BỘ GIÁO DỤC VÀ ĐÀO TẠO
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM**

---


**HOÀNG NGỌC DƯƠNG**

# PHÂN LOẠI VĂN BẢN DỰA TRÊN MÔ HÌNH ĐỒ THỊ

## LUẬN VĂN THẠC SĨ

Chuyên ngành : Công nghệ thông tin
Mã số ngành: 60480201

TP. HỒ CHÍ MINH, tháng 6 năm 2017

BỘ GIÁO DỤC VÀ ĐÀO TẠO
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM**

---


**HOÀNG NGỌC DƯƠNG**

# PHÂN LOẠI VĂN BẢN DỰA TRÊN MÔ HÌNH ĐỒ THỊ

## LUẬN VĂN THẠC SĨ

Chuyên ngành : Công nghệ thông tin
Mã số ngành: 60480201

**CÁN BỘ HƯỚNG DẪN KHOA HỌC: PGS.TS VÕ ĐÌNH BẢY**

TP. HỒ CHÍ MINH, tháng 6 năm 2017

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM**

Cán bộ hướng dẫn khoa học: PGS. TS VÕ ĐÌNH BẢY

Luận văn Thạc sĩ được bảo vệ tại Trường Đại học Công nghệ TP. HCM ngày … tháng … năm …

Thành phần Hội đồng đánh giá Luận văn Thạc sĩ gồm:
*(Ghi rõ họ, tên, học hàm, học vị của Hội đồng chấm bảo vệ Luận văn Thạc sĩ)*

| TT | Họ và tên | Chức danh Hội đồng |
|---|---|---|
| 1 | | Chủ tịch |
| 2 | | Phản biện 1 |
| 3 | | Phản biện 2 |
| 4 | | Ủy viên |
| 5 | | Ủy viên, Thư ký |

Xác nhận của Chủ tịch Hội đồng đánh giá Luận sau khi Luận văn đã được sửa chữa (nếu có).

**Chủ tịch Hội đồng đánh giá LV**

TRƯỜNG ĐH CÔNG NGHỆ TP. HCM
**PHÒNG QLKH – ĐTSĐH**

**CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**
**Độc lập – Tự do – Hạnh phúc**

*TP. HCM, ngày..... tháng….. năm 20.....*

# NHIỆM VỤ LUẬN VĂN THẠC SĨ

Họ tên học viên: HOÀNG NGỌC DƯƠNG Giới tính: Nam

Ngày, tháng, năm sinh: 05/10/1985 Nơi sinh: Vĩnh Thịnh, Vĩnh Lộc, Thanh Hóa

Chuyên ngành: Công nghệ thông tin MSHV: 1541860001

**I- Tên đề tài:**

**PHÂN LOẠI VĂN BẢN DỰA TRÊN MÔ HÌNH ĐỒ THỊ**

**II- Nhiệm vụ và nội dung:**

.....

.....

.....

.....

**III- Ngày giao nhiệm vụ:** 25/9/2016

**IV- Ngày hoàn thành nhiệm vụ:** 30/6/2017

**V- Cán bộ hướng dẫn: PGS. TS VÕ ĐÌNH BẢY**

**CÁN BỘ HƯỚNG DẪN** (Họ tên và chữ ký)

**KHOA QUẢN LÝ CHUYÊN NGÀNH** (Họ tên và chữ ký)

# LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi. Các số liệu, kết quả nêu trong Luận văn là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Tôi xin cam đoan rằng mọi sự giúp đỡ cho việc thực hiện Luận văn này đã được cảm ơn và các thông tin trích dẫn trong luận văn đã được chỉ rõ nguồn gốc.

**Học viên thực hiện Luận văn**

**Hoàng Ngọc Dương**

# LỜI CÁM ƠN

Tôi xin bày tỏ lòng biết ơn sâu sắc đến PGS. TS **Võ Đình Bảy**, người đã tận tình chỉ bảo và hướng dẫn tôi trong suốt quá trình thực hiện luận văn tốt nghiệp.

Tôi xin gởi lời cảm ơn đến trường Đại học Công nghệ TP.HCM đã tạo điều kiện và tổ chức khóa học này để tôi có điều kiện tiếp thu kiến thức mới và thời gian để hoàn thành luận văn cao học.

Tôi xin chân thành cảm ơn các thầy cô đã truyền đạt cho chúng tôi những kiến thức quý báu trong quá trình học Cao học và làm luận văn.

Xin chân thành cảm ơn những người thân trong gia đình, cùng các anh chị em, bạn bè, đồng nghiệp đã giúp đỡ và động viên tôi trong quá trình thực hiện và hoàn thành luận văn.

**Hoàng Ngọc Dương**

# TÓM TẮT

+ Họ và tên học viên: Hoàng Ngọc Dương

+ Chuyên ngành: Công nghệ thông tin  Lớp: 15SCT11

+ Cán bộ hướng dẫn: PGS.TS Võ Đình Bảy

+ Tên đề tài: Phân loại văn bản dựa trên mô hình đồ thị

Luận văn nghiên cứu về kỹ thuật phân loại văn bản dựa trên mô hình đồ thị. Cụ thể chúng tôi đã nghiên cứu các khái niệm cơ bản về lý thuyết đồ thị, bài toán phân loại văn bản, các thuật toán phân loại văn bản thông dụng, khai thác đồ thị con phổ biến, trong đó chúng tôi tập trung vào thuật toán khai phá đồ thị con phổ biến gSpan và thuật toán phân loại SVM cho bài toán phân phân loại văn bản dựa trên mô hình đồ thị.

Phương pháp tiếp cận bài toán phân loại văn bản của chúng tôi trải qua các bước sau:

Bước 1: Thực hiện việc tách từ và tính TF – IDF

Bước 2: Việc mô hình hóa văn bản thành đồ thị sẽ được thực hiện sau bước 1

Bước 3: Khai thác đồ thị con phổ biến bằng thuật toán gSpan

Bước 4: Vec tơ hóa đồ thị văn bản

Bước 5: Bước cuối cùng chúng tôi thực hiện là huấn luyện phân lớp bằng SVM

Với cách tiếp cận trên của chúng tôi, qua thực nghiệm trên bộ dữ liệu tiếng Việt là các bài báo được lấy từ các nguồn tin tức điện tử http://vnexpress.net/, http://dantri.com.vn/, http://tuoitre.vn/. Kết quả thực nghiệm cho thấy mô hình phân loại này của chúng tôi đạt độ chính xác tương đối cao trên 85%.

Với kết quả này, chúng tôi đã đóng góp cho việc xử lý văn bản tiếng Việt một hướng tiếp cận mới cho bài toán phân loại văn bản tiếng Việt. Đó là phương pháp phân loại văn bản dựa trên mô hình đồ thị. Qua đó làm giàu thêm các phương pháp phân loại văn bản tiếng Việt hơn nữa.

Luận văn này bao gồm 5 chương – trình bày chi tiết các ý tưởng, phương thức thực hiện, các thực nghiệm và kết luận cũng như hướng phát triển tiếp theo cho đề tài.

# MỤC LỤC

Trang

## DANH MỤC CÁC TỪ VIẾT TẮT

| Tiếng Anh | Từ viết tắt | Tiếng Việt |
|---|---|---|
| Term Frequency | TF | Tần suất thuật ngữ |
| Inverse Document Frequency | IDF | Nghịch đảo tần suất tài liệu |
| k-Nearest Neighbors | k-NN | k-láng giềng gần nhất |
| Support Vector Machine | SVM | Máy vectơ hỗ trợ |
| Naive Bayes | NB | Bayes |
| Depth First Search | DFS | Tìm kiếm theo chiều sâu |

# DANH MỤC CÁC BẢNG

# DANH MỤC CÁC HÌNH

# CHƯƠNG I: MỞ ĐẦU

## I.1 Giới thiệu

Phân loại văn bản là việc làm đã tồn tại từ lâu trong cuộc sống hàng ngày. Nhu cầu phân loại văn bản rất nhiều: phân loại sách trong thư viện, phân loại văn bản trong cơ quan, phân loại các trang báo, … Trước đây, việc phân loại những văn bản này chủ yếu dùng phương pháp thủ công.

Trong những năm gần đây, sự phát triển vượt bậc của công nghệ thông tin đã làm tăng số lượng giao dịch thông tin trên mạng Internet một cách đáng kể đặc biệt là thư viện điện tử, tin tức điện tử, ... Do đó số lượng văn bản xuất hiện trên mạng Internet cũng tăng với một tốc độ chóng mặt, và tốc độ thay đổi thông tin là cực kỳ nhanh chóng. Với số lượng thông tin đồ sộ như vậy, một yêu cầu lớn đặt ra là làm sao tổ chức và tìm kiếm thông tin, dữ liệu có hiệu quả nhất. Bài toán phân loại là một trong những giải pháp hợp lý cho yêu cầu trên. Nhưng trên thực tế khối lượng thông tin quá lớn, việc phân loại dữ liệu thủ công là điều không thể. Hướng giải quyết cho bài toán này là xây dựng một chương trình máy tính tự động phân loại các thông tin dữ liệu trên.

Đã có rất nhiều công trình nghiên cứu và ứng dụng thực tế dùng để thực hiện việc phân loại văn bản, tuy nhiên các ứng dụng đó cũng chưa thể đáp ứng hoàn toàn nhu cầu của người sử dụng, do vậy mà việc tìm kiếm, nghiên cứu các giải thuật, các phương pháp phân loại văn bản vẫn được tiếp tục nghiên cứu và hoàn thiện.

Với mục tiêu góp phần vào lĩnh vực nghiên cứu và ứng dụng phân loại văn bản vào cuộc sống, luận văn này sẽ thực hiện các công việc sau:

- Nghiên cứu và tổng hợp một số phương pháp phân loại văn bản đã làm và sau đó đưa ra 1 số nhận xét đánh giá.

- Nghiên cứu và đưa vào ứng dụng trong việc phân loại văn bản bằng lý thuyết khá mới hiện nay là phân loại văn bản dựa trên mô hình đồ thị.

- Đưa ra một chương trình máy tính để thử nghiệm và có kết quả đánh giá về phương pháp phân loại văn bản dựa trên mô hình đồ thị.

## I.2 Tổng quan về phân loại văn bản

Bài toán nhận dạng và phân loại văn bản là một trong những bài toán kinh điển trong lĩnh vực xử lý dữ liệu văn bản. Xử lý dữ liệu văn bản bao gồm một số bài toán:

- Kiểm tra lỗi chính tả (spelling-checker)

- Phân tích cú pháp (grammar analysis)

- Phân tích văn bản (text analyzer)

- Phân loại văn bản (text classification)

- Tóm tắt văn bản (text summarization)

- Khai thác văn bản và web (text & web mining), …

Phân loại văn bản là công việc phân tích nội dung của văn bản, sau đó đưa ra quyết định văn bản này thuộc nhóm nào trong các nhóm văn bản đã cho trước.

Phân loại văn bản chính là công việc khai thác dữ liệu văn bản. Trong lĩnh vực khai thác dữ liệu, các phương pháp tiếp cận chính như: Naïve Bayes, máy vectơ hỗ trợ (SVM), Cây quyết định, K láng giềng gần nhất (k-NN), mạng nơron.

Những phương pháp này đã cho kết quả chấp nhận được và được sử dụng trong thực tế, tuy nhiên việc nghiên cứu phân loại văn bản vẫn tiếp tục được nghiên cứu nhằm đưa ra những phương pháp mới cho kết quả tốt hơn.

## I.3 Mục tiêu luận văn

Do phạm vi bài toán khá rộng và thời gian làm đề tài tương đối hạn hẹp nên mục tiêu nghiên cứu của luận văn này sẽ được tập trung ở một số điểm sau:

- Nghiên cứu kỹ thuật phân loại văn bản và một số phương pháp phân loại văn bản, mô tả các yêu cầu chính yếu nhất của từng phương pháp và rút ra các ưu, khuyết điểm của từng phương pháp.

- Nghiên cứu một số thuật toán khai phá đồ thị con phổ biến thông dụng hiện nay như: FSG, gFSG, DPMine, gSpan, GASTON, gPrune, …

- Nghiên cứu cơ sở lý thuyết về phân loại văn bản dựa trên mô hình đồ thị và áp dụng phương pháp phân loại văn bản dựa trên môn hình đồ thị để xây dựng hệ thống tự động phân loại văn bản ứng dụng trong thực tế.

- Xây dựng thử nghiệm chương trình phân loại văn bản sử dụng thuật toán gSpan và SVM.

- Đưa ra các kết luận, đánh giá kết quả đạt được đồng thời cũng nêu ra phương hướng để giải quyết các vấn đề còn tồn tại.

