ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN HỆ THỐNG THÔNG TIN

LÊ NGUYỄN BÁ DUY –TRẦN MINH TRÍ

TÌM HIỂU CÁC HƯỚNG TIẾP CẬN PHÂN LOẠI EMAIL VÀ XÂY DỰNG PHẦN MỀM MAIL CLIENT HỖ TRỢ TIẾNG VIỆT

KHOÁ LUẬN CỬ NHÂN TIN HỌC

TP. HCM, NĂM 2005

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN HỆ THỐNG THÔNG TIN

LÊ NGUYỄN BÁ DUY -0112050
TRẦN MINH TRÍ -0112330

TÌM HIỂU CÁC HƯỚNG TIẾP CẬN PHÂN LOẠI EMAIL VÀ XÂY DỰNG PHẦN MỀM MAIL CLIENT HỖ TRỢ TIẾNG VIỆT

KHOÁ LUẬN CỬ NHÂN TIN HỌC

GIÁO VIÊN HƯỚNG DẪN

THẦY LÊ ĐỨC DUY NHÂN

NIÊN KHÓA 2001-2005

LỜI CẢM ƠN

Trước tiên, chúng tôi xin chân thành cảm ơn thầy Lê Đức Duy Nhân, người đã hướng dẫn chúng tôi thực hiện đề tài này. Nhờ có sự hướng dẫn, chỉ bảo tận tình của thầy, chúng tôi đã hoàn thành khoá luận này.

Chúng con xin kính gởi lòng biết ơn, kính trọng của chúng con đến ông bà, cha mẹ và các người thân trong gia đình đã hết lòng nuôi chúng con ăn học, luôn luôn ở bên chúng con, động viên giúp đỡ chúng con vượt qua khó khăn

Chúng em xin cảm ơn tất cả các thầy cô trường Đại học Khoa Học Tự Nhiên, đặc biệt là các thầy cô trong khoa Công Nghệ Thông Tin đã hết lòng giảng dạy, truyền đạt nhiều kiến thức và kinh nghiệm quý báu cho chúng em. Chúng em cũng xin chân thành cảm ơn khoa Công Nghệ Thông Tin, bộ môn Hệ Thống Thông Tin đã tạo mọi điều kiện thuận lợi trong quá trình thực hiện khoá luận của chúng em.

Chúng tôi xin chân thành cảm ơn bạn bè trong lớp cũng như các anh chị đi trước đã giúp đỡ, đóng góp ý kiến cho chúng tôi.

Với thời gian nghiên cứu ngắn, trong vòng 6 tháng và năng lực của những người làm đề tài, chắc chắn đề tài còn có nhiều thiếu sót. Chúng tôi rất mong nhận được những góp ý, nhận xét để đề tài được hoàn thiện hơn.

Thành phố Hồ Chí Minh

Tháng 7 năm 2005

Những người thực hiện:

Lê Nguyễn Bá Duy – Trần Minh Trí.

v Mục lục:

Danh mục các hình vẽ:

Danh mục các bảng:

Chương 1 : MỞ ĐẦU

1.1 Giới thiệu:

Thời đại ngày nay là thời đại bùng nổ thông tin, Internet đã trở nên quen thuộc và không thể thiếu đối với mỗi quốc gia và xã hội. Liên lạc qua Internet đã trở nên phổ biến, và email là một phương tiện liên lạc có chi phí thấp, nhanh chóng và hiệu quả nhất trên Internet. Hằng ngày mỗi người sử dụng email đều nhận được một lượng lớn email, tuy nhiên không phải tất cả các email mà ta nhận được đều chứa thông tin mà ta quan tâm. Những email mà ta không muốn nhận ấy là email Spam. Ngược lại, những email không phải là spam gọi là non-spam – email hợp lệ được ngườidùng chấp nhận.

Spam chính là những email được phát tán một cách rộng rãi không theo bất cứ một yêu cầu nào của người nhận với số lượng lớn (unsolicited bulk email (UBE)), hay những email quảng cáo được gởi mà không có yêu cầu của người nhận (unsolicited commercial email (UCE)) [1].

Nhiều người trong chúng ta nghĩ rằng spam là một vấn đề mới, nhưng thực ra nó đã xuất hiện khá lâu – ít nhất là từ năm 1975. Vào lúc khởi thủy, người dùng hầu hết là các chuyên gia về máy tính, họ có thể gởi hàng tá thậm chí hàng trăm email đến các nhóm tin (newsgroup) và spam hầu như chỉ liên quan đến các email gởi đến các nhóm tin Usenet, gây ra tình trạng không thể kiểm soát được các email nhận. Sau đó các biện pháp trừng trị về mặt xã hội và hành chính đã có tác dụng, thủ phạm đã bị trừng phạt , công khai hay bí mật, những người này nhanh chóng được đưa vào một danh sách, và một kĩ thuật lọc spam sớm nhất xuất hiện đó là ”bad sender” – lọc email của những người gởi được xem là xấu.

WWW(World-Wide Web) đã mang thế giới Internet đến nhiều người, và hệ quả của nó là nhiều người không phải là chuyên gia trong thế giới máy tính cũng được tiếp xúc nhiều với Internet, nó cho phép truy cập đến những thông tin và dịch vụ mà trước đây là không được phép. Chỉ trong vòng 2-3 năm chúng ta đã chứng kiến sự bùng nổ số người sử dụng Internet và tất nhiên là những cơ hội quảng cáo trên đấy. Và spam đã phát triển một cách nhanh chóng từ đây, những kĩ thuật ngăn

chặn spam trước đây đã không còn thích hợp. Spam thường theo sau những quảng cáo thương mại chèo kéo khách hàng (những email quảng cáo thương mại được gởi mà không có yêu cầu) [2]. Spam đã và đang gây tác hại đến người sử dụng Internet và tốc độ đường truyền Internet. Với người sử dụng email, spam gây cho họ cảm giác bực bội và phải mất thời gian và tiền bạc để xóa chúng, đôi khi họ có thể bị mất những email quan trọng chỉ vì xóa nhầm, tốc độ trên mạng xương sống của Internet (Internet Backbone) cũng bị spam là cho chậm lại vì số lượng spam được chuyển đi trên mạng là cực lớn [3]. Theo thống kê của ZDNet ở thời điểm năm 2004, mỗi ngày có khoảng 4 tỷ email spam được phát tán qua Internet, trên 40% lượng email trên mạng là spam[1], gần đây đã đạt con số 50%[2]. Cho dù được nhận diện là "kẻ thù của cộng đồng"("public enemy") Internet, nhưng spam đã và đang mang lại lợi nhuận. Trong số 100.000 email spam phát tán, chỉ cần một email có phản hồi là đã có thể bù đắp chi phí đầu tư [4].

Để ngăn chặn spam, nhiều nhà khoa học, các tổ chức, các cá nhân đã nghiên cứu và phát triển những kĩ thuật phân loại và lọc email, tuy nhiên các spammer - những người tạo nên spam và phát tán chúng cũng tìm mọi cách vượt qua các bộ lọc này. Cuộc chiến giữa các spammer và những người chống spam vẫn còn đang tiếp diễn và dường như không có hồi kết. Thực tế cho thấy, nhu cầu có một phương pháp và công cụ chống spam hữu hiệu là rất cần thiết.

Xuất phát từ thực trạng đó, nhóm chúng tôi chọn hướng nghiên cứu **"Tìm hiểu các hướng tiếp cận cho bài toán phân loại email và xây dựng phần mềm Mail Client hỗ trợ tiếng Việt "** với mục đích tìm hiểu, thử nghiệm các phương pháp tiếp cận cho bài toán phân loại email , từ đó thực hiện phân loại email giúp ngăn chặn email spam hiệu quả.

[1] http://zdnet.com.com/2100-1106-955842.html
[2] http://zdnet.com.com/2100-1105_2-1019528.html

1.2 Yêu cầu bài toán:

Yêu cầu đối với một hệ thống phân loại email và ngăn chặn email spam đương nhiên là phân loại được email là spam hay non-spam, từ đó sẽ có biện pháp ngăn chặn email spam, hiệu quả phân loại email phải khả quan, tuy nhiên không thể đánh đổi hiệu quả phân loại email spam cao mà bỏ qua lỗi sai cho rằng email non-spam là spam, bởi vì cùng với việc tăng khả năng phân loại email spam thì khả năng xảy ra lỗi nhận nhầm email non-spam thành email spam cũng tăng theo. Do đó yêu cầu đối với một hệ thống phân loại email spam là phải nhận ra được email spam càng nhiều càng tốt và giảm thiểu lỗi nhận sai email non-spam là email spam.

1.3 Bố cục khoá luận :

Chúng tôi chia khoá luận làm 9 chương

- Chương 1 Giới thiệu về đề tài, bài toán phân loại email.
- Chương 2 Tổng quan : trình bày một số hướng tiếp cận phân loại email và chống email spam, đồng thời có sự nhận xét đánh giá các phương pháp, từ đó có cơ sở để chọn lựa hướng tiếp cận giải quyết vấn đề.
- Chương 3 : Giới thiệu và mô tả về cơ sở dữ liệu dùng để học và kiểm thử

  Hai chương tiếp theo, chúng tôi trình bày cơ sở lý thuyết và thực hiện phân loại email theo phương pháp Bayesian.
- Chương 4: Trình bày cơ sở lý thuyết cho hướng tiếp cận dựa trên phương pháp Bayesian.
- Chương 5: Thực hiện phân loại email dựa trên phương pháp Bayesian và kiểm thử.

  Hai chương tiếp theo, chúng tôi trình bày cơ sở lý thuyết và thực hiện phân loại email theo phương pháp AdaBoost
- Chương 6: Trình bày cơ sở lý thuyết cho hướng tiếp cận dựa trên thuật toán AdaBoost.
- Chương 7: Thực hiện phân loại dựa trên phương pháp AdaBoost và kiểm thử.

- Chương 8: Xây dựng phần mềm email Client tiếng Việt hỗ trợ phân loại email
- Chương 9: Tổng kết, trình bày về những vấn đề đã thực hiện, những kết quả đạt được, đề xuất hướng mở rộng, phát triển trong tương lai.

Chương 2 : TỔNG QUAN

2.1 Các cách thức con người xử lý với spam :

Trên thế giới đã có nhiều tổ chức, công ty phát triển nhiều cách thức khác nhau để giải quyết vấn đề spam. Có nhiều hệ thống được xây dựng sẵn một "danh sách đen" (Blacklist) chứa các tên miền mà từ đó spam được tạo ra và phát tán, và dĩ nhiên là các email đến từ các tên miền này hoàn toàn bị khóa (block out). Một số hệ thống căn cứ vào header của email (những trường như nơi gởi (from), tiêu đề (subject)..) và loại bỏ những email có địa chỉ xuất phát từ những spammer (người phát tán spam). Vài hệ thống khác lại tìm kiếm trong nội dung của email, những dấu vết cho thấy có sự tồn tại của spam chẳng hạn email có quá nhiều dấu than, số chữ cái được viết hoa nhiều một cách bất bình thường …

Tuy nhiên các spammer ngày càng tinh vi, vì thế các kỹ thuật dùng để chống spam cũng phải được cải tiến, và chính những cải tiến này càng thôi thúc các spammer trở nên ranh ma và tinh vi hơn… Kết quả là như hiện nay, các email spam gần như giống với một email thông thường. Tuy nhiên email spam có một điều không bao giờ thay đổi đó là bản chất của nó. Bản chất đó chính là mục tiêu quảng cáo sản phẩm hay dịch vụ. Nó là cơ sở cho phương pháp lọc email dựa trên nội dung (content based filtering).Theo đó, chúng ta cố gắng phát hiện ra các ngôn ngữ quảng cáo (sales-pitch language) thay vì chú ý đến các chỉ số thống kê của email chẳng hạn như có bao nhiêu lần xuất hiện chữ "h0t chixxx!" …

Một điều quan trọng cần phải cân nhắc đến khi lọc spam là cái giá phải trả khi lọc sai. Nếu một bộ lọc từ chối nhận hầu hết các email gửi đến hoặc đánh dấu một email thật sự quan trọng nào đó là spam thì điều đó còn tệ hơn cả việc nhận tất cả email spam được gửi đến. Ngược lại, nếu có quá nhiều email spam vượt được bộ lọc thì rõ ràng bộ lọc hoạt động không hiệu quả, không đáp ứng được yêu cầu của người sử dụng.

2.2 Các phương pháp tiếp cận:

2.2.1 Complaining to Spammers' ISPs :

- **Ý tưởng :**

Tìm cách làm tăng chi phí gửi spam của các spammer bằng những lời than phiền, phản ánh đến các nơi cung cấp dịch vụ mạng (Internet Service Provider - ISP). Khi chúng ta biết chính xác những email spam thực sự được gửi đến từ dịch vụ ISP nào, ta sẽ phản ánh lại với dịch vụ đó và dịch vụ này sẽ từ chối cung cấp dịch vụ cho các spammer dùng gửi spam.

- **Đặc điểm :**

Đây cũng là giải pháp chống spam đầu tiên. Những lời than phiền cũng có tác dụng của nó. Những nơi gửi spam sẽ bị vô hiệu hóa, khi đó các spammer phải đăng ký một tài khoản mới với nhà cung cấp dịch vụ ISP để có thể tiếp tục phát tán các email spam của mình. Dần dần việc chuyển nơi cung cấp dịch vụ sẽ làm các spammer tốn nhiều chi phí và khi chúng ta phát hiện càng sớm thì chi phí trên của các spammer càng tăng nhiều.

Cách này cũng gặp phải những khó khăn đó là không thể biết chính xác những email spam này thực sự đến từ đâu do các spammer đã khéo léo che giấu đi phần header của email để ẩn đi nguồn gốc. Do đó cần phải hiểu biết về header của email để hiểu rõ email spam này thật sự đến từ đâu.

2.2.2 Mail Blacklists /Whitelists:

- **Ý tưởng:**

Một danh sách đen (Blacklist) các địa chỉ email hay các máy chủ email (mail server) chuyên dùng của các spammer sẽ được thiết

lập và dựa vào đó ta có thể ngăn chặn nhận email spam được phát tán từ những nơi này.

Việc thiết lập danh sách các địa chỉ email đen hay máy chủ gửi email này sẽ do một nhóm tình nguyện xác nhận. Một số nhà cung cấp dịch vụ mạng ISP sẽ dùng danh sách đen kiểu này và tự động từ chối nhận email từ những máy chủ hay email trong dánh sách đó. Như vậy, những email spam sẽ được phân loại và chặn ngay tại máy chủ nhận email.

- **Đặc điểm:**

Phương pháp này bước đầu loại được khoảng 50% [5] email spam.

Khuyết điểm của phương pháp này là chúng không thể đương đầu với hơn một nửa số server mà spam đang sử dụng hiện nay. Và nếu xác nhận sai danh sách đen này thì việc dùng nó đồng nghĩa với việc bỏ qua một lượng lớn email hợp lệ.

Phương pháp này có thể bị qua mặt nếu như các spammer gửi lại email thông qua một máy chủ SMTP (Simple email Transfer Protocol) có nguồn gốc hợp pháp không kể tên trong danh sách “Blacklist”.

Ngoài ra, danh sách này không chỉ từ chối nhận email từ các địa chỉ IP (Internet Protocol) từ những nơi chuyên dùng gửi spam mà nó còn từ chối luôn cả những email mà có tên miền nằm trong danh sách “Blacklist” này.

Cách này được áp dụng tại mức nhà cung cấp dịch vụ mạng (ISP), và thật sự hữu dụng với người dùng nếu họ sử dụng một ISP đáng tin cậy.

Ngược lại với việc thiết lập một danh sách đen "Blacklist" ta còn có thể thiết lập một danh sách "Whitelist". Với những địa chỉ gửi email (hoặc tên miền domains) nằm trong danh sách này sẽ được các ISP tự động chấp nhận email gửi từ nó. Mặc định tất cả những email khác sẽ bị từ chối..

Nếu các spammer gửi email spam với phần "sender" của email có cùng tên miền được chấp nhận trong "Whitelist" thì email spam vẫn có thể đến được tay người nhận.

2.2.3 Mail volume :

- **Ý tưởng:**

Bộ lọc sẽ sử dụng thuật toán để kiểm tra số lượng email nhận được từ một máy chủ (host) cụ thể trong các lần kết nối sau cùng (cách này đã được bộ lọc Spamshield [3] của Kai sử dụng. Nếu số lượng email nhận được lớn hơn một ngưỡng nào đó thì các email đó sẽ được phân loại là spam.

- **Đặc điểm:**

Bộ lọc tỏ ra hiệu quả trong việc phân loại đúng tất cả các email hợp lệ trong điều kiện với một ngưỡng phân loại đủ cao.Nếu bộ lọc được sử dụng cho cá nhân, thì nó hoạt động rất hiệu quả. Có thể xem đây là một ưu điểm của bộ lọc bởi vì với email cá nhân thì những kẻ gửi email quảng cáo phải thiết lập nhiều kết nối hơn để gửi một số lượng email giống nhau. Điều này làm cho các email quảng cáo đó dễ dàng bị phát hiện dựa trên việc phân tích số lượng email.

Mặt hạn chế của bộ lọc này là tỉ lệ chấp nhận phân loại sai FAR (false acceptance rate) của nó còn khá cao. Với:

[3] http://spamshield.conti.nu

$$FAR = \frac{n_{S \to N}}{n_S}$$

$n_{S \to N}$: Số email spam mà bộ lọc nhận là non-spam.

n_S : Số email spam thực sự đến bộ lọc..

2.2.4 Signature/ Checksum schemes:

- **Ý tưởng:**

Đây là một trong những phương pháp phân loại email dựa trên nội dung. Khi một email tới thì giá trị "Signature/ Checksum" sẽ được tính toán cho mỗi email này và so sánh nó với giá trị tính được từ những email spam đặc trưng trong từ những email spam có sẵn trên Internet. Nếu giá trị "signature/ checksum" của những email tới giống với bất kỳ giá trị nào trong cơ sở dữ liệu thì email đó được đánh giá là spam.

Một cách đơn giản để tính giá trị này là gán một giá trị cho mỗi kí tự, sau đó cộng tất cả chúng lại. Sẽ là không bình thường nếu 2 email khác nhau lại có chung một giá trị "signature/ checksum".

- **Đặc điểm:**

Cách tấn công một bộ lọc kiểu này là thêm vào ngẫu nhiên một vài ký tự hay một câu vô nghĩa trong mỗi email spam để tạo ra sự khác biệt của giá trị "signature". Khi bạn thấy những thứ hỗn tạp chèn ngẫu nhiên trong phần tiêu đề (subject) của email, đó chính là cách để tấn công bộ lọc dựa vào "signature/ checksum".

Các spammer dễ dàng đối phó đối với các bộ lọc dựa trên "signature/ checksum" bằng phương pháp trên. Khi mà những người viết các chương trình lọc email tìm được cách chống lại cách chèn

ngẫu nhiên này thì các spammer lại chuyển sang cách khác. Vì thế, cách chống spam dùng các bộ lọc "signature/checksum" chưa bao giờ là một cách tốt.

Bộ lọc này được ứng dụng tại mức server, được các nhà cung cấp dịch vụ mạng (ISP) sử dụng.

Theo P.Graham [5], bộ lọc kiểu này chỉ lọc khoảng 50-70% spam

Ưu điểm của bộ lọc này là ít khi phân loại sai email non-spam.

Brightmail[4] là phần mềm chống spam dựa trên hướng tiếp cận này. Cách hoạt động của nó là tạo ra một mạng lưới các địa chỉ email giả. Bất kì email nào được gửi đến những địa chỉ này thì đều là spam vì với những email hợp lệ thì hiếm khi lại được gửi đến những địa chỉ giả này. Vì vậy, khi bộ lọc nhận thấy những email giống nhau gửi đến một địa chỉ giả đã được tạo ra này thì nó sẽ lọc ra.. Bộ lọc phân biệt những email giống nhau dựa vào "signatures" của chúng.

2.2.5 Genetic Algorithms:

- **Ý tưởng:**

Bộ lọc dựa trên thuật toán di truyền (Genetic Algorithms) sử dụng các bộ nhận dạng đặc trưng ("fearture detectors") để ghi điểm (score) cho mỗi email. Thực tế, những "fearture detectors" này là một tập các luật được xây dựng dựa trên các kinh nghiệm đã có (empirical rules) và áp dụng vào mỗi email để thu về một giá trị số.

Thuật toán di truyền này được biểu diễn là những cây (*trees*) và được kết hợp với một tập huấn luyện cùng với một hàm thích hợp "fitness function".

[4]http://brightmail.com

Cơ chế tiến hóa (*Evolutionary mechanism*) của thuật toán :thuật tóan thực hiện hai thao tác cơ bản là phép lai "crossover" và đột biến "mutation". Mục đích tiến trình này là tìm ra được một giá trị "score" nhỏ nhất dựa vào hàm "fitness function". Giá trị "score" sau đó sẽ được sử dụng để phân loại email là spam hay non-spam.[6]

- **Đặc điểm:**

Đây là hướng tiếp cận phân loại email dựa trên nội dung.

Hướng tiếp cận hiệu quả nhất cho bộ lọc tại mức ISP được đánh giá là dựa trên thuật toán di truyền "Genetic Algorithms" [6]

Điểm không thuận lợi của thuật toán di truyền là đòi hỏi khả năng xử lý phải lớn.

Hướng tiếp cận này được ứng dụng trong trình lọc spam Spamassassin[5]. Nó hoạt động rất hiệu quả tại mức ISP và được nhiều người đánh giá là một trong những bộ lọc hoạt động hiệu quả nhất tại mức ISP.

Điểm yếu của trình lọc "Spamassassin" là hoạt động với hiệu quả chưa cao tại mức người dùng cá nhân.

2.2.6 Rule-Based (hay là Heuristic):

- **Ý tưởng:**

Dựa vào luật tìm kiếm các mẫu có dấu hiệu là spam như các từ và ngữ xác định, hàng loạt các chữ hoa và dấu chấm than, phần header của email sai định dạng, ngày trong email là ở tương lai hoặc quá khứ.Đó là cách hầu hết phần lớn các trình lọc spam hoạt động từ năm 2002.

