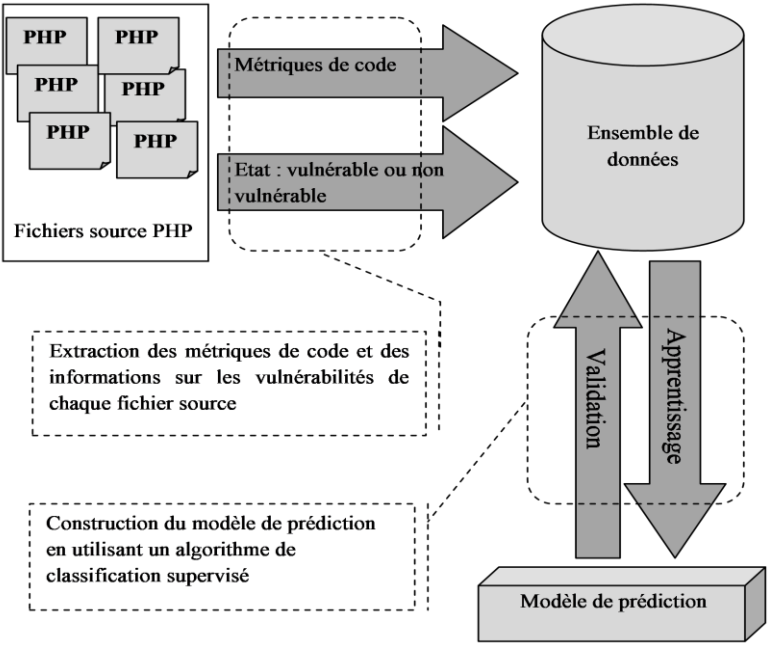

Fig.1 *: Approche adoptée.*

4.2. Méthodologie

Dans cette sous-section, nous présentons la méthodologie suivie pour réaliser notre étude : préparation de l'ensemble de données, construction des modèles de prédiction et évaluation de la puissance de prédiction de chaque modèle.

4.2.1. Préparation de l'ensemble de données

Pour faire l'apprentissage et la validation de nos modèles de prédiction, nous avons utilisé un ensemble de données extrait de l'ensemble de données développé par Walden et al. [2]. La nature de notre étude, nous a obligé de préparer une version qui répond à nos besoins. Cette réorganisation est résumée et présentée dans la figure 2.

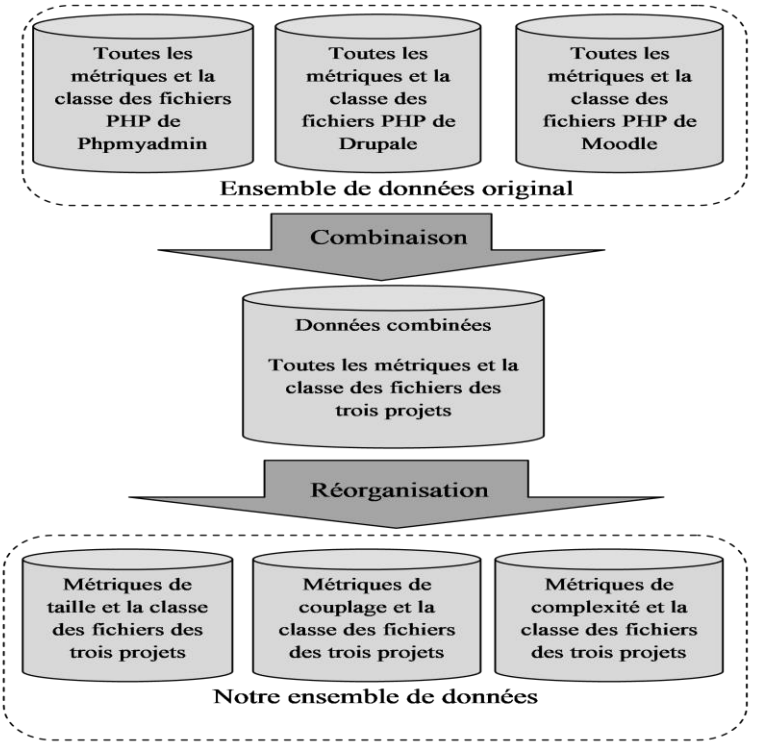

***Fig. 2** : Préparation de l'ensemble de données.*

4.2.3. Construction des modèles de prédiction

Les travaux [2–4] ont évalué les performances de plusieurs modèles en utilisant l'ensemble de données que nous avons utilisé dans notre étude. En se basant sur les résultats de ces travaux et d'autres travaux qui ont fait des études similaires [11, 12], nous avons choisi comme algorithme de classification pour construire nos modèles de prédiction, l'algorithme Random forest.

