TĂNG CƯỜNG TRỌNG SỐ BM25 KẾT HỢP MÔ HÌNH NGỮ CẢNH CHO VIỆC DÒ TÌM BÁO CÁO LỖI TRÙNG NHAU

IMPROVING BM25 WEIGHTING COMBINED WITH CONTEXTUAL MODEL FOR DUPLICATE BUG REPORT DETECTION

Nhan Minh Phúc[1*], Nguyễn Thừa Phát Tài[1], Nguyễn Hoàng Duy Thiện[1]

[1]*Trường Đại học Trà Vinh*

[*]Tác giả liên hệ: nhanminhphuc@tvu.edu.vn

(Nhận bài: 07/9/2020; Chấp nhận đăng: 28/6/2021)

Tóm tắt - Những báo cáo lỗi được những người sử dụng gửi thường được lưu trữ và quản lý bởi những hệ thống quản lý lỗi của những dự án phần mềm nguồn mở như Open Office, Mozilla Firefox, Eclipse... Những lập trình viên sẽ dựa vào những báo cáo lỗi này để xử lý lỗi. Tuy nhiên, có quá nhiều báo cáo lỗi gửi đến hệ thống, khi đó sẽ có những báo cáo lỗi trùng nhau. Do đó, việc phải xác định báo cáo lỗi vừa được gửi đến có bị trùng hay không sẽ mất nhiều thời gian và công sức của người được phân công xử lý lỗi. Trong bài báo này, nhóm tác giả giới thiệu một phương pháp mới tự động dò tìm những báo cáo lỗi trùng nhau bằng cách sử dụng mô hình LDA-NWF (Latent Dirichlet Allocation-New Weight Feature). Mô hình này là sự kết hợp giữa mô hình LDA với đặc điểm trọng số mới. Kết quả thực nghiệm trên ba hệ thống Open Office, Eclipse và Mozilla cho thấy, phương pháp được giới thiệu đạt tỉ lệ chính xác cao hơn các phương pháp trước đó từ khoảng 4-9%.

Từ khóa - Báo cáo lỗi; LDA; trọng số BM25; báo cáo lỗi trùng nhau; hệ thống báo cáo lỗi

Abstract - Bug reports submitted by users are usually stored and managed by issue management systems in open source software projects such as Open Office, Mozilla Firefox, Eclipse... The developers will rely on these bug reports to process bugs. However, there are too many bug reports sent to the system, which leads to the duplication of bug reports. Therefore, it will take time and effort of the person assigned to handle the bug for determining if the bug has been duplicated or not. In this paper, we introduce a new approach of detecting duplicate bug reports automatically using the Latent Dirichlet Allocation-New Weight Feature (LDA) model. This model is a combination of the LDA model with the new weighting feature. Experimental results on the three systems of Open Office, Eclipse and Mozilla show that, the introduced method achieves a higher accuracy rate than previous methods at about 4-9%.

Key words - Bug report; Latent Dirichlet Allocation (LDA); BM25 weighting; duplicate bug report; bug report system

1. Đặt vấn đề

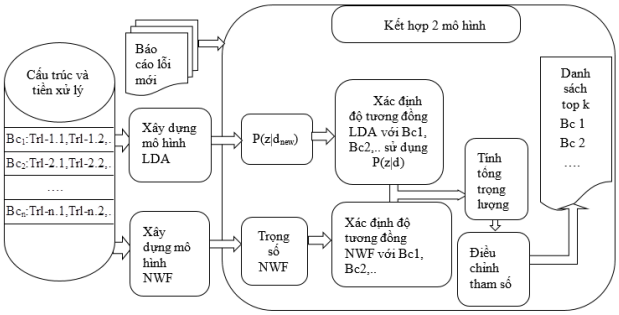
Những dự án mã nguồn mở lớn như Bugzilla, Open Office thường có phần mềm để lưu trữ và quản lý các lỗi do người dùng sử dụng gặp phải để xử lý. Những lỗi này được gửi bởi những người dùng trong quá trình họ sử dụng phần mềm giúp việc bảo trì và cải thiện tính năng của hệ thống tốt hơn [1]. Theo các nghiên cứu gần đây, với việc phát triển nhanh chóng của những hệ thống phần mềm, mỗi ngày có hàng trăm báo cáo lỗi được gửi đến. Khi đó sẽ xảy ra tình trạng báo cáo lỗi bị trùng, lý do là lỗi này đã được người dùng trước đó gửi đến hệ thống. Hay nói cách khác báo cáo lỗi bị trùng là do có nhiều hơn một người dùng gửi cùng một báo cáo lỗi giống nhau [2]. Những báo cáo lỗi thường được mô tả dùng ngôn ngữ tự nhiên do đó cùng một lỗi giống nhau có thể được được diễn tả bằng những từ ngữ khác nhau hay nhiều cách khác nhau. Bảng 1, Bảng 2 minh họa về hai báo cáo lỗi trùng nhau của hệ thống quản lý lỗi Open Office. Chúng ta dễ nhận thấy hai báo cáo lỗi này cùng báo cáo một lỗi tuy nhiên lại sử dụng bằng những từ ngữ khác nhau. Với số lượng báo cáo lỗi ngày càng tăng, việc dò tìm những báo cáo lỗi trùng nhau bằng thủ công là một việc gây lãng phí nhiều thời gian, tốn kém nhiều công sức con người. Vì vậy, trong những năm gần đây, nhiều phương pháp về việc tự động dò tìm báo cáo lỗi trùng lắp đã được nghiên cứu để giải quyết vấn đề này. Hiện tại có vài phương pháp được giới thiệu. Phương pháp sử dụng phổ biến trước đây là sử dụng kỹ thuật lấy thông tin (IR) sử dụng mô hình vector (Vector Space Model) [3, 4]. Một phương pháp khác cải tiến hơn là sử dụng kỹ thuật lấy thông tin kết hợp với phương pháp xử lý ngôn ngữ tự nhiên [5, 6]. Ngoài ra, còn một số phương pháp khác như sử dụng mô hình học máy [7], mô hình phân loại nhị phân [8]. Tuy nhiên, giới hạn của những phương pháp này chính là tỷ lệ chính xác của kết quả thực nghiệm vẫn còn thấp. Gần đây, một phương pháp cải tiến của kỹ thuật rút trích thông tin được nhóm tác giả Sun và cộng sự [9] giới thiệu cho thấy, có sự cải tiến trong phương pháp tự động dò tìm sự trùng nhau của các báo cáo lỗi. Phương pháp này sử dụng đặc điểm trọng số BM25F kết hợp với việc xem xét trên nhiều thuộc tính của tập tin báo cáo lỗi. Phương pháp này sau khi thực nghiệm cho thấy, kết quả có cải tiến hơn là do dựa vào sự tương đồng giữa các báo lỗi cao. Tuy nhiên, trong thực tế có nhiều báo cáo lỗi khác nhau sử dụng các từ (term) khác nhau để diễn tả cho cùng một lỗi. Do đó, khi so sánh những báo cáo lỗi này về độ tương đồng sẽ cho kết quả rất khác nhau. Trong trường hợp này phương pháp của Sun et al sẽ không cho kết quả tốt. Trong bài báo này, nhóm tác giả giới thiệu mô hình LDA-NWF, một mô hình dò tìm những báo cáo lỗi tự động để kiểm tra xem nó có bị trùng nhau hay không?, và tận dụng những ưu điểm không chỉ của kỹ thuật rút trích thông tin mà còn dựa vào mô hình đặc điểm chủ đề sử dụng LDA. Mô hình này được thiết kế để giải quyết bài toán cho hai báo cáo lỗi không tương đồng nhưng được xem là trùng nhau do họ cùng được báo cáo cho một lỗi giống nhau.

