Ministry of Higher Education and
Scientific Research
University of Mosul
College of Computer Science and
Mathematics
Department of Computer Science

Refactoring for Software Security Based on Artificial Neural Networks

**A Thesis Submitted to the Council of the College of
Computer Science and Mathematics
University of Mosul
as a Partial Fulfillment of Requirements
for the Degree of Doctor of Philosophy in
Computer Science**

By

Hiba Muneer Yahya Othman

Supervised by

Prof.Dr. Dujan Basheer Taha Ali

2024 A.D. **1446 A.H.**

Abstract

In recent years, the increasing focus on software quality has emerged as a fundamental factor for guaranteeing reliable, efficient, and secure technological solutions. In this context, code refactoring plays a vital role by significantly reducing technical debt and enhancing code readability and quality without altering its external behavior.

Refactoring is a process aimed at improving the existing code's design without changing its external functionality. By implementing refactoring techniques, code smells can be effectively addressed, reducing technical debt and improving software quality. This helps maintain the software's development and maintainability, ensuring its sustainability and adaptability to meet new requirements efficiently and effectively.

This thesis presents the design and development of a tool to enhance software security through the use of phases to clean the code programmed in Java, after parsing it, then using refactoring techniques, the performance of the proposed tool was evaluated by conducting many numbers of real experiments for the software systems by considering some important metrics. The obtained results revealed that the suggested security optimizer tool greatly enhanced security in terms of decreasing the number of insecure classes in a software.

Also, three deep learning models (RNN, LSTM, and GRU) are suggested to classify and detect bad code smells. Finally, a new dataset called (SQDS) was created that combine between quality and security metrics. This dataset used to detect God Class bad smell and if the code is secure or insecure by checking the software security that deals with the surface of attack.

In conclusion, the proposed optimizer security tool has improved the security of each insecure class, and has proven to be a viable solution for real-time applications even with the growing volume of classes in a software. It was applied on three different case studies in size and number of insecure classes. This tool was also able to provide a technical report on the assess of the class's security by providing general information about the class and calculating the rate of encapsulation, cohesion and total security of the class. According to the resulting values, there were recommendations given by the tool to further improve security, including encapsulation and cohesion factors.

The three suggested deep learning were able to classify and predicate three types of bad code smells, the result displays that LSTM model was the best model to classify and predicate God Class and Long Method bad smells with value of accuracy 0.96 and 0.91, while The GRU model is the best to classify and predicate Feature Envy with value of accuracy equal 0.95.The proposed SQDS dataset open a new region in software engineering research that identifies specific vulnerabilities associated with different code smells.

وزارة التعليم العالي والبحث العلمي
جامعة الموصل
كلية علوم الحاسوب والرياضيات
قسم علوم الحاسوب

إعادة بناء لأمن البرمجيات إعتماداً على الشبكات العصبية الاصطناعية

أطروحة مقدمة
الى مجلس كلية علوم الحاسوب والرياضيات في جامعة الموصل
كجزء من متطلبات نيل شهادة دكتوراه فلسفة في
علوم الحاسوب

من قبل

هبـــة منـــير يحـــيى عثمـــان

بإشراف
أ.د. دجان بشير طه علي

٢٠٢٤ م ١٤٤٦ هـ

الخلاصة

في السنوات الأخيرة، برز التركيز المتزايد على جودة البرمجيات كعامل أساسي لضمان حلول تكنولوجية موثوقة وفعالة وآمنة. في هذا السياق، تلعب إعادة هيكلة الكود دورًا حيويًا من خلال تقليل الديون الفنية بشكل كبير وتعزيز قابلية قراءة الكود وجودته دون تغيير سلوكه الخارجي.

إعادة الهيكلة هي عملية تهدف إلى تحسين تصميم الكود الحالي دون تغيير وظائفه الخارجية. من خلال تنفيذ تقنيات إعادة الهيكلة، يمكن معالجة روائح الكود بشكل فعال، وتقليل الديون الفنية وتحسين جودة البرنامج. يساعد هذا في الحفاظ على تطوير البرنامج وصيانته، وضمان استدامته وقدرته على التكيف لتلبية المتطلبات الجديدة بكفاءة وفعالية.

تقدم هذه الأطروحة تصميم وتطوير أداة لتعزيز أمان البرنامج من خلال استخدام المراحل لتنظيف الكود المبرمج فيJava ، بعد تحليله، ثم باستخدام تقنيات إعادة الهيكلة، تم تقييم أداء الأداة المقترحة من خلال إجراء العديد من التجارب الحقيقية لأنظمة البرامج من خلال النظر في بعض المقاييس المهمة. كشفت النتائج التي تم الحصول عليها أن أداة تحسين الأمان المقترحة عززت الأمان بشكل كبير من حيث تقليل عدد الفئات غير الآمنة في البرنامج.

كما تم اقتراح ثلاثة نماذج للتعلم العميق (RNN و LSTM و GRU) لتصنيف وكشف روائح الكود السيئة. وأخيرًا، تم إنشاء مجموعة بيانات جديدة تسمى (SQDS) تجمع بين مقاييس الجودة والأمان. تستخدم هذه المجموعة البيانات للكشف عن رائحة God Class السيئة وما إذا كان الكود آمنًا أم غير آمن من خلال التحقق من أمان البرنامج الذي يتعامل مع سطح الهجوم.

قامت أداة تحسين الأمان المقترحة بتحسين أمان كل فئة غير آمنة، وأثبتت أنها حل قابل للتطبيق للتطبيقات في الوقت الفعلي حتى مع الحجم المتزايد للفئات في البرنامج. عندما

تم تطبيقها على ثلاث دراسات حالة مختلفة من حيث الحجم وعدد الفئات غير الآمنة. تمكنت هذه الأداة أيضًا من تقديم تقرير فني عن تقييم أمان الفئة من خلال توفير معلومات عامة عن الفئة وحساب معدل التغليف والتماسك والأمان الكلي للفئة. ووفقًا للقيم الناتجة، كانت هناك توصيات قدمتها الأداة لمزيد من تحسين الأمان، بما في ذلك عوامل التغليف والتماسك.

تمكنت تقنيات التعلم الآلي العميقة الثلاثة المقترحة من تصنيف والتنبؤ بثلاثة أنواع من روائح الكود السيئة، وتُظهر النتيجة أن نموذج LSTM كان أفضل نموذج لتصنيف والتنبؤ بروائح God Class و Long Method السيئة بدقة ٠.٩٦ و٠.٩١، بينما نموذج GRU هو الأفضل للتصنيف والتنبؤ بـ Feature Envy بدقة تساوي ٠.٩٥. مجموعة بيانات SQDS المقترحة فتحت منطقة جديدة في أبحاث هندسة البرمجيات تحدد نقاط ضعف محددة مرتبطة بروائح الكود المختلفة.