- **Đặc điểm:**

[5] http://spamassassin.org

Hiệu suất của trình lọc dựa trên luật (rule-based filters) khác nhau rất nhiều. Cách đơn giản nhất là loại bỏ các email mà có chứa những từ xấu nào đó (ví dụ những từ mà thường xuất hiện nhiều hay chỉ xuất hiện trong spam). Nhưng đây cũng là điểm yếu để các spammer có thể lợi dụng để qua mặt các bộ lọc kiểu này bằng cách cố gắng tránh sử dụng những từ xấu và thay bằng những từ "tốt" - được sử dụng nhiều trong email non-spam. Trong khi đó các email non-spam thì bị loại bỏ nếu vô tình chứa một vài từ "xấu" dạng này. Điều này, dẫn đến khả năng lọc sai còn cao.

Một điều bất lợi khác là các luật dạng này đều là tĩnh. Khi các spammer tìm ra được một phương pháp mới để vượt qua thì những người viết trình lọc lại phải viết những luật mới để lọc các spam. Những spammer chuyên nghiệp thì có thể kiểm tra được những email trên các hệ thống lọc dựa trên luật trước khi gửi chúng đi.

Nếu bộ lọc được xây dựng dựa trên luật phức tạp thì vẫn phát huy tác dụng lọc spam hiệu quả. Ví dụ như trình lọc Spamassassin lọc lên đến 90-95% spam.

Một điều thuận lợi là bộ lọc dựa trên luật tĩnh thì dễ cài đặt.

2.2.7 Challenge-Response:

- **Ý tưởng:**

Khi bạn nhận được email từ ai đó mà chưa hề gửi cho bạn trước đó thì hệ thống lọc *challenge-response* [6] gửi ngược lại 1 email yêu cầu họ phải đến 1 trang web và điền đầy đủ thông tin vào form trước khi email chuyển cho người dùng.

- **Đặc điểm:**

[6] http://spamarrest.com/products

Lợi thế của hệ thống này là để lọt lưới rất ít spam. Điều bất lợi của nó can thiệp thô bạo đến người gửi. Bằng cách sử dụng hệ thống này, ta cần xác định rõ ai là người gửi email.

Một điểm bất lợi khác của hệ thống này là có nhiều email non-spam bị loại bỏ và thời gian trì hoãn quá lâu. Ví dụ như một người muốn mời bạn đi dự tiệc nhưng người bạn ấy sẽ chỉ thấy email trả lời của bạn vào ngày hôm sau và đến lúc đó thì đã quá trễ.

Nhiều trường hợp người gửi sẽ không trả lời cho các thông điệp kiểu này và email họ gởi sẽ bị thất lạc.

Sử dụng phương pháp dạng này chẳng khác nào ta đang tự cô lập chính mình với mọi người xung quanh. Hệ thống này sẽ giống như bức tường bao quanh thế giới luôn muốn gửi thông điệp cho ta.

2.2.8 Machine Learning (Máy học):

- **Ý tưởng:**

Áp dụng các phương pháp máy học trong các bài toán phân loại, đặc biệt là phân loại văn bản vào bài toán phân loại email, các thuật toán máy học như Naïve Bayesian [9],[17],[18] AdaBoost [13], Suppor Vector Machine[18],.., đã được sử dụng trong lĩnh vực phân loại văn bản, nhận dạng, …với hiệu quả cao. Ý tưởng là tìm cách xây dựng một bộ phân loại nhằm phân loại cho một mẫu mới bằng cách huấn luyện những mẫu đã có sẵn.

- **Đặc điểm**

Phương pháp này có thể áp dụng ở mức Server hay Client.

Hạn chế là cần phải có một kho ngữ liệu (corpus) huấn luyện ban đầu để cho máy học, việc huấn luyện mất nhiều thời gian. Một hạn chế nữa là hiệu quả phân loại phụ thuộc vào kho ngữ liệu dùng để huấn luyện.

2.3 Phương pháp lựa chọn :

Trong những hướng tiếp cận đã tìm hiểu, chúng tôi chọn hướng tiếp cận phân loại email bằng phương pháp máy học, phương pháp này có hiệu quả cao, đồng thời cũng rất khó bị các spammer vượt qua. Ngoài ra, hướng tiếp cận này có thể áp dụng được ở mức Client

Cụ thể hướng tiếp cận mà nhóm chúng tôi tìm hiểu và thử nghiệm là phân loại email dựa trên thuật toán huấn luyện Naïve Bayes và Adaboost, hai phương pháp này có một số ưu điểm sau:

- Hiệu quả phân loại trong các lĩnh phân loại văn bản, nhận dạng đã được kiểm chứng và khá cao
- Thích hợp cho từng người dùng cụ thể và ở mức Client
- Có khả năng tự học để phân loại đúng.
- Hướng tiếp cận còn khá mới.

2.4 Các chỉ số đánh giá hiệu quả phân loại email :

2.4.1 Spam Recall và Spam Precision:

Để tiện lợi cho việc so sánh, người ta đưa ra hai chỉ số đánh giá là spam recall và spam precision.

Spam recall là tỉ lệ phần trăm giữa số email – được bộ lọc coi là spam - bị chặn lại và tổng số email spam (thực sự) đến bộ lọc

Spam Precision là tỉ lệ phần trăm giữa số email bị chặn thực sự là spam với số email bị chặn - được bộ lọc coi là spam, spam precision đánh giá mức độ an toàn của bộ lọc.

Công thức tính Spam Recall (SR) và Spam Precision(SP) như sau:

$$SR = \frac{n_{S->S}}{n_{S->S} + n_{S->N}}$$

Công thức 2-1 :Công thức tính Spam Recall

$$SP = \frac{n_{S->S}}{n_{S->S} + n_{N->S}}$$

Công thức 2-2 : Công thức tính Spam Precesion

Với :

- $n_{S->S}$ là số email là spam mà bộ lọc nhận ra là spam
- $n_{S->N}$ là số email là spam mà bộ lọc nhận ra là email non-spam
- $n_{N->S}$ là số email non-spam mà bộ lọc nhận ra là spam

2.4.2 Tỉ lệ lỗi Err (Error) và tỉ lệ chính xác Acc(Accuracy) :

Trong việc phân loại email, hiệu quả phân loại dựa vào tỉ lệ chính xác (Acc) hoặc tỉ lệ lỗi (Err). Công thức tính tỉ lệ chính xác và tỉ lệ lỗi như sau :

$$Acc = \frac{n_{N->N} + n_{S->S}}{N_N + N_S}$$

Công thức 2-3 :công thức tính tỉ lệ chính xác

$$Err = \frac{n_{N->S} + n_{S->N}}{N_N + N_S}$$

Công thức 2-4 : công thức tính tỉ lệ lỗi

Với

- N_N và N_S là số email non-spam và số email spam cần phân loại
- $n_{N->N}$ là số email là non-spam và được bộ lọc nhận ra là non- spam
- $n_{N->S}$ là số email là non-spam mà bộ lọc nhận ra là spam
- $n_{S->S}$ là số email là spam mà được bộ lọc nhận ra là spam
- $n_{S->N}$ là số email là spam mà được bộ lọc nhận ra là non-spam

2.4.3 Tỉ lệ lỗi gia trọng WErr (Weighted Error) và tỉ lệ chính xác gia trọng (Weighted Accuracy):

Trong phân loại email có hai loại lỗi : lỗi nhận spam ra non-spam (false negative) và lỗi nhận non-spam ra spam(false positive) [3]. Lỗi thứ hai là lỗi

nghiêm trọng hơn, bởi người dùng có thể chấp nhận một email spam vượt qua bộ lọc nhưng khó mà chấp nhận một email hợp lệ lại bị bộ lọc chặn lại. Để biểu thị tác động của hai loại lỗi này đối với tỉ lệ chính xác và tỉ lệ lỗi, ta sẽ xem mỗi một email hợp lệ như là λ email hợp lệ. Do đó khi một email hợp lệ bị phân loại sai, thay vì xem như có một lỗi, ta xem như là λ lỗi, và khi phân loại đúng ta xem như là λ lần thành công. Ta có hai tỉ lệ : tỉ lệ chính xác gia trọng *WAcc* (Weighted Accuracy Rate) và tỉ lệ lỗi gia trọng *WErr* (Weighted Error Rate) (*WErr*=1 -*WAcc*).

$$WAcc = \frac{\lambda n_{N->N} + n_{S->S}}{\lambda N_N + N_S}$$

Công thức 2-5 Tỉ lệ chính xác gia trọng

$$WErr = \frac{\lambda n_{N->S} + n_{S->N}}{\lambda N_N + N_S}$$

Công thức 2-6 Tỉ lệ lỗi gia trọng

Với :

- N_N và N_S là số email non-spam và số email spam cần phân loại
- $n_{N->N}$ là số email là non-spam và được bộ lọc nhận ra là non- spam
- $n_{N->S}$ là số email là non-spam mà bộ lọc nhận ra là spam
- $n_{S->S}$ là số email là spam mà được bộ lọc nhận ra là spam
- $n_{S->N}$ là số email là spam mà được bộ lọc nhận ra là non-spam

2.4.4 Tỉ số chi phí tổng hợp TCR (Total Cost Ratio):

Giá trị của tỉ lệ chính xác và tỉ lệ lỗi thường có sự sai lệch cao. Để thấy rõ được hiệu quả của cách phân loại, người ta thường so sánh tỉ lệ chính xác hoặc tỉ lệ lỗi giữa bộ phân loại với trường hợp đơn giản nhất và được xem là trường hợp "ranh giới "(baseline)."Ranh giới" được chọn là trường hợp không sử dụng một bộ lọc nào, các email hợp lệ không bao giờ bị chặn lại và các email

là spam thì luôn luôn đi qua. Như vậy tỉ lệ chính xác gia trọng và tỉ lệ lỗi gia trọng của trường hợp “ranh giới “ là :

$$WAcc^b = \frac{\lambda N_N}{\lambda N_N + N_S}$$

Công thức 2-7: Tỉ lệ chính xác gia trọng của trường hợp "Ranh giới "

$$WErr^b = \frac{N_S}{\lambda N_N + N_S}$$

Công thức 2-8: Tỉ lệ lỗi gia trọng của trường hợp "Ranh giới "

Với :

N_N, N_S, $n_{N->N}$, $n_{N->S}$, $n_{S->S}$, $n_{S->N}$ có cùng ý nghĩa như ở mục 2.4.1 và 2.4.2

Tỉ số chi phí toàn bộ TCR (total cost ratio) cho phép ta so sánh được hiệu quả của trường hợp sử dụng bộ lọc so với trường hợp “ranh giới”:

$$TCR = \frac{WErr^b}{WErr} = \frac{N_S}{\lambda n_{N->S} + n_{S->N}}$$

Công thức 2-9 Công thức tính tỉ số chi phí tổng hợp

Giá trị TCR càng lớn thì hiệu quả phân loại càng cao, với TCR nhỏ hơn 1 thì rõ ràng không sử dụng bộ lọc còn tốt hơn.

.

Chương 3 : GIỚI THIỆU CÁC KHO NGỮ LIỆU DÙNG KIỂM THỬ PHÂN LOẠI EMAIL

3.1 Kho ngữ liệu PU (corpus PU):

3.1.1 Vài nét về kho ngữ liệu PU:

Các nghiên cứu về phân loại văn bản có nhiều thuận lợi vì có sẵn các kho ngữ liệu công cộng để dùng chung, tuy nhiên sử dụng những kho ngữ liệu này vào việc lọc spam lại gặp phải rắc rối bởi vấn đề tính riêng tư, cá nhân. Những email spam thì không có vấn đề gì, tuy nhiên không thể sử dụng những email hợp lệ mà không thể không vi phạm đến sự riêng tư của người gởi và người nhận của những email này.

Chúng tôi sử dụng kho ngữ liệu PU để học và kiểm thử[7]. PU là một kho ngữ liệu email chuẩn, gồm có bốn kho ngữ liệu nhỏ hơn bao gồm PU1, PU2, PU3 và PUA. Mỗi một token sẽ được thay thế tương ứng bằng một con số duy nhất như minh họa trong hình 3-1.

Subject: unlimited cash and millions in your mail	Subject: 40 9 7 24 18 45 21
dear friend ,	10 13 1
this is one of the great money making programs	38 20 29 28 36 15 25 23 34
over the net and the only one i have earned	31 36 27 7 36 30 29 17 16 12
money with . the advantage of this program is	25 42 2 36 5 28 38 33 20
that all the money is directed to your mailing	35 6 36 25 20 11 39 45 22
address without intrusion of a third party . [...]	4 43 19 28 3 37 32 2 [...]
my best wishes to you , george	26 8 41 39 44 1 14

Hình 3-1Email sau khi tách token và mã hoá (trong kho ngữ liệu pu)

Hàm ánh xạ từ văn bản sang các con số không được công bố, do đó việc khôi phục lại văn bản ban đầu là cực kỳ khó, điều này đảm bảo được tính bí mật, riêng tư của người gởi và người nhận. Những email giống nhau cũng được xem xét. Trong kho ngữ liệu PU1 và PU2, những email giống nhau và nhận trong cùng một ngày được xóa thủ công.Trong kho ngữ liệu PU3 và PUA quá trình này được thực hiện tự động, ở hai kho ngữ liệu này, khái niệm khác nhau của hai email được xem xét như sau :hai email được xem là khác nhau nếu chúng có ít nhất 5 dòng khác nhau.Tất cả những email giống nhau, bất kể ngày nhận, đều

[7] Để lấy cơ sở dữ liệu PU, vào trang web Internet CONtent Filtering Group, http://www.iit.demokritos.gr/skel/i-config/

bị xóa đi, chỉ giữ lại một email mà thôi.Cơ chế này được áp dụng cho cả email spam và email non-spam. Theo [18], trong quá trình tạo kho ngữ liệu PU, một vấn đề phát sinh đó là có một lượng lớn email là của những người gởi thường xuyên liên lạc với người tạo kho ngữ liệu - những email RC (Relative Correspondence), những email này cũng được loại bỏ.

3.1.2 Mô tả cấu trúc kho ngữ liệu PU:

Những email hợp lệ trong PU1 là những email hợp lệ người tạo đã nhận được trong vòng 36 tháng cho đến tháng 12 năm 2003, gồm có 1182 email. Những email hợp lệ không có nội dung và những email RC sẽ bị loại bỏ, kết quả là có 618 email hợp lệ. Những email spam trong PU1 là email spam người tạo đã nhận được trong khoảng thời gian 22 tháng cho đến thời điểm 12-2003, bao gồm những email không phải là email tiếng Anh và những email giống nhau nhận trong một ngày.

PU2 cũng tương tự như PU1, điểm khác nhau ở đây là những email RC

Ở PU3 và PUA,những email hợp lệ không phải là tiếng Anh vẫn được giữ lại

Tỉ lệ non-spam :spam của PU3 xấp xỉ PU1, tuy nhiên số lượng của PU3 nhiều gấp 4 lần PU1, trong PU2 tỉ lệ đó xấp xỉ 4:1, ở PUA tỉ lệ đó là 1:1

Trong tất cả các kho ngữ liệu PU, các tập tin đính kèm, các thẻ HTML, các trường khác trong header của email đều bị loại bỏ (ngoại trừ trường tiêu đề (subject). Các dấu chấm câu, các kí tự đặc biệt khác (!,$) cũng được xem xét .

Tên	Email hợp lệ ban đầu	Email RC	Email hợp lệ khác bị xóa	Email hợp lệ còn lại	Email spam	Tổng số email giữ lại	Tỉ lệ non-spam:spam
Pu1	1182	564		618	481	1099	1.28
Pu2	6207	5628		579	142	721	4.01
Pu3	8824	6253	258	2313	1826	4139	1.27
Pua	980	369	40	571	571	1142	1

Bảng 3-1Mô tả cấu trúc kho ngữ liệu PU

Mỗi kho ngữ liệu pu lại được chia ra làm 11 thư mục từ part 1 đến part 10, và một thư mục unused, mỗi thư mục từ part 1 đến part 10 chứa số lượng email như nhau và số lượng email spam và email hợp lệ trong mỗi thư mục part i (i=1,…,10) trên là như nhau, thư mục unused chứa những email không sử dụng. Chúng tôi sử dụng từ part 1 đến part 9 để phục vụ cho việc học. Đối với việc kiểm thử kết quả , chúng tôi sử dụng kho ngữ liệu đã được học (từ part 1 đến part 9) và kho ngữ liệu chưa được học để kiểm thử. Để thực hiện việc kiểm thử các thuật toán được tiện lợi, chúng tôi tiến hành chia nhóm kho ngữ liệu học.Với mỗi kho ngữ liệu PU, chúng tôi phân loại email thành hai thư mục, một thư mục chứa các email spam từ part 1 đến part 9, thư mục còn lại chứa email hợp lệ từ part 1 đến part 9, với part 10 chúng tôi cũng tiến hành phân loại tương tự như trên

3.2 Kho ngữ liệu email chữ:

Để tạo kho ngữ liệu email là chữ, chúng tôi lấy dữ liệu tại trang : Index of /publiccorpus http://spamassassin.apache.org/publiccorpus/. Ngữ liệu gồm những email được thu thập trong các năm 2002 và 2003, số lượng email spam 2398 là, số lượng email 6951

Chúng tôi tiến hành xử lý và phân loại email : loại bỏ những email có tập tin đính kèm, phân loại email html và email văn bản trơn (text/plain).

Số email spam là văn bản trơn sau khi đã xử lý khỏang 600 email, email non-spam là văn bản trơn sau khi đã xử lý là khoảng 2500 mail

Số email non-spam là email html sau khi đã xử lý là gần 200 mail, số email spam là email html sau khi đã xử lý khoảng 1000 mail. Sau đó chúng tôi tạo thành hai kho ngữ liệu email văn bản trơn (text/plain) và email html.

Việc tạo kho ngữ liệu email văn bản trơn (text/plain) thực hiện bằng cách chọn ngẫu nhiên các email từ kho ngữ liệu sau khi đã qua xử lý, số email spam dùng huấn luyện là 517, số lượng email spam để kiểm thử là 98. Với ngữ liệu email non-spam là văn bản trơn (text/plain) số lượng dùng huấn luyện là 528, số lượng dùng để kiểm thử là 100

Để tạo kho ngữ liệu email html, chúng tôi cũng xây dựng tương tự như trên. Với ngữ liệu email non-spam là html, chúng tôi dùng 141 email để huấn luyện, 50 email dùng để kiểm thử. Còn ngữ liệu emal spam là html, chúng tôi dùng 205 email để huấn luyện và 50 email để kiểm thử.

Chương 4 : PHƯƠNG PHÁP PHÂN LOẠI NAÏVE BAYESIAN VÀ ỨNG DỤNG PHÂN LOẠI EMAIL

4.1 Một vài khái niệm xác suất có liên quan

4.1.1 Định nghĩa biến cố, xác suất :

4.1.1.1 Khái niệm phép thử và biến cố:

Gieo một đồng tiền trên một mặt phẳng :đó là một phép thử

Kết quả có thể xảy ra khi gieo đồng tiền : "Xuất hiện mặt sấp" hoặc "Xuất hiện mặt ngữa"

"Xuất hiên mặt sấp" -Đó là một biến cố

"Xuất hiện mặt ngữa" -Đó là một biến cố

4.1.1.2 Định nghĩa xác suất:

Theo [8] có những định nghĩa xác suất sau:

Dạng cổ điển :

Xác suất của biến cố A là một số không âm,ký hiệu P(A), biểu thị khả năng xảy ra biến cố A và được xác định như sau :

$P(A) = \frac{m}{n} =$ Số trường hợp thuận lợi cho A / Số trường hợp có thể có khi phép thử thực hiện

(Những khả năng hoặc các biến cố sơ cấp – nếu chúng xảy ra thì suy ra A xảy ra – gọi là những trường hợp thuận lợi cho A).

Định nghĩa xác suất theo phương pháp thống kê :

Làm đi làm lại một phép thử nào đó n lần mà có m lần biến cố A xuất hiện thì tỷ số m/n gọi là tần suất của biến cố A

Khi n thay đổi,tần suất m/n cũng thay đổi nhưng nó luôn dao động quanh một số cố định đó. Số cố định ấy được gọi là xác suất của biến cố A theo nghĩa thống kê. Trên thực tế khi n đủ lớn ta xấp xỉ P(A) bởi m/n

4.1.2 Xác suất có điều kiện, công thức xác suất đầy đủ – công thức xác suất Bayes

4.1.2.1 Xác suất có điều kiện

Theo Đặng Hấn [8]:

Xác suất có điều kiện của biến cố A với điều kiện biến cố B đã xảy ra là một con số không âm, được ký hiệu P(A/B) nó biểu thị khả năng xảy ra biến cố A trong tình huống biến cố B đã xảy ra

$$P(A \mid B) = \frac{P(AB)}{P(B)}$$

Công thức 4-1: công thức tính xác suất có điều kiện

Suy ra:

$$P(A \mid B) \times P(B) = P(B \mid A) \times P(A) = P(AB)$$

Công thức 4-2

4.1.2.2 Công thức xác suất đầy đủ:

Giả sử $B_1, B_2, B_3, ..., B_n$ là một nhóm đầy đủ các biến cố. Xét biến cố A sao cho A xảy ra chỉ khi một trong các biến cố $B_1, B_2, B_3, ..., B_n$ xảy ra.

Khi đó :

$$P(A) = \sum_{i=1}^{n} P(B_i).P(A/B_i)$$

Công thức 4-3 :công thức xác suất đầy đủ

Công thức trên được gọi là công thức xác suất đầy đủ

4.1.2.3 Công thức xác suất Bayes:

Từ các công thức:Công thức 4-1, Công thức 4-2 và Công thức 4-3, ta có:

$$P(B_k \mid A) = \frac{P(AB_k)}{P(A)} = \frac{P(B_k).P(A/B_k)}{\sum_{i=1}^{n} P(B_i).P(A/B_i)}$$

Công thức 4-4 : công thức xác suất Bayes

4.2 Phương pháp phân loại Naïve Bayesian :

Phân loại Bayesian là phương pháp phân loại sử dụng tri thức các xác suất đã qua huấn luyện. Phương pháp này thích hợp với những lớp bài toán đòi hỏi phải dự đoán chính xác lớp của mẫu cần kiểm tra dựa trên những thông tin từ tập huấn luyện ban đầu [16].