[1] Tra Vinh University (Nhan Minh Phuc, Nguyen Thua Phat Tai, Nguyen Hoang Duy Thien)

2. Phương pháp giới thiệu

Phương pháp của nhóm tác giả gồm hai phần chính: Đầu tiên xây dựng mô hình LDA và tính độ tương đồng LDA; Tiếp theo xây dựng phương pháp tính đặc điểm trọng số mới (NWF). Sau đó, kết hợp hai mô hình này lại với nhau được gọi là LDA-NWF.

Hình 1 cho thấy phương pháp tổng thể của mô hình này.



Hình 1. *Mô hình tổng quát*

2.1. Cấu trúc và tiền xử lý báo cáo lỗi

Tất cả báo cáo lỗi trong kho quản lý lỗi được tổ chức theo cấu trúc dữ liệu kiểu danh sách. Cấu trúc này là một dạng kiểu dữ liệu bảng băm. Trong đó, mỗi danh sách chứa một báo cáo lỗi chính Bc (được xem là báo cáo lỗi đầu tiên). Những báo cáo lỗi Bc này được xem như một khóa chính của mỗi danh sách, và tất cả các báo cáo lỗi Trl trùng với báo cáo lỗi chính được xem như là những giá trị của danh sách và chứa cùng một loại lỗi với báo cáo chính. Điều này có nghĩa là mỗi danh sách sẽ chứa một lỗi khác nhau và tất cả những báo cáo lỗi trong cùng danh sách này có cùng một loại lỗi. Khi một báo cáo lỗi mới được người dùng gửi đến, nó sẽ được kiểm tra có trùng với báo cáo lỗi đã được gửi đến kho trước đó hay không. Nếu báo cáo mới này được phát hiện trùng, nó sẽ được thêm vào danh sách tương ứng với danh sách báo cáo lỗi chính mà nó trùng, ngược lại một danh sách mới sẽ được tạo ra và báo cáo lỗi này sẽ trở thành báo cáo chính của danh sách mới được tạo. Ngoài ra, nhóm tác giả cũng tiến hành các bước tiền xử lý với các báo cáo lỗi. Do một tập tin báo cáo lỗi thường chứa nhiều thông tin. Những thông tin này có thể có vài thông tin khác nhau đối với các hệ thống kho mã nguồn mở khác nhau. Nhưng nhìn chung họ hầu như giống nhau. Bảng 1 và Bảng 2 cho thấy cụ thể về những tập tin báo cáo lỗi của Open Office.

Bảng 1. *Báo cáo lỗi trên Open Office có mã lỗi: 9002*

Bug ID	9002
Pro	Math
Com	Code
Sum	formatting of font attributes
Des	The attributes: vector, check, bar, grave, tilde,… so on which are removed from the font. The problem seems to be used to define for Font. Widebar or widehat are works around. It is seems that this has a change from SVv4 that used according toTimes bold which is a conventional mathematical notation, and it is incidentally has better character kerning. Beside the 'wide' version almost don't exist for all properties. Font 'bold' is translated into some type of arial font which has characters which is poor spacing.

Bảng 2. *Báo cáo lỗi trên Open Ofice có mã lỗi 4524*

Bug ID	4524
Pro	Math
Com	UI
Sum	It is too big for spacing between a arrow and vector
Des	Dear, The space is too big making the formula so high between a arrow and its. It is clumsy when formular is a own paragraph. It is easy to make more clear is to copy this file in a sxw text after that insert the formula in middle the previous text like "vec u" as the formula "AB and widevec". This is compared with what you get to insert to the formula "overline AB". Thanks

Do tập tin báo cáo lỗi gốc thuộc dạng XML chứa nhiều thông tin dư thừa, nhóm tác giả sử dụng công cụ Java SAX để chuyển đổi và rút trích lấy bốn thông tin chính của tập tin báo cáo lỗi như: Thông tin về nội dung tóm tắt lỗi, thông tin dùng để mô tả chi tiết lỗi, loại lỗi và thông tin trùng lắp. Thông tin tóm tắt lỗi và phần mô tả chi tiết lỗi được chứa trong thông tin văn bản của tập tin báo cáo lỗi. Thông tin loại lỗi chứa bốn phần gồm: Loại lỗi, sản phẩm, thành phần và phiên bản. Thông tin trùng lắp được dùng để kiểm tra độ chính xác của kết quả thực nghiệm. Tiền xử lý là bước đầu tiên được thực hiện bằng việc trích dữ liệu bao gồm các bước: Làm sạch dữ liệu, tách từ, tìm từ gốc, tìm từ đồng nghĩa, và tìm những từ không có nghĩa để loại bỏ. Với bước tiền xử lý trong trong bài báo này nhóm tác giả sử dụng công cụ GATE [10] và Lucene [11] để làm việc này.