Nous avons utilisé la même technique de cross-validation qui a été utilisée et bien décrite dans [2] pour faire l'apprentissage et la validation de nos modèles de prédiction.

4.2.4. Evaluation des performances

Le résultat de prédiction d'un modèle peut être l'un des quatre cas suivants : True Positive (si un fichier vulnérable est prédit comme vulnérable par le modèle), True Negative(si un fichier non vulnérable est prédit comme non vulnérable par le modèle), False Positive (si un fichier non vulnérable est prédit comme vulnérable par le modèle), False Negative (si un fichier vulnérable est prédit comme non vulnérable par le modèle).

A partir des TP, TN, FP et FN, on peut calculer plusieurs mesures qui servent à évaluer les performances de prédiction de chaque modèle. Nous avons utilisé les indicateurs de performances suivants pour évaluer nos modèles de prédiction :

- **Recall** : cet indicateur est largement utilisé dans le domaine de la sécurité [2] il donne le pourcentage des fichiers vulnérables qui sont correctement classés par le modèle.

$$\text{Recall} = \frac{TP}{TP+FN} * 100 \qquad (1)$$

- **Inspection :** donne une indication sur le coût en matière du pourcentage des fichiers qu'on doit inspecter pour trouver les TP identifiés par le modèle.

$$\text{Inspection} = \frac{TP+FP}{TP+TN+FP+FN} * 100 \quad (2)$$

- **FP Rate** : Le taux des faux positifs mesure le pourcentage des fausses classifications positives parmi les négatifs réels.

$$\text{FP Rate} = \frac{FP}{FP+TN} * 100 \quad (3)$$

FN Rate : Le taux des faux négatifs mesure le pourcentage des négatifs faussement classifiés parmi les positifs réels.

$$\text{FN Rate} = \frac{FN}{FN+TP} * 100 \quad (4)$$

5. Expérimentations et discussion des résultats

5.1. Expérimentations

Nous avons construit trois modèles de prédiction en utilisant le même algorithme de classification : RF (Random Forest) et des données différentes comme il est montré dans le tableau 1.

Modèle de prédiction	Données d'apprentissage et de validation	Algorithme
Modèle 1	Métriques de taille	RF

Modèle 2	Métriques de complexité	RF
Modèle 3	Métriques de couplage	RF

Tableau 1 : *Modèles de prédiction construits.*

Les expérimentations sont réalisées en utilisant l'outil Weka [13] version 3.8.1 pour la construction et la validation des modèles, l'outil R [14] version 3.4.3 pour calculer les indicateurs de performance.

5.2. Discussion des résultats

La figure 3 montre les résultats obtenus par les trois modèles de prédiction. Les indicateurs de performance : recall (a), inspection (b), FP rate (c) et FN rate (d) nous permettent de comparer les performances de prédiction de chaque modèle.