**I.4 Nội dung nghiên cứu**

Dựa trên các mục tiêu của luận văn, việc nghiên cứu trong luận văn này sẽ tiến hành bám sát yêu cầu mục tiêu đòi hỏi:

- Nghiên cứu các phương pháp phân loại văn bản mới được đưa ra hoặc có tính phổ biến được sử dụng nhiều trong thực tế hiện nay

- Dựa trên các kết quả đã nghiên cứu về phân loại văn bản ở trên thì luận văn sẽ chọn lựa một phương pháp mới trong việc phân loại văn bản đó là phương pháp phân loại văn bản dựa trên mô hình đồ thị.

- Trong quá trình thực hiện chương trình, để tăng nhanh tốc độ lập trình và hiệu quả của phương pháp làm, sẽ có sử dụng lại các chương trình tính toán được cung cấp ở dạng mã mở. Cụ thể là chương trình tính toán máy vectơ hỗ trợ (Support vector machine – SVM) là chương trình được cho tại địa chỉ http://www.csie.ntu.edu.tw/~cjlin

- Việc kết luận chủ yếu sẽ là đưa ra các kết luận thực nghiệm khi sử dụng, xác định được những thông số để có thể sử dụng các kết quả này nhằm có thể đánh giá được với các phương pháp khác.

**I.5 Kết quả đạt được**

Sau quá trình nghiên cứu và thực hiện luận văn đã đạt được các kết quả như sau:

- Đã nghiên cứu và tiếp thu các kỹ thuật phân loại văn bản đang được sử dụng trong thực tế hiện nay.

- Nắm được phương pháp phân loại văn bản dựa trên mô hình đồ thị.

- Đã xây dựng thử nghiệm một chương trình phân loại văn bản cho các file văn bản.

- Có những kết luận và nêu ra phương hướng để giải quyết các vấn đề còn tồn tại.

## I.6 Bố cục của luận văn

Bố cục của luận văn được chia làm 5 chương:

Chương 1: "**Mở đầu**" trình bày tổng quan về phân loại văn bản, mục tiêu, nội dung nghiên cứu cũng như kết quả đạt được của luận văn.

Chương 2: "**Cơ sở lý thuyết**" Trình bày cơ sở lý thuyết mô hình biểu diễn văn bản, các phương pháp phân loại văn bản, lý thuyết đồ thị, đồ thị con, khai thác đồ thị con phổ biến.

Chương 3: "**Mô tả bài toán và xử lý bài toán**" trình bày các bước tiến hành phân loại văn bản dựa trên mô hình đồ thị.

Chương 4: "**Thực nghiệm**" tiến hành thu thập bộ dữ liệu tiếng Việt, cài đặt chương trình và thực nghiệm đánh giá kết quả đạt được của bài toán với bộ dữ liệu trên.

Chương 5: "**Kết luận**" tổng hợp lại các vấn đề đã nghiên cứu được, các kết quả đạt được của luận văn, những vấn đề còn tồn tại và phương hướng giải quyết, phát triển của luận văn trong thời gian tới.

# CHƯƠNG II: CƠ SỞ LÝ THUYẾT

## II.1 Tổng quan

### II.1.1 Định nghĩa phân loại văn bản

Phân loại văn bản là một trong nhiêu lĩnh vực được chú ý nhất và đã được nghiên cứu trong những năm gần đây.

Phân loại văn bản là quá trình gán các văn bản vào một hay nhiều lớp văn bản đã được xác định từ trước. Người ta có thể phân loại các văn bản một cách thủ công, tức là đọc nội dung từng văn bản và gán nó vào một loại nào đó. Hệ thống quản lý tập gồm nhiều văn bản cho nên cách này sẽ tốn nhiều thời gian, công sức và do đó là không khả thi. Do vậy mà phải có các phương pháp phân loại tự động. Để phân loại tự động, người ta sử dụng các phương pháp học máy trong trí tuệ nhân tạo như: Cây quyết định, Naïve Bayes, K láng giềng gần nhất ...

Một trong những ứng dụng quan trọng nhất của phân loại văn bản tự động là ứng dụng trong các hệ thống tìm kiếm văn bản. Từ một tập con văn bản đã phân loại sẵn, tất cả các văn bản trong miền tìm kiếm sẽ được gán chỉ số lớp tương ứng. Trong câu hỏi của mình, người dùng có thể xác định chủ đề hoặc loại văn bản mà mình mong muốn tìm kiếm để hệ thống cung cấp đúng yêu cầu của mình.

Một ứng dụng khác của phân loại văn bán là trong lĩnh vực hiểu văn bản. Phân loại văn bản có thể được sử dụng để lọc các văn bản hoặc một phần văn bản chứa dữ liệu cần tìm mà không làm mất đi tính phức tạp của ngôn ngữ tự nhiên.

Ngoài ra phân loại văn bản còn xuất hiện trong nhiều ứng dụng: lọc e-mail, định hướng mail, lọc thư rác, giám sát tin, chỉ mục tự động các bài báo khoa học, …

### I.1.2 Đặc trưng văn bản

Các phương pháp rút trích thông tin cổ điển thì coi mỗi một văn bản như là tập các từ khóa và gọi tập các từ khóa này là tập các term. Một phần tử trong tập term thì đơn giản là một từ, mà ngữ nghĩa của từ này giúp tạo thành nên nội dung của văn bản. Vì vậy, tập term được sử dụng để tạo các chỉ mục và tóm lược nội dung của văn bản.

Giả sử cho một tập term của một văn bản nào đó, chúng ta có thể nhận thấy rằng không phải tất cả các từ trong tập term này đều có mức độ quan trọng như nhau trong việc

mô tả nội dung văn bản. Ví dụ, bây giờ chúng ta xét một tập gồm một trăm ngàn văn bản, giả sử có một từ A nào đó xuất hiện trong một trăm ngàn văn bản này thì chúng ta có thể khẳng định rằng từ A này không quan trọng và chúng ta sẽ không quan tâm đến nó, bởi vì chắc chắn là nó sẽ không cho chúng ta biết được về nội dung của các văn bản này. Vì vậy từ A sẽ bị loại ra khỏi tập các term, khi chúng ta xây dựng tập term cho văn bản để miêu tả nội dung ngữ nghĩa của các văn bản này. Kết quả này có được thông qua thao tác xác định trọng số cho mỗi một từ trong tập term của một văn bản .

Đặt $k_i$ là từ thứ i trong tập term, $d_j$ là văn bản j, và $W_{ij} \geq 0$ là trọng số của từ $k_i$ trong văn bản $d_j$. Giá trị của trọng số này thì rất là quan trọng trong việc miêu tả nội dung của văn bản.

Đặt t là số lượng các từ trong tập term của hệ thống. $K = \{k_1, k_2, k_3, ..., k_t\}$ là tập tất cả các từ trong tập term, trong đó $k_i$ là từ thứ i trong tập term. Trọng số $W_{ij} > 0$ là trọng số của từ $k_i$ trong văn bản $d_j$. Với mỗi một từ, nếu nó không xuất hiện trong văn bản thì $W_{ij} = 0$. Do đó, văn bản $d_j$ thì được biểu diễn bằng vector $d_j$, trong đó vector $d_j = \{W_{j1}\ W_{j2}\ W_{j3}, ..., W_{jt}\}$.

Các đặc trưng của văn bản khi biểu diễn dưới dạng vector:

Số nhiều không gian đặc trưng thường lớn

Các đặc trưng độc lập nhau

Các đặc trưng rời rạc: vector đặc trưng $d_i$ có thể  có nhiều thành phần mang giá trị 0 do có nhiều đặc trưng không xuất hiện trong văn bản $d_i$ (nếu chúng ta tiếp cận theo cách sử dụng giá trị nhị phân 1, 0 để biểu diễn cho việc có xuất hiện hay không một đặc trưng nào đó trong văn bản đang được biểu diễn thành vector), tuy nhiên cách tiếp cận sử dụng giá trị nhị phân 0, 1 này thì kết quả phân loại phần nào hạn chế là do có thể đặc trưng đó không có trong văn bản đang xét nhưng trong văn bản đang xét lại có từ khóa khác với từ đặc trưng nhưng có ngữ nghĩa giống với từ đặc trưng này, do đó một cách tiếp cận khác là không sử dụng số nhị phân 0, 1 mà sử dụng giá trị số thực để phần nào giảm bớt sự rời rạc trong vector văn bản

## II.2 Mô hình biểu diễn văn bản

Có nhiều cách biểu diễn văn bản, luận văn trình bày một số phương pháp biểu diễn văn bản phổ biến.

### II.2.1 Mô hình logic

Theo mô hình này, các từ có nghĩa trong văn bản sẽ được đánh chỉ số và nội dung văn bản được quản lý theo các chỉ số Index đó. Mỗi văn bản được đánh chỉ số theo quy tắc liệt kê các từ có nghĩa trong các văn bản với vị trí xuất hiện của nó trong văn bản. Từ có nghĩa là từ mang thông tin chính về các văn bản lưu trữ, khi nhìn vào nó, người ta có thể biết chủ đề của văn bản cần biểu diễn.

Khi đó chúng ta tiến hành Index các văn bản đưa vào theo danh sách các từ khóa nói trên. Với mỗi từ khóa người ta sẽ đánh số thứ tự vị trí xuất hiện của nó và lưu lại chỉ số đó cùng với mã văn bản chứa nó. Cách biểu diễn này cũng được các máy tìm kiếm ưa dùng.

Ví dụ: Có 2 văn bản với mã tương ứng là VB1, VB2:

VB1 là: “Đại hội chi bộ thành công”

VB2 là: “Chi bộ hoàn thành nhiệm vụ”

Khi đó, ta có cách biểu diễn như sau:

Bảng 2.1: Biểu diễn văn bản trong mô hình Logic

| Từ mục | Mã VB_Vị trí xuất hiện |
|---|---|
| Đại | VB1(1) |
| Hội | VB1(2) |
| Chi | VB1(3), VB2(1) |
| Bộ | VB1(4), VB2(2) |
| Thành | VB1(5),VB2(4) |

| | |
|---|---|
| Công | VB1(6) |
| Hoàn | VB2(3) |
| Nhiệm | VB2(5) |
| Vụ | VB2(6) |

**Ưu điểm, nhược điểm của mô hình logic:**

**+ Ưu điểm**

Việc tìm kiếm trở nên nhanh chóng và đơn giản. Cần tìm kiếm từ "computer". Hệ thống sẽ duyệt trên bảng Index để trỏ đến chỉ số Index tương ứng nếu từ "computer" tồn tại trên hệ thống. Việc tìm kiếm này khá nhanh và đơn giản khi trước đó ta đã sắp xếp bảng Index theo vẫn chữ cái. Phép tìm kiếm trên có độ phức tạp cấp $\theta(n\log_2 n)$, với n là số từ trong bảng Index. Tương ứng với chỉ số index trên sẽ cho ta biết các tài liệu chứa từ khóa tìm kiếm. Như vậy, việc tìm kiếm liên quan đến k từ thì các phép toán cần thực hiện là k*n* $\log_2 n$ (với n là số từ trong bảng Index).

**+ Nhược điểm**

Với phương pháp này đòi hỏi người sử dụng phải có kinh nghiệm và chuyên môn trong lĩnh vực tìm kiếm vì câu hỏi đưa vào dưới dạng Logic nên kết quả cũng có giá trị Logic (Boolean). Một số tài liệu sẽ được trả lại khi thỏa mãn mọi điều kiện đưa vào. Như vậy muốn tìm được tài liệu theo nội dung thì phải biết đích xác về tài liệu.

Việc Index các tài liệu rất phức tạp và làm tốn nhiều thời gian, đồng thời cũng tốn không gian để lưu trữ các bảng Index.

Các tài liệu tìm được không được sắp xếp theo độ chính xác của chúng. Các bảng Index không linh hoạt vì khi các từ vựng thay đổi (thêm, sửa, xóa, ...) dẫn tới chỉ số Index cũng phải thay đổi theo.

### II.2.2 Mô hình phân tích cú pháp

Trong mô hình này, mỗi văn bản đều phải được phân tích cú pháp và trả lại thông tin chi tiết về chủ đề của văn bản đó. Sau đó, người ta tiến hành Index các chủ đề của từng văn bản. Cách Index trên chủ đề cũng giống như Index trên văn bản nhưng chỉ Index trên các từ xuất hiện trong chủ đề.

Các văn bản được quản lý thông qua các chủ đề này để có thể tìm kiếm được khi có yêu cầu, câu hỏi tìm kiếm sẽ dựa trên các chủ đề trên.

**Một số ưu điểm, nhược điểm của phương pháp này:**

**+ Ưu điểm**

Tìm kiếm theo phương pháp này khá hiệu quả và đơn giản, do tìm kiếm nhanh và chính xác.

Đối với những ngôn ngữ đơn giản về mặt ngữ pháp thì việc phân tích trên có thể đạt được mức độ chính xác cao và chấp nhận được.

