HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Thị Quỳnh Anh

XÂY DỰNG QUY TRÌNH SINH MÃ KIỂM THỬ CHỨC NĂNG TỰ ĐỘNG SỬ DỤNG LLM VÀ ÁP DỤNG CHO HỆ THỐNG VNEDU

Chuyên ngành: Hệ thống thông tin
Mã số: 8.48.01.04 (Hệ thống thông tin)

TÓM TẮT ĐỀ ÁN TỐT NGHIỆP THẠC SĨ

HÀ NỘI - NĂM 2025

Đề án tốt nghiệp được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học: Tiến sĩ Đỗ Thị Bích Ngọc
(Ghi rõ học hàm, học vị)

Phản biện 1: PGS.TS Nguyễn Quang Uy

Phản biện 2: PGS.TS Đỗ Năng Toàn

Đề án tốt nghiệp sẽ được bảo vệ trước Hội đồng chấm đề án tốt nghiệp thạc sĩ tại Học viện Công nghệ Bưu chính Viễn thông
Vào lúc: 10 giờ 30 ngày 24 tháng 01 năm 2026

Có thể tìm hiểu đề án tốt nghiệp tại:

– Thư viện của Học viện Công nghệ Bưu chính Viễn thông.

MỞ ĐẦU

1. Tính cấp thiết của đề án

Các phương pháp kiểm thử tự động thông thường vẫn phụ thuộc nhiều vào việc kiểm thử viên (tester) tự viết mã kiểm thử và dần bộc lộ hạn chế khi quy mô hệ thống lớn và thay đổi liên tục. Việc chỉ áp dụng mô hình LLM sinh mã kiểm thử chức năng tự động mà không có một quy trình phù hợp để quản lý đầu vào và kiểm soát đầu ra có thể dẫn đến nhiều rủi ro.

2. Tổng quan vấn đề nghiên cứu

Ngày càng nhiều các dự án phần mềm yêu cầu kiểm thử tự động song song với kiểm thử thủ công. Các công cụ kiểm thử tự động yêu cầu kiểm thử viên tự lập trình và viết các kịch bản kiểm thử dưới dạng mã lệnh, từ việc xác định phần tử giao diện đến xây dựng các bước thao tác và kiểm tra kết quả đòi hỏi kiểm thử viên phải có kỹ năng lập trình và tốn nhiều thời gian viết khi hệ thống thay đổi. Để đảm bảo quy trình sinh mã kiểm thử chức năng tự động bằng mô hình (LLM) đạt được hiệu quả và tính ứng dụng thực tế, cần xem xét các yếu tố sau đây:

- Thu thập các loại dữ liệu đầu vào

- Phân loại đầu vào
- Xây dựng mẫu prompt
- Tiền xử lý và liên kết dữ liệu
- Sinh mã kiểm thử chức năng tự động
- Phân tích và chỉnh sửa mã kiểm thử chức năng
- Lưu trữ và quản lý mã

3. Mục tiêu nghiên cứu

Mục tiêu của đề án này là xây dựng một quy trình sinh mã kiểm thử chức năng tự động sử dụng mô hình LLM kết hợp kỹ thuật Prompt Engineering, nhằm chuyển đổi mô tả các bước kiểm thử thủ công thành mã kiểm thử chức năng tự động

4. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu:

Quy trình sinh mã kiểm thử chức năng tự động sử dụng LLM, trong đó tập trung vào việc khai thác khả năng sinh mã của các mô hình LLM và kỹ thuật Prompt Engineering để hỗ trợ kiểm thử chức năng tự động cho giao diện web

- Phạm vi nghiên cứu: Tập hợp các bước kiểm thử chức năng được mô tả bằng ngôn ngữ tự nhiên

NỘI DUNG

CHƯƠNG I: KIỂM THỬ CHỨC NĂNG TỰ ĐỘNG VÀ HIỆN TRẠNG CỦA HỆ THỐNG VNEDU

1.1. Kiểm thử chức năng tự động

Kiểm thử chức năng (Functional Testing) là phương pháp kiểm thử phần mềm nhằm đảm bảo các tính năng của hệ thống hoạt động đúng như mô tả trong tài liệu yêu cầu hoặc đặc tả nghiệp vụ.

Kiểm thử tự động (Automated Testing) là phương pháp kiểm thử phần mềm trong đó kiểm thử viên đọc các trường hợp kiểm thử (test case) sau đó dựa trên test case và giao diện của chức năng để viết các kịch bản kiểm thử tự động (script) dưới dạng mã, các mã này được thực thi tự động thông qua các công cụ kiểm thử tự động thay vì thực hiện thủ công bởi con người.