Theo Charles Elkan [16] cho $X_1,...,X_n$ là các thuộc tính với các giá trị rời rạc được dùng để dự đoán một lớp riêng biệt C cho một mẫu, tập các lớp mà mẫu có thể thuộc về là $C=\{c_1,c_2,...,c_m\}$. Cho một mẫu huấn luyện với giá trị các thuộc tính tương ứng là $x_1,...,x_n$, dự đoán mẫu thuộc về lớp $c\in C$ khi xác suất $P(C=c \mid X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n)$ có giá trị lớn nhất. Sử dụng công thức xác suất Bayes ta có :

$$P(C=c \mid X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n)=\frac{P(X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n \mid C=c)}{P(X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n)}P(C=c)$$

Xác suất $P(C=c)$ được tính dễ dàng từ tập dữ liệu huấn luyện. Xác suất $P(X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n)$ không thích hợp để dùng cho việc quyết định lớp của C bởi vì giá trị này như nhau đối với mỗi lớp *c*. Như vậy căn cứ để dự đoán lớp của C là dựa vào xác suất $P(X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n \mid C=c)$.Tuy nhiên việc tính toán xác suất này rất phức tạp [9] . Một phương pháp đơn giản và được đưa ra sớm nhất là phương pháp phân loại Naïve Bayesian, theo đó giả thiết rằng mỗi X_i độc lập với các X_j $(i \neq j)$, như vậy ta sẽ có:

$$P(X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n \mid C=c)=\prod_{i=1}^{n} P(X_i=x_i \mid C=c)$$

Thật vậy, sử dụng công thức xác suất Bayes ta có :

$$P(X_1=x_1 \wedge X_2=x_2 \wedge ... \wedge X_n=x_n \mid C=c)$$
$$=P(X_1=x_1 \mid X_2=x_2 \wedge ... \wedge X_n=x_n, C=c)\,P(X_2=x_2 \wedge ... \wedge X_n=x_n \mid C=c)$$

Bằng cách đệ qui, viết thừa số thứ hai trong tích trên như sau :

$P(X_2 = x_2 \wedge ... \wedge X_n = x_n \mid C = c) =$
$P(X_2 = x_2 \mid X_3 = x_3 \wedge ... \wedge X_n = x_n, C = c) P(X_3 = x_3 \wedge ... \wedge X_n = x_n \mid C = c)$ và cứ tiếp tục như vậy. Phương pháp phân loại Naïve Bayesian giả thiết rằng với mỗi X_i kết quả tác động của nó là độc lập với các X_j khác, như vậy chúng ta thừa nhận rằng:

$P(X_1 = x_1 \mid X_2 = x_2 \wedge ... \wedge X_n = x_n, C = c) = P(X_1 = x_1 \mid C = c)$ và tương tự như vậy đối với $X_2, .., X_n$.

Như vậy xác suất $P(X_1 = x_1 \wedge X_2 = x_2 \wedge ... \wedge X_n = x_n \mid C = c) =$

$P(X_1 = x_1 \mid C = c) P(X_2 = x_2 \mid C = c) ... P(X_n = x_n \mid C = c) = \prod_i^n P(X_i = x_i \mid C = c)$

Mỗi một thừa số trong tích trên có thể được tính dễ dàng từ tập huấn luyện ban đầu, như vậy phương pháp Naïve Bayesian giảm sự phức tạp của việc tính toán giá trị xác suất $P(X_1 = x_1 \wedge X_2 = x_2 \wedge ... \wedge X_n = x_n \mid C = c)$

4.3 Phân loại email bằng phương pháp Naïve Bayesian :

Ở đây mỗi mẫu mà ta xét chính là mỗi một email, tập các lớp mà mỗi email có thể thuộc về là C ={spam, non-spam}

Khi ta nhận được một email, nếu ta không biết một thông tin gì về nó, do đó khó có thể quyết định chính xác email này là spam hay không .

Nếu như ta có thêm đặc điểm hay thuộc tính nào đó của email thì ta có thể nâng cao hiệu quả nhận được email là spam Một email có nhiều đặc điểm như : tiêu đề, nội dung, có đính kèm tập tin hay không,…Ta có thể dựa vào các thông tin này để nâng cao hiệu quả phân loại email spam. Một ví dụ đơn giản : nếu ta biết được rằng 95 % email html là email spam, và ta lại nhận được một email html, như vậy có thể dựa vào xác suất biết trước 95% email html là email spam để tính được xác suất email mà ta nhận được là spam, nếu xác suất này lớn hơn xác suất email đó là non-spam, có thể kết

luận rằng email đó là spam, tuy nhiên kết luận này không chính xác lắm Nhưng nếu ta có được nhiều xác suất biết trước như vậy, thì kết luận sẽ trở nên đáng tin cậy hơn. Để có được các xác suất biết trước này, sử dụng phương pháp Naïve Bayesian huấn luyện tập mẫu (email) ban đầu, sau đó sẽ sử dụng các xác suất này ứng dụng vào phân loại một mẫu (email) mới.

4.3.1 Phân loại email dựa trên thuật toán Naïve Bayesian

Giả thiết mỗi một email được đại diện bởi một vector thuộc tính đặc trưng $x=(x_1,x_2,...,x_n)$ với $x_1,x_2,...,x_n$, là giá trị của các thuộc tính $X_1,X_2,..,X_n$ tương ứng trong không gian vector đặc trưng X. Theo M Sahami et al [9] ta sử dụng các giá trị nhị phân, X_i=1 nếu các đặc điểm của X_i có trong email, ngược lại X_i=0.

Ta tính giá trị tương hỗ MI (X,C) (Mutual Information) mà mỗi một đại diện của X thuộc về loại C như sau:

$$MI(X,C)=\sum_{x\in\{0,1\}} P(X=x,C=c).\log\frac{P(X=x,C=c)}{P(X=x)P(C=c)}$$

$$c\in\{spam,non-spam\}$$

Công thức 4-5 :công thức tính độ tương hỗ MI

Sau đó ta chọn các thuộc tính có giá trị MI cao nhất.Các xác suất *P(X), P(C), P(X,C)* được tính dựa trên dữ liệu học

Dựa vào công thức xác suất Bayes và công thức xác suất đầy đủ ta có được xác suất một email với vector đặc trưng x thuộc về loại c là:

$$P(C=c\mid X=x)=\frac{P(C=c).P(X=x\mid C=c)}{\sum_{k\in\{spam,non-spam\}} P(C=k).P(X=x\mid C=k)}$$

Với *C* là e email được xét, $c\in\{spam,nonspam\}$

Công thức 4-6

Thực tế thì rất khó tính được xác suất $P(X \mid C)$ bởi vì giá trị số lượng của các vector rất nhiều và nhiều vector hiếm khi hay thậm chí không xuất hiện trong tập dữ liệu huấn luyện.Như đã nói, phương pháp Naïve Bayesian giả thiết rằng $X_1, X_2, .., X_n$ là những biến cố độc lập, do đó chúng ta có thể tính được xác suất ở trên như sau:

$$P(C = c \mid X = x) = \frac{P(C = c).\prod_{i=1}^{n} P(X_i = x_i \mid C = c)}{\sum_{k \in \{spam, non-spam\}} P(C = k).\prod_{i=1}^{n} P(X_i = x_i \mid C = k)}$$

Công thức 4-7

Với $P(X_i \mid C)$ và $P(C)$ được tính dựa trên dữ liệu học, việc tính này dựa vào tập huấn luyện ban đầu.

Từ xác suất này, ta so sánh với một giá trị ngưỡng *t* (trình bày ở mục) mà ta cho là ngưỡng để phân loại email spam hay không, nếu xác suất này lớn hơn *t*, ta cho là email đó là spam, ngược lại ta xem email đó là non-spam.

4.3.2 Chọn ngưỡng phân loại email :

Trong phân loại email, có hai loại sai lầm : sai lầm nhận một email là spam mặc dù thực tế nó là non-spam (false positive) và sai lầm thứ hai là nhận một email là non-spam mặc dù nó là spam (false negative). Rõ ràng là sai lầm thứ nhất là nghiêm trọng hơn bởi vì người sử dụng có thể chấp nhận một email spam vượt qua bộ lọc nhưng không chấp nhận một email hợp lệ quan trọng lại bị bộ lọc chặn lại.

Giả sử $N \rightarrow S$ và $S \rightarrow N$ tương ứng với hai lỗi sai trên đây Sử dụng luật quyết định Bayes dựa trên chi phí [9], ta giả sử rằng lỗi $N \rightarrow S$ có chi phí gấp λ lần lỗi $S \rightarrow N$, chúng ta phân loại một email là spam dựa vào tiêu chuẩn sau:

$$\frac{P(C = spam) \mid \bar{X} = x)}{P(C = non - spam \mid X = x)} > \lambda$$

Công thức 4-8

Mà $P(C = spam \mid \bar{X} = x) = 1 - P(C = non - spam \mid \bar{X} = x)$

Nên ta có:

$$P(C = spam \mid \bar{X} = x) > t \text{ với } t = \frac{\lambda}{\lambda + 1} \text{ và } \lambda = \frac{t}{1 - t}$$

Như vậy ngưỡng phân loại được chọn là *t* tùy thuộc vào giá trị λ

Chương 5 : THỰC HIỆN VÀ KIỂM THỬ PHÂN LOẠI EMAIL DỰA TRÊN PHƯƠNG PHÁP PHÂN LOẠI NAÏVE BAYESIAN

5.1 Cài đặt chương trình phân loại email dựa trên phương pháp phân loại Naïve Bayesian:

5.1.1 Khái niệm "Token" :

Để xem xét nội dung email chúng tôi dùng khái niệm "token"

Các "token" có thể xem như là các từ cần xem xét mà ta tách ra từ nội dung của email. Với các kí tự chữ, kí tự số, kí tự '$', kí tự gạch ngang '-', kí tự gạch dưới '_', kí tự nháy đơn '' là những kí tự cấu tạo thành token. Còn những kí tự còn lại như khoảng trắng, kí tự '*', kí tự ':', … được xem là kí tự để tách từ hay phân cách các từ. Với những từ tách được mà gồm toàn kí số thì không được xem là token (ví dụ: "12345").

Ví dụ ta có các token sau:

"qvp0045", " indira", "mx-05", "$7500", "3d0725", " platinum".

Nếu ta có một chuỗi sau: "http://www.27meg.com/foo" thì ta sẽ có các token tương ứng là: "http", "www", "27meg", "com", "foo".

5.1.2 Vector thuộc tính :

Như đã nói ở mục 4.3.1, ta chuyển mỗi một email sang một vector $x=(x_1, x_2, .., x_n)$ với $x_1, x_2, .., x_n$ là giá trị các thuộc tính $X_1, X_2, .., X_n$ trong không gian vector đặc trưng X. Các thuộc tính có thể là một token , nhóm các token …Trong trường hợp đơn giản nhất, mỗi một thuộc tính được thể hiện bởi một token đơn và tất cả các thuộc tính có giá trị luận lý (Boolean), như vậy X_i=1 nếu email chứa token, trường hợp ngược lại X_i=0.

Chúng tôi chọn thuộc tính là token đơn, nhưng thay vì giá trị của các thuộc tính là giá trị luận lý (boolean), chúng tôi chọn là xác suất spam của mỗi token. Xác suất spam của mỗi token sẽ có giá trị trong đoạn [0, 1].Xác suất cho ta nhiều thông tin hơn so với giá trị luận lý.Ví dụ : xét

token “$” xuất hiện trong email, nếu ta sử dụng giá trị luận lý, ta không đủ cơ sở để nghi ngờ email này là email spam, và nếu email này khá dài thì càng khó kết luận rằng nó là spam. Tuy nhiên sử dụng xác suất, ta có thể biết được khả năng email đó là spam là bao nhiêu, điều này hợp lý hơn là chỉ sử dụng hai giá trị 0 và 1.Với không gian vector đặc trưng X, chúng tôi chọn n là số các thuộc tính của X để thử nghiệm lần lượt là 10, 15 và 20. Chọn n sao cho không lớn quá, nếu n lớn có khả năng những thuộc tính không phải là đặc trưng, như vậy sẽ làm “nhiễu “ khả năng phân loại đúng.Ngược lại nếu chọn n quá nhỏ, ta sẽ không có được số cần thiết các thuộc tính.

5.1.3 Chọn ngưỡng phân loại :

Chúng tôi tiến hành thử nghiệm với giá trị λ lần lượt là 1, 9 và 999, như vậy ngưỡng phân loại t xác định một email là spam lần lượt là 0.5, 0.9, 0.999.

5.1.4 Cách thực hiện :

Chúng ta sẽ bắt đầu với hai kho ngữ liệu email : kho ngữ liệu email spam và kho ngữ liệu email non-spam. Số lượng email trong mỗi kho ngữ liệu ban đầu không hạn chế. Nếu kho ngữ liệu càng lớn thì hiệu quả lọc email sẽ càng cao. Từ hai kho ngữ liệu này, chúng tôi phân tích và duyệt qua tất cả các token bao gồm cả phần tiêu đề của email. Đối với những email html, chúng tôi thực hiện bóc tách các thẻ html để lấy nội dung giữa các thẻ.

Sau đó ta tính xác suất spam của mỗi token đã được phân tích, xác suất này chính là xác suất một email chỉ chứa token đó và là email spam.

Như vậy mấu chốt ở đây là ta phải tính ra được xác suất spam của mỗi token. Theo Paulgraham [7], xác suất spam của mỗi token được tính dựa trên số lần xuất hiện của mỗi token trong mỗi kho ngữ liệu học ban đầu. Ví dụ một token w có số lần xuất hiện trong kho ngữ liệu spam là *s*,

trong kho ngữ liệu non-spam là *n*, số email tổng cộng của hai kho ngữ liệu spam và non-spam lần lượt là N_S và N_N, thế thì xác suất spam của token w được tính như sau:

$$P(X = w, C = spam) = \frac{\frac{s}{N_S}}{\frac{s}{N_S} + \frac{n}{N_N}}$$

Công thức 5-1

Tuy nhiên, vì số lần xuất hiện của một token trong mỗi kho ngữ liệu học có khả năng vượt quá kích thước của kho ngữ liệu học đó (tổng số email) do đó, trong công thức trên, thay $\frac{s}{N_S}$ bằng Min(1, $\frac{s}{N_S}$) và $\frac{n}{N_N}$ bằng Min(1, $\frac{n}{N_N}$)

Do đó Công thức 5-1viết lại như sau:

$$P(X = w, C = spam) = \frac{Min(1, \frac{s}{N_S})}{Min(1, \frac{s}{N_S}) + Min(1, \frac{n}{N_N})}$$

công thức 5-2

Theo cách trên thì chúng ta đánh giá khả năng spam của một token xuất hiện trong một kho ngữ liệu học 100 lần ở 100 email khác nhau là bằng với khả năng spam của một token xuất hiện trong một kho ngữ liệu học 100 lần nhưng chỉ ở trong một email

Chúng tôi đề xuất một cách tính xác suất spam của token khác như sau: thay vì dựa vào số lần xuất hiện của token trong từng kho ngữ liệu học, chúng tôi dựa vào số email chứa token trong từng kho ngữ liệu học. Công thức tính như sau :

$$P(X = w, C = spam) = \frac{\frac{n_S}{N_S}}{\frac{n_S}{N_S} + \frac{n_N}{N_N}}$$

công thức 5-3

Với :

- n_S là số email có chứa token trong kho ngữ liệu email spam
- n_N là số email có chứa token trong kho ngữ liệu email non-spam
- N_S là tổng số email của kho ngữ liệu học spam
- N_N là tổng số email của kho ngữ liệu học non-spam

Tuy nhiên, ta nhận thấy rằng công thức trên đã đánh giá khả năng spam của mỗi token là như nhau với token xuất hiện 1 lần trong 1 email và token xuất hiện 100 lần trong 1 email, bởi vì ở cả hai trường hợp, ta đều chỉ tính thêm vào số email chứa token là 1 mà thôi

Chúng ta có thể kết hợp hai cách tính ở trên, để có thể sử dụng được nhiều thông tin về token hơn. Chúng tôi đề xuất thêm một công thức nữa - được xem là sự kết hợp giữa hai công thức trên

$$P(X = w, C = spam) = \frac{\frac{n_S}{N_S} * b}{\frac{n_S}{N_S} * b + \frac{n_N}{N_N} * g}$$

công thức 5-4

Với

- n_S là số email có chứa token trong kho ngữ liệu email spam
- n_N là số email có chứa token trong kho ngữ liệu email non-spam
- N_S là tổng số email của kho ngữ liệu học spam
- N_N là tổng số email của kho ngữ liệu học non-spam

- ü *b* là số lần xuất hiện của token trong kho ngữ liệu email spam
- ü *g* là số lần xuất hiện của token trong kho ngữ liệu email non-spam

Còn đối với các token chỉ xuất hiện kho ngữ liệu này mà không xuất hiện ở kho ngữ liệu kia thì ta không thể kết luận rằng một token chỉ xuất hiện ở kho ngữ liệu spam thì không bao giờ xuất hiện trong một email non-spam, và ngược lại. Cách thích hợp ở đây là ta sẽ gán cho chúng một giá trị phù hợp [7] Như vậy, với những token chỉ xuất hiện trong kho ngữ liệu email spam thì ta sẽ gán khả năng xác suất spam cho nó là giá trị N gần với 1 (chẳng hạn 0.9999)và ngược lại thì gán xác suất spam là giá trị M gần với 0 (chẳng hạn 0.0001).

Như vậy ta đã xác định được xác suất spam của một email có chuá một token nào đó hay xác suất spam của một token như sau:

Tính theo công thức 5-2, ta có :

$$P = Max\left(M, Min\left(N, \frac{Min(1, \frac{s}{N_S})}{Min(1, \frac{s}{N_S}) + Min(1, \frac{n}{N_N})}\right)\right)$$

Công thức 5-5 :công thức tính xác suất spam của token dựa trên số lần xuất hiện

Tính theo công thức 5-3, ta có :

$$P = Max\left(M, Min\left(N, \frac{\frac{n_S}{N_S}}{\frac{n_S}{N_S} + \frac{n_N}{N_N}}\right)\right)$$

Công thức 5-6 :công thức tính xác suất spam của token dựa trên số email chứa token

Tính theo công thức 5-4

$$P = Max\left(M, Min\left(N, \frac{\frac{n_S}{N_S} * s}{\frac{n_S}{N_S} * s + \frac{n_N}{N_N} * n} \right) \right)$$

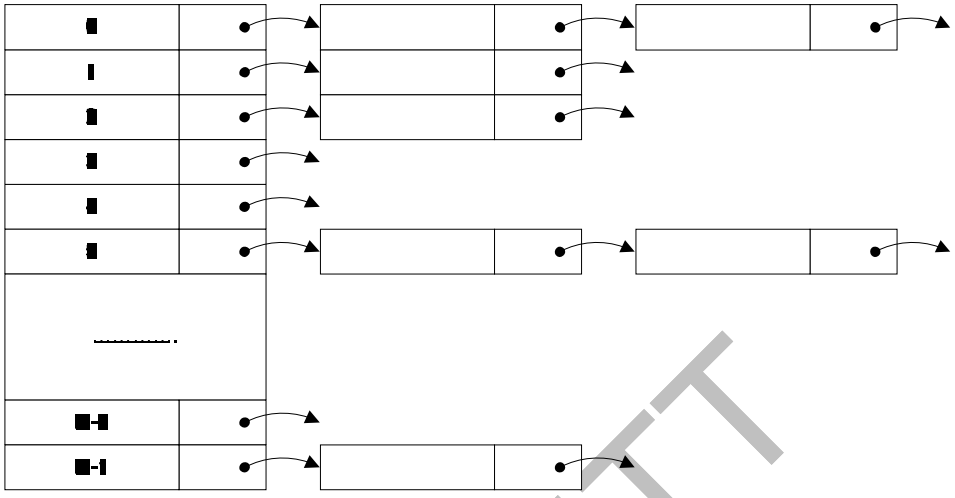
Công thức 5-7 :ctính xác suất spam của token dựa trên số lần xuất hiện và số email chứa nó

Với :

- ü s là số lần xuất hiện của token trong kho ngữ liệu học spam
- ü n là số lần xuất hiện của token trong kho ngữ liệu học non-spam
- ü n_S là số email chứa token trong kho ngữ liệu học spam
- ü n_N là số email chứa token trong kho ngữ liệu học non-spam
- ü N_S là tổng số email chứa trong kho ngữ liệu học spam
- ü N_N là tổng số email chứa trong kho ngữ liệu học non-spam

Một vấn đề phức tạp mà chúng tôi gặp phải trong quá trình thực hiện phân loại email dựa trên thuật toán Naïve Bayesian là việc tách token và tính xác suất spam của token, bởi vì số token là khá lớn, ở đây chúng tôi sử dụng cấu trúc dữ liệu là bảng băm.Ứng với mỗi kho ngữ liệu email spam và non-spam chúng tôi xây dựng một bảng băm tương ứng.Bảng băm này sẽ bao gồm token và số email chứa token hoặc số lần xuất hiện của token trong từng kho ngữ liệu tương ứng, hoặc có thể đồng thời chứa ba thông tin này – tùy theo chúng ta áp dụng cách tính xác suất spam nào cho mỗi token. Như vậy mỗi token sẽ có một giá trị băm (xác định bằng hàm băm tự định nghĩa) tương ứng với vị trí trên bảng băm để ta có thể truy xuất nhanh đến phần tử token trên bảng. Mục đích xây dựng bảng băm là để tối ưu hóa tốc độ truy xuất các token trích từ email cũng như tối ưu thời gian xác định một email là spam hay không. Mỗi phần tử của bảng băm lưu trữ token, số lần xuất hiện (hoặc số email có chứa token đó), hoặc xác suất spam của nó, tùy theo mục đích xử lý cụ thể mà mỗi phần tử

của bảng băm sẽ mang những thông tin khác nhau. Bảng băm được mô tả như sau:



Hình 5-1Mô tả cấu trúc bảng băm

Sau khi có 2 bảng băm tương ứng với hai kho ngữ liệu email, ta sẽ xây dựng bảng băm thứ ba. Mỗi phần tử trong bảng băm này sẽ lưu những thông tin gồm: token và khả năng (xác suất) spam của token.Tuy nhiên để việc thực hiện tiện lợi và không phải xét quá nhiều token, chúng tôi chỉ xem xét những token mà số lần xuất hiện của nó hoặc số email chứa nó trong cơ sở dữ học ban đầu lớn hơn một ngưỡng nào đó, với những token mà tổng số lần xuất hiện hoặc tổng số email chứa nó nhỏ hơn ngưỡng này, chúng tôi không tính xác suất cho token đó. Điều này là hợp lý bởi vì những token có tổng số lần xuất hiện (hoặc tổng số email chứa nó quá ít thì cũng không đáng để xem xét đến, do đó sẽ giúp giảm bớt số token cần tính xác suất cũng như dung lượng lưu trữ cho dữ liệu ở bảng băm thứ ba này.Ở đây chúng tôi thử nghiệm lần lượt hai ngưỡng 3 và 5, kết quả thực hiện ở hai ngưỡng này gần như là tương đương nhau, cuối cùng chúng tôi chọn giá trị 3.