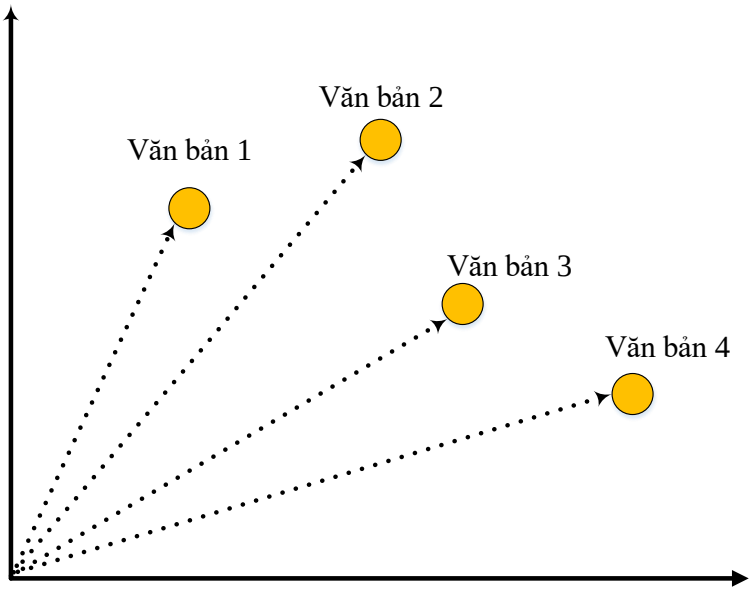
**+ Nhược điểm**

Chất lượng của hệ thống theo phương pháp này hoàn toàn phụ thược vào chất lượng của hệ thống phân tích cú pháp và đoán nhận nội dung tài liệu. Trên thực tế, việc xây dựng hệ thống này rất phức tạp, phụ thuộc vào đặc điểm của từng ngôn ngữ và đa số chưa đạt đến độ chính xác cao.

### II.2.3 Mô hình không gian vector

Cách biểu diễn văn bản thông dụng nhất là thông qua vector biểu diễn theo mô hình không gian vector (Vector Space Model). Đây là một cách biểu diễn tương đối đơn giản và hiệu quả.

Theo mô hình này, mỗi văn bản được biểu diễn thành một vector. Mỗi thành phần của vector là một từ khóa riêng biệt trong tập văn bản gốc và được gán một giá trị là hàm f chỉ mật độ xuất hiện của từ khóa trong văn bản.




Hình 2.1 Biểu diễn vector văn bản trong không gian 2 chiều

Giả sử ta có một văn bản và nó được biểu diễn bởi vector $V(v_1, v_2, ..., v_n)$. Trong đó, $v_i$ là số lần xuất hiện của từ khóa thứ i trong văn bản. Ta xét 2 văn bản sau:

VB1: Đại hội chi bộ

VB2: Đại hội đã thành công

Sau khi qua bước tiền xử lý văn bản, ta biểu diễn như sau:

Bảng 2.2: Biểu diễn văn bản mô hình Vector

| Từ | Vector_VB1 | Vector_VB2 |
|---|---|---|
| Đại | 1 | 1 |
| Hội | 1 | 1 |
| Chi | 1 | 0 |
| Bộ | 1 | 0 |
| Thành | 0 | 1 |
| Công | 0 | 1 |

Trong các cơ sở dữ liệu văn bản, mô hình vector là mô hình biểu diễn văn bản được sử dụng phổ biến nhất hiện nay. Mối quan hệ giữa các văn bản được thực hiện thông qua việc tính toán trên các vector biểu diễn vì vậy được thi hành khá hiệu quả.

### II.2.4 Mô hình boolean

Một mô hình biểu diễn vector với hàm f cho ra giá trị rời rạc với duy nhất hai giá trị đúng và sai (true và false, hoặc 0 và 1) gọi là mô hình Boolean. Hàm f tương ứng với từ khóa $t_i$ sẽ cho ra giá trị đúng nếu và chỉ nếu từ khóa $t_i$ xuất hiện trong văn bản đó.

Mô hình Boolean được xác định như sau:

Giả sử có một cơ sở dữ liệu gồm ***m*** văn bản, ***D = {d₁, d₂, ..., dₘ}***. Mỗi văn bản được biểu diễn dưới dạng một vector gồm ***n*** từ khóa ***T = {t₁, t₂, ..., tₙ}***. Gọi ***W = {Wᵢⱼ}*** là ma trận trọng số, trong đó ***Wᵢⱼ*** là giá trị trọng số của từ khóa ***tᵢ*** trong văn bản ***dj***.

$$W_{ij} = \begin{cases} 1 \text{ nếu } t_i \text{ có mặt trong } d_j \\ 0 \text{ nếu ngược lại} \end{cases}$$

Trở lại với 2 văn bản trên, áp dụng mô hình Boolean ta có biểu diễn như sau:

Bảng 2.3: Biểu diễn văn bản mô hình Boolean

| Từ | Vector_VB1 | Vector_VB2 |
|---|---|---|
| Đại | 1 | 1 |
| Hội | 1 | 1 |
| Chi | 1 | 0 |
| Bộ | 1 | 0 |
| Thành | 0 | 1 |
| Công | 0 | 1 |

### II.2.5 Mô hình tần suất

Trong mô hình tần suất, ma trận W = $\{W_{ij}\}$ được xác định dựa trên tần số xuất hiện của từ khóa $t_i$ trong văn bản $d_j$ hoặc tần số xuất hiện của từ khóa $\boldsymbol{t_i}$ trong toàn bộ cơ sở dữ liệu. Sau đây là một số phương pháp phổ biến:

#### II.2.5.1 Phương pháp dựa trên tần số từ khóa (TF - Term Frequency)

Các giá trị $w_{ij}$ được tính dựa trên tần số (hay số lần) xuất hiện của từ khóa trong văn bản. Gọi $f_{ij}$ là số lần xuất hiện của từ khóa $t_i$ trong văn bản $d_j$, khi đó $w_{ij}$ được tính bởi một trong ba công thức:

$$w_{ij} = f_{ij}$$

$$w_{ij} = 1 + \log(f_{ij})$$

$$w_{ij} = \sqrt{f_{ij}}$$

Với phương pháp này, trọng số $w_{ij}$ tỷ lệ thuận với số lần xuất hiện của từ khóa $t_i$ trong văn bản $d_j$. Khi số lần xuất hiện từ khóa $t_i$ trong văn bản $d_j$ càng lớn thì điều đó có nghĩa là văn bản $d_j$ càng phụ thuộc vào từ khóa $t_i$ hay nói cách khác từ khóa $t_i$ mang nhiều thông tin trong văn bản $d_j$.

Ví dụ, khi văn bản xuất hiện nhiều từ khóa máy tính, điều đó có nghĩa là văn bản đang xét chủ yếu liên quan đến lĩnh vực tin học.

Nhưng suy luận trên không phải lúc nào cũng đúng. Một ví dụ điển hình là từ "và" xuất hiện nhiều trong hầu hết các văn bản, nhưng trên thực tế từ này lại không mang nhiều ý nghĩa như tần suất xuất hiện của nó. Hoặc có những từ không xuất hiện trong văn bản này nhưng lại xuất hiện trong văn bản khác, khi đó ta sẽ không tính được giá trị của $\log(f_{ij})$. Một phương pháp khác ra đời khắc phục được nhược điểm của phương pháp TF, đó là phương pháp IDF.

#### II.2.5.2 Phương pháp dựa trên nghịch đảo tần số văn bản (IDF - Inverse Document Frequency)

Trong phương pháp này, giá trị $w_{ij}$ được tính theo công thức sau:

$$W_{ij} = \begin{cases} \log\frac{m}{h_i} = \log(m) - \log(h_i) & \text{neáu } t_i \text{ coùmaët trong } d_j \\ 0 & \text{neáu ngöôïc lai} \end{cases}$$

Trong đó m là số lượng văn bản, $h_i$ là số lượng văn bản mà từ khóa $t_i$ xuất hiện.

Trọng số $w_{ij}$ trong công thức này được tính dựa trên độ quan trọng của từ khóa $t_i$ trong văn bản $d_j$. Nếu $t_i$ xuất hiện trong càng ít văn bản, điều đó có nghĩa là khi nó xuất hiện trong $d_j$ thì trọng số của nó đối với văn bản $d_j$ càng lớn hay nó là điểm quan trọng để phân biệt văn bản $d_j$ với các văn bản khác và hàm lượng thông tin trong nó càng lớn.

**II.2.5.3 Phương pháp TF x IDF**

Phương pháp này là tổng hợp của hai phương pháp TF và IDF, giá trị của ma trận trọng số được tính như sau:

$$W_{ij} = \begin{cases} \left[1 + \log(f_{ij})\right]\log(\frac{m}{h_i}) & \text{neáu } f_{ij} \geq 1 \\ 0 & \text{neáu ngöôïc lai} \end{cases}$$

Đây là phương pháp kết hợp được ưu điểm của cả hai phương pháp trên. Trọng số $w_{ij}$ được tính bằng tần số xuất hiện của từ khóa $t_i$ trong văn bản $d_j$ và độ hiếm của từ khóa $t_i$ trong toàn bộ cơ sở dữ liệu.

Một số ưu, nhược điểm của phương pháp biểu diễn này:

+ Ưu điểm

Các tài liệu có thể được sắp xếp theo mức độ liên quan đến nội dung yêu cầu.

Tiến hành lưu trữ và tìm kiếm đơn giản hơn phương pháp Logic.

+ Nhược điểm

Việc xử lý sẽ chậm khi hệ thống các từ vựng là lớn do phải tính toán trên toàn bộ các vector của tài liệu.

Khi biểu diễn các vector với các hệ số là số tự nhiên sẽ làm tăng mức độ chính xác của việc tìm kiếm nhưng làm tốc độ tính toán giảm đi rất nhiều do các phép nhân vector

phải tiến hành trên các số tự nhiên hoặc số thực, hơn nữa việc lưu trữ các vector sẽ tốn kém và phức tạp.

Hệ thống không linh hoạt khi lưu trữ các từ khóa. Chỉ cần một thay đổi rất nhỏ trong bảng từ vựng sẽ kéo theo hoặc là vector hóa lại toàn bộ các tài liệu lưu trữ, hoặc là sẽ bỏ qua các từ có nghĩa bổ sung trong các tài liệu được mã hóa trước đó.

Một nhược điểm nữa, chiều của mỗi Vector theo cách biểu diễn này là rất lớn, bởi vì chiều của nó được xác định bằng số lượng các từ khác nhau trong tập hợp văn bản.

## II.3 Các phương pháp phân loại văn bản

### II.3.1 Phương pháp Naïve Bayes (NB)

NB là phương pháp phân loại dựa vào xác suất được sử dụng rộng rãi trong lĩnh vực máy học (Mitchell trình bày năm 1996, Joachims trình bày năm 1997 và Jason năm 2001) được sử dụng lần đầu tiên trong lĩnh vực phân loại bởi Maron vào năm 1961, sau đó trở nên phổ biến dùng trong nhiều lĩnh vực như trong các công cụ tìm kiếm (được mô tả năm 1970 bởi Rijsbergen), các bộ lọc mail (mô tả năm 1998 bởi Sahami), …

Ý tưởng cơ bản của cách tiếp cận này là sử dụng xác suất có điều kiện giữa từ và chủ đề để dự đoán xác suất chủ đề của một văn bản cần phân loại. Điểm quan trọng của phương pháp này chính là ở chỗ giả định rằng sự xuất hiện của tất cả các từ trong văn bản đều độc lập với nhau. Với giả định này NB không sử dụng sự phụ thuộc của nhiều từ vào một chủ đề, không sử dụng việc kết hợp các từ để đưa ra phán đoán chủ đề và do đó việc tính toán NB chạy nhanh hơn các phương pháp khác với độ phức tạp theo hàm số mũ.