Test case là một trường hợp cần kiểm thử, nó bao gồm các bước kiểm thử là các thao tác với giao diện, tiền điều kiện, các giá trị đầu vào, và kết quả mong đợi.

DOM (Document Object Model) là mô hình biểu

diễn cấu trúc của một tài liệu HTML hoặc XML dưới dạng cây phân cấp (tree structure)

Xpath là một cú pháp được sử dụng để tìm vị trí của bất kỳ phần tử nào trên trang web trong cấu trúc DOM

Kịch bản kiểm thử (test script) là một tập hợp các mã lệnh được kiểm thử viên viết bằng công cụ kiểm thử tự động để tự động hóa quá trình thực thi một trường hợp kiểm thử (test case). Trong test script các bước thực hiện trong test case sẽ được ánh xạ thành các keyword, dữ liệu đầu vào của test case được truyền vào keyword dưới dạng tham số, các keyword kết hợp với các XPath để xác định phần tử giao diện cần tương tác trên web

Keyword-driven testing: là phương pháp được sử dụng phổ biến trong kiểm thử giao diện web mục đích tạo các test script từ tập từ khóa (Keyword)

1.2. Giới thiệu về hệ thống VNEDU

VNEDU là hệ thống quản lý giáo dục số được xây dựng và phát triển bởi Trung tâm giải pháp Giáo dục số – VNPT IT phục vụ hơn 20 triệu người dùng, trên 34 tỉnh thành với hơn 3 triệu hồ sơ học sinh, 9000 trường học, bao gồm hơn 30 phân hệ nghiệp vụ

1.3. Hiện trạng hoạt động kiểm thử tự động của hệ thống VNEDU

Hệ thống VNEDU hiện đang vận hành trên 200 test case nghiệp vụ chính trong đó 150 test case được tự động hóa tương ứng với 150 kịch bản kiểm thử tự động. Các chức năng của hệ thống xoay quanh một số dạng chức năng như sau:

- Nhóm chức năng năng CRUD (Create - Read - Update - Delete)
- Nhóm chức năng tra cứu
- Nhóm chức năng thống kê báo cáo
- Nhóm chức năng cấu hình hệ thống
- Nhóm chức năng phân quyền và quản lý tài khoản
- Nhóm tính năng nhập và xuất dữ liệu

Một số vấn đề trong hoạt động kiểm thử tự động cần xem xét trước khi áp dụng các công nghệ mới để sinh mã kiểm thử chức năng như sau:

- Vấn đề 1: Đầu vào của hoạt động kiểm thử tự động chưa được thu thập và lưu trữ tập trung
- Vấn đề 2: Test case có cấu trúc tách biệt nhưng test script lại gộp toàn bộ thông tin vào một dạng duy

nhất

- Vấn đề 3: Sự đa dạng và phức tạp của Xpath, Keyword, Test case
- Vấn đề 4: Việc tạo ra test script (cấu trúc dự án, XPath, keyword, test case) phụ thuộc hoàn toàn vào kinh nghiệm cá nhân của từng kiểm thử viên
- Vấn đề 5: Test case bao gồm các thông tin về phần tử giao diện, keyword, dữ liệu kiểm tra không được đánh dấu hay tách biệt dưới bất kỳ hình thức nào
- Vấn đề 6: Thao tác lặp lại giữa các testcase không được tách thành keyword dùng chung
- Vấn đề 7: Cơ chế lưu trữ mã kiểm thử chức năng theo cấu trúc
- Vấn đề 8: Kết quả kiểm thử tự động sau mỗi lần chạy chưa được hệ thống phân tích hay phân loại một cách tự động

CHƯƠNG 2: LÝ THUYẾT LLM VÀ SỬ DỤNG CÔNG CỤ LLM TRONG KIỂM THỬ CHỨC NĂNG TỰ ĐỘNG

2.1. Giới thiệu về LLM và kỹ thuật Prompt Engineering

2.1.1. Large Language Models (LLMs):

Large Language Models (LLM) là các mô hình học sâu quy mô lớn được huấn luyện bằng phương pháp tự giám sát trên lượng dữ liệu văn bản rất lớn, từ đó học được các mẫu ngôn ngữ, cấu trúc cú pháp, ngữ nghĩa và khả năng suy luận

2.1.2. Prompt Engineering:

Prompt Engineering là kỹ thuật thiết kế tương tác đầu vào nhằm đạt được đầu ra chính xác và ổn định từ LLM. Một số nguyên tắc thiết kế prompt:

- Mô tả nhiệm vụ rõ ràng
- Hướng dẫn mô hình tư duy
- Cung cấp thông tin tham chiếu
- Quan tâm đến định dạng đầu ra