2.2. Xây dựng mô hình LDA

Vấn đề chính của mô hình LDA là làm thế nào để tạo ra các chủ đề từ những tập tin báo cáo lỗi và phân tích nó. Trong LDA, những thuật ngữ (term) hay từ trong tất cả tập tin báo cáo lỗi sẽ được thu thập thành tập các từ vựng, nhóm tác giả gọi là V. Một chủ đề có thể được tạo ra từ các từ khác nhau trong tập tập các từ vựng này. Khi đó mỗi từ trong tập các từ vựng V sẽ có tầng suất xuất hiện khác nhau trong việc tạo ra chủ đề k, và một chủ đề có thể được tạo ra thông qua một hay nhiều từ. Để làm được điều này, LDA sử dụng một vector chọn từ gọi là $\emptyset_k$có kích thước V cho chủ đề k. Mỗi thành phần của vector $\emptyset_k$dựa vào phân bố xác suất của từ, và tương ứng với nó là vị trí của thành phần đó trong tập từ vựng V được dùng để tạo chủ đề k. Mỗi thành phần ϑ trong $\emptyset_k$ có giá trị trong khoảng [0-1]. Giả sử đối với chủ đề 1, $\emptyset_1$=[0,24; 0,23; 0,14;...] như Hình 2, điều này cho thấy, việc phân bố tầng suất xuất hiện của từ đầu tiên trong tập từ vựng được sử dụng để tạo chủ đề k chiếm 24%, trong khi đó đối với từ thứ hai chỉ chiếm 23%, tương tự 14% đối với từ thứ ba,... Một chủ đề sẽ được tạo ra từ tập các từ tùy vào sự phân bố xác suất của chúng. Khi đó, ta có ma trận $\emptyset$=K x V dùng để chọn từ dựa vào việc phân bố từ cho mỗi chủ đề.

Topic 0	$\emptyset_1$
editor	0.24
open	0.23
file	0.14
……	

Topic 1	$\emptyset_1$
repository	0.26
revision	0.18
remote	0.13
……	

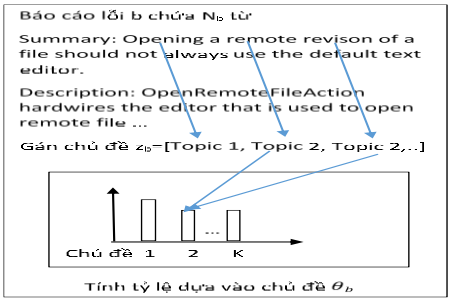
…

Topic K	$\emptyset_K$
navigator	0.24
browser	0.23
display	0.14
……	

Hình 2. *Chủ đề và cách chọn chủ đề*

2.2.1. Sử dụng LDA xử lý tập tin báo cáo lỗi

Nhóm tác giả sẽ tiến hành rút trích tất cả dữ liệu từ hai trường của một báo cáo lỗi: Mô tả (descriptions) và tóm tắt (summaries). Khi đó, tập tin báo cáo lỗi b chứa N_b từ, để sử dụng LDA với báo cáo lỗi này cần hai tham số chính. Đầu tiên là vector dùng để gán chủ đề Z_b. Đối với mỗi vị trí của $N_b$trong báo cáo lỗi b sẽ được xem xét gán cho một chủ đề. Vector $Z_b$dùng để gán chủ đề cho báo cáo lỗi b có kích thước N_b. Mỗi thành phần của vector Z_b là một chỉ mục cho một chủ đề. Tham số thứ hai là θ, đối với một báo cáo lỗi b có thể có nhiều chủ đề, khi đó thuật toán LDA sử dụng tham số θ để xác định tỷ lệ xác suất cho các chủ đề trong báo cáo lỗi b. θ_bcủa báo cáo lỗi b được trình bày bởi một vector với K thành phần. Mỗi thành phần là một giá trị nằm trong khoảng [0-1] để mô hình hóa tỷ lệ của một chủ đề trong báo báo lỗi b. Mỗi giá trị đề cập đến một chủ đề và tổng của chúng là 100%. Giá trị của $\theta_b[k]$ càng cao thì sẽ có càng nhiều từ thuộc chủ đề k có trong báo cáo lỗi b. Ví dụ trong báo cáo lỗi Hình 3, nếu θ_b=[0,20; 0,24; 0,13; ...], có nghĩa là 20% trong báo cáo b có chứa từ "editing", 24% chứa từ "versioning", …



Hình 3. *Mô hình dữ liệu*

2.2.2. Mô hình sinh

LDA là một dạng học máy và thường được gọi là mô hình sinh (generative model). Từ khía cạnh sinh của nó, một báo cáo lỗi b được xem như một đối tượng được tạo bởi ba tham số $Z_b, \theta_b, \emptyset$ như Hình 3. Ví dụ đối với báo cáo lỗi b, mô hình sẽ sinh vector Z_bđể xác định chủ đề cho mỗi vị trí dựa vào xác suất $\theta_b$tính tỉ lệ từ của b. Mỗi chỉ mục sẽ tương ứng với từ $w_b$theo chủ đề được gán và việc phân bổ từ trên mỗi chủ đề $\emptyset_i$tương ứng với chủ đề đó, quá trình này gọi là tiến trình sinh của mô hình LDA. Khi một báo cáo lỗi mới do người dùng gửi đến, mô hình LDA sẽ gán chủ đề $Z_{b_{new}}$ và tỷ lệ $\theta_{b_{new}}$ của những chủ đề này cho b_{new}. Nhóm tác giả sẽ training mô hình này với dữ liệu đã tồn tại trong các kho quản lý lỗi bao gồm những tập tin báo cáo lỗi và thông tin trùng lặp của nó. Những thông tin này sẽ được sử dụng để ước lượng cho vector dùng để gán chủ đề của tất cả các tập tin báo cáo lỗi cũng như tỷ lệ xác suất của chủ đề mà nó có thể chia sẻ có cùng một lỗi. Để dự đoán lỗi, nhóm tác giả sử dụng phương pháp này cho một báo cáo lỗi mới vừa được gửi đến. Khi đó tham số huấn luyện để ước lượng tỷ lệ của từng chủ đề $\theta_{b_{new}}$ của b_{new}. Việc tính độ tương đồng dựa vào tỷ lệ các chủ đề giữa b_{new}và nhóm báo lỗi trùng lắp G được tính như sau:

$$topicsim(b_{new}, G) = (topicsim(b_{new}, b_i)) \quad (1)$$

Trong đó, topicSim(b_{new}, b_i) là độ tương đồng chủ đề giữa hai báo cáo lỗi b_{new}, và b_i. Điều này có nghĩa là độ tương đồng cao nhất sẽ được lấy giữa báo cáo lỗi b_{new}và nhóm báo cáo lỗi trùng lắp G. Nhóm tác giả dùng phương pháp của Jensen-Shannon divergence để làm việc này. Cuối cùng tất cả những nhóm báo cáo lỗi trùng nhau Gj sẽ được sắp xếp lại và nhóm có độ tương đồng cao nhất theo sắp xếp top-k sẽ được xem như là một ứng viên trùng với báo cáo lỗi mới b_{new}.