Theo Paulgraham [7] thì chúng ta cần hạn chế loại lỗi false positive (nhận email non-spam thành email spam), do đó số lần xuất hiện của các token hoặc số email chứa token trong kho ngữ liệu non-spam sẽ được

nhân với một trọng số W, điều này giúp phân biệt được giữa những token thỉnh thoảng xuất hiện trong các email hợp lệ với những token hầu như không xuất hiện, chúng tôi thử nghiệm lần lượt với hai giá trị 1 và 2.

Ví dụ thông tin bảng băm thứ 3:

Token:	Khả năng spam :
madam	0.99
promotion	0.99
republic	0.99
shortest	0.047225013
mandatory	0.047225013
standardization	0.07347802

Cách tính xác suất spam cho mỗi token được thực hiện theo các công thức như đã nói ở trên.

Cuối cùng để xác định một email mới đến có phải là spam không thì chúng tôi trích ra *n* token ở trong email đó.Cách chọn mẫu tập thuộc tính để xét thông thường là chọn ra *n* token một cách ngẫu nhiên, tuy nhiên nhận thấy rằng những token trung tính (khả năng spam là 0.4-0.6 thì không có tác dụng lắm trong việc nhận dạng email spam) nên ta chọn *n* token này với định hướng là chọn những token đặc trưng cho một email spam và email non-spam, chúng tôi chọn những token có khả năng spam cao nhất và thấp nhất. Như vậy chúng tôi chọn *n* token có khoảng cách giữa xác suất spam của chúng với giá trị trung tính 0.5 là cao nhất Chúng ta gọi giá trị này là giá trị "đặc trưng". Như vậy ta sẽ chọn được những token hoặc là có khả năng spam cao nhất (xác suất spam cao nhất) hoặc là những token có khả năng non-spam cao nhất (xác suất spam thấp nhất). Nếu có k ($k \geq 2$) token có cùng giá trị "đặc trưng ", bởi vì khả năng xuất hiện của k token này ngang nhau, do đó hoàn toàn không mất tính tổng quát, chúng tôi chọn token đầu tiên trong k token có cùng giá trị " đặc trưng "này. Sau khi chọn được *n* token này chúng tôi sẽ tra trong bảng

băm thứ 3 (lưu token và khả năng spam của nó) để lấy ra khả năng spam riêng của mỗi token. Nếu không tìm thấy khả năng spam riêng cho token trong bảng băm,có nghĩa là token này là mới – chưa có trong cơ sở dữ liệu token của ta.Một token chưa từng xuất hiện trong kho ngữ liệu học thì khả năng spam của nó tương đối thấp [7], chúng tôi lấy giá trị trung tính 0.4. Từ đó chúng tôi tính khả năng tổng hợp một email chứa n token này là spam.

Cách tính khả năng tổng hợp :chúng tôi dựa vào Công thức 4-7

$$P(C=c \mid X=x) = \frac{P(C=c).\prod_{i=1}^{n} P(X_i = x_i \mid C=c)}{\sum_{k\in\{spam,non-spam\}} P(C=k).\prod_{i=1}^{n} P(X_i = x_i \mid C=k)}$$

Thế thì xác suất spam tổng hợp của một email *C* được xét là :

$$P(C=spam \mid X=x) = \frac{P(C=spam)\prod_{i=1}^{n} P(X_i = x_i \mid C=c)}{\sum_{k\in\{spam,non-spam\}} P(C=k).\prod_{i=1}^{n} P(X_i = x_i \mid C=k)}$$

Ví dụ

Token:	X*ác suất (Probability)*:
madam	0.99
promotion	0.99
shorstest	0.047225013

Xác suất một email là Spam là :0.6

à Khả năng kết hợp

$$= \frac{0.99*0.99*0.047225013*0.6}{0.6*0.99*0.99*0.047225013 + (1-0.6)*(1-0.99)(1-0.99)(1-0.047225013)}$$

Sau khi có khả năng tổng hợp, chúng tôi so sánh với các giá trị ngưỡng (đã nói ở mục 4.3.1) để phân loại email spam hay non-spam, nếu xác suất spam tổng hợp của email lớn hơn ngưỡng *t* chúng tôi kết luân email đó là spam, ngược lại email đó là non-spam.

5.2 Thử nghiệm hiệu quả phân loại

5.2.1 Thử nghiệm với kho ngữ liệu pu:

Bởi vì kho ngữ liệu học và kiểm thử là số, do đó chúng tôi thay đổi về cách lấy token, ở đây chúng tôi xem token là các con số, và dấu hiệu tách token là các khoảng trắng.

5.2.1.1 Kịch bản kiểm thử :

Chúng tôi thử nghiệm nhân trọng số non-spam W với 1 và 2

Với mỗi W, chúng tôi thử nghiệm với λ lần lượt với các giá trị 1, 9, và 999

Tương ứng với mỗi giá trị λ và W chúng tôi thực hiện tính xác suất spam theo các công thức :Công thức 5-5, Công thức 5-6 và Công thức 5-7

Số token được lấy lần lượt là 10, 15, 20

Chúng tôi kiểm tra với các kho ngữ liệu pu1, pu2, pu3 và puA

Tương ứng với mỗi kho ngữ liệu trên chúng tôi cho học từ part1 đến part 9, sau đó chúng tôi thử nghiệm phân loại trên part10, chứa những email chưa được học.

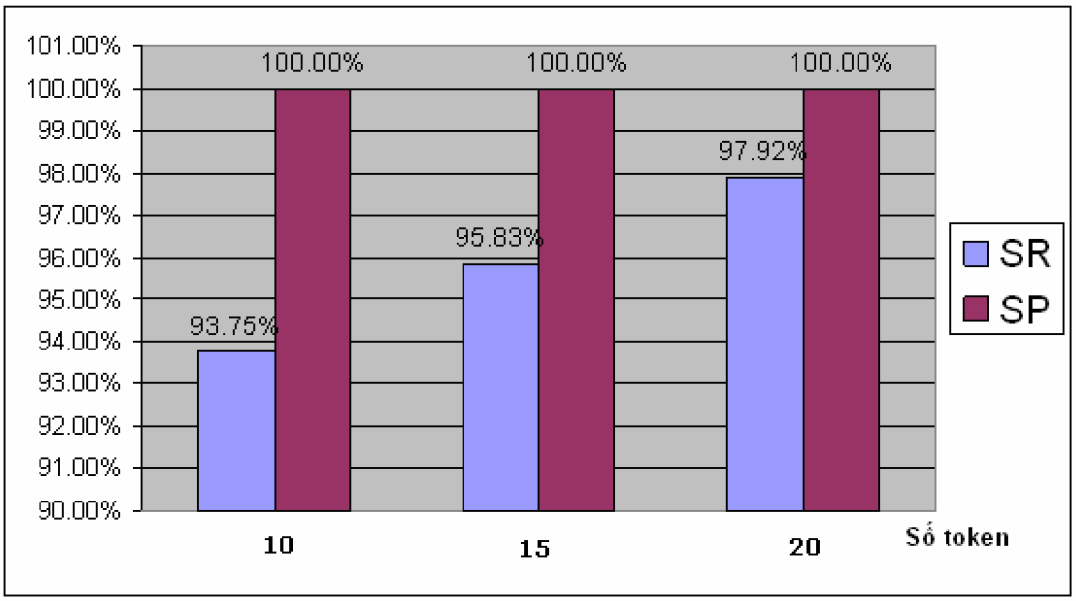
5.2.1.2 Kết quả thử nghiệm với kho ngữ liệu pu :

Kết quả thực hiện: chúng tôi trình bày kết quả thực hiện với trường hợp nhân trọng số non-spam W=2, kết quả chi tiết với W=1 xin xem phần phụ lục.

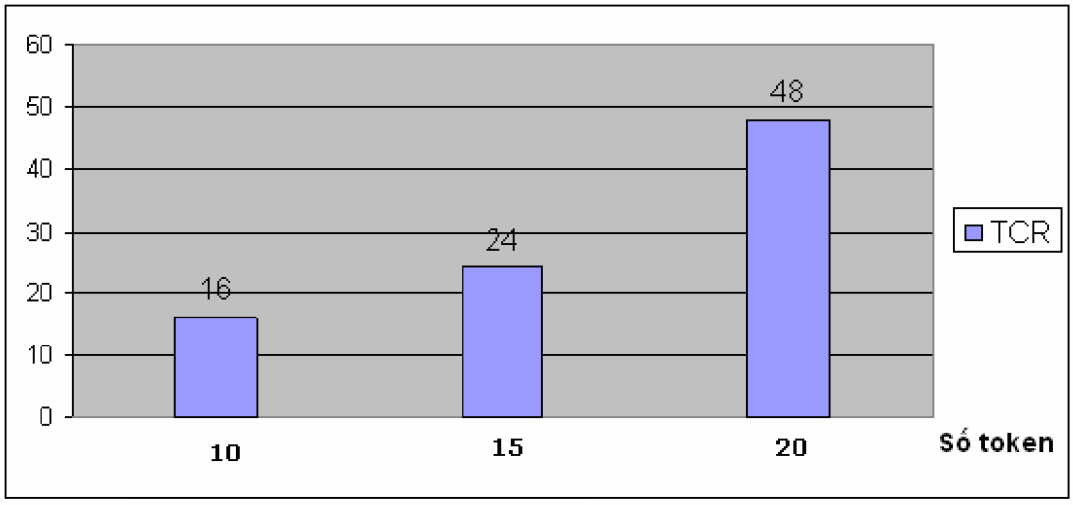
v Kết quả kiểm thử trên PU1:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	44	45	45	45	45	44	46	46	47
	S→N	4	3	3	3	3	4	2	2	1
	N→N	61	61	61	61	61	61	61	61	61
	N→S	0	0	0	0	0	0	0	0	0
	SR	91.67%	93.75%	93.75%	93.75%	93.75%	91.67%	95.83%	95.83%	97.92%
	SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	TCR	12	16	16	16	16	12	24	24	48
9	S→S	44	45	45	44	44	44	45	46	47
	S→N	4	3	3	4	4	4	3	2	1
	N→N	61	61	61	61	61	61	61	61	61
	N→S	0	0	0	0	0	0	0	0	0
	SR	91.67%	93.75%	93.75%	91.67%	91.67%	91.67%	93.75%	95.83%	97.92%
	SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	TCR	12	16	16	12	12	12	16	24	48
999	S→S	43	43	43	43	43	43	45	45	47
	S→N	5	5	5	5	5	5	3	3	1
	N→N	61	61	61	61	61	61	61	61	61
	N→S	0	0	0	0	0	0	0	0	0
	SR	89.58%	89.58%	89.58%	89.58%	89.58%	89.58%	93.75%	93.75%	97.92%
	SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	TCR	9.6	9.6	9.6	9.6	9.6	9.6	16	16	48

Bảng 5-1 Kết quả kiểm thử phân lọai email bằng phương pháp phân lọai Naïve Bayesian trên kho ngữ liệu PU1



Hình 5-2 Lược đồ so sánh các chỉ số spam recall (SR) và spam precision (SP) theo số token thử nghiệm trên kho ngữ liệu PU1 với công thức 5-7 ($\lambda = 9$)

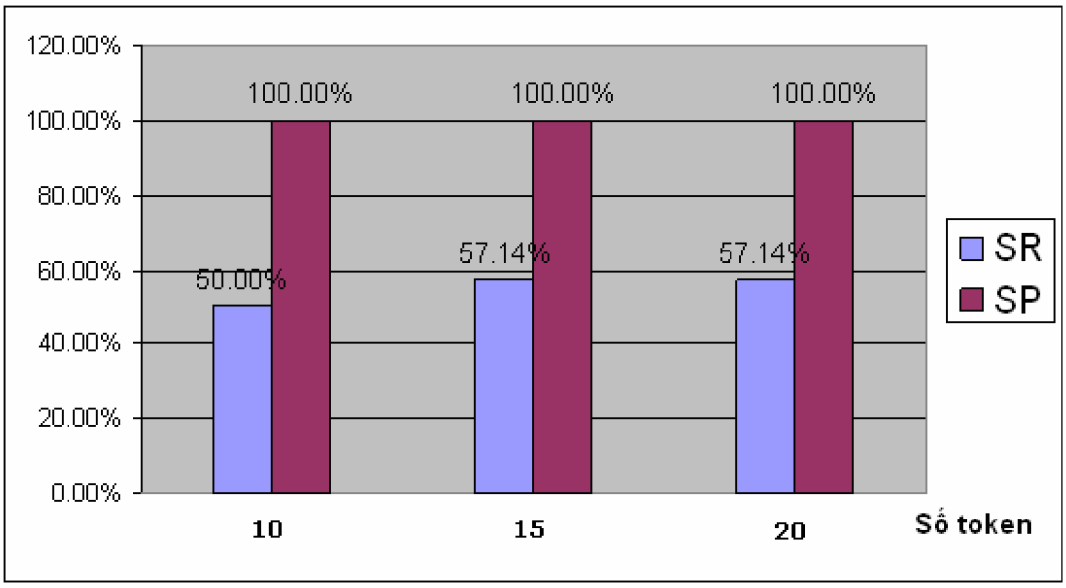


Hình 5-3 Lược đồ chỉ số TCR theo số token thử nghiệm trên kho ngữ liệu PU1 với công thức 5-7 ($\lambda = 9$)

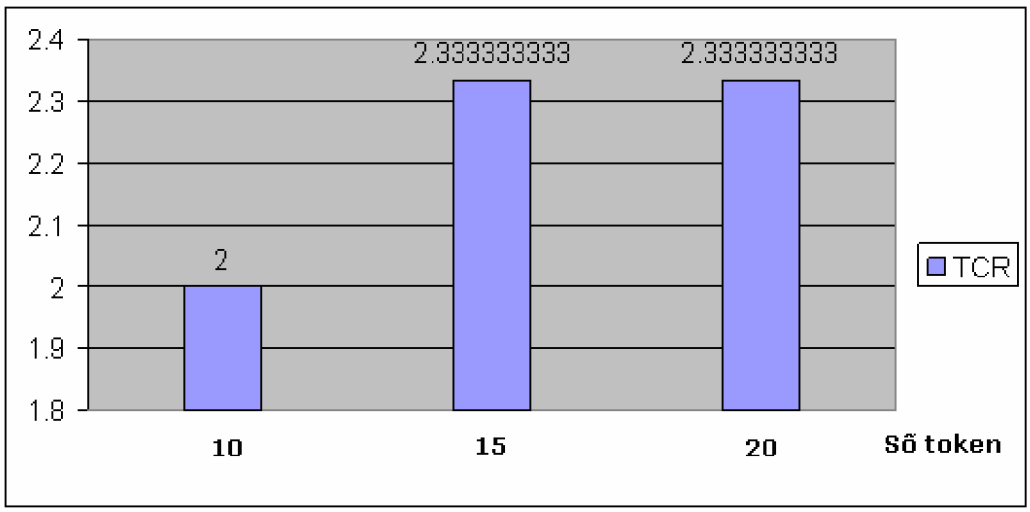
v Kết quả kiểm thử trên PU2:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	7	8	9	7	8	8	8	9	5
	S→N	7	6	5	7	6	6	6	5	9
	N→N	57	57	57	57	57	57	57	57	57
	N→S	0	0	0	0	0	0	0	0	0
	SR	50.00%	57.14%	64.29%	50.00%	57.14%	57.14%	57.14%	64.29%	35.71%
	SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	TCR	2	2.333333	2.8	2	2.333333	2.333333	2.333333	2.8	1.555556
9	S→S	7	8	8	7	8	8	8	8	5
	S→N	7	6	6	7	5	6	6	6	9
	N→N	57	57	57	57	57	57	57	57	57
	N→S	0	0	0	0	0	0	0	0	0
	SR	50.00%	57.14%	57.14%	50.00%	61.54%	57.14%	57.14%	57.14%	35.71%
	SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	TCR	2	2.333333	2.333333	2	2.6	2.333333	2.333333	2.333333	1.555556
999	S→S	7	8	8	7	6	7	8	5	5
	S→N	7	6	6	7	8	7	6	9	9
	N→N	57	57	57	57	57	57	57	57	57
	N→S	0	0	0	0	0	0	0	0	0
	SR	50.00%	57.14%	57.14%	50.00%	42.86%	50.00%	57.14%	35.71%	35.71%
	SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	TCR	2	2.333333	2.333333	2	1.75	2	2.333333	1.555556	1.555556

Bảng 5-2 Kết quả kiểm thử phân lọai email bằng phương pháp phân lọai Naïve Bayesian trên kho ngữ liệu PU2



Hình 5-4 Lược đồ so sánh các chỉ số spam recall (SR) và spam precision (SP) theo số token thử nghiệm trên kho ngữ liệu PU2 với công thức 5-5 ($\lambda = 9$)

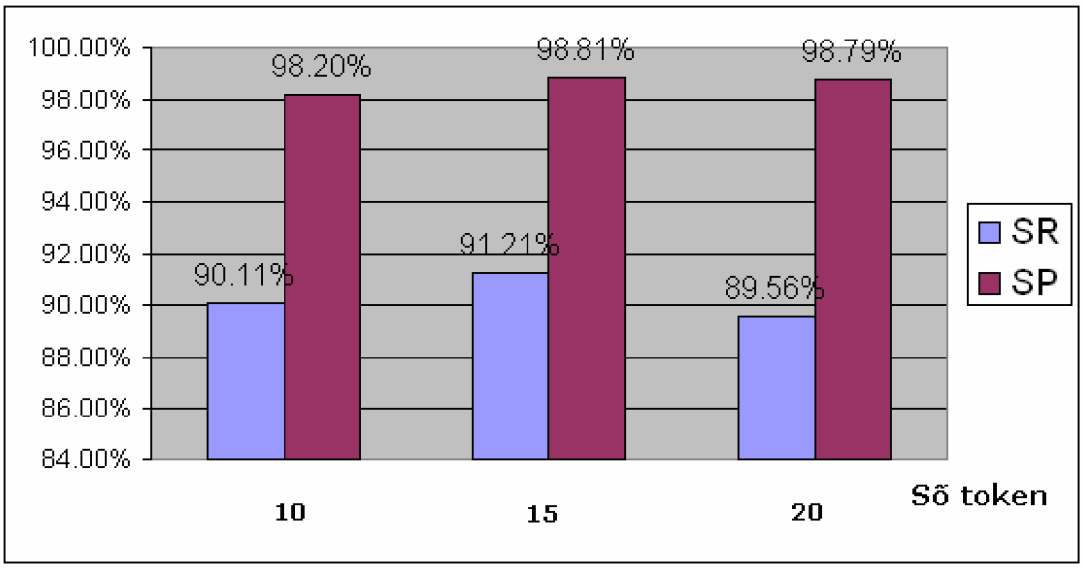


Hình 5-5 Lược đồ chỉ số TCR theo số token thử nghiệm trên kho ngữ liệu PU2 với công thức 5-5 ($\lambda = 9$)

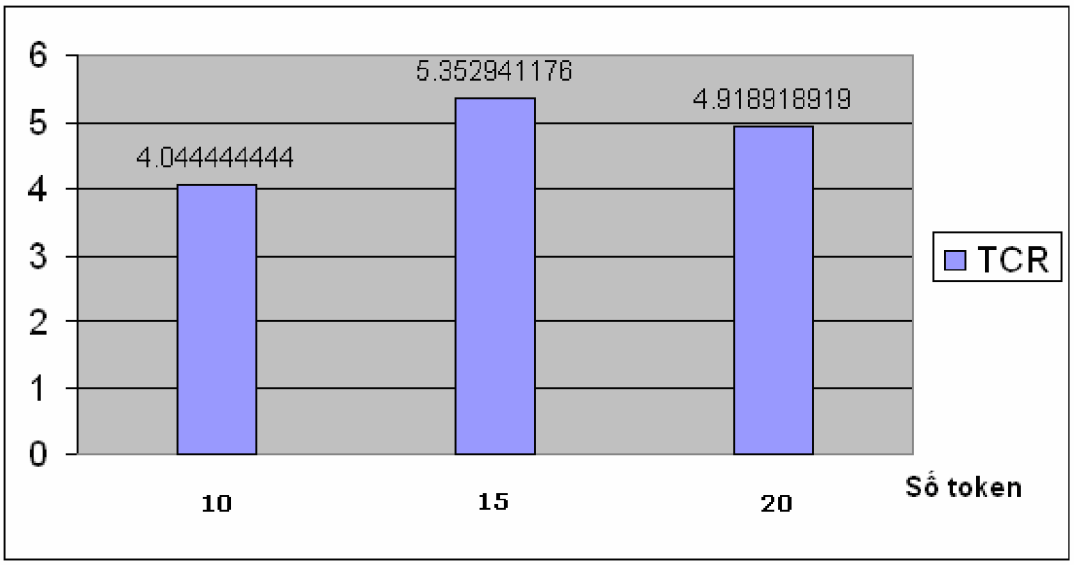
v Kết quả kiểm thử trên PU3:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	169	168	168	167	169	165	165	172	170
	S→N	13	14	14	15	13	17	17	10	12
	N→N	228	228	227	228	228	229	226	222	224
	N→S	3	3	4	3	3	2	5	9	7
	SR	92.86%	92.31%	92.31%	91.76%	92.86%	90.66%	90.66%	94.51%	93.41%
	SP	98.26%	98.25%	97.67%	98.24%	98.26%	98.80%	97.06%	95.03%	96.05%
	TCR	11.375	10.70588	10.11111	10.11111	11.375	9.578947	8.272727	9.578947	9.578947
9	S→S	167	168	168	164	166	163	165	171	170
	S→N	15	14	14	18	16	19	17	11	12
	N→N	229	228	227	228	229	229	227	222	225
	N→S	2	3	4	3	2	2	4	9	6
	SR	91.76%	92.31%	92.31%	90.11%	91.21%	89.56%	90.66%	93.96%	93.41%
	SP	98.82%	98.25%	97.67%	98.20%	98.81%	98.79%	97.63%	95.00%	96.59%
	TCR	5.515152	4.439024	3.64	4.044444	5.352941	4.918919	3.433962	1.978261	2.757576
999	S→S	163	163	165	160	156	156	163	168	169
	S→N	19	19	17	22	26	26	19	14	13
	N→N	229	229	229	229	229	229	227	225	225
	N→S	2	2	2	2	2	2	4	6	6
	SR	89.56%	89.56%	90.66%	87.91%	85.71%	85.71%	89.56%	92.31%	92.86%
	SP	98.79%	98.79%	98.80%	98.77%	98.73%	98.73%	97.60%	96.55%	96.57%
	TCR	0.090233	0.090233	0.090323	0.090099	0.089921	0.089921	0.04533	0.030293	0.030298