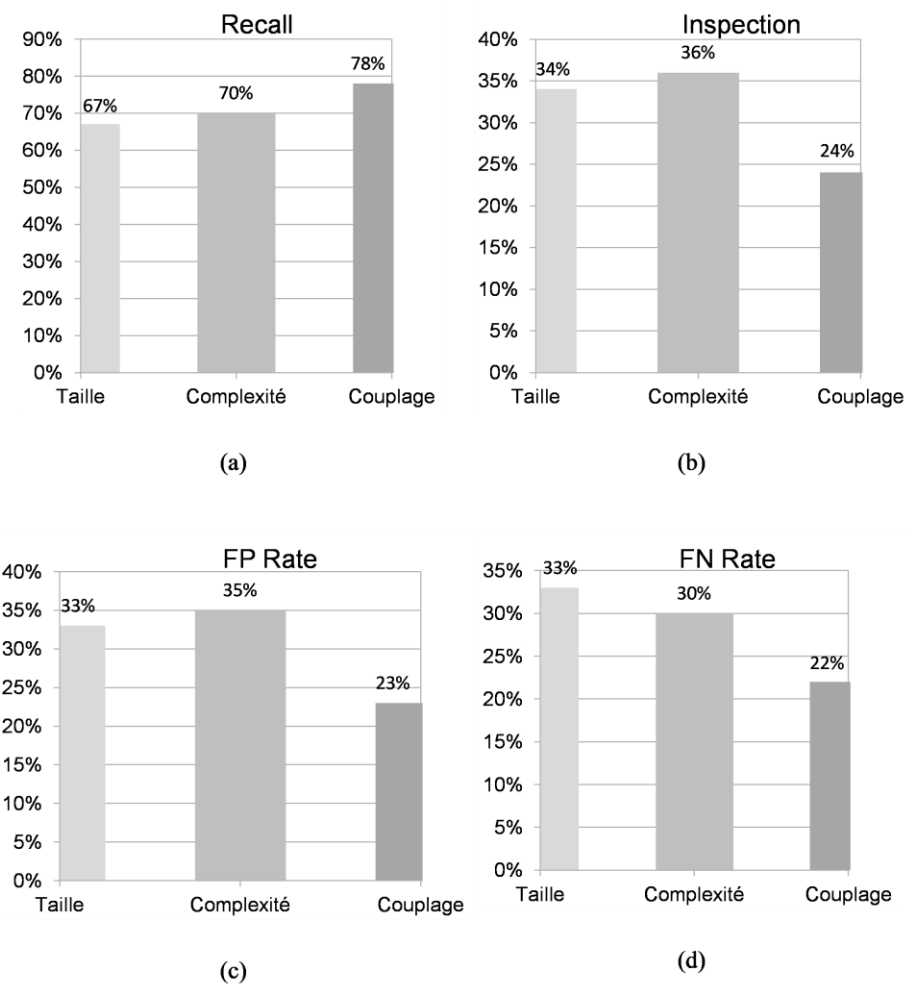

***Fig. 3** : Comparaison de la puissance de prédiction des trois types de métrique.*

Contrairement à ce que nous attendions dans l'hypothèse 1 et l'hypothèse 2, Les meilleures performances de prédiction n'ont pas été données par les métriques de complexité, ni par les métrique de taille. En effet, les métriques de couplage ont donné les bonnes valeurs dans tous les indicateurs de performance : haute recall 78 % contre 70 % pour les métriques de complexité et 67 % pour les métriques de la taille, et moins de coût en matière d'inspection 24 % avec une différence de -12 % par rapport aux métriques de complexité et -10 % par rapport aux métriques de taille. La même chose pour le FP rate et FN rate, les bonnes valeurs de ces indicateurs ont été obtenues par les métriques de couplage. Les valeurs des indicateurs de performance pour les métriques de complexité et les métriques de la taille ont été très proches. Les métriques de complexité ont été meilleures dans les indicateurs recall et FN rate et les métriques de la taille dans les indicateurs Inspection et FP rate.

Les résultats obtenus dans notre étude empirique, nous permettent de valider l'hypothèse 3 et conclure que : les métriques de couplage sont plus puissantes dans la prédiction des vulnérabilités dans les applications web. Nous pouvons interpréter ces résultats comme suit : les applications web se diffèrent de plusieurs façons des autres types d'applications : développement, utilisation, contexte d'exécution etc., c'est pour cette raison qu'un type de métrique ne peut pas donner les mêmes performances de prédiction dans les applications web comme les autres types d'applications, en plus un couplage élevé que ce soit interne ou externe augmente la complexité du code qui est l'ennemie de la sécurité dans un système informatique.

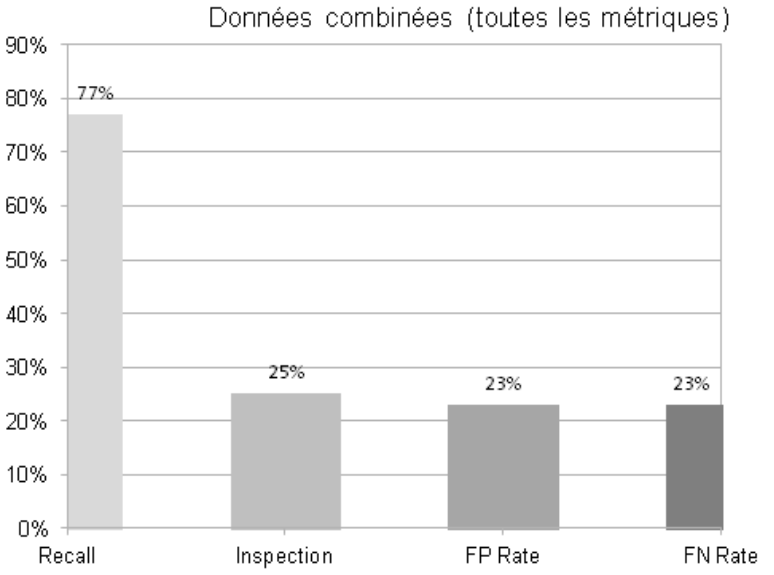

Fig. 4 : *Résultats obtenus en combinant les trois types de métriques*

La figure 4 montre les résultats obtenus en combinant les trois types de métrique, il est claire que ces résultats sont très proches aux résultats obtenus par les métriques de couplage seul. Ceci renforce aussi nos conclusions concernant la puissance de prédiction des vulnérabilités des métriques de couplage.