Công thức tính $Pr(C_j,d')$: Mục đích chính là tính được xác suất $Pr(C_j,d')$, xác suất để văn bản $d'$ nằm trong lớp $C_j$. Theo luật Bayes, văn bản $d'$ sẽ được gán vào lớp $C_j$ nào có xác suất $Pr(C_j, d')$ cao nhất. Công thức sau dùng để tính $Pr(C_j,d')$ (do Joachims đề xuất năm 1997)

$$H_{BAYES} = \arg\max_{C_j \in C}\left(\frac{\Pr(C_j).\prod_{i=1}^{d'}\Pr(w_i / C_j)}{\sum_{C' \in C}\Pr(C').\prod_{i=1}^{|d'|}\Pr(w_i / C')}\right) = \arg\max_{C_j \in C}\left(\frac{\Pr(C_j).\prod_{i=1}^{d'}\Pr(w_i / C_j)^{IF(w,d')}}{\sum_{C' \in C}\Pr(C').\prod_{w \in F}\Pr(w_i / C')^{IF(w,d')}}\right)$$

Trong đó:

- (TF,d') là số lần xuất hiện của từ $w_j$ trong văn bản d′

- |d′| là số lượng các từ trong văn bản d′

- $w_j$ là một từ trong không gian đặc trưng F với số chiều là |F|

- $Pr(C_j)$ được tính dựa trên tỷ lệ phần trăm của số văn bản mỗi lớp tương ứng trong tập dữ liệu huấn luyện:

$$\Pr(C_j) = \frac{\square C_j \square}{\square C \square} = \frac{\square C_j \square}{\sum_{C' \in C} \square C' \square}$$

- $Pr(w_j|Cj)$ được tính sử dụng phép ước lượng Laplace (do Napnik trình bày năm 1982)

$$\Pr(w_i \mid C_j) = \frac{1 + TF(w_i, C_j)}{|F| + \sum_{w' \in |F|} TF(w', C_j)}$$

Ngoài ra còn có các phương pháp NB khác có thể kể ra như sau ML Naive Bayes, MAP Naive Bayes, Expected Naive Bayes, Bayesian Naive Bayes (Jason mô tả năm 2001). Naive Bayes là một công cụ rất hiệu quả trong một số trường hợp. Kết quả có thể rất tồi nếu dữ liệu huấn luyện nghèo nàn và các tham số dự đoán (như không gian đặc trưng) có chất lượng kém. Nhìn chung đây là một thuật toán phân loại tuyến tính thích hợp trong phân loại văn bản nhiều chủ đề. NB có ưu điểm là cài đặt đơn giản, tốc độ nhanh, dễ dàng cập nhật dữ liệu huấn luyện mới và có tính độc lập cao với tập huấn luyện, có thể sử dụng kết hợp nhiều tập huấn luyện khác nhau. Tuy nhiên NB ngoài giả định tính độc lập giữa các từ còn phải cần đến một ngưỡng tối ưu để cho kết quả khả quan. Nhằm mục đích cải thiện hiệu năng của NB, các phương pháp như multiclass-boosting, ECOC (do Berger trình bày năm 1999 và Ghani mô tả lại năm 2000) có thể được dùng kết hợp.

### II.3.2 Phương pháp K-Nearest Neighbor (k-NN)

Đây là phương pháp truyền thống khá nổi tiếng về hướng tiếp cận dựa trên thống kê đã được nghiên cứu trong nhận dạng mẫu hơn bốn thập kỷ qua (theo tài liệu của Dasarathy năm 1991). kNN được đánh giá là một trong những phương pháp tốt nhất (áp dụng trên tập dữ liệu Reuters phiên bản 21450), được sử dụng từ những thời kỳ đầu của

việc phân loại văn bản (được trình bày bởi Marsand năm 1992, Yang năm 1994, Iwayama năm 1995)

Ý tưởng chủ đạo của phương pháp này là khi cần phân loại một văn bản mới, thuật toán sẽ tính khoảng cách (khoảng cách Euclide, Cosine ...) của tất cả các văn bản trong tập huấn luyện đến văn bản này để tìm ra k văn bản gần nhất (gọi là k "láng giềng"), sau đó dùng các khoảng cách này đánh trọng số cho tất cả chủ đề. Trọng số của một chủ đề chính là tổng tất cả khoảng cách ở trên của các văn bản trong k láng giềng có cùng chủ đề, chủ đề nào không xuất hiện trong k láng giềng sẽ có trọng số bằng 0. Sau đó các chủ đề sẽ được sắp xếp theo mức độ trọng số giảm dần và các chủ đề có trọng số cao sẽ được chọn là chủ đề của văn bản cần phân loại.

Khi đó trọng số của chủ đề $c_j$ đối với văn bản $\vec{x}$:

$$W\left(\vec{x}, c_j\right) = \sum sim\left(\vec{x}, \vec{d}_i\right).y\left(\vec{x}, c_j\right) - b_j$$

Trong đó:

$y(\vec{d}_i, c_j) \in \{0,1\}$, với $y = 0$: văn bản $\vec{d}_i$ không thuộc về chủ đề $c_j$, $y = 1$: văn bản $\vec{d}_i$ thuộc về chủ đề $c_j$

$sim(\vec{x}, \vec{d}_i)$: độ giống nhau giữa văn bản cần phân loại $\vec{x}$ và văn bản $\vec{d}_i$. Có thể sử dụng độ đo cosine để tính $sim(\vec{x}, \vec{d}_i)$

$$sim(\vec{x}, \vec{d}_i) = \cos\left(\vec{x}, \vec{d}_i\right) = \frac{\vec{x}, \vec{d}_i}{\| \vec{x} \| . \| \vec{d}_i \|}$$

$b_j$ là ngưỡng phân loại của chủ đề $c_j$ được tự động học sử dụng một tập văn bản hợp lệ được chọn ra từ tập huấn luyện.

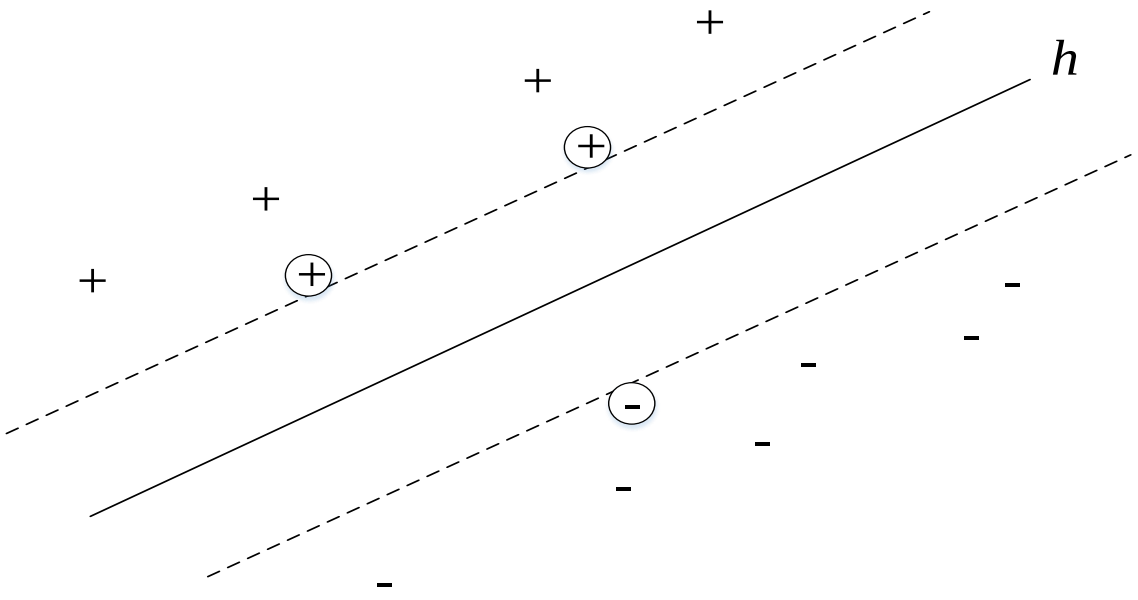
Để chọn được tham số k tốt nhất cho việc phân loại, thuật toán phải được chạy thử nghiệm trên nhiều giá trị k khác nhau, giá trị k càng lớn thì thuật toán càng ổn định và sai

sót càng thấp [29]. Giá trị tốt nhất được sử dụng tương ứng trên hai bộ dữ liệu Reuter và Oshumed là k = 45.

### II.3.3 Phương pháp Support vector Machine (SVM)

Support vector Machine (SVM) là phương pháp tiếp cận phân loại rất hiệu quả được Vapnik giới thiệu năm 1995 [30]

Ý tưởng của phương pháp này là cho trước một tập huấn luyện được biểu diễn trong không gian vector trong đó mỗi tài liệu là một điểm, phương pháp này tìm ra một siêu mặt phẳng h quyết định tốt nhất có thể chia các điểm trên không gian này thành hai lớp riêng biệt tương ứng lớp + (cộng) và lớp – (trừ). Chất lượng của siêu mặt phẳng này được quyết định bởi khoảng cách (gọi là biên) của điểm dữ liệu gần nhất của mỗi lớp đến mặt phẳng này. Khoảng cách biên càng lớn thì mặt phẳng quyết định càng tốt đồng thời việc phân loại càng chính xác. Mục đích thuật toán SVM tìm được khoảng cách biên lớn nhất để tạo được kết quả phân loại tốt.




Hình 2.2: Mô hình SVM

Có thể nói SVM thực chất là một bài toán tối ưu, mục tiêu của thuật toán là tìm được một không gian H và siêu mặt phẳng quyết định h trên H sao cho sai số khi phân loại là thấp nhất, nghĩa là kết qủa phân loại sẽ cho kết quả tốt nhất.

Phương trình siêu mặt phẳng chứa vector $d_i$ trong không gian như sau:

$$\vec{d_i}.\vec{w}+b=0$$

Khi đó đặt

$$h\left(\vec{d_i}\right)=sign\left(\vec{d_i}.\vec{w}\right)=\begin{cases} +, \vec{d_i}.\vec{w}+b>0 \\ -, \vec{d_i}.w+\vec{b}<0 \end{cases}$$

Như vậy vector h($d_i$) biểu diễn sự phân lớp của vector $d_i$ vào hai lớp. Gọi $Y_i$ mang giá trị +1 hoặc -1, khi đó $Y_i$ = +1 văn bản tương ứng với vector $d_i$ thuộc lớp (+) và ngược lại nó sẽ thuộc vào lớp (-). Khi đó để có siê mặt phẳng h, ta quay trở lại bài giải bài toán sau: Tìm Min $\|\vec{w}\|$ với $\vec{w}$ và b thỏa mãn điều kiện: $\forall i \in 1,n : y_i(sign(d_i - w+b)) \geq 1$

Điểm đặc biệt ở phương pháp SVM là mặt phẳng quyết định chỉ phụ thuộc vào các vector hỗ trợ (Support Vector) có khoảng cách đến mặt phẳng quyết định là $\frac{1}{\|\vec{w}\|}$ Khi các điểm khác bị xóa đi thì thuật toán vẫn cho kết quả giống như ban đầu. Chính đặc điểm này làm cho SVM khác với các thuật toán khác như kNN,LLSF, NNet và NB vì tất cả dữ liệu trong tập huấn luyện đều được dùng để tối ưu hóa kết quả. Các phiên bản SVM tốt có thể kể đến là SVM Light (Joachims trình bày năm 1998) [31] và Sequential Minimal Optimization (SMO) (Platt trình bày năm 1998) [32]

### II.3.4 Phương pháp Phương pháp Linear Least Square Fit (LLSF)

LLSF là một cách tiếp cận ánh xạ được phát triển bởi Yang và Chute vào năm 1992. Ban đầu LLSF được thử nghiệm trong lĩnh vực xác định từ đồng nghĩa sau đó sử dụng trong phân loại vào năm 1994. Các thử nghiệm cho thấy hiệu suất phân loại của LLSF có thể ngang bằng với phương pháp K-NN kinh điển.

Ý tưởng của LLSF là sử dụng phương pháp hồi quy để học từ tập huấn luyện và các chủ đề có sẵn.

Tập huấn luyện được biểu diễn dưới dạng một cặp vector đầu vào và đầu ra như sau:

- Vector đầu vào là một văn bản bao gồm các từ và trọng số.

- Vector đầu ra gồm các chủ đề cùng với trọng số nhị phân của văn bản ứng với vector đầu vào.

- Giải phương trình các cặp vector đầu vào, đầu ra chúng ta sẽ thu được ma trận đồng hiện của hệ số hồi quy của từ và chủ đề.

Phương pháp này sử dụng công thức: $F_{LS} = \arg\min = \| FA - B \|^2$

Trong đó :

- A, B là ma trận đại diện tập dữ liệu huấn luyện (các cột trong ma trận tương ứng là các vector đầu vào và đầu ra).

- $F_{LS}$ là ma trận kết quả chỉ ra một ánh xạ từ một văn bản bất kỳ vào vector của chủ đề đã gán trọng số.

Nhờ vào việc sắp xếp trọng số của các chủ đề, chúng ta được một danh sách chủ đề có thể gán cho văn bản cần phân loại. Nhờ đặt ngưỡng lên trọng số của các chủ đề mà ta tìm được chủ đề thích hợp cho văn bản đầu vào. Hệ thống tự động học các ngưỡng tối ưu cho từng chủ đề, giống với KNN. Mặc dù LLSF và KNN khác nhau về mặt thống kê, nhưng chúng ta vẫn tìm thấy điểm chung trong cách làm của hai phương pháp này là quá trình học ngưỡng tối ưu.

### II.3.5 Phương pháp Centroid - based vector

Là một phương pháp phân loại đơn giản, dễ cài đặt và tốc độ nhanh do có độ phức tạp tuyến tính O(n).

Ý tưởng của cách tiếp cận này là mỗi lớp trong dữ liệu huấn luyện sẽ được biểu diễn bằng một vector trọng tâm. Việc xác định lớp của một văn bản bất kỳ sẽ thông qua việc tìm vector trọng tâm nào gần với vector biểu diễn văn bản thứ nhất. Lớp của văn bản chính là lớp mà vector trọng tâm đại diện và khoảng cách được xác định theo độ đo Cosine.