Bảng 5-3 Kết quả kiểm thử phân loại email bằng phương pháp phân lọai Naïve Bayesian trên kho ngữ liệu PU3



Hình 5-6 Lược đồ so sánh các chỉ số spam recall (SR) và spam precision (SP) theo số token thử nghiệm trên kho ngữ liệu PU3 với công thức 5-6 ($\lambda = 9$)

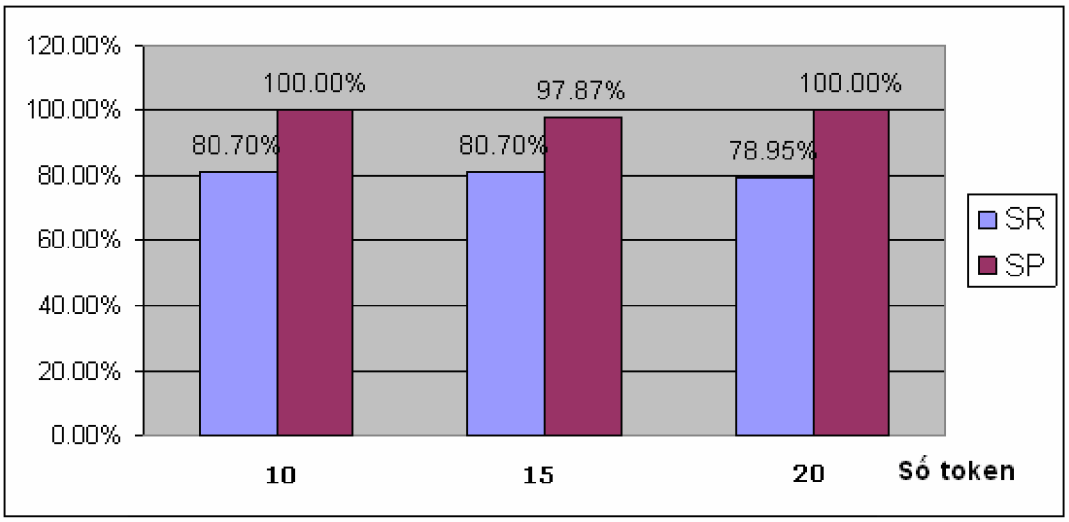


Hình 5-7 Lược đồ chỉ số TCR theo số token thử nghiệm trên kho ngữ liệu PU3 với công thức 5-6 ($\lambda = 9$)

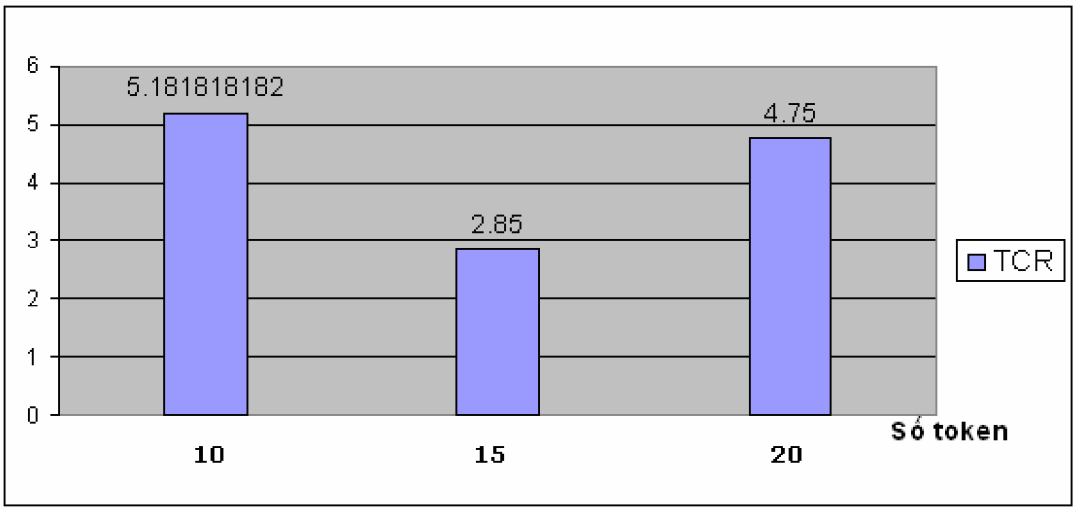
v Kết quả kiểm thử trên PUA:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	46	46	46	43	42	41	50	48	46
	S→N	11	11	11	14	15	16	7	9	11
	N→N	57	56	57	57	57	57	56	56	57
	N→S	0	1	0	0	0	0	1	1	0
	SR	80.70%	80.70%	80.70%	75.44%	73.68%	71.93%	87.72%	84.21%	80.70%
	SP	100.00%	97.87%	100.00%	100.00%	100.00%	100.00%	98.04%	97.96%	100.00%
	TCR	5.181818	4.75	5.181818	4.071429	3.8	3.5625	7.125	5.7	5.181818
9	S→S	46	46	45	42	41	38	49	46	45
	S→N	11	11	12	15	16	19	8	11	12
	N→N	57	56	57	57	57	57	56	55	57
	N→S	0	1	0	0	0	0	1	2	0
	SR	80.70%	80.70%	78.95%	73.68%	71.93%	66.67%	85.96%	80.70%	78.95%
	SP	100.00%	97.87%	100.00%	100.00%	100.00%	100.00%	98.00%	95.83%	100.00%
	TCR	5.181818	2.85	4.75	3.8	3.5625	3	3.352941	1.965517	4.75
999	S→S	43	43	42	41	37	35	47	45	44
	S→N	14	14	15	16	20	2	10	12	13
	N→N	57	57	57	57	57	57	56	57	57
	N→S	0	0	0	0	0	0	1	0	0
	SR	75.44%	75.44%	73.68%	71.93%	64.91%	94.59%	82.46%	78.95%	77.19%
	SP	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	97.92%	100.00%	100.00%
	TCR	4.071429	4.071429	3.8	3.5625	2.85	18.5	0.056492	4.75	4.384615

Bảng 5-4 Kết quả kiểm thử phân loại email bằng phương pháp phân loại Naïve Bayesian trên kho ngữ liệu PUA



Hình 5-8 Lược đồ so sánh các chỉ số spam recall (SR) và spam precision (SP) theo số token thử nghiệm trên kho ngữ liệu PUA với công thức 5-5 ($\lambda = 9$)



Hình 5-9 Lược đồ chỉ số TCR theo số token thử nghiệm trên kho ngữ liệu PUA với công thức 5-5 ($\lambda = 9$)

Nhận xét :kết quả kiểm thử trên các kho ngữ liệu PU là khá tốt, hiệu quả phân loại giữa các công thức là không quá khác biệt, với cách chọn $\lambda = 9$ và $\lambda = 1$ hiệu quả hơn với $\lambda = 999$, theo chúng tôi thì kho ngữ liệu không lớn lắm nên sử dụng $\lambda = 999$ thì không hiệu quả bằng. Về cách chọn số token, hiệu quả phân loại khi chọn số token là 10, 15 hay 20 cũng không khác biệt lắm.

5.2.2 Thử nghiệm với kho ngữ liệu email chữ :

5.2.2.1 Kịch bản kiểm thử :

Sau khi đã thử nghiệm với kho ngữ liệu số, chúng tôi chọn một bộ (λ , n, W) để kiểm thử với kho ngữ liệu email chữ.

Chúng tôi thử nghiệm với bộ dữ liệu λ = 9, số token là 15, trọng số non-spam là 2.

Ngữ liệu học và kiểm thử ở đây gồm ngữ liệu email là email văn bản trơn (text/plain), và ngữ liệu email html. Ngữ liệu email văn bản trơn có số email dùng để huấn luyện là :517 email non-spam, 528 email spam. Ngữ liệu dung để kiểm thử gồm 98 email spam, 100 email non-spam. Ngữ liệu email html có số email dùng để huấn luyện là 141 email non-spam, 155 email spam, số email dung để kiểm thử là 50 email spam, 50 email non-spam.

5.2.2.2 Kết quả kiểm thử :

Ngữ liệu email văn bản trơn:

- Ngữ liệu học :số email spam :517, số email non-spam:528
- Ngữ liệu kiểm thử :số email spam :98, số email non-spam :100

Ngữ liệu email html, số email kiểm thử :Spam =50, non-spam=50

		Công thức 5-5	Công thức 5-6	Công thức 5-7
TEXT	S→S	96	94	96
	S→N	2	4	2
	N→N	99	99	99
	N→S	1	1	1
	SR	97.96%	95.92%	97.96%
	SP	98.97%	98.95%	98.97%
	TCR	32.66667	19.6	32.66667
HTML	S→S	32	24	23
	S→N	18	26	27
	N→N	50	50	50
	N→S	0	0	0
	SR	64.00%	48.00%	46.00%
	SP	100.00%	100.00%	100.00%
	TCR	2.777778	1.923077	1.851852

Bảng 5-5 Kết quả kiểm thử phân lọai email bằng phương pháp phân lọai Bayesian trên kho ngữ liệu email chữ

Kết quả thực hiện với ngữ liệu email văn bản (text/plain) khá tốt, các chỉ số spam recall, spam precision khá cao, tuy nhiên thực hiện với kho ngữ liệu email html thì chỉ số spam recall không được cao trong khi chỉ số spam precision vẫn tốt. Kết quả này một phần vì kho ngữ liệu email html của chúng tôi không được lớn lắm, số lượng email html dùng để huấn luyện tương đối ít. Email html có đặc điểm là nội dung của nó hầu hết là các thẻ html, những thẻ html này không cung cấp được nhiều thông tin trong việc phân loại, nội dung chữ thật sự tương đối ít, điều này cũng ảnh hưởng đến kết quả thực hiện của thuật tóan Naïve Bayesian

5.3 Ưu – nhược điểm của phương pháp phân loại Naïve Bayesian:

5.3.1 Ưu điểm :

- Một ưu điểm của bộ lọc Bayes là nó cho phép học spam. Nghĩa là, khi có một email spam vượt qua được bộ lọc thì người dùng có thể đánh dấu spam cho email đó và bộ lọc sẽ tự phân tích email spam đó và cập nhật thêm vào kho ngữ liệu spam.
- Hiệu quả phân loại là khá cao

- Thời gian huấn luyện nhanh, theo Charles Elkan [16] với *e* mẫu huấn luyện và số thuộc tính là *f* thời gian cần thiết để huấn luyện phân loại Naïve Bayesian là hàm tuyến tính *O*(*ef*), không có thuật tóan máy học nào có thể khảo sát với cùng dữ liệu huấn luyện đó nhanh hơn Naïve Bayesian
- Ngoài những bộ lọc được xây dựng sẵn dựa trên kho ngữ liệu email có trước, thì với mỗi người dùng sẽ có bộ lọc riêng được xây dựng dựa trên kho ngữ liệu học email của chính họ. Do đó, việc đưa spam vượt qua được bộ lọc xây dựng sẵn thì không chắc là nó có thể vượt qua được bộ lọc của từng người dùng cụ thể.

5.3.2 Khuyết điểm :

- Khuyết diểm của phương pháp phân loại Bayes chính là việc phải huấn luyện cho nó
- Khuyết điểm thứ hai là hiệu quả phân loại phụ thuộc vào kho ngữ liệu huấn luyện ban đầu, nếu ngữ liệu không đủ lớn, kết quả phân loại sẽ bị ảnh hưởng.
- Dữ liệu đã qua huấn luyện là khá nhiều, làm tăng dung lượng lưu trữ.

Chương 6 : PHƯƠNG PHÁP ADABOOST VÀ ỨNG DỤNG PHÂN LOẠI EMAIL

Thuật toán Adaboost (Adaptive Boost)[15], được giới thiệu lần đầu vào năm 1995 bởi Freund và Schapire [10],[11], là một trong các thuật toán theo phương pháp Boosting. Boosting là một trong các phương pháp mới nhất được đề xuất dùng để nâng cao khả năng dự đoán đúng [12]. Boosting được áp dụng trong các bài toán phân loại và hồi qui. Boosting kết hợp các luật "yếu " (weak rule) có độ chính xác dự đoán thấp để cho ra một luật có độ chính xác dự đoán cao [12]. Thông thường mỗi một luật yếu là một luật đơn giản, có thể dựa vào đó để dự đoán đối tượng được xét thuộc về loại nào, tuy nhiên độ chính xác của dự đoán là không cao. Các luật yếu sẽ được huấn luyện tuần tự dựa trên các mẫu huấn luyện rất khó dự đoán đúng nếu chỉ căn cứ vào các luật đã có trước. Các thuật toán boost gồm AdaBoost, Uboost, LPBoost, LPUBoost ….Ở đây chúng tôi tập trung vào AdaBoost, và AdaBoost ứng dụng trong lĩnh vực phân loại văn bản

6.1 Thuật toán AdaBoost :

Mô tả phác thảo thuật toán như sau:

- Cho một tập huấn luyện $(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)$
- $y_i \in \{-1, +1\}$ là nhãn đúng của mỗi x_i trong tập X
- Với mỗi t, t=1,…T
  - Xây dựng hàm phân phối $D_t(i)$ tương ứng $i = 1...m$
  - Chọn một luật yếu $h_t : X \to \{-1, +1\}$ với lỗi

    $\varepsilon_t = \Pr_{D_t}[h_t(x_i) \neq y_i]$ là nhỏ nhất bằng thủ tục WeakLearner
- Ra : hàm dùng để dự đoán H

Hình 6-1 Mô tả thuật toán AdaBoost

Thuật toán sẽ tìm ra một tập các luật yếu bằng cách gọi thực hiện lặp đi lặp lại một thủ tục WeakLearner với số lần T. Những luật yếu này sẽ được kết hợp tuyến tính lại để cho ra một luật phân loại mạnh hơn : $H(x) = \sum_{t=1}^{T} h_t(x)$. Thủ tục

WeakLearner dùng để chọn ra luật yếu có lỗi phân loại sai nhỏ nhất tương ứng trong mỗi bước chạy t=1, … T, kết quả là ta có được tập luật đã huấn luyện gồm T luật yếu. Lỗi xảy ra khi $H(x_i) \neq y_i$, thuật toán AdaBoost xây dựng thủ tục WeakLearner chọn lựa luật yếu sao cho tương ứng với luật yếu được chọn lỗi sai ε được giảm tối đa. Thuật toán tìm cách duy trì một tập các trọng số tương ứng với tập mẫu huấn luyện. Mức ảnh hưởng của trọng số với mẫu học i ở lần t là $D_t(i)$ Trong quá trình học, các trọng số sẽ được cập nhật động. Nếu phân loại sai trọng số sẽ tăng lên nhấn mạnh những mẫu huấn luyện bị phân loại sai, ngược lại, trọng số giảm xuống [15].

6.2 AdaBoost trong phân loại văn bản nhiều lớp :

Một trong các lĩnh vực ứng dụng quan trọng của thuật toán AdaBoost là phân loại văn bản. Trong phân loại văn bản với nhiều lớp, có hai thuật toán AdaBoost mới nhất là AdaBoost.MH và AdaBoost.MR, trong phạm vi luận văn này chúng tôi tập trung nghiên cứu thuật toán AdaBoost.MH.

Xét bài toán phân loại văn bản nhiều lớp (nhãn), X biểu thị tập các văn bản và **Y** là tập có giới hạn các nhãn hoặc lớp. Định nghĩa kích thuớc của **Y** là k= | **Y** |. Trong trường hợp phân loại nhiều lớp, mỗi một văn bản $x \in$ X được gán nhiều nhãn trong **Y**. Một ví dụ dễ thấy là phân loại tin tức là một dạng phân loại văn bản nhiều lớp, mỗi một tin có thể phù hợp với nhiều lớp, chẳng hạn một tin có thể thuộc về nhiều loại như tin xã hội, kinh tế, văn hoá... Như vậy mỗi mẫu sẽ được gán nhãn là một cặp (x,Y) với $Y \subseteq$ **Y** là một tập các nhãn được gán cho x.

Với $Y \in$ **Y**, định nghĩa $Y[l]$ cho $l \in$ **Y** là

Y[l]= +1 nếu $l \in$ Y

-1 nếu $l \notin$ Y

Phân loại nhiều lớp ở đây là tìm cách xếp hạng các nhãn mà x có thể có.Mục đích của việc huấn luyện là thu được một hàm f: X× **Y** → R sao cho với mỗi văn bản x, những nhãn trong Y sẽ được sắp xếp theo thứ tự $f(x,.)$. Như vậy, nếu

$f(x, l_1) > f(x, l_2)$ thì l_1 được xem là có thứ hạng ưu tiên xếp loại cao hơn l_2. Thuật toán huấn luyện được xem là thành công nếu với mỗi *x* có tập nhãn tương ứng là Y thì thuật toán sẽ xếp hạng các nhãn trong Y cao hơn các nhãn không có trong Y

Thuật toán AdaBoost MH phân loại văn bản nhiều lớp :

Cho S là tập mẫu huấn luyện $\langle (x_1, Y_1), (x_2, Y_2), ..., (x_m, Y_m) \rangle$ với $x_i \in X$ và $Y_i \subseteq Y$. Ở mỗi bước thực hiện *t*, thủ tục WeakLearner sẽ chọn một luật yếu *h*: $X \times Y \to R$,dấu của *h*(*x*, *l*) cho biết nhãn *l* được gán hay không gán cho *x*, còn giá trị | *h*(*x*, *l*)| được xem là độ tin cậy của dự đoán

Thuật toán AdaBoost MH phân loại văn bản với nhiều lớp [14]

Cho $(x_1, Y_1), (x_2, Y_2), ..., (x_m, Y_m), x_i \in X$ và $Y_i \subseteq Y$.

- Khởi tạo $D_1(i,l) = \frac{1}{mk}$
- Với mỗi $t = 1, \ldots, T$
  - Huấn luyện tập học yếu sử dụng D_t
  - Chọn một luật yếu $h_t : X \times Y \to R$ bằng thủ tục WeakLearner
  - Chọn $\alpha_t \in R$
  - Cập nhật

$$D_{t+1}(i,l) = \frac{D_t(i,l)\exp\left(-\alpha_t Y_i[l] h_t(x_i,l)\right)}{Z_t}$$

Z_t được chọn sao cho D_{t+1} là hàm phân phối

Ra : $H(x) = sign(f(x)), f(x) = \sum^{T} \alpha_t h_t(x)$

Hình 6-2 Mô tả thuật toán AdaBoost MH phân loại văn bản nhiều lớp

Luật yếu h_t được chọn sao cho giá trị $Z_t = \sum_{i=1}^{m} \sum_{l} D_t(i,l)\exp\left(-\alpha_t Y_i[l] h_t(x_i,l)\right)$ nhỏ nhất

6.3 Ứng dụng AdaBoost trong phân loại email:

Bài toán chúng ta đang xét là phân loại email, ở đây chúng ta chỉ phân loại email hoặc là loại spam hoặc là loại non-spam. Như vậy bài toán phân loại email là trường hợp đặc biệt của phân loại văn bản nhiều lớp, khi mỗi mẫu huấn luyện chỉ

nhận một nhãn đơn – thay vì một tập nhãn. Khi đó phân loại email với hai lớp spam và non-spam trở thành bài tóan phân loại văn bản nhị phân.

6.3.1 Thuật toán AdaBoost.MH trong trường hợp phân loại nhị phân

Xét bài toán hai lớp, mẫu huấn luyện là tập $S = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$ gồm m bộ (x_i, y_i) đã được gán nhãn, x_i là các mẫu huấn luyện thuộc về không gian mẫu huấn luyện *X* và $y_i \in$ {-1, +1} là lớp hay nhãn tương ứng gắn với x_i. Mục đích của việc học là có được một hàm *H:* $X \to R$, sao cho với mỗi mẫu *x*, dấu của *H(x)* cho biết lớp mà *x* thuộc về (-1 hay +1), và độ lớn |*H(x)*| cho biết độ tin cậy được dự đoán. Một hàm như thế có thể dùng cho việc phân loại hay xếp loại các mẫu chưa gặp.

Mã giả của AdaBoost.MH trong trường hợp phân loại nhị phân như sau:

Thủ tục Adaboost

Vào :S= $\{(x_i, y_i)\}_{i=1}^{m}$

S là tập mẫu huấn luyện

khởi tạo hàm phân phối D_1 (Cho tất cả i, $1 \le i \le$ m)

$D_1(i)$= 1/m

Lặp :với mọi t : t:=1 …T

Huấn luyện tập học yếu sử dụng phân bố D_t

Chọn một luật yếu $h_t : X \to$ R dùng thủ tục WeakLearner

Chọn $\alpha_t \in$ R,

Cập nhật D_t (cho tất cả i, $1 \le i \le$ m)

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Z_t là hệ số chuẩn hoá (Z_t được chọn sao cho D_{t+1} là hàm

#phân phối

Hết lặp

Ra: luật kết hợp $H(x) = sign\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right)$

Hình 6-3 Mô tả thuật toán AdaBoost.MH phân loại nhị phân

Ban đầu các trọng số D_1 là như nhau, nhưng thuật toán boost cập nhật trọng số ở mỗi lần t theo hàm mũ, luật yếu h_t được chọn sao cho giá trị

$Z_t = \sum_{i=1}^{m} D_t(i)\left(\exp\left(-\alpha_t y_i h_t(x_i)\right)\right)$ đạt cực tiểu.