2.2. Sử dụng công cụ LLM trong sinh mã kiểm thử chức năng tự động

Trong thực tiễn triển khai, các công cụ dựa trên mô hình LLM thường được phân chia theo cách thức vận hành của mô hình, bao gồm: LLM chạy trên cloud và LLM chạy cục bộ. Nhóm công cụ LLM chạy trên cloud đáp ứng đặc trưng phù hợp cho bài toán sinh mã kiểm thử chức năng tự động. Một số công cụ LLM chạy trên cloud điển hình: Cursor, Chat GPT, Github Copilot. Dựa so sánh các công cụ, GitHub Copilot là công cụ phù hợp nhất để áp dụng vào bài toán sinh mã kiểm thử chức năng tự động.

2.3. Công nghệ và công cụ sử dụng

Học viên lựa chọn Github Copilot Chat chế độ Agent sử dụng mô hình LLM mặc định GPT4-o đóng vai trò là công cụ LLM sinh mã kiểm thử chức năng, Visual Studio Code (VS Code) làm môi trường phát triển để tổ chức cấu trúc dự án kiểm thử, Robot Framework làm khung kiểm thử (testing framework) để thực thi các kịch bản sinh ra

CHƯƠNG III: XÂY DỰNG QUY TRÌNH SINH MÃ KIỂM THỬ CHỨC NĂNG TỰ ĐỘNG SỬ DỤNG LLM

3.1. Tổng quan các thành phần cần thiết để thực hiện sinh mã chức năng tự động

Các thành phần như thu thập đầu vào, phân loại đầu vào, xây dựng prompt mẫu, tiền xử lý và liên kết dữ liệu, phân tích chỉnh sửa kết quả sinh mã được thực hiện bởi kiểm thử viên, nhằm đảm bảo dữ liệu đầu vào rõ ràng, ngữ cảnh đầy đủ cho mô hình và đảm bảo chất lượng mã do LLM sinh ra. Hai thành phần sinh mã kiểm thử và lưu trữ quản lý mã sinh ra được tạo ra bởi mô hình LLM

3.2. Thu thập đầu vào dựa trên đánh giá thực trạng

- Thu thập HTML của từng màn hình giao diện
- Thu thập test case, kiểm thử viên thu thập test case và gộp các phần trong test case (tiền điều kiện, các bước kiểm thử, dữ liệu kiểm tra và kết quả mong đợi) thành các tệp dạng txt thư mục STEP_TESTCASE chứa bước theo đúng trình tự

nghiệp vụ. Các bước thao tác lặp lại được gom nhóm thành các keyword dùng để tái sử dụng trong nhiều test case, mỗi keyword đại diện cho một cụm hành động được lưu trữ dưới dạng các file txt tại thư mục STEP_KEYWORD.

3.3. Phân loại đầu vào

3.3.1. Nguyên tắc phân loại đầu vào

- Phân loại dựa trên đặc trưng cấu trúc: XPath tĩnh có cấu trúc cố định và không chứa tham số, XPath động có cấu trúc biến đổi theo tham số hoặc ngữ cảnh
- Phân loại dựa trên cách thức hình thành: Các keyword thao tác giao diện (như click, nhập liệu, chọn giá trị) được hình thành từ các hành động mang tính thao tác; trong khi đó các keyword kiểm tra giao diện lại hình thành từ các bước xác nhận trạng thái và mang tính kiểm tra.
- Phân loại dựa trên phạm vi và mức độ phức tạp: Các test case một tính năng thường có luồng nghiệp vụ nhỏ, số lượng bước không lớn, test case nhiều tính năng bao gồm nhiều nghiệp vụ, liên kết nhiều màn hình.

3.3.2. Phân loại đầu vào

- Đối với nhóm XPath, dữ liệu được chia thành hai nhóm: Xpath tĩnh, Xpath động
- Nhóm Keyword được chia thành hai loại: Keyword tương tác giao diện, Keyword kiểm tra giao diện
- Nhóm test case, dữ liệu được phân chia thành hai dạng: Nhóm test case chứa một tính năng, Nhóm test case chứa nhiều tính năng

3.4. Xây dựng mẫu prompt

- Prompt theo tầng:
    - Prompt Project
    - Prompt Xpath
    - Prompt Keyword
    - Prompt Testcase
- Prompt theo nhóm:
    - Prompt cho XPath tĩnh, prompt cho XPath động
    - Prompt dành cho keyword tương tác giao diện, Prompt dành cho keyword kiểm tra giao diện
    - Prompt cho test case chứa một tính năng,