Công thức tính vector trọng tâm của lớp i : $\vec{C}_i = \frac{1}{\| \{i\} \|} \sum_{d_j \in \{i\}} \vec{d}_j$

Độ đo khoảng cách giữa vector x và vector $C_i$: $\cos\left(\vec{x},\vec{C_i}\right)=\frac{\vec{x}.\vec{C_i}}{\|\vec{x}\|.\|\vec{C_i}\|}$

Trong đó : x là vector văn bản cần phân loại

{i} là tập hợp các văn bản thuộc chủ đề $C_i$

Chủ đề của vector x là $C_x$ thõa mãn $Cos(x, C_x) = \arg\max (Cos(x, C_i))$

### II.4 Khai thác đồ thị

Mục này chúng tôi sẽ trình bày một số khái niệm, định nghĩa quan trọng liên quan tới lý thuyết đồ thị, phục vụ cho bài toán phân loại văn bản dựa trên mô hình đồ thị đồ thị và khai phá đồ thị con phổ biến sẽ trình bày ở các mục tiếp theo.

#### II.4.1 Một số định nghĩa

##### II.4.1.1 Graph

Cho một nhãn node bằng chữ cái (alphabet) $L_V$ và một nhãn cạnh bằng chữ cái $L_E$ đồ thị $G$ (có hướng) được định nghĩa bằng bộ gồm 4 thành phần $G=(V,E,\mu,v)$, trong đó:

$V$: biểu diễn một tập hữu hạn các node.

$E \subseteq V \times V$ biểu diễn một tập các cạnh.

$\mu : V \to L_V$ biểu diễn một hàm ghi nhãn node.

$v : \mathrm{E} \to L_E$ biểu diễn một hàm ghi nhãn cạnh.

Tập $V$ có thể được coi là một tập các định danh nút và thường được chọn bằng $V = \{1, ..., | V |\}$. Trong khi $V$ xác định các nút, tập các cạnh $E$ thể hiện cấu trúc của đồ thị. Đó là một nút $u \in V$ được kết nối với một nút $v \in V$ bằng một cạnh= $(u, v)$ nếu $(u, v) \in E$. Hàm ghi nhãn có thể được sử dụng để tích hợp thông tin về các node và các cạnh vào trong các đồ thị bằng cách gán các thuộc tính từ $L_V$ và $L_E$ tới các node và các cạnh tương ứng.

Đồ thị được định nghĩa ở trên bao gồm một số trường hợp đặc biệt. Để định nghĩa đồ thị vô hướng, cho một thể hiện yêu cầu $(u,v) \in E$ cho mỗi cạnh $(u,v) \in E$ sao cho

$v(u,v) = v(v,u)$ Trong trường hợp đồ thị không thuộc tính, bảng chữ cái nhãn được xác định bởi $L_V, L_E = \varnothing$, bởi vậy mỗi node và mỗi cạnh được gán nhãn *null* nhãn $\varnothing$ . Đồ thị rỗng được định nghĩa bằng $G_\varepsilon = (\varnothing, \varnothing, \mu_\varepsilon, \nu_\varepsilon)$

### II.4.1.2 Đồ thị được gán nhãn

Đồ thị được gán nhãn: là một đồ thị được biểu diễn bởi $G = (V, E, L_V, L_E, \varphi_v, \varphi_e)$, trong đó:

$V$: tập hợp của các đỉnh đồ thị.

$E \subseteq V \times V$ : tập hợp các cạnh của đồ thị.

$L_V$: các nhãn đỉnh tương ứng.

$L_E$: các nhãn cạnh tương ứng.

$\mu_v : V \to L_V$ : hàm ánh xạ một đỉnh với nhãn của nó

$\mu_e : \mathrm{E} \to L_E$ : hàm ánh xạ một cạnh với nhãn của nó

Ta có một số thuật ngữ cơ bản liên quan tới đồ thị như sau:

Đường đi: là một dãy các đỉnh được sắp xếp theo thứ tự sao cho giữa hai đỉnh cạnh nhau được nối với nhau bởi một cạnh.

Chiều dài của đường đi: là số cạnh bên trong đường đi đó.

Chu trình: là một đường đi bắt đầu và kết thúc ở cùng một đỉnh.

Khuyên: là một cạnh được kết nối một đỉnh và chính nó.

Các cạnh bội: là có hai hay nhiều cạnh nối giữa hai đỉnh.

Đồ thị không chu trình: là đồ thị mà không có chu trình nào.

Đồ thị đầy đủ: là đồ thị mà có cạnh nối giữa mọi cặp đỉnh với nhau.

Đồ thị có hướng: là đồ thị mà các cạnh được mô tả bởi thứ tự của các đỉnh.

Đồ thị vô hướng: là đồ thị mà các cạnh được biểu diễn không quan tâm tới thứ tự của các đỉnh trong cạnh.

### II.4.1.3 Đồ thị con

Cho trước hai đồ thị $G_1 = (V_1, E_1, L_{V1}, L_{E1}, \varphi_{v1}, \varphi_{e1})$ và $G_2 = (V_2, E_2, L_{V2}, L_{E2}, \varphi_{v2}, \varphi_{e2})$

$G_1$ là một đồ thị con của $G_2$ nếu $G_1$ thỏa mãn các điều kiện sau:

$$V_1 \subseteq V_2, \forall v \in V_1, \varphi v_1(v) = \varphi v_2(v)$$

$$E_1 \subseteq E_2, \forall (u,v) \in E_1, \varphi_{E_1}(u,v) = \varphi_{E_1}(u,v)$$

$G_2$ cũng được gọi là siêu đồ thị của $G_1$. Ngoài ra, $G_1$ được gọi là đồ thị con cảm sinh của $G_2$ nếu $G_1$ thỏa mãn thêm điều kiện dưới đây:

$$\forall u,v \in V_1, (u,v) \in E_1 \Leftrightarrow (u,v) \in E_2$$

Định nghĩa đồ thị con cảm sinh ám chỉ rằng với một đồ thị con $G_1$ của $G_2$ được gọi là cảm sinh nếu với mọi cặp đỉnh xuất hiện đồng thời trong $G_1$ và $G_2$ thì các cạnh giữa các đỉnh này cũng phải xuất hiện đồng thời trong $G_1$ và $G_2$. Theo cách diễn đạt khác, một đồ thị con cảm sinh là một đồ thị con với một số các ràng buộc.

### II.4.1.4 Đồ thị đẳng cấu

Cho trước hai đồ thị $G_1 = (V_1, E_1, L_{V1}, L_{E1}, \varphi_{v1}, \varphi_{e1})$ và $G_2 = (V_2, E_2, L_{V2}, L_{E2}, \varphi_{v2}, \varphi_{e2})$

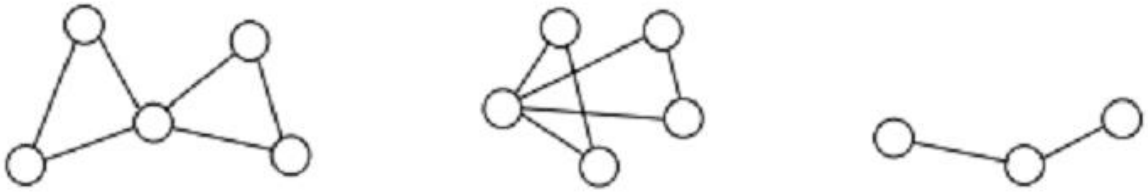
$G_1$ và $G_2$ được gọi là đẳng cấu nếu và chỉ nếu tồn tại một song ánh $f : V_1 \to V_2$ thỏa mãn các điều kiện sau:

$$\forall u \in V_1, \varphi v_1(u) = \varphi v_2(f(u))$$

$$\forall u,v \in V_1, (u,v) \in E_1 \Leftrightarrow (f(u), f(v)) \in E_2$$

$$\forall u,v \in E_1, \varphi_{E_1}(u,v) = \varphi_{E_1}(f(u), f(v))$$

Song ánh $f$ là một đẳng cấu giữa $G_1$ và $G_2$. Một đồ thị $G_1$ là một đồ thị con đẳng cấu với một đồ thị $G_2$, ký hiệu là: $G_1 \subseteq_{sub} G_2$ , nếu và chỉ nếu tồn tại một đồ thị đẳng cấu giữa chúng.



(a) (b) (c)

Hình 2.3 (b) đẳng cấu với (a), (c) đẳng cấu với một đồ thị con của (a)

### II.4.2 Phân lớp đồ thị

#### II.4.2.1. Giới thiệu về phân lớp đồ thị

Biểu diễn bằng đồ thị rất phù hợp cho việc biểu diễn các kiểu dữ liệu có cấu trúc như các trình tự sinh học, các hợp chất hóa học, các protein, RNA, các mạng xã hội, các văn bản, tài liệu bán cấu trúc như HTML, XML... Chủ đề về khai phá dữ liệu đồ thị đã không còn quá mới mẻ. Trong thời gian qua, đã có nhiều công trình nghiên cứu các phương pháp mới để xử lý dữ liệu đồ thị xuất phát từ nhu cầu thực tiễn của các lĩnh vực tin sinh học, tin hóa học, thị giác máy tính, phân tích các mạng xã hội... Ví dụ, NCBI PubChem có một tập hợp hàng triệu hợp chất hóa học được biểu diễn dưới dạng đồ thị các phân tử.

Phân lớp sử dụng đồ thị gồm có hai bài toán:

- Phân lớp đồ thị: gán nhãn cho toàn bộ một đồ thị. Được thực hiện giữa các đồ thị trong một tập hợp các đồ thị.

- Phân lớp đỉnh đồ thị: gán nhãn cho các đỉnh bên trong một đồ thị. Được thực hiện với một đồ thị đơn có kích thước lớn.

Bài toán phân lớp đồ thị lần đầu được nghiên cứu bởi [20], người đề xuất thuật toán SUBDUECL cho nhiệm vụ này và đã đạt được những kết quả đáng hứa hẹn. Tiếp cận sử dụng tìm kiếm tham lam cho các đồ thị con nhằm phân biệt một lớp các đồ thị với các lớp khác. Từ đó, rất nhiều các thuật toán mới cũng như cách tiếp cận mới đã được nghiên cứu cho bài toán này. [14] áp dụng hệ thống FSG để khai phá các đồ thị con phổ biến trong một cơ sở dữ liệu đồ thị và biểu diễn chúng dưới dạng các véc tơ đặc trưng, và một bộ phân lớp SVM được sử dụng để phân loại các véc tơ đặc trưng này. [15] đề xuất thuật toán DT-CLGBI sử dụng cây quyết định cho việc phân lớp đồ thị trong đó mỗi nút liên quan

tới một đồ thị con. [21] đề xuất một tiếp cận dựa trên SVM để phân lớp đồ thị với việc phát triển các nhân đồ thị.

### II.4.2.2. Một số kỹ thuật phân lớp đồ thị

#### a. SUBDUE

Thuật toán Subdue được đề xuất bởi Agrawal và Srikant [20] là thuật toán khởi đầu cho bài toán phân lớp đồ thị. Điểm mấu chốt của thuật toán này là dựa trên tư tưởng tham lam, tìm kiếm heuristic cho các đồ thị con xuất hiện trong các mẫu dương và vắng mặt trong các mẫu âm. Không gian tìm kiếm của Subdue được tạo thành bởi tất cả các đồ thị con liên thông của tất cả các đồ thị được gán nhãn đầu vào nằm trong các mẫu dương.

Subdue thực hiện việc tìm kiếm bắt đầu từ các đồ thị con được cấu thành từ tất cả các đỉnh với các nhãn đơn nhất. Các đồ thị con được mở rộng bởi một đỉnh và một cạnh hoặc một cạnh trong số tất cả các cách có thể dựa trên đồ thị đầu vào, để sinh ra các đồ thị con ứng cử. Subdue duy trì các đồ thị con này và sử dụng tính đẳng cấu đồ thị để xác định các cấu trúc con ứng cử của đồ thị đầu vào. Các cấu trúc con ứng cử được đánh giá dựa trên độ chính xác của việc phân lớp hay khái niệm độ dài mô tả cực tiểu được giới thiệu trong [22].

Độ dài của tia tìm kiếm xác định số lượng cấu trúc con ứng cử cho việc mở rộng sau này. Thủ tục này được lặp lại cho tới khi tất cả các cấu trúc được xem xét hay độ phức tạp tính toán vượt quá ngưỡng quy định bởi người dùng. Kết thúc thủ tục này, các mẫu dương được khám phá bởi cấu trúc tốt nhất bị loại bỏ. Tiến trình tìm kiếm các cấu trúc con và loại bỏ các mẫu dương tiếp tục cho tới khi tất cả các mẫu dương được khám phá.