2.3. Mô hình đặc điểm trọng số mới (NWF)

2.3.1. Trọng số BM25

Sau khi rút trích toàn bộ các từ trong tập tin báo cáo lỗi sang mô hình vector, trong đó mỗi từ được xem tương ứng một chiều trong mô hình vector, giá trị trọng số tùy thuộc vào xác suất từ đó xuất hiện trong một file báo cáo lỗi. Việc xác định độ tương đồng giữa hai báo cáo lỗi được tính dựa vào khoảng cách của những giá trị trọng số trong mô hình vector. Phương pháp cổ điển trước đây thường dùng là sử dụng mô hình TF-IDF. Tuy nhiên, phương pháp này còn nhiều hạn chế. Gần đây một vài nghiên cứu đã giới thiệu một mô hình tính trọng số mới được gọi BM25 [12]. Phương pháp này cho thấy, sự hiệu quả của nó thông qua kết quả dựa vào thực nghiệm trên các hệ thống báo cáo lỗi mã nguồn mở như Mozila, Open Office. BM25 là một mô hình dùng để sắp xếp thứ hạng và được phát triển cho việc sử dụng trong hệ thống tìm kiếm và truy xuất thông tin Okapi [12]. Đối với BM25, các dữ liệu được đánh giá và sắp xếp dựa vào số lần truy xuất từng từ, và nó xem mỗi từ như là một câu lệnh truy vấn để xác định sự phụ thuộc của nó dựa trên xác suất tính được của các từ đối với các file báo cáo lỗi. Đơn giản hơn, có thể hiểu BM25 xác định mối quan hệ bên trong giữa các từ trong câu truy vấn của các file báo cáo lỗi, thay vì xác định mối quan hệ giữa các từ trong truy vấn đối với một báo cáo lỗi. Ngoài những đặc điểm trên, BM25 còn được sử dụng để biểu diễn cho một số hàm sử dụng giá trị trọng số của những biến thể đối với các báo cáo lỗi khác nhau để dùng thay đổi giá trị các tham số cho các phương pháp dùng để truy xuất dữ liệu tương ứng. Trong bài báo này, nhóm tác giả sử dụng giá trị trọng số BM25 cho câu truy vấn q trong file báo cáo lỗi d và được định nghĩa như bên dưới:

$$s(q,d) = \sum_{i=1}^{|q|} idf(q_i) \frac{tf(q_i,d)\cdot(k_1+1)}{tf(q_i,d)+k_1\cdot\left(1-b+b\cdot\frac{|d|}{dl_{avg}}\right)} \quad (2)$$

Trong đó, tf (q_i, d) được hiểu là tần số dùng cho việc lặp lại của từ q_i để xác định sự xuất hiện của nó trong báo cáo lỗi, |d| là tổng độ dài của báo cáo lỗi d và được xác dịnh dựa vào thuật ngữ hay từ, dl_{avg} là độ dài trung bình của các báo cáo lỗi của toàn bộ tập dữ liệu trong kho báo cáo lỗi. Ngoài ra, tham số k_1 và tham số b được sử dụng như các tham số heuristic dùng để điều chỉnh giá trị trọng số để tính tần suất số lần xuất hiện của một thuật ngữ, cũng như chiều dài giá trị của nó trong các báo cáo lỗi. Giá trị được sử dụng phổ biến đối với tham số k_1 là $1,2 \le k_1 \le 2,0$ và đối với tham số b là $0,5 \le b \le 0,8$. Ngoài ra, *idf(qi)* dùng để hiện số lần báo cáo lỗi nghịch đảo được tính dựa vào truy vấn q_i. Nó được định nghĩa theo công thức bên dưới:

$$idf(q_i) = log\left(\frac{N-df(q_i+0.5}{df(q_i)+0.5}\right) \quad (3)$$

Trong (3), N được tính dựa vào tổng số báo cáo lỗi được xác định trong tập dữ liệu của kho báo cáo lỗi dùng cho thực nghiệm và *df(q_i)* dùng để xác định số báo cáo lỗi mà nó chứa những từ hay thuật ngữ trong truy vấn q_i.

2.3.2. Giới thiệu đặc điểm trọng số mới (NWF)

Mặc dù, BM25 cho thấy sự hiệu quả của nó đối với việc tính đặc điểm trọng số. Tuy nhiên, theo [9] BM25 vẫn còn những hạn chế. Cụ thể, đối với việc tính cosine ranking, BM25 cho kết quả tốt hơn đối với câu truy vấn ngắn. Ngược lại, đối với câu truy vấn dài thuật toán này chưa cho thấy hiệu quả của nó. Điều này cũng gặp phải đối với một vài thuật toán xếp hạng không chỉ cho BM25. Để khắc phục hạn chế này, sau khi quan sát đánh giá dữ liệu từ những tập tin báo cáo lỗi, nhóm tác giả nhận thấy BM25 không được chứa giá trị âm, và chỉ cho kết quả tốt đối với câu truy vấn ngắn. Sau khi phân tích, nhóm tác giả nhận thấy điều này liên quan đến thành phần IDF:

$$rsv_q = \sum_{t \in q} log\left(\frac{N+1}{df_t+0.5}\right).\frac{(k_1+1).tf_{td}}{k_1\left((1-b)+b.\left(\frac{L_d}{L_{avg}}\right)\right)+tf_{td}} \quad (4)$$

Khi đó nhóm tác giả đề xuất lại IDF bằng cách sắp xếp lại để có:

$$rsv_q = \sum_{t \in q} log\left(\frac{N+1}{df_t+0.5}\right).\frac{(k_1+1).C_{td}}{k_1+C_{td}} \quad (5)$$

Trong đó:

$$C_{td} = \frac{tf_{td}}{1-b+b.\left(\frac{L_d}{L_{avg}}\right)} \quad (6)$$

Đối với công thức này, nhóm tác giả quan tâm đến sự ảnh hưởng của thành phần C_{td}, để khắc phục trường hợp đối với câu truy vấn dài. Giải pháp đưa ra là bổ sung thêm một hằng số nguyên dương O, điều này có tác dụng làm thay đổi chức năng ưu tiên số nhỏ hơn (nghĩa là mẫu số lớn, giá trị Ld lớn hoặc tài liệu dài).