Prompt cho test case chứa nhiều tính năng

3.5. Tiền xử lý và liên kết dữ liệu hỗ trợ

- Tiền xử lý Keyword:
    - Mỗi keyword được viết theo một cấu trúc bán chuẩn
    - Mỗi keyword bán chuẩn chỉ biểu diễn một hành động duy nhất
    - Các mô tả tương ứng với phần tử giao diện được đánh dấu trong dấu [], các mô tả tương ứng với các dữ liệu thay đổi được đánh dấu trong ()
- Tiền xử lý Test case: Để giúp mô hình nhận biết được đâu là keyword, các cụm hành động trong testcase được đặt trong dấu nháy đơn ‘ ’

3.6. Sinh mã kiểm thử chức năng tự động

- Tầng Project: sinh ra cấu trúc dự án kiểm thử tự động
- Tầng XPath sinh ra Xpath tĩnh và Xpath động
- Tầng Keyword sinh keyword thao tác giao diện và keyword kiểm tra giao diện
- Tầng Prompt Testcase sinh test case 1 chức năng và

test case nhiều chức năng

3.7. Phân tích, chỉnh sửa mã kiểm thử chức năng tự động

Người kiểm thử rà soát lại mã được tạo tự động bằng cách chạy thử test script để đảm bảo sự phù hợp với giao diện thực tế và luồng nghiệp vụ. Kiểm thử viên cung cấp cho mô hình cả dòng lỗi thực tế, nguyên nhân lỗi, và phiên bản đúng của đoạn mã cần thay thế nhằm hỗ trợ mô hình có thể cập nhật phần mã bị sai trong các tệp, đồng thời học được quy tắc sửa lỗi đúng từ cặp dữ liệu "lỗi sai – mã đúng"

3.8. Lưu trữ và quản lý mã

Các mã kiểm thử chức năng do mô hình LLM sinh ra không được ghi trực tiếp vào dự án, mà trước tiên được lưu dưới dạng thay đổi tạm thời (temporary changes). Kiểm thử viên sẽ xem lại nội dung, xác nhận sự chính xác của từng phần rồi mới chấp nhận lưu vào đúng thư mục dựa theo cấu trúc dự án sinh ra trong tầng Prompt Project (Repository, Resource, Testcases) và yêu cầu trong từng prompt theo loại ở các tầng

3.9. Quy trình sinh mã kiểm thử chức năng tự động

Quy trình sinh mã kiểm thử chức năng tự động sử dụng LLM được triển khai theo bảy thành phần liên tiếp từ thu thập đầu vào, phân loại đầu vào, xây dựng prompt, sinh mã, tiền xử lý và liên kết dữ liệu hỗ trợ, phân tích và chỉnh sửa mã đến quản lý và lưu trữ kết quả.

CHƯƠNG IV: THỰC NGHIỆM VÀ ĐÁNH GIÁ TRÊN HỆ THỐNG VNEDU

4.1. Thực nghiệm quy trình trên hệ thống VNEDU:

4.1.1. Thu thập đầu vào:

Gồm 2 loại đầu vào là các bước kiểm thử thủ công (Test case) và các file HTML chứa cấu trúc DOM giao diện của hệ thống VNEDU

Test case được lựa chọn dựa trên một số nguyên tắc:

- Nguyên tắc 1: Bao phủ các nhóm tính năng của hệ thống
- Nguyên tắc 2: Bao phủ các nhóm keyword
- Nguyên tắc 3: Bao phủ các nhóm xpath
- Nguyên tắc 4: Bao phủ cả test case một tính năng và test case nhiều tính năng
- Nguyên tắc 5: Lựa chọn test case cần ưu tiên những phân hệ chức năng có tần suất sử dụng cao

Áp dụng quy trình sinh mã kiểm thử chức năng tự động cho toàn bộ 150 test case trong đó lựa chọn 75 test case đại diện từ năm nhóm (CRUD, tra cứu, thống kê báo cáo, cấu hình hệ thống và phân quyền người dùng) để mô

tả, phân tích và đánh giá chi tiết trong đề án. Trong đó chia ra 36 test case một tính năng và 39 test case nhiều tính năng.