Mô hình được học bởi Subdue được cấu thành bởi một danh sách quyết định, mỗi thành viên trong đó là đồ thị liên thông. Áp dụng mô hình này để phân loại các mẫu chưa biết bao gồm việc kiểm tra tính đẳng cấu của đồ thị con; nếu một đồ thị bất kỳ trong danh sách quyết định được xuất hiện trong mẫu, nó sẽ được dự đoán là dương, nếu tất cả đồ thị trong danh sách quyết định đều vắng  mặt trong mẫu đó, nó được dự đoán là âm.

#### b. Khai phá đồ thị con phổ biến kết hợp với SVM

Tiếp cận được giới thiệu bởi T. Asai và cộng sự [14] cho việc phân lớp đồ thị bao gồm hai công việc chính: khai phá đồ thị con phổ biến kết hợp với phân lớp SVM.

Bài toán khai phá đồ thị con phổ biến sẽ được trình bày chi tiết trong mục II.4.3, với các thuật toán như AGM, FSG, gSpan, GASTON...

Ý tưởng chính của việc kết hợp khai phá đồ thị con phổ biến và SVM để thực hiện việc phân lớp đồ thị đó là sử dụng một hệ thống khai phá đồ thị con phổ biến để xác định các đồ thị con phổ biến trong tập các mẫu cho trước, sau đó cấu trúc thành các véc tơ đặc trưng cho các mẫu này, trong đó mỗi đặc trưng là sự xuất hiện hay không của các đồ thị con cụ thể trong mẫu và huấn luyện một bộ SVM để phân lớp cho các véc tơ đặc trưng này.

Mô hình được xây dựng dựa trên tiếp cận này bao gồm một danh sách các đồ thị và một mô hình được tạo ra bởi SVM. Áp dụng mô hình này đểphân lớp các mẫu chưa biết bao gồm việc kiểm tra tính đẳng cấu của đồ thị con; một véc tơ đặc trưng cho mẫu chưa biết được tạo ra trong đó mỗi đặc trưng được biểu diễn bởi việc xuất hiện hay vắng mặt của một đồ thị trong danh sách trong đồ thị mẫu chưa biết và véc tơ đặc trưng này được phân lớp là âm hay dương bởi mô hình được xây dựng bởi SVM.

**c. Khai phá đồ thị con phổ biến kết hợp với AdaBoost**

Tiếp cận đề xuất bởi A. Inokuchi và cộng sự [23] bao gồm việc kết hợp khai phá đồ thị con phổ biến và AdaBoost. Việc khai phá đồ thị con phổ biến được sử dụng thuật toán gSpan. Việc phân lớp đồ thị được sử dụng bộ phân lớp mạnh AdaBoost.

### II.4.2.3. Các ứng dụng của phân lớp đồ thị

Phân lớp đồ thị được ứng dụng trong nhiều lĩnh vực khác nhau. Có thể kể đến một số nghiên cứu nổi bật sử dụng phân lớp đồ thị như:

- Phân lớp cấu trúc protein 3D với việc sử dụng phương pháp nhân đồ thị.

- Phân lớp RNA sinh học.

- Phân lớp hình ảnh sử dụng nhân đồ thị, trong đó mỗi khu vực tương ứng với một nút và các mối quan hệ giữa các vùng là các cạnh.

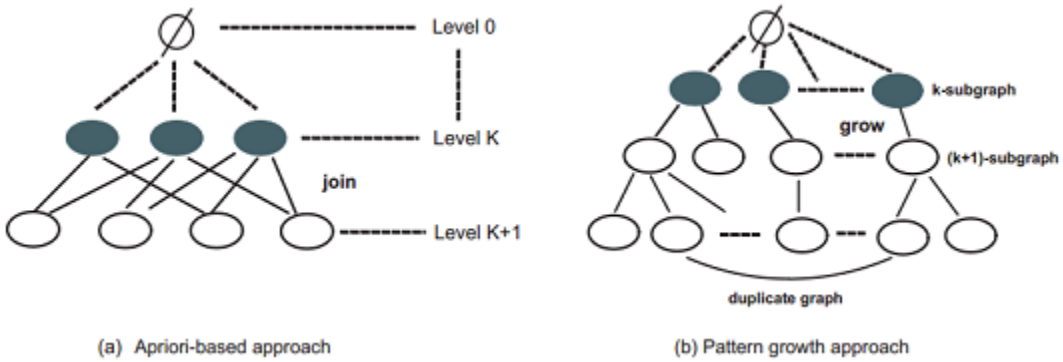
- Phân lớp các hợp chất hóa học nhỏ.

- Phân lớp văn bản ...

### II.4.3 Khai phá đồ thị con phổ biến

#### II.4.3.1 Tổng quan về khai phá đồ thị con phổ biến

Khai phá đồ thị con phổ biến (FSM) là vấn đề trọng tâm của khai phá đồ thị, được ứng dụng rộng rãi trong nhiều nhiệm vụ khác nhau như: phân loại hợp chất hóa học, phân loại hình ảnh, phân loại văn bản, đánh chỉ mục đồ thị, tìm kiếm đồ thị và nhiều lĩnh vực khác.




Hình 2.4 Hai cách tiếp cận của FSM

Các nhà nghiên cứu đều thống nhất chia các kỹ thuật FSM thành hai loại:

+ Tiếp cận dựa trên Apriori (còn được gọi là tiếp cận dựa trên chiến lược tìm kiếm theo chiều rộng - BFS).

+Tiếp cận dựa trên phát triển mẫu.

Hai cách tiếp cận trên cũng tương tự như trong khai phá luật kết hợp, với thuật toán Apriori [20] và thuật toán phát triển mẫu phổ biến (FP-growth) [24].

**Algorithm 2.1: Apriori – based approach**

**Input**: GD = a graph dataset, $\sigma$ = minimum support

**Output**: $F_1, F_2, ..., F_k$, a set of frequent subgraph sets

1 $F_1 \leftarrow$ detect all frequent 1 – subgraphs in GD

2 $k \leftarrow 2$

3 **While** $F_{k-1} \neq \phi$ **do**

4 $F_k \leftarrow \phi$

5 $C_k \leftarrow candidate - gen(F_{k-1})$

6 **foreach** $candidate\, g_k \in C_k$ **do**

7 $g_k.count \leftarrow 0$

8 **foreach** $G_i \in$ GD **do**

9 **if** subgraph-insomorphism($g_k$, $G_i$) **then**

10 $g_k.count \leftarrow g_k.count + 1$

11 **end**

12 **end**

13 **if** $g_k.count \geq \sigma|GD| \wedge g_k \notin F_k$ **then**

14 $F_k = F_k \cup g_k$

15 **end**

16 **end**

17 $k \leftarrow k + 1$

18 **end**

Với cách tiếp cận dựa trên Apriori sử dụng chiến lược tìm kiếm theo chiều rộng (BFS) để khám phá ra các đồ thị con từ cơ sở dữ liệu cho trước. Thuật toán sinh các đồ thị con ứng cử từ các đồ thị con phổ biến đã biết ở bước trước, sử dụng kỹ thuật tỉa để bỏ đi các đồ thị con ứng cử không thỏa mãn ngưỡng hỗ trợ cho trước. Ý tưởng chính của thuật toán:

- Dùng tập đồ thị con phổ biến kích thước (k-1) để tạo các tập ứng cử kích thước k.

- Duyệt cơ sở dữ liệu và đối sánh mẫu (phát hiện đồ thị đẳng cấu) để đếm số lần xuất hiện của tập đồ thị con ứng cử trong các đồ thị giao tác. Nếu số lần xuất hiện của đồ thị con ứng cử lớn hơn hoặc bằng độ hỗ trợ tối thiểu minsup thì nó là đồ thị con phổ biến và ngược lại.

- Quá trình được lặp lại cho đến khi không còn tập đồ thị con phổ biến nào được tạo ra.

**Algorithm 2.2: Pattern growth approach**

**Input**: $g$ = *a* frequent subgraph, $\sigma$ = minimum support, GD = *a* graph dataset

**Output**: $F$, *a* set of frequent subgraphs

1 $F \leftarrow \emptyset$

2 $F_1 \leftarrow$ detect all frequent 1 – subgraph in GD

3 $k \leftarrow 1$

4 **foreach** g $\in F_1$ **do**

5 Pattern – growth($g$, GD, $\sigma, F$)

6 **end**

7 **Function**: Pattern – growth($g$, GD, $\sigma, F$)

8 $k \leftarrow k+1$

9 $C_k \leftarrow \phi$

10 **if** $g \in F$ **then**

11 return

12 **else**

13 $F \leftarrow F \cup g$

14 **end**

15 scan GD, find all the edges $e$ such that $g$ can be extended to $g \cup e, g \leftarrow g \cup e$, and insert $g$ into $C_k$

16 **foreach** $g_k \in C_k$ **do**

17 **if** $g_k.count \geq \sigma|GD|$ **then**

18 Pattern – growth($g_k, GD, \sigma, F$)

19 **else**

20 **return**

21 **end**

22 **end**

Tiếp cận dựa trên phát triển mẫu có thể sử dụng đồng thời cả hai chiến lược tìm kiếm theo chiều rộng (BFS) và chiều sâu (DFS), tuy nhiên nó thường xuyên được sử dụng hơn vì nó tiết kiệm bộ nhớ hơn. Trong tiếp cận Apriori thì các đồ thị con ứng cử kích thước k+1 được sinh ra bởi việc kết hợp hai đồ thị con phổ biến kích thước k, trong khi

với tiếp cận phát triển mẫu, tập đồ thị con ứng cử kích thước k+1 được tạo ra bởi việc mở rộng một đồ thị con phổ biến kích thước k. Trong thuật toán trên, với mỗi đồ thị con phổ biến g được khai phá, thuật toán sẽ phát triển g một cách đệ quy cho tới khi nào tất cả các đồ thị con phổ biến của g được khám phá.

### II.4.3.2. Một số thuật toán khai phá đồ thị con phổ biến

a. Chiến lược tìm kiếm theo chiều rộng (BFS)

Chiến lược tìm kiếm BFS được sử dụng rộng rãi do nó được mở rộng từ tư tưởng của thuật toán Apriori cho việc khai phá tập mục phổ biến. Điểm mấu chốt của Apriori nằm ở chỗ nó nhận thấy thực tế rằng tất cả các tập con của một tập mục phổ biến đều là phổ biến. Các thuật toán FGM dựa trên BFS cũng phát triển ý tưởng dựa trên tư tưởng của Apriori này. Một số thuật toán FGM sử dụng chiến lược BFS:

+ AGM: sử dụng ma trận kề để biểu diễn các đồ thị và tìm kiếm theo bậc để khám phá các đồ thị con phổ biến. Nó giả sử rằng tất cả các đỉnh trong đồ thị là phân biệt. AGM không chỉ khám phá các đồ thị con liên thông mà còn cả các đồ thị con không liên thông. Để ứng dụng trong các bài toán thực tế, một phiên bản cải tiến của AGM là AcGM được đề xuất để khai phá các đồ thị liên thông phổ biến. Thuật toán này vẫn sử dụng các khái niệm và biểu diễn đồ thị tương tự như AGM nhưng có đưa thêm vào một số cải tiến. Kết quả thực nghiệm chỉ ra rằng AcGM nhanh hơn đáng kể so với AGM và FSG.

+ FSG: là thuật toán nổi tiếng khác dựa trên tư tưởng Apriori, được đề xuất bởi Kuramochi [21]. FSG tập trung vào việc tìm tất cả các đồ thị con phổ biến liên thông trong một tập dữ liệu đồ thị rất lớn. FSG sử dụng chiến lược BFS để phát triển tập ứng cử, trong đó các cặp đồ thị con phổ biến mức k được hợp với nhau để sinh ra các đồ thị con mức k+1. FSG sử dụng phương pháp gán nhãn kinh điển để so sánh và tính toán độ hỗ trợ của các mẫu sử dụng danh sách ID giao dịch (TID - biểu diễn dữ liệu một đỉnh). Mỗi danh sách TID biểu diễn một đồ thị con được khai phá dựa trên tập các giao dịch (TIDs) của các đồ thị. Để xác định độ hỗ trợ của một đồ thị con cấp k, trước tiên cần tính toán giao của các danh sách TID của các đồ thị con phổ biến cấp k -1 của nó. Nếu kích thước giao nhỏ hơn ngưỡng thì đồ thị con cấp k này không phổ biến và sẽ được cắt tỉa, ngược lại độ hỗ trợ được tính toán dựa trên việc phát hiện đồ thị con đẳng cấu tương ứng với giao của

các danh sách TID. Đếm độ hỗ trợ dựa trên danh sách TID làm giảm việc quá tải của tính toán, tuy nhiên thực nghiệm cho thấy FSG không hoạt động tốt khi các đồ thị giao dịch chứa nhiều đỉnh và cạnh.