2.4. Kết hợp LDA với đặc điểm trọng số mới

Trong phần này, đề cập đến phương pháp sử dụng kết hợp giữa LDA với NWF cho việc xác định những báo cáo lỗi được xem là trùng nhau. Trong mô hình này, nhóm tác giả đưa ra hai kỹ thuật dự đoán p1 và p2. Kỹ thuật p1 dựa vào mô hình chủ đề (LDA), và p2 dựa vào đặc điểm trọng số mới. Hai kỹ thuật này có những ưu điểm khác nhau trong mô hình dự đoán những báo cáo lỗi trùng nhau. Kỹ thuật NWF sẽ cho kết quả chính xác hơn nếu hai báo cáo lỗi có độ tương đồng về thuật ngữ trong tập tin báo cáo lỗi. Tuy nhiên, kỹ thuật này sẽ cho kết quả thấp nếu hai báo cáo lỗi khác nhau nhưng lại mô tả cùng một lỗi (trùng nhau), bằng những thuật ngữ khác nhau trong hai báo cáo lỗi. Ngược lại, phương pháp sử dụng mô hình LDA xác định hai báo cáo lỗi có trùng nhau hay không dựa vào độ tương đồng chủ đề, thậm chí trong trường hợp hai báo cáo lỗi này không có độ tương đồng về thuật ngữ. Điều này có nghĩa là nếu hai báo cáo lỗi dùng những thuật ngữ khác nhau viết báo cáo lỗi, nhưng cùng diễn tả về một lỗi phát sinh giống nhau, khi đó phương pháp này cho kết quả dò tìm tốt hơn phương pháp p1. Tuy nhiên, do phương pháp LDA dựa vào độ tương đồng chủ đề giữa hai báo lỗi, mà mô hình chủ đề thường là một sự tóm tắt nội dung mô tả lỗi, do đó kết quả tính độ tương đồng của nó cũng sẽ không hiệu quả bằng việc so sánh giữa các thuật ngữ như cách tính đặc điểm trọng số của từ. Với việc kết hợp cả hai kỹ thuật, có thể tận dụng ưu điểm của cả hai phương pháp để bổ trợ cho nhau trong việc xác định sự tương đồng giữa hai báo cáo lỗi, những báo cáo lỗi trùng nhau cho kết quả tốt hơn. Để thực hiện việc này, nhóm tác giả sử dụng phương pháp học máy gọi là Ensemble Average, đây là một phương pháp tuyến tính. Việc kết hợp này được thực hiện như sau:

$$p=\alpha1 \text{ x } p1 + \alpha2 \text{ x } p2 \quad (7)$$

Trong đó, α1 và α2 là những tham số trong việc ước lượng xem những báo cáo lỗi có trùng nhau hay không và phải thỏa điều kiện α1+α2=1. Điều này có nghĩa, nếu α1=1 và α2=0, khi đó chỉ áp dụng kỹ thuật một nghĩa là sử dụng mô hình LAD. Ngược lại, nếu α1=0 và α2=1 khi đó phương pháp dò tìm sử dụng kỹ thuật xác định độ tương đồng khi so sánh giữa hai báo cáo lỗi dựa vào đặc điểm trọng số NW. Việc chọn giá trị để tối ưu cho hai tham số này sẽ được thảo luận trong phần kết quả thực nghiệm.

3. Thuật toán

3.1. Thuật toán cho mô hình LDA

Mục đích của thuật toán này là để ước lượng những tham số cho mô hình LDA như z_b, θ_b, và θ_{br} với dữ liệu training là kho báo cáo lỗi B và tập hợp những nhóm báo cáo lỗi trùng nhau {Gj(b)}. Nhóm tác giả sử dụng thuật toán Gibbs sampling để làm điều này. Đầu tiên hai tham số $z_b$và θ_{br}được gán cho những giá trị ngẫu nhiên. Khi đó một vòng lặp được thực hiện để ước lượng mỗi tham số dựa vào việc tính phân bổ chủ đề từ những giá trị có sẵn. Vòng lặp sẽ kết thúc khi tổng của sự khác nhau giữa việc phân bổ chủ đề được ước lượng hiện tại và sự phân bổ chủ đề được ước lượng trước đó nhỏ hơn hoặc bằng ngưỡng. Ý tưởng thuật toán được mô tả như sau:

3.1.1. Ước lượng việc gán chủ đề cho những báo cáo lỗi trong B

Với mỗi báo cáo lỗi b trong B, mô hình LDA ước lượng việc gán chủ đề $z_b[i]$ cho vị trí i. Cho mỗi chủ đề k trong K chủ đề, nó sẽ ước lượng dựa vào xác suất mà chủ đề k được gán cho vị trí i trong báo cáo lỗi b. Khi đó nó gán một chủ đề dựa vào giá trị xác suất của k. Có hai trường hợp xảy ra:

Trường hợp thứ nhất, một báo cáo lỗi không có báo cáo lỗi nào trùng với nó, khi đó việc ước lượng gán chủ đề được thực hiện theo thuật toán của Gibbs sampling trong mô hình LDA như sau:

$$p(z_b[-i]], w_b) = \frac{(N_b[-i,k]+\alpha).(N_{BR,k}[-i,w_i]+\beta)}{(N_b-1,+K\alpha).(N_{BR,k}-1+V\beta)} \quad (8)$$

Trong đó, $N_b[-i,k]$ là số từ trong b (ngoại trừ vị trí hiện tại thứ i) được gán đến chủ đề k. N_b là tổng số từ trong b. $(N_{BR,k}[-i,w_i]$ là số từ wi trong tất cả những báo cáo lỗi B (ngoại trừ vị trí hiện tại) được gán đến chủ đề k. $N_{BR,k}$ là tổng số từ trong B được gán đến chủ đề k.