Giới hạn lỗi huấn luyện sai :

Đặt $f(x) = \sum_{t=1}^{T}\alpha_t h_t(x)$, thế thì $H(x)$=sign$(f(x))$, đặt $\left[\left[\pi\right]\right]=1$ nếu π đúng, ngược lại $\left[\left[\pi\right]\right]=0$

Theo (Schapire &Singer 1998)[12] , giới hạn lỗi huấn luyện sai của hàm H là :

$$\frac{1}{m}|\{i: H(x_i \neq y_i)\}| \leq \prod_{t=1}^{T} Z_t$$

Thật vậy, vì $D_1 = \frac{1}{m}$, bằng qui nạp theo *t* ta có :

$$D_{t+1}(i) = \frac{\exp\left(-\sum_t \alpha_t y_i h_t(x_i)\right)}{m\prod_t Z_t}$$

$$= \frac{\exp(-y_i f(x_i))}{m\prod_t Z_t} \qquad (1)$$

Nếu $H(x_i) \neq y_i$ (có lỗi xảy ra) thì $\left[\left[H(x_i) \neq y_i\right]\right] = 1$ và $y_i f(x_i) \leq 0$ nên $\exp(-y_i f(x_i)) \geq 1$

Nếu $H(x_i) = y_i$ thì $\left[\left[H(x_i) \neq y_i\right]\right] = 0$

Như vậy luôn có được :

$$\left[\left[H(x_i) \neq y_i\right]\right] \leq \exp(-y_i f(x_i)) \qquad (2)$$

Kết hợp (1) và (2), ta có giới hạn trên của lỗi sai :

$$\frac{1}{m}\left[\left[i: H(x_i) \neq y_i\right]\right] \leq \frac{1}{m}\sum_{i=1} \exp(-y_i f(x_i))$$

$$= \sum_i \left(\prod_t Z_t\right) D_{T+1}(i)$$

$$= \prod_t Z_t$$

Hệ quả quan trọng của công thức trên là : thay vì phải cực tiểu lỗi huấn luyện, ta chỉ cần cực tiểu giới hạn trên Z_t trong mỗi lần thực hiện boost, ta có thể áp dụng giới hạn này phục vụ cho việc chọn giá trị α_t và chọn luật yếu h_t ở mỗi bước chạy *t*

6.3.2 Phương pháp lựa chọn luật yếu :

Ở mỗi bước chạy *t*, luật yếu được lựa chọn sao cho lỗi sai được cực tiểu, dựa vào giới hạn trên của lỗi sai, thay vì chọn h_t sao cho lỗi huấnluyện là nhỏ nhất, ta chọn h_t sao cho Z_t là nhỏ nhất

Với mỗi từ *w*, định nghĩa $w \in x$ tương đương với *w* có trong văn bản *x*. Định nghĩa luật yếu *h* như sau:

$h(x) = c_0$ nếu $w \notin x$ và $h(x) = c_1$ nếu $w \in x$

Theo (Schapire &Singer)[14], có ba phương pháp lựa chọn luật yếu với thuật toán AdaBoost MH như sau:

6.3.2.1 AdaBoost.MH with discrete predictions :

Với cách thực hiện này , $c_j \left(j \in \{0,1\} \right)$ sẽ có giá trị +1 hoặc -1, với một luật *w* ta có thể cực tiểu giá trị Z_t bằng cách sau :

Đặt $X_0 = \{x : w \notin x\}$ và $X_1 = \{x : w \in x\}$

Với giá trị phân phối hiện tại là D_t, ta có những giá trị tương ứng với mỗi $j \in \{0,1\}$ và với mỗi $b \in \{-1,+1\}$ như sau:

$$W_b^j = \sum_{i=1}^{m} D_t(i) \left[\left[x_i \in X_j \wedge y_i = b \right] \right]$$

Như vậy W_b^j là trọng số, ứng với phân phối D_t của mẫu huấn luyện trong tập $X_j \left(j \in \{0,1\} \right)$ thuộc về loại $b \left(b \in \{-1,+1\} \right)$.

Thiết lập $c_j = sign\left(W_+^j - W_-^j \right)$

Đặt $r_t = \sum_{j \in \{0,1\}} |W_+^j - W_-^j|$

(Schapire &Singer, 1998)[12] chỉ ra rằng để cực tiểu giá trị Z_t, ta chọn

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 + r_t}{1 - r_t} \right)$$

Dẫn đến $Z_t = \sqrt{1-r^2}$

6.3.2.2 AdaBoost.MH with real -value predictions:

Khác với thuật toán AdaBoost vừa trình bày, ở đây $c_j\left(j \in \{0,1\}\right)$ có giá trị thực chứ không như phương pháp vừa nói là $c_j\left(j \in \{0,1\}\right)$có giá tri là +1 hoặc -1. Để cực tiểu giá trị Z_t, giá trị $c_j\left(j \in \{0,1\}\right)$ với mỗi luật được tính như sau:

Theo (Schapire &Singer,1998) [12], Z_t đạt giá trị cực tiểu nếu chọn

$$c_j = \frac{1}{2}\ln\left(\frac{W_+^j}{W_-^j}\right)$$

Thiết lập $\alpha_t = 1$, suy ra $Z_t = 2\sum_{j\in\{0,1\}}\sqrt{W_+^j W_-^j}$

Như vậy, luật yếu h_t được chọn sao cho giá trị $Z_t = 2\sum_{j\in\{0,1\}}\sqrt{W_+^j W_-^j}$ là nhỏ nhất, còn α_t trong trường hợp này là 1

Tuy nhiên, các giá trị W_+^j, W_-^j có thể rất nhỏ hay bằng 0, điều này sẽ dẫn đến các giá trị $c_j\left(j \in \{0,1\}\right)$ có giá trị rất lớn hay vô hạn.Trong thực tế những giá trị này có thể gây ra các vấn đề phức tạp trong tính toán, gây tràn số. Theo (Schapire &Singer)[14]để giới hạn các giá trị $c_j\left(j \in \{0,1\}\right)$ không quá lớn, $c_j\left(j \in \{0,1\}\right)$ sẽ được tính như sau :

$$c_j = \frac{1}{2}\ln\left(\frac{W_+^j + \varepsilon}{W_-^j + \varepsilon}\right)$$

Với $\varepsilon = \frac{1}{m}$

6.3.2.3 AdaBoost.MH with real -value predictions and abstainings

Thuật toán AdaBoost với giá trị dự đoán thực (AdaBoost.MH with real -value predictions) gán một giá trị biểu thị độ tin cậy trong cả hai trường hợp luật xuất hiện hay không. Như vậy nó ngầm cho rằng một luật không thoả trong văn bản cũng chứa đựng thông tin về loại của văn bản đó.Ta có thể loại bỏ giả thiết này và ép luật yếu không nhận giá trị gì khi luật không thoả văn bản. Điều này được thực hiện một cách đơn giản chỉ bằng cách gán cho mỗi luật yếu giá trị độ tin cậy là 0 nếu không thoả văn bản.

Với một luật *h*, thuật toán sẽ cho giá trị dự đoán c_1 với những văn bản (ở đây là email) thoả luật *h*, với các văn bản còn lại, giá trị dự đoán c_0 sẽ có giá trị là 0.Do đó, luật *h* sẽ không có tác dụng gì đến việc phân loại nếu văn bản không thỏa.

Thiết lập $\alpha_t = 1$

Xem $W_0 = \sum_{i, x_i \in X_0} D_t(i)$ là trọng số của tất cả các văn bản không thoả *h*

Theo (Schapire &Singer, 1998)[12].thì

$$Z_t = W_0 + 2 \sum_{j \in \{0,1\}} \sqrt{W_+^j W_-^j}$$

Như vậy ở mỗi bước chạy *t*, luật yếu được chọn sao cho $Z_t = W_0 + 2 \sum_{j \in \{0,1\}} \sqrt{W_+^j W_-^j}$ nhỏ nhất Một ưu điểm của phương pháp này so với cách thực hiện trước là cải thiện tốc độ thực hiện, thực tế tốc độ thực hiện của phương pháp này nhanh hơn 15% so với phương pháp thực hiện theo thuật toán. AdaBoost.MH with real -value predictions.

Chương 7 : THỰC HIỆN VÀ KIỂM THỬ PHÂN LOẠI EMAIL DỰA TRÊN PHƯƠNG PHÁP ADABOOST

7.1 Cài đặt bộ phân loại email dựa trên phương pháp AdaBoost:

Chúng tôi tiến hành cài đặt bộ phân loại email dựa trên thuật toán AdaBoost với ba cách

- Ø Cách 1 : cài đặt theo thuật toán AdaBoost MH With Discrete Value Prediction
- Ø Cách 2: cài đặt theo thuật toán AdaBoost MH With Real Value Prediction

Sau khi thực hiện, chúng tôi lưu lại T luật đã được chọn để phân loại cho các mẫu mới

Chúng tôi xây dựng một cấu trúc dữ liệu luật như sau :

```
Struct rule
{
    Token :chuỗi            //lưu token
    c0 :số thực             //giá trị của luật khi token không có trong
                            //email được xét
    c1 :số thực             // giá trị của luật khi token có trong email
                            //được xét
}
```

7.1.1 Tập huấn luyện mẫu và tập nhãn :

Tập huấn luyện mẫu chính là các email spam và email non-spam được dung để huấn luyện, tập nhãn là Y={-1,+1}, ở đây chúng tôi qui định -1 là spam và +1 là non-spam

7.1.2 Xây dựng tập luật yếu ban đầu :

Với mỗi token[8] w , định nghĩa $w \in$ x tương đương với *w* có trong email x.Định nghĩa luật yếu *h* như sau:

$h(x) = c_0$ nếu w $\notin$ x và $h(x) = c_1$ nếu w $\in$ x

Chúng tôi tiến hành cài đặt thử nghiệm thuật toán AdaBoost với hai cách khác nhau, do đó tương ứng với mỗi cách, cách lấy giá trị c_0 và c_1 khác nhau, các giá trị c_0, c_1 mà $h(x)$ có thể nhận được tính như đã nói ở các mục 6.3.2.1 và mục 6.3.2.2.

Số lượng của tập luật yếu được dùng để huấn luyện theo nguyên tắc là không hạn chế, như vậy chúng ta có thể lấy tất cả các token trong tập học. Tuy nhiên, chúng tôi nhận thấy để lấy hết tất cả các token thì rất mất thời gian và tốc độ huấn luyện cũng chậm đi, vì thế chúng tôi chỉ chọn ra một số các token thoả mãn một tiêu chí nào đó để xây dựng luật yếu. Mỗi luật yếu được chọn như sau :chúng tôi duyệt qua tất cả các mẫu học, tính số lần xuất hiện của mỗi token, những token có số lần xuất hiện lớn hơn một giá trị ngưỡng nào đó (được qui định) sẽ được lựa chọn, việc lựa chọn ngưỡng để quyết định luật có được chọn hay không tuỳ thuộc vào kho ngữ liệu học. Chúng tôi chia thành hai tập riêng, một tập gồm các token xuất hiện trong các email spam, tập kia gồm các token xuất hiện trong email non-spam.Cách xây dựng tập luật yếu như vậy làm giảm đáng kể số luật cần xét Khi huấn luyện, chúng tôi sẽ quyết định số lượng các luật yếu cần chọn, khi đó chúng tôi sẽ chọn tập luật yếu bằng cách lần lượt chọn một token chưa có trong tập được chọn từ tập các token spam, rồi lại chọn một token chưa có trong tập được chọn từ tập các token non-spam cho đến khi đủ số lượng yêu cầu

Để thực hiện việc duyệt các token và tìm kiếm một token với tốc độ nhanh, tương tự như thực hiện thuật toán huấn luyện Naïve Bayesian chúng tôi

[8] Xem định nghĩa token ở mục 5.1.1

cũng xây dựng bảng băm tương tự như bảng băm đã được sử dụng ở cách thực hiện theo phương pháp Naïve Bayesian.

7.1.3 Thủ tục WeakLearner chọn luật yếu:

Thủ tục WeakLearner được xây dựng nhằm tìm luật yếu h_t như sau : chọn luật yếu h_t ở bước chạy *t* sao cho Z_t nhỏ nhất, cách chọn Z_t và α_t đã được đề cập ở các mục 6.3.2.1 và 6.3.2.2

7.1.4 Phân loại email :

Khi nhận được một email *x*, chúng tôi sẽ tiến hành so khớp các luật từ kho ngữ liệu các luật được chọn sau quá trình huấn luyện , từ đó tính giá trị *f*(*x*), nếu *f*(*x*) >0 (cùng dấu với +1) chúng tôi cho email đó là non-spam, ngược lại (cùng dấu với -1) chúng tôi cho email đó là spam.

7.2 Thử nghiệm hiệu quả phân loại :

7.2.1 Thử nghiệm với kho ngữ liệu pu:

7.2.1.1 Kịch bản kiểm thử:

Vói mỗi phiên bản AdaBoost đã cài đặt, chúng tôi chọn tập luật yếu với số lượng là 2500 luật, những luật được xem là ứng cử viên nếu số lần xuất hiện của token lớn hơn hay bằng 10 lần. Nếu số luật yếu ban đầu không đủ 2500, chúng tôi sẽ lấy tất cả số sẵn có.Chúng tôi thử nghiệm với T lần lượt là 5, 10, 50, 100, 200 và 500.

Chúng tôi lần lượt kiểm thử với các pu, với mỗi pu, chúng tôi cho học từ part 1- đến part 9. Đối với việc kiểm thử chúng tôi kiểm thử trên kho ngữ liệu chưa được huấn luyện là part 10 của mỗi pu

7.2.1.2 Kết quả kiểm thử:

Chúng tôi trình bày kết quả kiểm thử với T=500, về chi tiết kết quả kiểm thử, xem phần phụ lục

- Kết quả thực hiện kiểm thử với thuật toán ADaBoost with real value predictions

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	48	0	58	3	100.00%	94.12%
			432	549	432	0	549	0	100.00%	100.00%
PU2	126	513	14	57	12	2	56	1	85.71%	92.31%
			126	513	126	0	513	0	100.00%	100.00%
PU3	1638	2079	182	231	176	6	216	15	96.70%	92.15%
			1638	2079	1638	0	2079	0	100.00%	100.00%
PUA	513	513	57	57	56	1	38	19	98.25%	74.67%
			513	513	513	0	513	0	100.00%	100.00%

Bảng 7-1 Kết quả thử nghiệm phân loại email với ngữ liệu số PU bằng thuật toán AdaBoost with real -value predictions

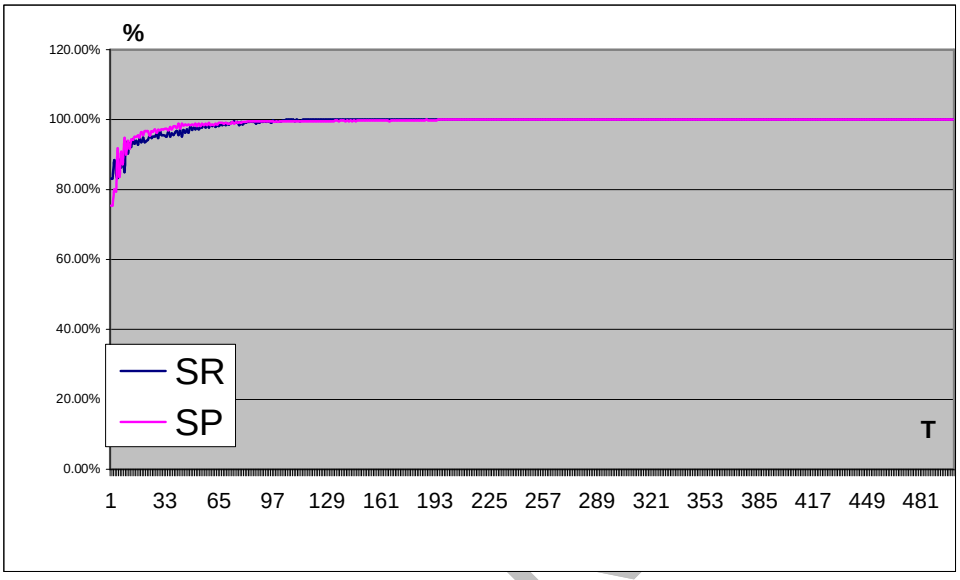
- Kết quả thực hiện kiểm thử với thuật toán ADaBoost with discrete predictions

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	46	2	57	4	95.83%	92.00%
			432	549	432	0	549	0	100.00%	100.00%
PU2	126	513	14	57	13	1	57	0	92.86%	100.00%
			126	513	126	0	513	0	100.00%	100.00%
PUA	513	513	57	57	53	4	45	12	92.98%	81.54%
	513	513	513	513	513	0	513	0	100.00%	100.00%
PU3	1638	2079	182	231	173	9	216	15	95.05%	92.02%
			1638	2079	1624	14	2074	5	99.15%	99.69%

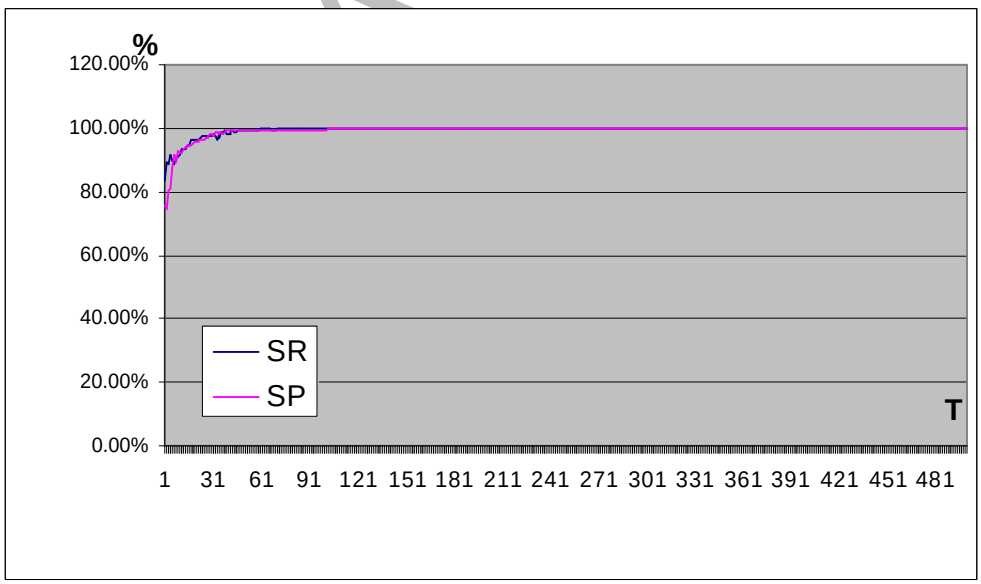
Bảng 7-2 Kết quả thử nghiệm phân loại email với ngữ liệu số PU bằng thuật toán AdaBoost with discrete predictions

Nhận xét : theo Schapire & Singer [14], hiệu quả phân loại của thuật toán AdaBoost with real value predictions cao hơn của thuật toán AdaBoost with discrete predictions, tuy nhiên ở đây ta thấy điều đó không rõ rệt. Hiệu quả phân loại của cả hai thuật toán trên các kho ngữ liệu là khá cao. Với thuật toán AdaBoost, lỗi phân loại sai trên các kho ngữ liệu đã huấn luyện sẽ ngày càng giảm khi T ngày càng tăng, tương ứng với các chỉ số

spam recall và spam precision ngày càng tăng, dưới đây là biểu đồ thể hiện điều đó



Hình 7-1 Đồ thị biểu diễn sự biến thiên của spam recall (SR) và spam precision (SP) theo T (thuật tóan AdaBoost.MH with discrete predictions)



Hình 7-2 Đồ thị biểu diễn sự biến thiên của spam recall (SR) và spam precision (SP) theo T (thuật tóan AdaBoost MH with real value predictions)

7.2.2 Thử nghiệm với kho ngữ liệu email chữ:

7.2.2.1 Kịch bản kiểm thử:

Chúng tôi thử nghiệm hai thuật toán AdaBoost đã cài đặt với T được chọn lần lượt là 5, 10, 50, 100, 200, và 500.

7.2.2.2 Kết quả kiểm thử:

Ngữ liệu email văn bản trơn, số email kiểm thử : Spam =98, non-spam=100

Ngữ liệu email html, số email kiểm thử :Spam =50, non-spam=50

- Kết quả thực hiện kiểm thử với thuật toán ADaBoost with real value predictions

Ngữ liệu		T=5	T=10	T=50	T=100	T=200	T=500
HTML	S à S	48	48	49	49	49	49
	S à N	2	2	1	1	1	1
	N à N	49	49	49	49	49	49
	N à S	1	1	1	1	1	1
	SR	96.00%	96.00%	98.00%	98.00%	98.00%	98.00%
	SP	97.96%	97.96%	98.00%	98.00%	98.00%	98.00%
TEXT	S à S	84	93	98	98	98	98
	S à N	14	5	0	0	0	0
	N à N	98	97	98	99	99	99
	N à S	2	3	2	1	1	1
	SR	85.71%	94.90%	100.00%	100.00%	100.00%	100.00%
	SP	97.67%	96.88%	98.00%	98.99%	98.99%	98.99%

Bảng 7-3 kết quả thử nghiệm phân loại email với ngữ liệu email chữ bằng thuật toán AdaBoost with real-value predictions

- Kết quả thực hiện kiểm thử với thuật toán ADaBoost with discrete predictions

Ngữ liệu		T=5	T=10	T=50	T=100	T=200	T=500
HTML	S à S	48	49	50	50	50	50
	S à N	2	1	0	0	0	0
	N à N	49	49	49	49	49	49
	N à S	1	1	1	1	1	1
	SR	96.00%	98.00%	100.00%	100.00%	100.00%	100.00%
	SP	97.96%	98.00%	98.04%	98.04%	98.04%	98.04%

TEXT	S à S	91	91	95	97	96	97
	S à N	7	7	3	1	2	1
	N à N	98	98	98	98	99	99
	N à S	2	2	2	2	1	1
	SR	92.86%	92.86%	96.94%	98.98%	97.96%	98.98%
	SP	97.85%	97.85%	97.94%	97.98%	98.97%	98.98%

Bảng 7-4 Kết quả thử nghiệm phân loại email với ngữ liệu email chữ bằng thuật toán AdaBoost with discrete predictions

Nhận xét : hiệu quả phân loại trên ngữ liệu email là chữ của thuật toán AdaBoost khá tốt, so với phương pháp phân loại Naïve Bayesian thì ADaBoost phân loại email html tốt hơn, hiệu quả phân loại trên email là văn bản trơn cũng tương đương với Naïve Bayesian.