+ gFSG: thuật toán FSG ở trên thường áp dụng cho các đồ thị trong không gian hai chiều (đỉnh và cạnh), trong khi bài toán phân tích hợp chất hóa học đòi hỏi phải quan tâm tới tọa độ của các đỉnh trong không gian hai hoặc ba chiều. gFSG được mở rộng cho FSG để khai phá các đồ thị con hình học phổ biến. Các đặc trưng hình học ở đây gồm việc quay, tỷ lệ, sự dịch chuyển. Thực nghiệm cho thấy hiệu năng của gFSG trên bộ dữ liệu hóa học gồm 20.000 hợp chất hóa học (các đồ thị phân tử) rất tốt khi giá trị ngưỡng hỗ trợ nhỏ và tuyến tính với kích thước của dữ liệu.

**b. Chiến lược tìm kiếm theo chiều sâu (DFS)**

Các thuật toán dựa trên DFS cần ít bộ nhớ hơn bởi chúng duyệt qua không gian tất cả các đồ thị con phổ biến theo chiến lược tìm kiếm chiều sâu. Một thuật toán nổi bật:

+ MoFa: được đề xuất bởi Borgelt, MoFa tập trung khai phá các cấu trúc con (đồ thị con liên thông) phổ biến bên trong các phân tử. Thuật toán lưu trữ danh sách các đồ thị con đã tìm thấy ở bước trước. MoFa cũng sử dụng cấu trúc cắt tỉa và các kiến thức nền tảng để làm giảm phép tính toán độ hỗ trợ. Tuy nhiên, MoFa sinh ra nhiều đồ thị con ứng cử trùng lặp, do đó dẫn tới các phép tính toán độ hỗ trợ không cần thiết.

+ gSpan: được đề xuất bởi Yan và Han [18], sử dụng biểu diễn kinh điển M-DFSC, để biểu diễn một đồ thị con đơn nhất. Thuật toán sử dụng thứ tự từ điển DFS để cấu trúc một lưới dạng cây trên toàn bộ các mẫu khả thi, kết quả tạo ra một không gian tìm kiếm phân cấp gọi là cây mã DFS. Mỗi nút trongcây tìm kiếm này được biểu diễn bởi một mã DFS. Bậc thứ k+1 của cây có các nút chứa các mã DFS của các đồ thị con mức k. Các đồ thị con mức k được sinh ra bởi việc mở rộng một cạnh từ bậc k của cây. Cây tìm kiếm này được duyệt theo chiều sâu (DFS) và tất cả cac đồ thị con có mã DFS không cực tiểu sẽ bị cắt tỉa, do đó giúp giảm được các đồ thị con ứng cử dư thừa. Thay bằng việc lưu danh sách cho mỗi đồ thị con được khai phá, gSpan chỉ lưu danh sách giao dịch của mỗi mẫu đã khai phá và việc phát hiện các đồ thị con đẳng cấu chỉ áp dụng cho các đồ thị bên trong danh

sách này. Do vậy, gSpan tiết kiệm bộ nhớ sử dụng. gSpan là một trong số các thuật toán FGM tốt nhất hiện nay và cũng là thuật toán cơ sở để em xây dựng mô hình phân loại văn bản dựa vào mô hình đồ thị. Chi tiết thuật toán gSpan sẽ được trình bày trong chương 3 của luận văn.

+ FFSM: các thuật toán như FSG và gSpan thường hoạt động rất hiệu quả với các đồ thị nhỏ các nhiều đỉnh với nhãn khác nhau. Tuy nhiên, chúng lại hoạt động kém hiệu quả khi đồ thị lớn, mật độ dày và chỉ với ít nhãn đỉnh, chẳng hạn như cấu trúc các protein. Huan et al [13] đã đề xuất một thuật toán mới gọi là FFSM, để áp dụng cho các đồ thị dạng này. FFSM sử dụng biểu diễn kinh điển CAM. Đánh giá hiệu năng thuật toán trên các bộ dữ liệu hóa học cho thấy FFSM hoạt động tốt hơn gSpan.

+ GASTON: áp dụng cho các đồ thị ở dạng các cây tự do. Thuật toán chia thành ba pha chính: khai phá đường đi, khai phá cây con và khai phá đồ thị con. GASTON sử dụng một hàm băm và phát hiện đồ thị đẳng cấu để xác định sự trùng lặp của ứng cử. GASTON cũng ghi lại danh sách này để phát triển các mẫu trong đồ thị giao dịch, giúp tiết kiệm các phép kiểm tra đẳng cấu không cần thiết. Còn rất nhiều thuật toán khai phá đồ thị con phổ biến được sử dụng cho bài toán khai thác đồ thị con phổ biến.

**II.5 Kết luận**

Trong chương 2, chúng tôi đã trình bày các kiến thức nền tảng liên quan đến phân loại văn bản, một số phương pháp phân loại văn bản thường được sử dụng. Các thuật toán phân loại trên đều có điểm chung là yêu cầu văn bản phải được biểu diễn dưới dạng vector đặc trưng. Ngoài ra các thuật toán như KNN, NB đều phải sử dụng các ước lượng tham số và ngưỡng tối ưu khi phân loại văn bản. Bên cạnh đó cần một tập dữ liệu huấn luyện chuẩn và đủ lớn để cho thuật toán học phân loại. Ngoài ra các thuật toán này không có tính tăng cường (incremental function) nghĩa là phân loại lại toàn bộ tập văn bản khi thêm một số văn bản mới vào tập dữ liệu.

Tiếp đến là các kiến thức nền tảng liên quan tới bài toán phân loại văn bản dựa trên mô hình đồ thị. Cụ thể, em trình bày các khái niệm cơ bản liên quan đến đồ thị như: đồ thị được gán nhãn, các tính chất của đồ thị, các khái niệm về đồ thị đẳng cấu, đồ thị con. Một số thuật toán tiểu biểu cho bài toán phân lớp đồ thị cũng được em trình bày ở

đây. Cuối cùng là một thuật toán nổi tiếng áp dụng cho bài toán khai phá đồ thị con phổ biến như: thuật toán gSpan, thuật toán FSG, gFSG, GASTON ... Phân tích, đánh giá các thuật toán về độ phức tạp, phạm vi áp dụng, em nhận thấy gSpan rất phù hợp cho bài toán phân loại văn bản dựa trên mô hình đồ thị Từ những nghiên cứu này, luận văn xây dựng mô hình phân loại văn bản sử dụng thuật toán gSpan và bộ phân lớp SVM, sẽ được trình bày chi tiết trong chương 3.

# CHƯƠNG III: MÔ TẢ BÀI TOÁN và XỬ LÝ BÀI TOÁN

## III.1 Giới thiệu

Phân loại văn bản là quá trình gán văn bản vào một hoặc nhiều chủ đề đã xác định trước. Phân loại văn bản tự động là một lĩnh vực nghiên cứu được quan tâm trong nhiều năm qua do khả năng ứng dụng rộng rãi. Rất nhiều phương pháp phân loại như Naïve Bayes, cây quyết định, k-láng giềng gần nhất (k-NN), mạng nơron, máy vectơ hỗ trợ (SVM) đã áp dụng vào bài toán loại văn bản. Các phương pháp này đều sử dụng mô hình không gian vectơ khi biểu diễn văn bản.

Mô hình không gian vectơ là phương pháp biểu diễn văn bản phổ biến. Trong đó, mỗi từ trong văn bản có thể trở thành đặc trưng (hay chiều của vectơ biểu diễn văn bản). Mặc dù mô hình này cho kết quả phân loại tốt, nhưng nó cũng tồn tại các hạn chế. Mô hình không gian vectơ truyền thống chỉ tập trung vào tần suất xuất hiện của từ và không nắm bắt được các mối quan hệ của từ trong văn bản.

Trong những năm gần đây mô hình biểu diễn văn bản bằng đồ thị phát triển mạnh và áp dụng trong phân loại văn bản. Mô hình đồ thị có khả năng hạn chế nhược điểm của biểu diễn vectơ truyền thống khi lưu lại được mối quan hệ giữa các từ trong văn bản.

Trong luận văn, chúng tôi trình bày một phương pháp mới trong việc phân loại văn bản dựa trên biểu diễn đồ thị và kỹ thuật khai thác đồ thị. Các văn bản được biểu diễn dưới dạng đồ thị đơn giản. Kỹ thuật khai thác đồ thị con phổ biến gSpan đưa ra các đồ thị con phổ biến, kết hợp phương pháp phân loại SVM để phân loại văn bản.

## III.2 Quy trình phân loại văn bản dựa trên mô hình đồ thị

### III.2.1 Tiền xử lý văn bản

Các văn bản thuộc cùng một lớp tạo nên tập dữ liệu huấn luyện nhằm xác định các mẫu phổ biến. Việc đầu tiên chúng tôi làm là tách các từ. Để tách các từ trong từng văn bản trong luận văn chúng tôi sử dụng bộ thư viện tách từ Jvntextpro được phát triển bởi tác giả Nguyễn Cẩm Tú được cho tại địa chỉ: http://jvntextpro.sourceforge.net/ Đây là bộ thư viện mã nguồn mở trong java. Bước tiếp theo hệ thống sẽ tính tần suất xuất hiện của các từ trong tập văn bản. Để giảm kích thước của đồ thị và thời gian tính toán đồ thị con phổ biến, chỉ những từ có tần suất xuất hiện cao được giữ lại. Những từ có tần suất xuất

hiện ít sẽ được loại bỏ. Phương pháp thống kê tần suất xuất hiện và tính trọng số của từ được chúng tôi sử dụng trong luận văn là phương pháp TF - IDF. Sau khi loại bỏ những từ có tần suất xuất hiện thấp chúng tôi thu được bộ từ khóa cho từng văn bản. Bộ từ khóa này sẽ được dùng để xây dựng đồ thị văn bản sau này.

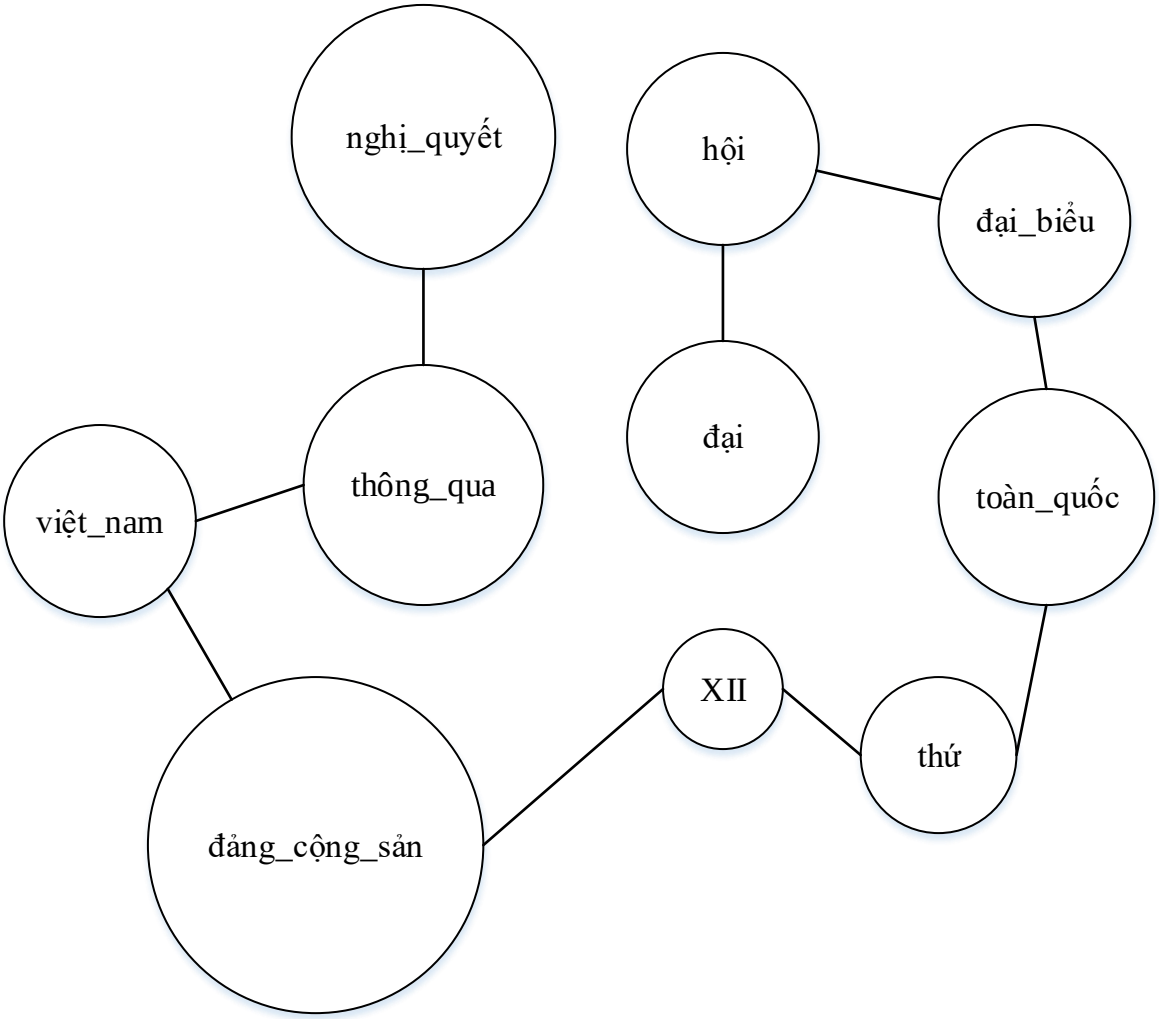
### III.2.2 Mô hình hóa văn bản thành đồ thị

Ưu điểm chính của mô hình biểu diễn văn bản bằng đồ thị là mô hình này có thể lưu giữ các mối quan hệ của các từ trong văn bản ban đầu. Có nhiều phương pháp xây dựng đồ thị từ văn bản như: mô hình đồ thị hình sao, mô hình đồ thị tần số vô hướng, mô hình đồ thị đơn giản, .... Nhìn chung các kiểu biểu diễn văn bản bằng đồ thị đều sử dụng sự liền kề của các từ trong văn bản.