Trường hợp thứ 2, nếu một báo cáo lỗi b được xác định trùng nhau với nhóm báo cáo lỗi G_j, nghĩa là nó sẽ có cùng chủ đề với nhóm này. Công thức bên dưới thể hiện điều này:

$$p(z_b[-i]], w_b) = \frac{(N*_b[-i,k]+\alpha).(N_{BR,k}[-i,w_i]+\beta)}{(N*_b-i,+K\alpha).(N_{BR,k}-1+V\beta)} \quad (9)$$

Trong đó, $(N_{BR,k}[-i,w_i]$ là số từ w_i trong tất cả báo cáo lỗi trong B, ngoài trừ vị trí hiện tại được gán đến k, và $N_{BR,k}$ là số từ trong S đang mô tả thông tin k. So với công thức (4), do một báo cáo lỗi trùng nhau có cùng chủ đề với những báo cáo lỗi trong cùng nhóm. Tỷ lệ θ của một chủ đề k được mô tả trong báo cáo lỗi, bao gồm tỷ lệ của chủ đề θ_b và tỷ lệ chủ đề được chia sẽ θ_{F_j} của nhóm báo cáo lỗi trùng nhau G_j.

Từ 3 và 4 ta có $N *_b [-i,k] = N_b[-i,k]N_{G_j}k$ và $N *_b [-i] = (N_b - 1)N_{G_j}k$. Trong đó, $N_b[-i,k]$ là số thuật ngữ trong b ngoại trừ cho vị trí hiện tại mà nó được gán trong chủ đề k; N_b là tổng số từ trong b; $N_{G_j}k$ là tổng số vị trí được gán cho chủ đề k trong tập dữ liệu báo cáo lỗi trùng nhau trong nhóm G_j và N_{Gj} là tổng của các báo cáo lỗi này dựa vào chiều dài của nó. Công thức này tác động đến việc chia sẽ chủ đề trong việc ước lượng của $\theta_b[k]$ làm $\theta_{F_i}[k]$ phản ánh và được ước lượng thông qua tỉ lệ $N_{G_j}k/N_{G_j}$.

3.1.2. Ước lượng cho chủ đề dựa vào $\theta_b$cho báo cáo lỗi b

Sau khi việc gán chủ đề cho tất cả vị trí trong b được ước lượng, tỷ lệ ước lượng $\theta_b[k]$ của chủ đề k trong b có thể được tính đơn giản bằng tỷ lệ giữa số từ được mô tả trong chủ đề k và chiều dài của báo cáo lỗi.

3.1.3. Ước lượng việc phân bố từ θ_{BR}

Đây là bước cuối cùng được dùng để ước lượng việc phân bố từ trên mỗi chủ đề. Đối với mỗi từ w_i trong V_{oc} và chủ đề k. $\theta_k[w_i]$ được tính dựa vào tỷ lệ giữa số lần mà từ đó xuất hiện tại vị trí thứ i trong V_{oc} và được dùng để diễn tả chủ đề k và tổng số lần mà bất kỳ thuật ngữ được sử dụng để mô tả cho chủ đề k.

3.2. Mô hình LDA-NWF

Mô hình LDA-NWF kết hợp mô hình LDA và mô hình đặc điểm trọng số mới. Khi đó, ta cần xác định α1 và α2 để tính độ tương đồng giữa báo cáo lỗi mới và các nhóm báo cáo lỗi được phân loại là trùng nhau. Đầu tiên nhóm tác giả training mô hình LDA và NWF. Những tham số của mô hình được training dùng để ước lượng mức độ tương tự nhau giữa hai báo cáo lỗi và mức độ giống nhau chủ đề của một báo cáo lỗi đối với các nhóm báo cáo lỗi giống nhau. Những mức độ tương đồng này được kết hợp sử dụng sim(Btest, Gtest) thông qua một đặc điểm trọng lượng khác nhau α1. Việc kết hợp những giá trị tương đồng được dùng để xếp hạng giữa báo cáo lỗi mới và nhóm báo cáo lỗi được phân loại trùng nhau. Danh sách xếp hạng Lpred được sử dụng để đánh giá chức năng của MAP(Gtest, Lpred) được sử dụng để tìm giá trị tối ưu cho α1. Giá trị α1 sẽ nhận được từ giá trị cao nhất được trả về từ MAP(Gtest, Lpred). Hàm này được sử dụng để tính độ chính xác trung bình [3] và được định nghĩa như sau:

$$MAP(L_{test}, L_{pred}) = \frac{1}{|L_{test}|}\sum_{i=1}^{|L_{test}|}\frac{1}{index_i} \quad (10)$$

Trong đó, Ltest là những liên kết đến các báo cáo lỗi trùng nhau trong kho dữ liệu dùng cho testing. Lpred là danh sách được sắp xếp của những liên kết đực dự đoán. Indexi là vị trí của nhóm báo cáo lỗi trùng nhau đúng được lấy từ truy vấn thứ i. Do MAP được sử dụng để đo lường độ chính xác của thuật toán sắp xếp đối với những liên kết đúng nên có thể coi nó như chức năng chính trong việc training cho mô hình LDA-NWF. Trọng lượng từ α1 và α2 được training dùng để tính trong việc kết hợp giữa một báo cáo lỗi và độ tương đồng chủ đề và được tính như sau:

$$Sim=\alpha1*sim1+\alpha2*sim2 \quad (11)$$

Trong đó, sim1 và sim2 là tập tin báo cáo lỗi và độ tương đồng chủ đề giữa một báo cáo lỗi bnew và nhóm báo cáo lỗi G. Độ tương đồng được kết hợp càng cao thì bnew càng được xem là trùng với những báo cáo lỗi trong nhóm G. Thuật toán được thể hiện Hình 4.

```
1  //thuật toán
2  function PredictModel(φ_BR,BugReport b_new, DuplicateGroups G_j)
3    // ước lượng tỷ lệ chủ đề của báo cáo lỗi mới b_new
4    repeat
5      θ'_{b_new} ← θ_{b_new}
6      for (i = 1 to N_b)
7        θ_{b_new} = EstimateZB2(b_new, i) //ước lượng topic tại vị trí i
8      end
9      θ_{b_new}[k] = N_{b_new}[k]/N_{bnew} //ước lượng tỷ lệ chủ đề
     until (|θ_{b_new} − θ'_{b'_new}| <= ϵ)
11   // Calculate topic similarity between bug report b_new and G_j
12   for (DuplicateGroups G_j ∈ B)
13     sim_2(b_new, G_j) = TopicSim(b_new, G_j)
14   end
15   return list (sim_2(b_new, G_j))
16 end
17 // ——————— Gán chủ đề cho vị trí i trong b ——————
18 function EstimateB2(BugReport b_new,, int i)
19   p(z_{b_new}[i] = k) ← ((N_{b_new}[−i,k]+α)/(N_{b_new}−1+Kα)) ((N_{BR,k}[−i,w_i]+β)/(N_{BR,k}−1+Vβ))
20   z_{b_new}[i] ← sample(p(z_{b_new}[i]))
21 end
22 //Tính độ tương đồng of b_new và nhóm báo cáo lỗi
23 function TopicSim(b_new, G_j)
24   for (BugReports b_i ∈ G_j)
25    TopicSim(b_new, b_i) = 1 − JSDivergence(θ_{b_new}, θ_{b_i})
26   end
27   TopicSim(b_new, G_j) = max_{b_i ∈ G_j} (TopicSim(b_new, b_i))
28   return TopicSim(b_new, G_j)
29 end
```

Hình 4. *Thuật toán*

4. Kết quả đánh giá

4.1. Tập dữ liệu và tham số K

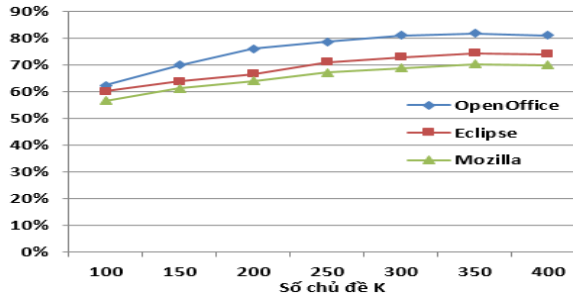
Để đánh giá của phương pháp được giới thiệu, nhóm tác giả sử dụng cùng tập dữ liệu báo cáo lỗi được công bố trong [9]. Hai thông tin quan trọng trong file báo cáo lỗi là thông tin dùng để tóm tắt (summary) và phần mô tả (description) sau khi được rút trích từ các tập tin báo cáo lỗi sẽ được lưu trong cùng một tập tin dữ liệu. Sau đó, nhóm tác giả thực hiện tiền xử lý với những kỹ thuật như tách từ, phục hồi từ gốc, bỏ những từ không có nghĩa… Khi đó, tất cả những thuật ngữ còn lại được đánh chỉ mục. Sau giai đoạn này một báo cáo lỗi được xem như một vector trong đó những từ trong nó được chỉ mục tương ứng. Tất cả báo cáo lỗi được sắp xếp theo trình tự thời gian. Nhóm tác giả chia tập dữ liệu sang hai phần: Phần dùng cho huấn luyện và phần dùng cho kiểm tra. Phần dùng để huấn luyện bao gồm M báo cáo lỗi được xác định đầu tiên, và 200 báo cáo lỗi trong số này trùng nhau, và nó được dùng để huấn luyện cho mô hình LDA và NWF. Những báo cáo còn lại được dùng cho việc kiểm tra đánh giá. Sau khi nhóm tác giả thực nghiệm cho phần kiểm tra (testing), nếu xác định một báo cáo trùng nhau b, sẽ trả về một danh sách top-k ứng viên những nhóm báo cáo lỗi trùng nhau. Nếu một báo cáo lỗi được xác định trùng nhau với nhóm lỗi G trong danh sách top-k, nhóm tác giả đếm nó như có một xác định đúng. Khi đó, sẽ thêm báo cáo lỗi b đến nhóm đó để huấn luyện sau này. Độ chính xác top-k hay còn gọi là recall rate được tính bằng tỷ lệ số báo cáo xác định đúng trên tổng số những báo cáo lỗi đang xem xét.

$$Recal\ rate = \frac{\text{Số những dự đoán đúng}}{\text{Tổng số những báo cáo lỗi trùng nhau}} \quad (12)$$

Ngoài ra, nhóm tác giả cũng xem xét sự tác động liên quan đến việc chọn số chủ đề K. Nhóm tác giả thực nghiệm trên tập dữ liệu Eclipse trong khoảng 20 đến 400 với khoảng cách là 20 và kết quả lấy trong top-10. Kết quả như Hình 5.

Từ việc quan sát kết quả ta có thể thấy, K càng nhỏ (K<60) cho độ chính xác thấp. Lý do số đặc điểm đặc trưng của các báo lỗi quá nhỏ để phân biệt lỗi do có quá nhiều báo cáo lỗi được phân loại cùng một nhóm chủ đề. Khi đó, số chủ đề tăng đồng nghĩa độ chính xác cũng tăng lên. Tuy nhiên, độ ổn định có sự

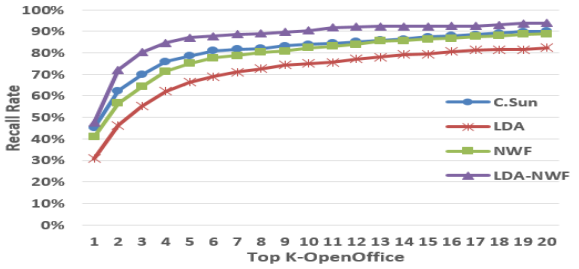
khác nhau chút ít đối với các kho lỗi. Ví dụ, K=[140-320] đối với Eclipse, K=[120-300] cho Open Office, và K=[100-240] cho Mozila. Điều này cũng cho thấy, với giá trị của K cho mỗi dự án phần mềm khác nhau có giá trị cao sẽ cho kết quả ổn định về độ chính xác trong việc xác định báo cáo lỗi trùng nhau. Lý do số lượng chủ đề lớn phản ánh tốt số lỗi trong những tập tin báo cáo lỗi. Ngoài ra, nhóm tác giả cũng quan sát thấy trong trường hợp K>380 độ chính xác bắt đầu giảm bởi vì khi đó số chủ đề lớn có thể dẫn đến sự chồng lắp ngữ nghĩa và một báo cáo lỗi có nhiều chủ đề với tỷ lệ tương đồng gần giống nhau. Điều này ảnh hưởng đến tỷ lệ xác định mức độ chính xác trong việc xác định các báo cáo lỗi trùng nhau.