7.3 Ưu – nhược điểm của phương pháp phân loại AdaBoost:

7.3.1 Ưu điểm :

- Một ưu điểm của AdaBoost giống với phương pháp phân loại Naïve Bayes là nó cho phép học cập nhật, nghĩa là khi một email spam vượt qua được bộ lọc thì người dung có thể đánh dấu email đó là spam và huấn luyện lại bộ lọc
- Hiệu quả phân loại là khá cao
- Việc lưu trữ tập luật đã qua huấn luyện khá gọn nhẹ, trong khi đó với phương pháp phân loại Naïve Bayes thì dữ liệu sau khi học là khá lớn n. Với phương pháp phân loại Naïve Bayesian, dữ liệu huấn luyện sẽ phình to sau mỗi lần huấn luyện cập nhật thêm, điều này với cách thực hiện theo phương pháp AdaBoost là không đáng kể.

7.3.2 Khuyết điểm :

- Cũng giống như các phương pháp máy học của phương pháp phân loại dựa trên thuật toán AdaBoost chính là việc phải huấn luyện cho nó, việc huấn luyện hiệu quả hay không còn phải phụ thuộc vào kho ngữ liệu huấn luyện ban đầu

- Khuyết điểm thứ hai là thời gian huấn luyện, so với Naïve Bayesian, để huấn luyện cùng một kho ngữ liệu thì phương pháp AdaBoost cần thời gian lâu hơn rất nhiều, theo chúng tôi nhận thấy thì sự chênh lệch ấy khá lớn.

Chương 8 : XÂY DỰNG CHƯƠNG TRÌNH MAIL CLIENT TIẾNG VIỆT HỖ TRỢ PHÂN LOẠI EMAIL

8.1 Chức năng:

Chúng tôi xây dựng phần mềm Mail Client với các chức năng chính như sau:

- Ø Chức năng gởi nhận email
- Ø Luu trữ email tương ứng với từng mục
- Ø Soạn email
- Ø Xây dựng sổ địa chỉ
- Ø Lọc email spam
- Ø Quản lý email như sao chép, chuyển, xóa … email
- Ø Và một số công cụ hỗ trợ khác khác : …

Để hỗ trợ cho việc kiểm thử Mail Client chúng tôi xây dựng chương trình Flood Mail gởi mail hàng loạt đến một địa chỉ nhận nào đó.

8.2 Xây dựng bộ lọc email spam :

Chúng tôi sử dụng bộ lọc dựa trên thuật toán học Naïve Bayes và AdaBoost, với Naivê Bayes chúng tôi sử dụng cách cài đặt theo cách tính xác suất spam cho mỗi token dựa trên số lần xuất hiện trong tập huấn luyện ban đầu, chọn số token để duyệt một email là 15, chọn $\lambda = 9$ do đó ngưỡng phân loại email spam là t=0.9. Với bộ lọc dựa trên AdaBoost chúng tôi chọn cách cài đặt theo AdaBoost.MH with real value predictions. Chúng tôi xây dựng thành các component tích hợp vào chương trình dưới dạng các dll.

Chúng tôi cũng xây dựng chức năng lọc email theo phương pháp BlackList và luậ do người dùng tự định nghĩa, phương pháp này sẽ hỗ trợ cho bộ lọc email ngăn chặn email spam.

8.3 Tổ chức dữ liệu cho chương trình :

Dữ liệu chương trình :gồm nội dung các email, các luật do người dùng thiết lập.

Lưu trữ nội dung các email gởi và nhận : được lưu dưới dạng các tập tin văn bản, với mỗi thư mục tương ứng như hộp thư đến, hộp thư đi,.. sẽ có một tập tin lưu nội dung các email trong các thu mục này, lưu trữ dưới dạng xml, cấu trúc tập tin như sau :

<?xml version=1.0?”
<MessageList NumberUnReadMail="1">
<Message MessageID=” ”>
<From>…</From>
<To>…</To>
<Cc> …</Cc>
<BCC>…</BCC>
<Subject>… </Subject>
<Body>…</Body>
<Date>…</Date>
<Attach>…</Attach>
<Priority>…</Priority>
<Read>…</Read>
</Message>
……
</MessageList>

Các thông tin liên quan đến một email mà chúng tôi lưu trữ gồm có : thuộc tính định danh email (trường MessageID), tiêu đề email (Subject), địa chỉ người gởi (trường from), địa chỉ đồng gởi (trường Cc), địa chỉ đồng gởi ẩn (trường Bcc), nội dung email (trường body),có đính kèm tập tin (trường Attach), mức độ quan trọng (Prority), ngày tháng (Date)

Các luật do người dùng thiết lập cũng được lưu trữ duới dạng xml

- Ưu điểm của cách tổ chức dữ liệu xml:

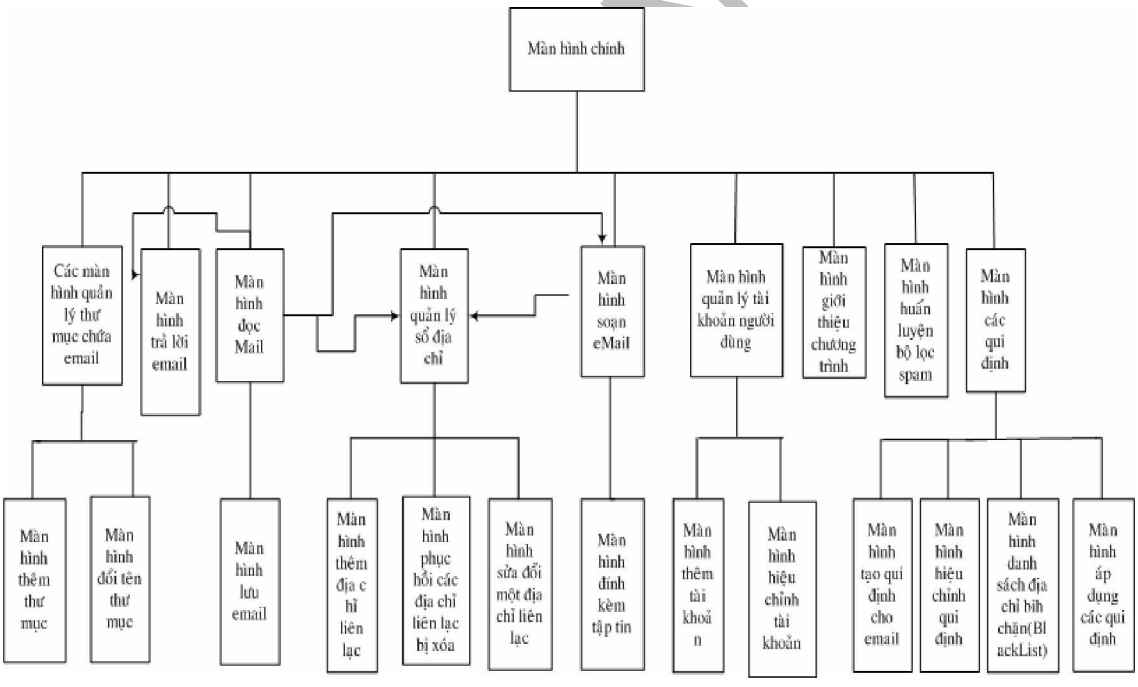
Xml là cách lưu trữ dữ liệu được tổ chức với cấu trúc cây, xml được các ngôn ngữ lập trình hiện đại hỗ trợ khá tốt, như vậy việc thao tác với dữ liệu chương trình rất thuận lợi.

Xml là chuẩn giao tiếp giữa các hệ thống với các cách lưu trữ dữ liệu khác nhau, sử dụng xml tiện lợi cho việc giao tiếp với hệ thống bên ngoài như chuyển đổi hay thu nhận thông tin.

- Khuyết điểm :Dữ liệu được lưu dưới dạng văn bản, không bảo mật

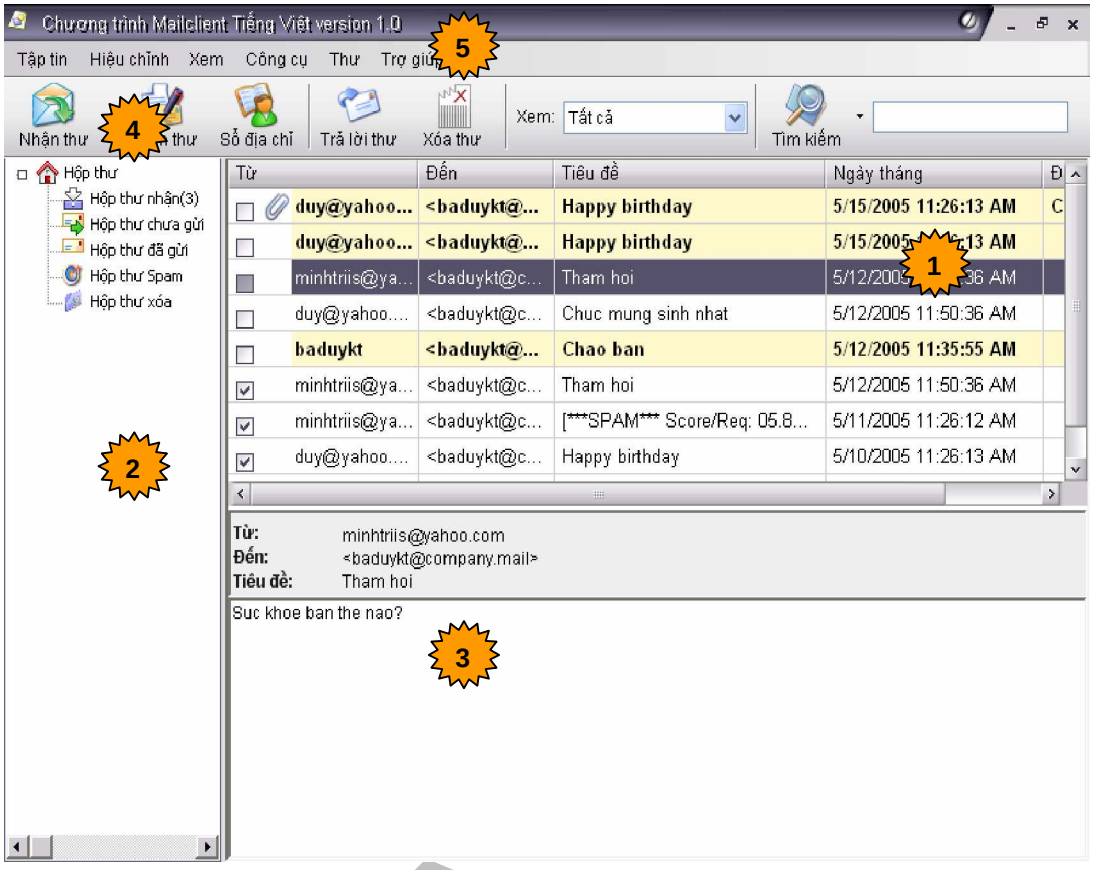
8.4 Giao diện người dùng :

8.4.1 Sơ đồ màn hình :



Hình 8-1:Sơ đồ màn hình của chương trình

8.4.2 Một số màn hình chính :



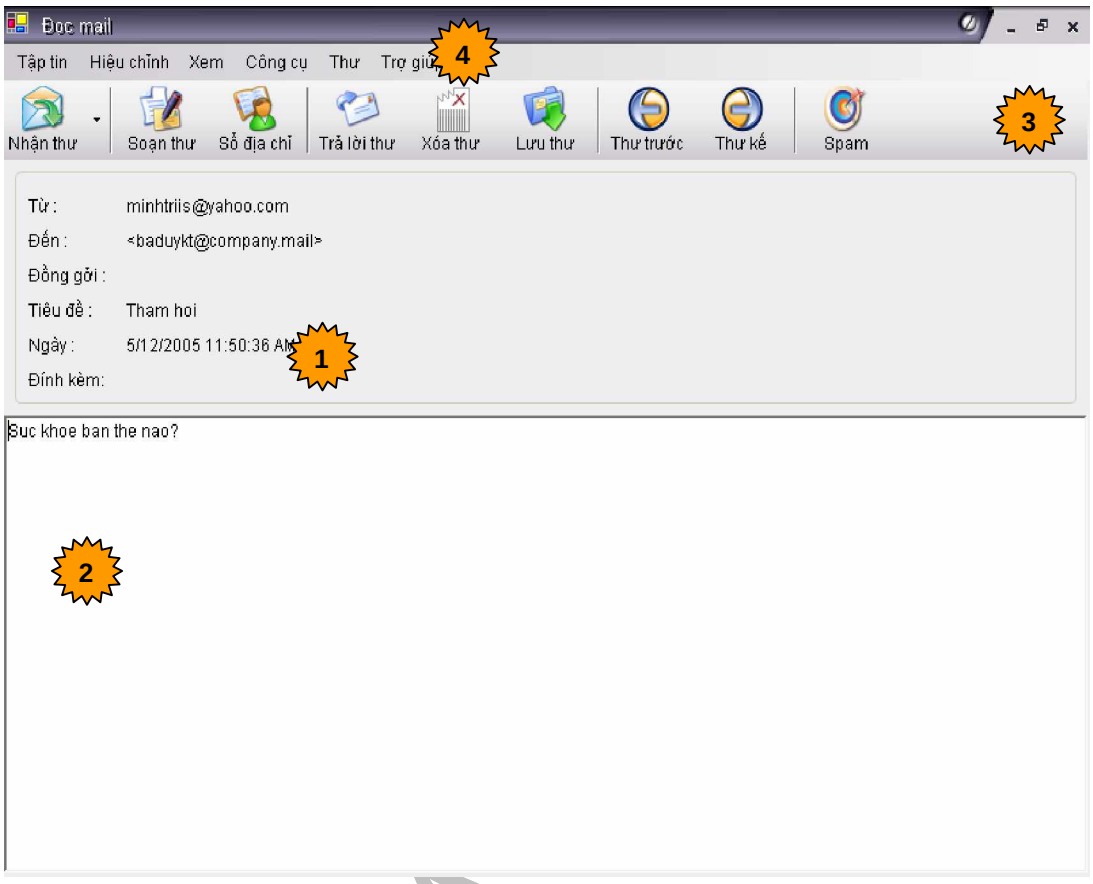
Hình 8-2 Màn hình chính của chương trình Mail Client

Bảng chú thích cho màn hình chính:

	Diễn giải
1	Lưới hiển thị danh sách các email gửi cho người dùng. Các sự kiện đi kèm: § *Nhấp đơn chuột trên dòng*: Đọc email nhanh. § *Nhấp đôi chuột trên dòng*: Đọc email chi tiết. § ☑ *Đánh dấu chọn trên dòng*: Đánh dấu email cần xóa. § : Thông báo thư có đính kèm. § : Thông báo email chưa đọc.
2	Khung hiển thị cây thư mục lưu trữ các hộp thư của người dùng:

	§ *Hộp thư nhận*: Lưu thư gửi đến cho người dùng. §  *Hộp thư chưa gửi*: Lưu thư đã soạn nhưng chưa gửi § *Hộp thư đã gửi*: Lưu thư đã được gửi đi. §  *Hộp thư spam*: Lưu thư spam (tự động) §  *Hộp thư xóa*: Lưu thư bị xóa bởi người dùng.
3	Khung hiển thị nhanh nội dung email khi người dùng click chọn một email trên lưới hiển thị danh sách email.
4	Thanh công cụ. § Nhận thư: Nhận thư từ email server. § Soạn thư: Soạn thư mới. § Sổ địa chỉ: Tra cứu sổ địa chỉ liên lạc. § Trả lời thư: Soạn thư trả lời. § Xóa thư: Xóa các thư được đánh dấu chọn. § Xem: Hiển thị danh sách email trên lưới theo tiêu chí xem. § Tìm kiếm: Tìm kiếm email theo tiêu đề /nội dung /người gởi.
5	Thực đơn chính. Tập tin: § Tạo mới thư: Soạn thư mới. § Tạo mới thư mục: Tạo thư mục mới (hộp thư mới) trên cây thư mục. § Đổi tên thư mục: Đổi tên thư mục (hộp thư) trên cây thư mục. § Xóa thư mục: Xóa thư mục (hộp thư) trên cây thư mục (Xóa luôn nội dung bên trong thư mục). § Mở thư đã lưu: Mở thư đã lưu dạng tập tin (.eml) Hiệu chỉnh: § Chọn tất cả: Chọn tất cả thư trên lưới hiển thị thư gửi cho

	người dùng. § Tìm kiếm: Tìm kiếm email theo tiêu đề /nội dung /người gởi. § Chuyển đến thư mục: Chuyển thư đến thư mục được chọn § Sao chép đến thư mục: Tạo bản sao thư đến thư mục được chọn. § Xóa thư: Xoá thư được chọn. § Xóa thư trong thư mục xóa: Xóa tất cả thư có trong hộp thư xoá. Công cụ: § Sổ địa chỉ: Tra cứu danh sách địa chỉ liên lạc. § Thêm liên lạc: Thêm liên lạc mới(tên liên lạc, địa chỉ email…) § Qui định (Rules): Qui định lọc thư tới vào thư mục định trước (hoặc xóa). Thư: § Soạn thư mới: Soạn thư để gửi đi. § Trả lời thư: Trả lời thư đến người gửi thư tới. § Thêm qui định: Thêm qui định nhận thư gửi tới. § Chặn người gửi: Không nhận thư của người gửi có trong danh sách. Trợ giúp: § Giới thiệu: Người thực hiện. § Hương dẫn: Hướng dẫn sử dụng chương trình.

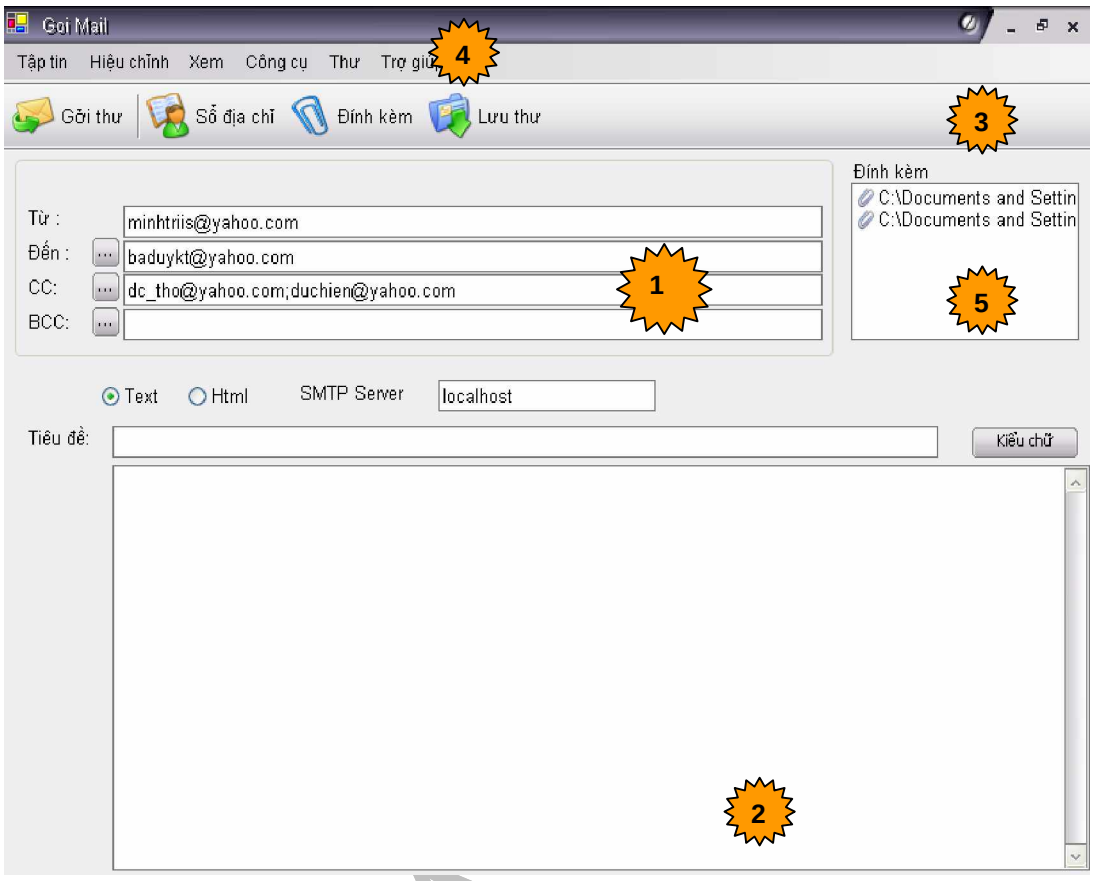


Hình 8-3 Màn hình "Đọc email"

- **Bảng chú thích cho màn hình "Đọc email":**

Mã	Diễn giải
1	Hiển thị thông tin về email.
2	Khung hiển thị nội dung email.
3	Thanh công cụ. § Nhận thư: Nhận thư từ email server. § Soạn thư: Soạn thư mới. § Sổ địa chỉ: Tra cứu sổ địa chỉ liên lạc. § Trả lời thư: Soạn thư trả lời. § Xóa thư: Xóa các thư được đánh dấu chọn. § Lưu thư: Lưu thư xuống ổ cứng dạng tập tin(**.eml**).

	§ Thư trước: Đọc thư liền trước. § Thư kế:Đọc thư liền sau. § Spam: Đánh dấu spam vượt qua bộ lọc (Yêu cầu học spam).
4	Thực đơn chính. Tập tin: § Tạo mới thư: § Tạo mới liên lạc: § Mở thư đã lưu: § Lưu thư: Hiệu chỉnh: § Tìm kiếm thư: § Chuyển đến thư mục: § Sao chép đến thư mục: § Xóa thư: Xóa thư được chọn. Công cụ: § Sổ địa chỉ: § Thêm liên lạc: § Qui định (Rules): Thư: § Soạn thư mới § Trả lời thư: § Thư trước: § Thư kế: § Lưu tập tin đính kèm: Lưu tập tin đính kèm trong thư xuống đĩa cứng. § Xóa tập tin đính kèm: Xóa tập tin đính kèm được chọn trong danh sách đính kèm.