6. Conclusion

Dans cet article, nous avons comparé la puissance de prédiction des vulnérabilités de trois types de métriques (taille, complexité et couplage) dans les applications web. Nous avons utilisé un ensemble de données construit de trois applications web open source (Moodle, Drupal et

PHPMyAdmin). Nous avons traité le problème de prédiction comme un problème de classification binaire supervisé.

Nos résultats ont montré qu'au contraire des autres types d'applications où les métriques de complexité permettent de prédire mieux les vulnérabilités, les métriques de couplage ont donné des meilleures performances de prédiction que les métriques de complexité et les métriques de taille.

Dans cette étude empirique, nous avons utilisé un ensemble de métriques de code bien connues et largement utilisées. Toutefois, les travaux futurs seront consacrés à d'autres études similaires sur d'autres systèmes, prenant en compte d'autres types de métriques et en utilisant des techniques avancées d'apprentissage profond.

Références

[1] S. Zhang, D. Caragea, and X. Ou, "An empirical study on using the national vulnerability database to predict software vulnerabilities," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6860 LNCS, no. PART 1, pp. 217–231, 2011.

[2] J. Walden, J. Stuckman, and R. Scandariato, "Predicting vulnerable components: Software metrics vs text mining," *Proc. - Int. Symp. Softw. Reliab. Eng. ISSRE*, pp. 23–33, 2014.

[3] M. Alenezi and I. Abunadi, "Evaluating software metrics as predictors of software vulnerabilities," *Int. J. Secur. its Appl.*, vol. 9, no. 10, pp. 231–240, 2015.

[4] I. Abunadi and M. Alenezi, "Towards Cross Project Vulnerability Prediction in Open Source Web Applications," in *Proceedings of the The International Conference on Engineering & MIS 2015 - ICEMIS '15*, 2015, pp. 1–5.

[5] K. Gao, T. M. Khoshgoftaar, H. Wang, and N. Seliya, "Choosing software metrics for defect prediction: an investigation on feature selection techniques," *Softw. Pract. Exp.*, vol. 41, no. 5, pp. 579–606, Apr. 2011.

[6] T. Menzies, J. Greenwald, and A. Frank, "Data Mining Static Code Attributes to Learn Defect Predictors," *IEEE Trans. Softw. Eng.*, vol. 33, no. 1, pp. 2–14, 2007.

[7] H. Watson, T. J. McCabe, and D. R. Wallace, "Structured Testing: A Testing Methodology Using the Cyclomatic Complexity Metric," *NIST Spec. Publ.*, pp. 1–114, 1996.

[8] V. Y. Shen, S. D. Conte, and H. E. Dunsmore, "Software Science Revisited: A Critical Analysis of the Theory and Its Empirical Support," *IEEE Trans. Softw. Eng.*, vol. SE-9, no. 2, pp. 155–165, 1983.

[9] B. Turhan, G. Kocak, and A. Bener, "Data mining source code for locating software bugs: A case study in telecommunication industry," *Expert Syst. Appl.*, vol. 36, no. 6, pp. 9986–9990, 2009.

[10] P. Morrison, K. Herzig, B. Murphy, and L. Williams, "Challenges with applying vulnerability prediction models," in *Proceedings of the 2015 Symposium and Bootcamp on the Science of Security - HotSoS '15*, 2015, vol. 14, no. 2, pp. 1–9.

[11] Y. Shin, A. Meneely, L. Williams, and J. A. Osborne, "Evaluating complexity, code churn, and developer activity metrics as indicators of software vulnerabilities," *IEEE Trans. Softw. Eng.*, vol. 37, no. 6, pp. 772–787, 2011.

[12] S. Moshtari and A. Sami, "Evaluating and comparing complexity, coupling and a new proposed set of coupling metrics in cross-project vulnerability prediction," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing - SAC '16*, 2016, pp. 1415–1421.

[13] G. Holmes, A. Donkin, and I. H. Witten, "WEKA: a machine learning workbench," in *Proceedings of ANZIIS '94 - Australian New Zealnd Intelligent Information Systems Conference*, pp. 357–361.

[14] R. Ihaka and R. Gentleman, “R: A Language for Data Analysis and Graphics,” *J. Comput. Graph. Stat.*, vol. 5, no. 3, p. 299, Sep. 1996.