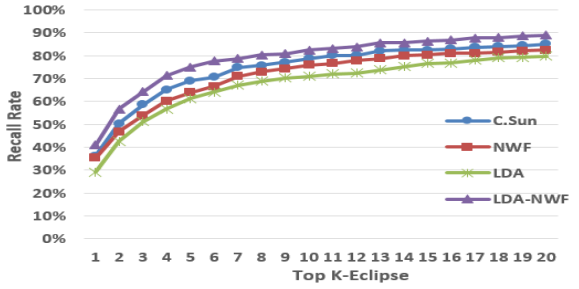
Hình 5. *So sánh số chủ đề K*

4.2. So sánh với các phương pháp khác

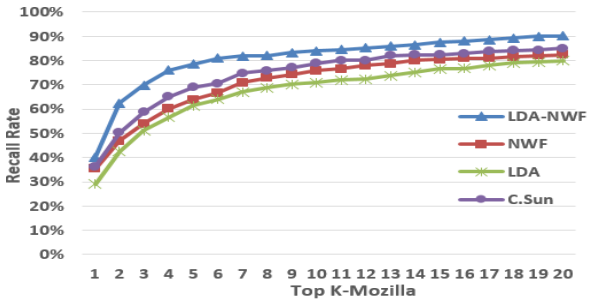
Để đánh giá hiệu quả của phương pháp, nhóm tác giả làm thực nghiệm để so sánh phương pháp được giới thiệu với các kỹ thuật đã được công bố gần đây. Cụ thể, trong bố trong [9] sử dụng mô hình trọng số BM25F. Mặc dù, phương pháp này cho kết quả dò tìm dựa trên kết quả thực nghiệm có cải tiến so với các phương pháp khác. Tuy nhiên, nó chỉ hiệu quả đối với các báo cáo lỗi được diễn tả bằng những từ hay thuật ngữ (term) tương tự nhau khi mô tả lỗi, và không hiệu quả trong trường hợp các báo cáo lỗi sử dụng những từ hay thuật ngữ khác nhau nhưng mô tả cùng một lỗi. Ngoài ra, để so sánh kết quả đánh giá của phương pháp được giới thiệu với các phương pháp khác, nhóm tác giả cũng thực hiện so sánh với mô hình LDA và NWF riêng biệt. Kết quả quan sát cho thấy, phương pháp LDA-NWF cho kết quả tốt hơn phương pháp của C.Sun, LDA và NWF như được thấy trong Hình 6 đến Hình 8.



Hình 6. *So sánh các phương pháp trước trên Open Office*



Hình 7. *So sánh các phương pháp trước trên Eclipse*



Hình 8. *So sánh các phương pháp trước trên Mozilla*

5. Kết luận

Bài báo này sử dụng một kỹ thuật mới LDA-NWF trong việc xác định những báo cáo lỗi trùng nhau. Phương pháp này không chỉ dựa vào kỹ thuật lấy thông tin mà còn dựa vào mô hình chủ đề LDA. Mô hình này tận dụng ưu điểm của mô hình kỹ thuật lấy thông tin đối với file báo cáo lỗi có cùng độ tương đồng, và mô hình LAD sử dụng cho file báo cáo lỗi với độ tương đồng kém. Kết quả sau khi thực nghiệm cho thấy, mô hình LAD-NWF cho kết quả dò tìm tốt hơn các kỹ thuật trước đây đã công bố khi so sánh trên cả ba hệ thống mã nguồn mở từ 4-9%.

TÀI LIỆU THAM KHẢO

[1] J. Lerch and M. Mezini, "Finding Duplicates of Your Yet Unwritten Bug Report," *2013 17th European Conference on Software Maintenance and Reengineering*, pp. 69-78, doi: 10.1109/CSMR.2013.17.

[2] N. S. a. I. Ciordia, "Bugzilla, ITracker, and other bug", 2013, 17th European Conference on Software Maintenance and Reengineering. IEEE, 69–78, 2005.

[3] M. &. B. C.-P. &. H. A. E. Rakha, "Revisiting the Performance Evaluation of Automated Approaches for the Retrieval of Duplicate Issue Reports", *in IEEE Transactions on Software Engineering*. PP. 10.1109/TSE.2017.2755005., 2017.

[4] S. L. D. Chengnian, X. Wang, J. Jiang and S.-C. Khoo, "A discriminative model approach for accurate duplicate bug report retrieval", *in Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering*, ACM, pp. 45-54., 2010.

[5] Y. Tian, C. Sun and D. Lo, "Improved Duplicate Bug Report Identification", *2012 16th European Conference on Software Maintenance and Reengineering*, 2012, pp. 385-390, doi: 10.1109/CSMR.2012.48.

[6] L. Z. T. X. J. A. a. J. S. X. Wang, "An approach to detecting duplicate bug reports using natural language and execution information", *in ACM/IEEE 30th International Conference on Software Engineering, Leipzig*, 2008, pp. 461-470, 2008.

[7] Meng-Jie Lin, Cheng-Zen Yang, Chao-Yuan Lee, Chun-Chang Chen, "Enhancements for duplication detection in bug reports with manifold correlation features", *Journal of Systems and Software, Elservier*, vol. Volume 121, no. November, pp. Pages 223-233, 2016.

[8] N. J. a. W. Weimer, "Automated duplicate detection for bug tracking systems", *in IEEE International Conference on Dependable Systems and Networks with FTCS and DCC (DSN), Anchorage, AK*, 2008, pp. 52-61, doi: 10.1109/DSN.2008.4630070.

[9] D. L., K. a. J. J. C. Sun, "Towards more accurate retrieval of duplicate bug reports", *in 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2016)*, pp. 253-262, Lawrence, KS, 2017.

[10] D. M. K. B. H. Cunningham, "GATE: an architecture for development of robust HLT applications", *in Proceedings of the 40th annual meeting on association for computational linguistics*, pp.168–175, 2002.

[11] M. O. Gospodnetic and E. Hatcher, " Lucene in Action", *Manning Publications Co., Greenwich, CT* 2005.

[12] J. Whissell and C. Clarke, "Improving document clustering using Okapi BM25 feature weighting", *information Retrieval Journal*, vol. Vol. 14, no. Issue 5, pp. p466-487, 2011.