4.1.2. Phân loại đầu vào:

Kiểm thử viên tiến hành phân tích thông tin về XPath (bao gồm XPath tĩnh, XPath động), thông tin Keyword (bao gồm keyword thao tác giao diện và keyword kiểm tra giao diện), các test case thành các keyword dùng chung

4.1.3. Xây dựng prompt mẫu:

Xây dựng bộ prompt mẫu gồm 7 mẫu (Prompt Project, Prompt Xpath tĩnh, Prompt Xpath động, Prompt Keyword thao tác với giao diện, Prompt Keyword kiểm tra giao diện, Prompt test case một tính năng, Prompt test case nhiều tính năng) nhằm hỗ trợ mô hình LLM xử lý tuần tự các loại dữ liệu đầu vào và sinh ra mã kiểm thử chức năng theo từng loại dữ liệu

4.1.4. Tiền xử lý và liên kết dữ liệu hỗ trợ:

Tiền xử lý test case dựa trên các quy tắc đã phân tích trong chương 3.4

4.1.5. Sinh mã kiểm thử chức năng tự động:

Thực nghiệm triển khai sinh cấu trúc dự án và 6 loại dữ liệu, sử dụng các prompt và các tệp ngữ cảnh làm đầu vào cho mô hình LLM sinh mã kiểm thử chức năng tự động cho kết quả tương ứng với 6 loại dữ liệu (Xpath tĩnh, Xpath động, Keyword thao tác giao diện, Keyword kiểm tra giao diện, Test case 1 tính năng, Test case nhiều tính năng)

4.1.6. Phân tích và chỉnh sửa mã:

Người kiểm thử rà soát lại mã được tạo tự động bằng cách chạy thử test script để đảm bảo sự phù hợp với giao diện thực tế và luồng nghiệp vụ. Kiểm thử viên cung cấp cho mô hình cả dòng lỗi thực tế, nguyên nhân lỗi, và phiên bản đúng của đoạn mã cần thay thế theo quy tắc sửa lỗi đúng từ cặp dữ liệu "lỗi sai – mã đúng"

4.1.7. Lưu trữ và quản lý mã:

Sau mô hình LLM sinh mã kiểm thử chức năng, các đoạn mã không lưu trực tiếp vào các tệp mà lưu dưới dạng chờ xác nhận, kiểm thử viên sẽ vào phân tích và chỉnh sửa mã sau đó mới lưu lại các mã

4.2. Phân tích kết quả thực nghiệm:

Kết quả thực nghiệm được đánh giá dựa trên năm

tiêu chí chính: Accuracy, Test Coverage, Generation Time, Execution Time và Pass Rate

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Chương này sẽ trình bày kết quả thu được sau quá trình sử dụng và hướng phát triển quy trình sinh mã kiểm thử tự động sử dụng LLM trong tương lai

Kết quả khi áp dụng quy trình sinh mã kiểm thử chức năng tự động sử dụng LLM so với kiểm thử tự động thông thường: kiểm thử viên không phải tự viết mã kiểm thử, mô hình LLM dựa trên prompt mẫu, các bước kiểm thử thủ công và các tệp HTML chứa cấu trúc giao diện tự động sinh mã kiểm thử chức năng bao gồm 6 loại dữ liệu (Xpath tĩnh, XPath động, Keyword thao tác giao diện, Keyword kiểm tra giao diện, Test case một tính năng, Test case nhiều tính năng) và cấu trúc dự án kiểm thử tự động.

Kết quả khi áp dụng quy trình sinh mã kiểm thử tự động sử dụng LLM trong mô hình phát triển phần mềm Agile/Scrum: Khi đưa quy trình vào bối cảnh phát triển phần mềm theo mô hình Agile/Scrum, kiểm thử viên có thể sử dụng mô hình LLM sinh mã kiểm thử chức năng tự động ngay khi có mô tả các bước kiểm thử và giao diện HTML mới, rút ngắn thời gian xây dựng kịch bản kiểm thử tự động trong mỗi Sprint

Hướng phát triển tương lai:

- Tự động kiểm tra lại (validate) mã kiểm thử sau khi sinh: Một hướng phát triển thực tế của quy trình là bổ sung prompt tự động phân tích và đề xuất sửa lỗi cho mã kiểm thử chức năng mà mô hình LLM đã sinh ra. Trong tương lai, có thể yêu cầu mô hình LLM phân tích lại đoạn mã kiểm thử chức năng đã sinh ra theo các tiêu chí cụ thể như tính hợp lệ của cú pháp, sự tồn tại của các biến, cấu trúc tham số của keyword hoặc những lỗi thường gặp trong test case.
- Tự động sinh Test Data: Mở rộng khả năng sinh mã kiểm thử chức năng sang phạm vi tự động sinh dữ liệu kiểm tra (test data). Hiện tại quy trình chỉ sinh biến khai báo dữ liệu kiểm tra (test data). Trong tương lai, mô hình có thể được sử dụng để tự động đề xuất giá trị phù hợp cho từng trường dữ liệu dựa trên quy luật dữ liệu phổ biến hoặc label của phần tử.