Hình 8-4 Màn hình gởi email

∨ **Bảng chú giải cho màn hình "Gởi email":**

Mã	Diễn giải
1	Khung nhập thông tin về email: gửi từ đâu, gửi đến đâu, gửi cho nhiều người (CC), gửi nhiều người nhưng ẩn địa chỉ gửi (BCC). Chức năng đi kèm: § Chọn địa chỉ gửi đến từ danh sách. § Chọn danh sách địa chỉ gửi cùng lúc. § Chọn danh sách địa chỉ gửi cùng lúc (ẩn địa chỉ người gửi).
2	Khung soạn thảo email.

3	Thanh công cụ. § Gửi thư: Thực hiện gửi thư đến người nhận. § Sổ địa chỉ: Tra cứu sổ địa chỉ liên lạc. § Lưu thư: Lưu thư xuống ổ cứng dạng tập tin(**.eml**). § Đính kèm: Mở và thêm tập tin đính kèm.
4	Thực đơn chính. Tập tin: § Tạo thư mới: § Mở thư đã lưu: § Lưu thư: § Lưu mới thư: Lưu lại thư xuống đĩa cứng với tên mới. Hiệu chỉnh: § Chọn tất cả: Chọn tất cả nội dung văn bản (text). § Tìm kiếm thư: § Chuyển đến thư mục: § Sao chép đến thư mục: § Kiểu chữ: Chọn kiểu chữ cho văn bản soạn. Xem: § Hiển thị thanh công cụ: Chọn hiển thị hay ẩn thanh công cụ. Công cụ: § Sổ địa chỉ: § Thêm liên lạc: Thư: § Soạn thư mới: § Lưu thư: § Gởi thư: Gửi thư đến người nhận. § Thêm tập tin đính kèm: Thêm tập tin đinh kèm vào trong thư gửi đi.

	§ Xóa tập tin đính kèm: Trợ giúp: § Giới thiệu: § Hương dẫn:
5	Danh sách tập tin đính kèm sẽ gửi.

Chương 9 : TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

9.1 Các việc đã thực hiện được :

Trong khoá luận này chúng tôi đã trình bày các hướng nghiên cứu, tiếp cận trong phân loại email và chống spam. Chúng tôi cũng đã tập trung đi sâu vào hướng tiếp cận phân loại email dựa trên nội dung.Ở đây chúng tôi trình bày hai phương pháp phân loại email khá mới và hiệu quả là phân loại email dựa trên thuật toán huấn luyện Naïve Bayes và dựa trên thuật toán AdaBoost.Kết quả thử nghiệm với dữ liệu số và dữ liệu văn bản trơn là khá hiệu quả, tuy nhiên đối với email html thì vẫn chưa được như mong muốn, điều này là do kho ngữ liệu email html chưa đủ lớn, mặt khác email html có những đặc điểm của riêng nó mà chúng tôi chưa khắc phục được như nội dung chủ yếu là các hình ảnh.

Chúng tôi cũng đã xây dựng thử nghiệm phần mềm Mail Client hỗ trợ lọc email. Bộ lọc email được tính hợp vào chương trình được xây dựng theo những hướng đã tiếp cận.Chương trình hỗ trợ một số chức năng chính của một phần mềm Mail Client thông thường như gởi, nhận email, tìm kiếm, quản lý email.....

9.2 Hướng cải tiến, mở rộng :

Vì thời gian có hạn, do đó vẫn còn những điều chúng tôi muốn thực hiện nhưng chưa thể thực hiện được.Dựa trên những kết quả đã đạt được, chúng tôi đề xuất những hướng cải tiến, mở rộng cho chương trình

9.2.1 Về phân loại và lọc email spam:

a) Về cách rút trích các token :

Có thể cải tiến cách lấy token, thay vì cách chọn token đơn, có thể chọn token như là một ngữ (gồm nhiều từ) – token gồm hai hay nhiều token đơn tạo thành, điều này giúp việc nhận biết chính xác hơn.

b) Mở rộng với email là tiếng Việt thay vì chỉ thực hiện với email tiếng Anh , tuy nhiên vấn đề phân loại email tiếng Việt có một số điểm khó khăn là không có sẵn một kho ngữ liệu email tiếng Việt phục vụ cho việc học Thêm nữa tiếng Việt là một tương đối ngôn ngữ phức tạp và đa dạng, do đó việc phân loại email tiếng Việt lại liên quan đến vấn đề tách từ (tách token), đây là bài toán phức tạp.

c) Có thể xây dựng bộ lọc thành các phần mềm riêng rẽ và tích hợp (plug in) vào các phần mềm email Client hiện có như Outlook Express, Mozzila ThunderBird.

d) Áp dụng bộ lọc email tại mức Server, ngăn chặn email spam ngay tại các Server email.

e) Có thể sử dụng kết hợp hai bộ lọc theo hai phương pháp Naïve Bayesian và AdaBoost, khi đó việc xây dựng tập luật yếu dùng để chọn lọc ban đầu có thể dựa vào những token có xác suất spam cao và xác suất non-spam thấp từ dữ liệu huấn luyện của Naïve Bayesian.

9.2.2 Về chương trình Mail Client:

Chương trình hiện chỉ mới được xây dựng với một vài chức năng chính, vẫn còn nhiều hạn chế. Với mong muốn xây dựng hoàn thiện một phần mềm Mail Client hỗ trợ tiếng Việt thì bên cạnh việc hoàn thiện những cái đã có , chúng tôi dự định xây dựng thêm một số chức năng:

- Ø Hỗ trợ bảo mật : dữ liệu của chương trình được lưu dạng tập tin văn bản, điều đó không bảo mật. Có thể cài tiến điều này bằng cách mã hoá tập tin, lưu dưới dạng nhị phân
- Ø Hỗ trợ nhiều tài khoản (Account) trên MailClient, hiện tại chương trình chỉ hỗ trợ một tài khoản .

TÀI LIỆU THAM KHẢO

Tiếng Việt :

[4] Hoàng Hoài Sơn, Thư rác nỗi khổ chung, báo THể thao Văn hoá, số 28 6-4-2004, Tr 34.

[8] Đặng Hấn (1992), "Xác suất thống kê ", Nhà xuất bản Giáo Dục

Tiếng Anh :

[1] Monty Python's Flying Circus. *Just the words*, volume 2, chapter 25, pages 27–28.Methuen, London, 1989.

[2] B. Leiba and N. Borenstein. A Multi-Faceted Approach to Spam Prevention, *Proceedings of the First Conference on E-mail and Anti-Spam*, 2004.

[3] Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras

and Constantine D. Spyropoulos, An Evaluation Bayes Antispam Filtering, *Proceedings of the workshop on Machine Learning in the New Information Age*

[5] P.Graham, Stopping Spam, *http://paulgraham.com/stoppingspam.html*, August 2003

[6] Flavio D. Garcia.Spam Filter Analysis Arxiv. *preprint cs.CR/0402046, 2004 - arxiv.org*

[7] P. Graham, A Plan for Spam, *http://paulgraham.com/spam.html*, August 2002

[9] M. Sahami, S. Dumais, D. Heckerman and E. Horvitz. A Bayesian Approach to Filtering Junk E-Mail *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[10]A short Introduction to Boosting *Journal of Japanese Society for Artificial Intelligence, 14(5):771-780, September, 1999*

[11] Meir, R., and Ratsch, G. 2003. An introduction to boosting and leveraging. Advanced lectures on machine learning, Springer-Verlag New York, Inc., New York, NY

[12] Schapire, R. E. and Y. Singer (1998). Improved boosting algorithms using confidence-rated predictions. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*.

[13] Carreras, X., and Marquez, L. (2001) Boosting trees for anti-spam email filtering. *In Proceedings of RANLP-01, 4th International Conference on Recent Advances in Natural Language Processing*.

[14] Robert E. Schapire and Yoram Singer. BoosTexter : A boosting-based system for text categorization. *MachineLearning*.135-168, 2000

[15] Schapire, R. (2001) The boosting approach to machine learning: an overview. *In MSRI Workshop on Nonlinear Estimation and Classification*

[16] Charles Elkan, Boosting and Naive Bayesian learning. Technical Report CS97-557, University of California, San Diego, 1997

[17]Androutsopoulos.I., et al.(2000) Learning to filter spam e-mail : acomparison of a NaiveBayesian and A memory-based approach. *In 4th PKDD'sWorkshop on MachineLearning and Textual Information Access.*

[18] I.Androutsopoulos,G.Paliouras,and E.Michelakis.Learning to filter unsolicited commercial e-mail.Technical report,National Centre for Scientific Research"Demokritos",2004.

Phụ lục

Phụ lục 1 : Kết quả thử nghiệm phân loại email bằng phương pháp Bayesian với kho ngữ liệu học và kiểm thử pu

Kết quả thử nghiệm nhân trọng số non-spam W=1:

Kết quả thử nghiệm với PU1:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	47	47	48	47	48	48	48	48	48
	S→N	1	1	0	1	0	0	0	0	0
	N→N	60	60	60	60	60	59	59	59	59
	N→S	1	1	1	1	1	2	2	2	2
	SR	97.92%	97.92%	100.00%	97.92%	100.00%	100.00%	100.00%	100.00%	100.00%
	SP	97.92%	97.92%	97.96%	97.92%	97.96%	96.00%	96.00%	96.00%	96.00%
	TCR	24	24	48	24	48	48	24	24	24
9	S→S	47	47	48	47	48	48	48	48	48
	S→N	1	1	0	1	0	0	0	0	0
	N→N	61	61	60	60	61	60	59	59	59
	N→S	0	0	1	1	0	1	2	2	2
	SR	97.92%	97.92%	100.00%	97.92%	100.00%	100.00%	100.00%	100.00%	100.00%
	SP	100.00%	100.00%	97.96%	97.92%	100.00%	97.96%	96.00%	96.00%	96.00%
	TCR	48	48	5.333333	4.8	#DIV/0!	5.333333	2.666667	2.666667	2.666667
999	S→S	47	47	48	46	47	48	48	48	48
	S→N	1	1	0	2	1	0	0	0	0
	N→N	61	61	60	61	61	60	59	59	60
	N→S	0	0	1	0	0	1	2	2	1
	SR	97.92%	97.92%	100.00%	95.83%	97.92%	100.00%	100.00%	100.00%	100.00%
	SP	100.00%	100.00%	97.96%	100.00%	100.00%	97.96%	96.00%	96.00%	97.96%
	TCR	48	48	0.048048	24	48	0.048048	0.024024	0.024024	0.048048

Kết quả thử nghiệm với PU2:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	9	10	11	10	10	13	11	11	11
	S→N	5	4	3	4	4	1	3	3	3
	N→N	56	57	57	57	57	57	56	56	56
	N→S	1	0	0	0	0	0	1	1	1
	SR	64.29%	71.43%	78.57%	71.43%	71.43%	92.86%	78.57%	78.57%	78.57%
	SP	90.00%	100.00%	100.00%	100.00%	100.00%	100.00%	91.67%	91.67%	91.67%
	TCR	2.333333	3.5	4.666667	3.5	3.5	14	3.5	3.5	3.5
9	S→S	9	9	11	10	10	12	11	11	11
	S→N	5	5	3	4	4	2	3	3	3
	N→N	56	57	57	57	57	57	56	56	56
	N→S	1	0	0	0	0	0	1	1	1
	SR	64.29%	64.29%	78.57%	71.43%	71.43%	85.71%	78.57%	78.57%	78.57%
	SP	90.00%	100.00%	100.00%	100.00%	100.00%	100.00%	91.67%	91.67%	91.67%
	TCR	1	2.8	4.666667	3.5	3.5	7	1.166667	1.166667	1.166667
999	S→S	9	9	10	8	10	10	11	11	11
	S→N	5	5	4	6	4	4	3	3	3
	N→N	56	57	57	57	57	57	56	56	56
	N→S	1	0	0	0	0	0	1	1	1
	SR	64.29%	64.29%	71.43%	57.14%	71.43%	71.43%	78.57%	78.57%	78.57%
	SP	90.00%	100.00%	100.00%	100.00%	100.00%	100.00%	91.67%	91.67%	91.67%
	TCR	0.013944	2.8	3.5	2.333333	3.5	3.5	0.013972	0.013972	0.013972

Kết quả thử nghiệm với PU3:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	177	178	178	178	179	178	174	178	178
	S→N	5	4	4	4	3	4	8	4	4
	N→N	215	210	206	214	206	207	215	211	208
	N→S	16	21	25	17	25	24	16	20	23
	SR	97.25%	97.80%	97.80%	97.80%	98.35%	97.80%	95.60%	97.80%	97.80%
	SP	91.71%	89.45%	87.68%	91.28%	87.75%	88.12%	91.58%	89.90%	88.56%
	TCR	8.666667	7.28	6.275862	8.666667	6.5	6.5	7.583333	7.583333	6.740741
9	S→S	175	178	178	178	178	178	173	178	178
	S→N	7	4	4	4	4	4	9	4	4
	N→N	218	213	211	218	212	209	216	211	208
	N→S	13	18	20	13	19	22	15	20	23
	SR	96.15%	97.80%	97.80%	97.80%	97.80%	97.80%	95.05%	97.80%	97.80%
	SP	93.09%	90.82%	89.90%	93.19%	90.36%	89.00%	92.02%	89.90%	88.56%
	TCR	1.467742	1.096386	0.98913	1.504132	1.04	0.90099	1.263889	0.98913	0.862559
999	S→S	173	176	177	175	175	177	172	177	177
	S→N	9	6	5	7	7	5	10	5	5
	N→N	222	219	216	222	218	215	219	214	215
	N→S	9	12	15	9	13	16	12	17	16
	SR	95.05%	96.70%	97.25%	96.15%	96.15%	97.25%	94.51%	97.25%	97.25%
	SP	95.05%	93.62%	92.19%	95.11%	93.09%	91.71%	93.48%	91.24%	91.71%
	TCR	0.020222	0.015174	0.012141	0.020227	0.014006	0.011383	0.015169	0.010713	0.011383

Kết quả thử nghiệm với PUA:

		Công thức 5-5			Công thức 5-6			Công thức 5-7		
λ		10	15	20	10	15	20	10	15	20
1	S→S	57	56	56	56	56	55	56	56	56
	S→N	0	1	1	1	1	2	1	2	1
	N→N	55	53	54	56	55	55	54	54	53
	N→S	2	4	3	1	2	2	3	3	4
	SR	100.00%	98.25%	98.25%	98.25%	98.25%	96.49%	98.25%	96.55%	98.25%
	SP	96.61%	93.33%	94.92%	98.25%	96.55%	96.49%	94.92%	94.92%	93.33%
	TCR	28.5	11.4	14.25	28.5	19	14.25	14.25	11.6	11.4
9	S→S	56	56	56	54	55	55	55	55	55
	S→N	1	1	1	3	2	2	2	2	2
	N→N	56	53	54	56	55	55	54	54	53
	N→S	1	4	3	1	2	2	3	3	4
	SR	98.25%	98.25%	98.25%	94.74%	96.49%	96.49%	96.49%	96.49%	96.49%
	SP	98.25%	93.33%	94.92%	98.18%	96.49%	96.49%	94.83%	94.83%	93.22%
	TCR	5.7	1.540541	2.035714	4.75	2.85	2.85	1.965517	1.965517	1.5
999	S→S	52	54	54	52	51	54	55	55	55
	S→N	5	3	3	5	6	3	2	2	2
	N→N	56	54	54	56	55	56	55	54	53
	N→S	1	3	3	1	2	1	2	3	4
	SR	91.23%	94.74%	94.74%	91.23%	89.47%	94.74%	96.49%	96.49%	96.49%
	SP	98.11%	94.74%	94.74%	98.11%	96.23%	98.18%	96.49%	94.83%	93.22%
	TCR	0.056773	0.019	0.019	0.056773	0.028443	0.056886	0.0285	0.019006	0.014257

Phụ lục 2 : Kết quả thử nghiệm phân loại email bằng phương pháp AdaBoost với kho ngữ liệu học và kiểm thử pu

1. Kết quả thực hiện với thuật toán AdaBoost with real value predictions:

a) T=500

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	48	0	58	3	100.00%	94.12%
			432	549	432	0	549	0	100.00%	100.00%
PU2	126	513	14	57	12	2	56	1	85.71%	92.31%
			126	513	126	0	513	0	100.00%	100.00%
PU3	1638	2079	182	231	176	6	216	15	96.70%	92.15%
			1638	2079	1638	0	2079	0	100.00%	100.00%
PUA	513	513	57	57	56	1	38	19	98.25%	74.67%
			513	513	513	0	513	0	100.00%	100.00%

b) T=200

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	48	0	58	3	100.00%	94.12%
			432	549	432	0	549	0	100.00%	100.00%
PU2	126	513	14	57	12	2	57	0	85.71%	100.00%
			126	513	126	0	513	0	100.00%	100.00%
PU3	1638	2079	182	231	178	4	217	14	97.80%	92.71%
			1638	2079	1634	4	2079	0	99.76%	100.00%
PUA	513	513	57	57	56	1	40	17	98.25%	76.71%
			513	513	513	0	513	0	100.00%	100.00%

c) T=100

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	48	0	59	2	97.96%	96.00%
			432	549	432	0	549	0	100.00%	100.00%
PU2	126	513	14	57	12	2	56	1	85.71%	92.31%
			126	513	126	0	513	0	100.00%	100.00%
PU3	1638	2079	182	231	174	8	215	16	95.60%	91.58%
			1638	2079	1618	20	2067	12	98.78%	99.26%
PUA	513	513	57	57	56	1	38	19	98.25%	74.67%
			513	513	513	0	513	0	100.00%	100.00%

d) T=50

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	47	1	57	4	97.92%	92.16%
			432	549	431	1	547	2	99.77%	99.54%
PU2	126	513	14	57	11	3	57	0	78.57%	100.00%
			126	513	126	0	513	0	100.00%	100.00%
PU3	1638	2079	182	231	174	8	214	17	95.60%	91.10%
			1638	2079	1592	46	2046	33	97.19%	97.97%
PUA	513	513	57	57	57	0	37	20	100.00%	74.03%
			513	513	512	1	510	3	99.81%	99.42%

e) T=10

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	45	3	56	5	93.75%	90.00%
			432	549	395	37	515	34	91.44%	92.07%
PU2	126	513	14	57	10	4	57	0	71.43%	100.00%
			126	513	102	24	502	11	80.95%	90.27%
PU3	1638	2079	182	231	157	25	218	13	86.26%	92.35%
			1638	2079	1419	219	2018	61	86.63%	95.88%
PUA	513	513	57	57	56	1	29	28	98.25%	66.67%
			513	513	510	3	437	76	99.42%	87.03%

f) T=5

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	44	4	53	8	91.67%	84.62%
			432	549	388	44	493	56	89.81%	87.39%

PU2	126	513	14	57	9	5	57	0	64.29%	100.00%
			126	513	74	52	497	16	58.73%	82.22%
PU3	1638	2079	182	231	143	39	214	17	78.57%	89.38%
			1638	2079	1352	286	1994	85	82.54%	94.08%
PUA	513	513	57	57	55	2	38	19	96.49%	74.32%
			513	513	495	18	412	101	96.49%	83.05%

2. Kết quả thực hiện với thuật toán AdaBoost with discrete predictions

a) T=500

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	46	2	57	4	95.83%	92.00%
			432	549	432	0	549	0	100.00%	100.00%
PU2	126	513	14	57	13	1	57	0	92.86%	100.00%
			126	513	126	0	513	0	100.00%	100.00%
PUA	513	513	57	57	53	4	45	12	92.98%	81.54%
	513	513	513	513	513	0	513	0	100.00%	100.00%
PU3	1638	2079	182	231	173	9	216	15	95.05%	92.02%
			1638	2079	1624	14	2074	5	99.15%	99.69%

b) T=200

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	45	3	58	3	93.75%	93.75%
			432	549	432	0	549	0	100.00%	100.00%
PU2	126	513	14	57	13	1	57	0	92.86%	100.00%
			126	513	126	0	513	0	100.00%	100.00%
PUA	513	513	57	57	53	4	45	12	92.98%	81.54%
	513	513	513	513	513	0	512	1	100.00%	99.81%
PU3	1638	2079	182	231	172	10	217	14	94.51%	92.47%
			1638	2079	1596	42	2062	17	97.44%	98.95%

c) T=100

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	46	2	57	4	95.83%	92.00%
			432	549	430	2	546	3	99.54%	99.31%

PU2	126	513	14	57	12	2	57	0	85.71%	100.00%
			126	513	126	0	513	0	100.00%	100.00%
PUA	513	513	57	57	54	3	45	12	94.74%	81.82%
	513	513	513	513	507	6	505	8	98.83%	98.45%
PU3	1638	2079	182	231	173	9	214	17	95.05%	91.05%
			1638	2079	1580	58	2035	44	96.46%	97.29%

d) T=50

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	46	2	54	7	95.83%	86.79%
			432	549	422	10	542	7	97.69%	98.37%
PU2	126	513	14	57	12	2	57	0	85.71%	100.00%
			126	513	126	0	513	0	100.00%	100.00%
PUA	513	513	57	57	56	1	44	13	98.25%	81.16%
	513	513	513	513	495	18	488	25	96.49%	95.19%
PU3	1638	2079	182	231	173	9	218	13	95.05%	93.01%
			1638	2079	1557	81	2018	61	95.05%	96.23%

e) T=10

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						
PU1	432	549	48	61	47	1	404	28	97.92%	62.67%
			432	549	432	0	504	45	100.00%	90.57%
PU2	126	513	14	57	11	3	56	1	78.57%	91.67%
			126	513	97	29	304	209	76.98%	31.70%
PUA	513	513	57	57	53	4	45	12	92.98%	81.54%
	513	513	513	513	470	43	449	64	91.62%	88.01%
PU3	1638	2079	182	231	173	9	218	13	95.05%	93.01%
			1638	2079	1557	81	2018	61	95.05%	96.23%

f) T=5

Ngữ liệu	Số email học		Số email kiểm thử		S->S	S->N	N->N	N->S	SR	SP
	Spam	Non-spam	Spam	Non-spam						Spam
PU1	432	549	48	61	39	9	56	5	81.25%	88.64%
			432	549	360	72	517	32	83.33%	91.84%
PU2	126	513	14	57	9	5	56	1	64.29%	90.00%
			126	513	106	20	305	163	84.13%	39.41%
PUA	513	513	57	57	54	3	38	19	94.74%	73.97%
	513	513	513	513	484	29	396	117	94.35%	80.53%
PU3	1638	2079	182	231	171	11	200	31	93.96%	84.65%
			1638	2079	1387	81	2018	61	94.48%	95.79%