Trong luận văn chúng tôi sử dụng mô hình đồ thị vô hướng để biểu diễn văn bản. Trong phương pháp biểu diễn văn bản bằng đồ thị này đỉnh của đồ thị biểu diễn các "từ" trong văn bản, các đỉnh được gán nhãn duy nhất là tên của "từ". Sau bước tiền xử lý văn bản, nếu từ *a* đứng ngay trước từ *b* thì sẽ tồn tại cạnh nối từ đỉnh *a* đến đỉnh *b* (không kể các trường hợp phân cách bởi dấu câu, dấu phẩy).

Ví dụ ta có văn bản sau: Đại hội đại biểu toàn quốc lần thứ XII Đảng Cộng sản Việt Nam đã thông qua Nghị quyết.

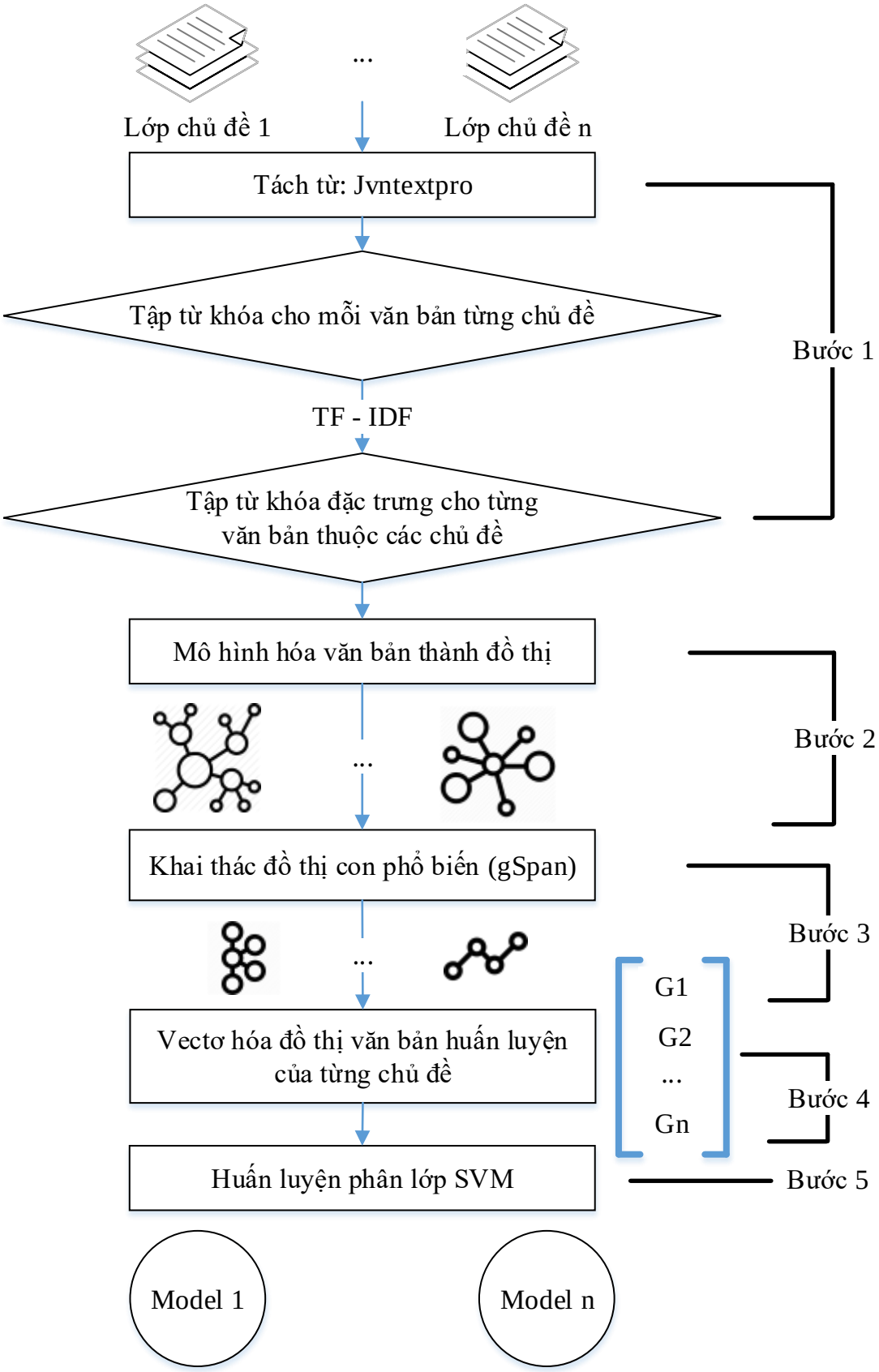
Sau bước tiền xử lý văn bản và mô hình hóa văn bản thành đồ thị chúng tôi thu được đồ thị như sau:




Hình 3.1 Ví dụ mô hình đồ thị văn bản chủ đề Chính trị - xã hội

### III.2.4 Mô hình phân loại văn bản dựa trên kỹ thuật khai thác đồ thị

Ý tưởng chính của phương pháp phân loại văn bản dựa trên mô hình đồ thị được chúng tôi mô tả trong hình 3.2 dưới đây:




Hình 3.2 Huấn luyện phân loại văn bản dựa trên mô hình đồ thị

**Giải thích mô hình:**

**Trong pha huấn luyện phân loại (TRAINING):**

- Đầu vào là dữ liệu dạng văn bản, trong pha huấn luyện trải qua các bước:

+ **Bước 1:** Tập văn bản đưa qua bộ tách từ Jvntextpro được phát triển bởi tác giả Nguyễn Cẩm Tú tại địa chỉ: http://jvntextpro.sourceforge.net/ Đây là bộ thư viện mã nguồn mở trong java.

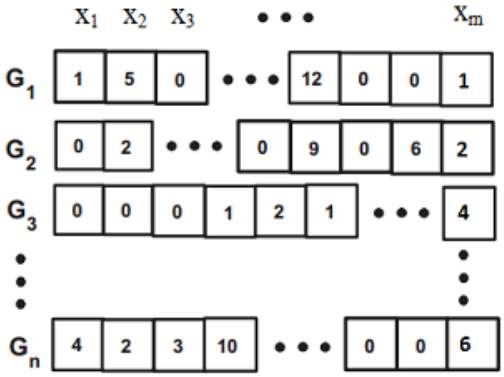
Sau bước tách từ, chúng tôi thu được tập dữ liệu là bộ từ khóa cho mỗi văn bản của từng chủ đề. Khi thu được bộ từ khóa cho từng văn bản trong mỗi chủ đề chúng tôi tính tần suất xuất hiện và trọng số của từ trong văn bản. Trong luận văn tác giả sử dụng phương pháp thống kê tần suất xuất hiện và tính trọng số của từ theo phương pháp TF-IDF. Sau bước này các từ không vượt ngưỡng bị loại bỏ. Cuối cùng chúng ta thu được tập dữ liệu là bô từ khóa đặc trưng cho từng văn bản thuộc các chủ đề.

+ **Bước 2 :** Sau khi thu được bộ từ khóa cho từng văn bản, chúng ta sẽ mô hình hóa văn bản thành đồ thị. Luận văn sử dụng đồ thị vô hướng đơn giản, mỗi văn bản là một đồ thị. Đỉnh biểu diễn "từ" trong văn bản. Các đỉnh được gán nhãn duy nhất là tên của "từ" trong văn bản. Sau bước tiền xử lý văn bản, nếu từ *a* đứng ngay trước từ *b* thì sẽ tồn tại cạnh nối từ đỉnh *a* đến đỉnh *b*. Để hệ thống hoạt động tốt thì đầu vào tập dữ liệu huấn luyện phải chứa mẫu của tất cả các lớp cần phân loại. Sau bước "Mô hình hóa văn bản thành đồ thị" chúng ta thu được tập cơ sở dữ liệu đồ thị, với các đỉnh và cạnh đã được gán nhãn, cùng với lớp của văn bản (đã biết trước đồ thị thuộc lớp nào).

+ **Bước 3:** Module "Khai phá đồ thị con phổ biến": thực hiện thuật toán gSpan để tìm tất cả các đồ thị con phổ biến của tập đồ thị đã được mô hình hóa với một độ hỗ trợ minsup. Ta được tập đồ thị con phổ biến $S = \{S_1, S_2, ..., S_n\}$ cho tất cả các chủ đề.

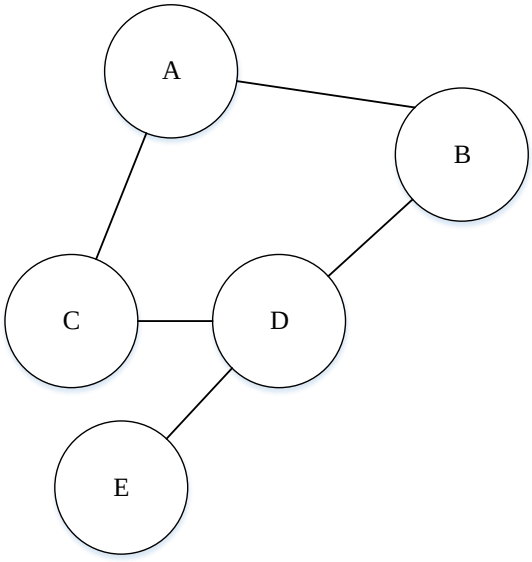
+ **Bước 4:** Lần lượt vec tơ hóa các đồ thị của từng chủ đề $G = \{G_1, G_2, ...,G_n\}$. Tập đồ thị của từng chủ đề được chiếu lên không gian đặc trưng của tập đồ thị con phổ biến S để nhận được các vec tơ đặc trưng tương ứng. Biểu diễn dưới dạng vec tơ đặc trưng của các đồ thị $G_i = \{x_1, x_2, ..., x_m\}$ được mô tả ở hình 3.3. Trong đó, $x_j$ chính là tần suất (số lần

xuất hiện) của đồ thị con phổ biến $S_j$ trong đồ thị $G_i$. Ở đây, số lần xuất hiện cần phải xét cả đến tính đẳng cấu của $S_j$ với các đồ thị con của $G_i$. Nếu vec tơ biểu diễn dưới dạng nhị phân thì $x_j = \{0, 1\}$, trong đó, $x_j = 0$ nếu $S_j$ không xuất hiện trong $G_i$ và $x_j = 1$ nếu tồn tại $S_j$ trong $G_i$. Có thể thấy, nếu biểu diễn dưới dạng vec tơ nhị phân thì sẽ tiết kiệm được thời gian tính toán. Số chiều vec tơ $G_i$ chính bằng số lực lượng của tập đồ thị con phổ biến S.




Hình 3.3 Cấu trúc các vec tơ đặc trưng của đồ thị

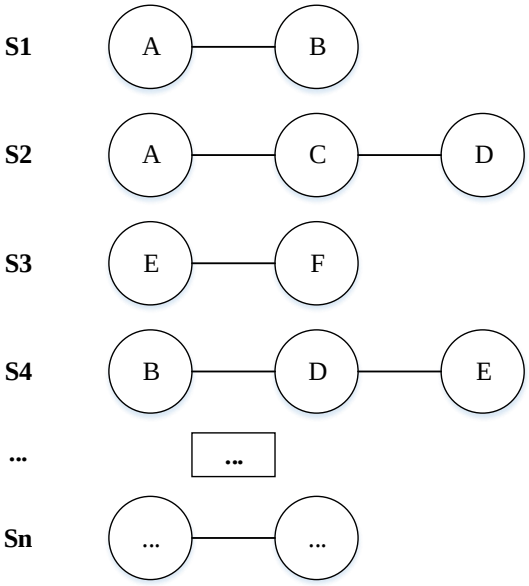
Ví dụ về vec tơ hóa đồ thị:




**Đồ thị văn bản G1**

**Vec tơ đồ thị văn bản G1 thu được**
**[1:1 2:1 3:0 4:1 ... n:?]**

**Tập đồ thị con phổ biến S = {S1, S2, ... , Sn}**

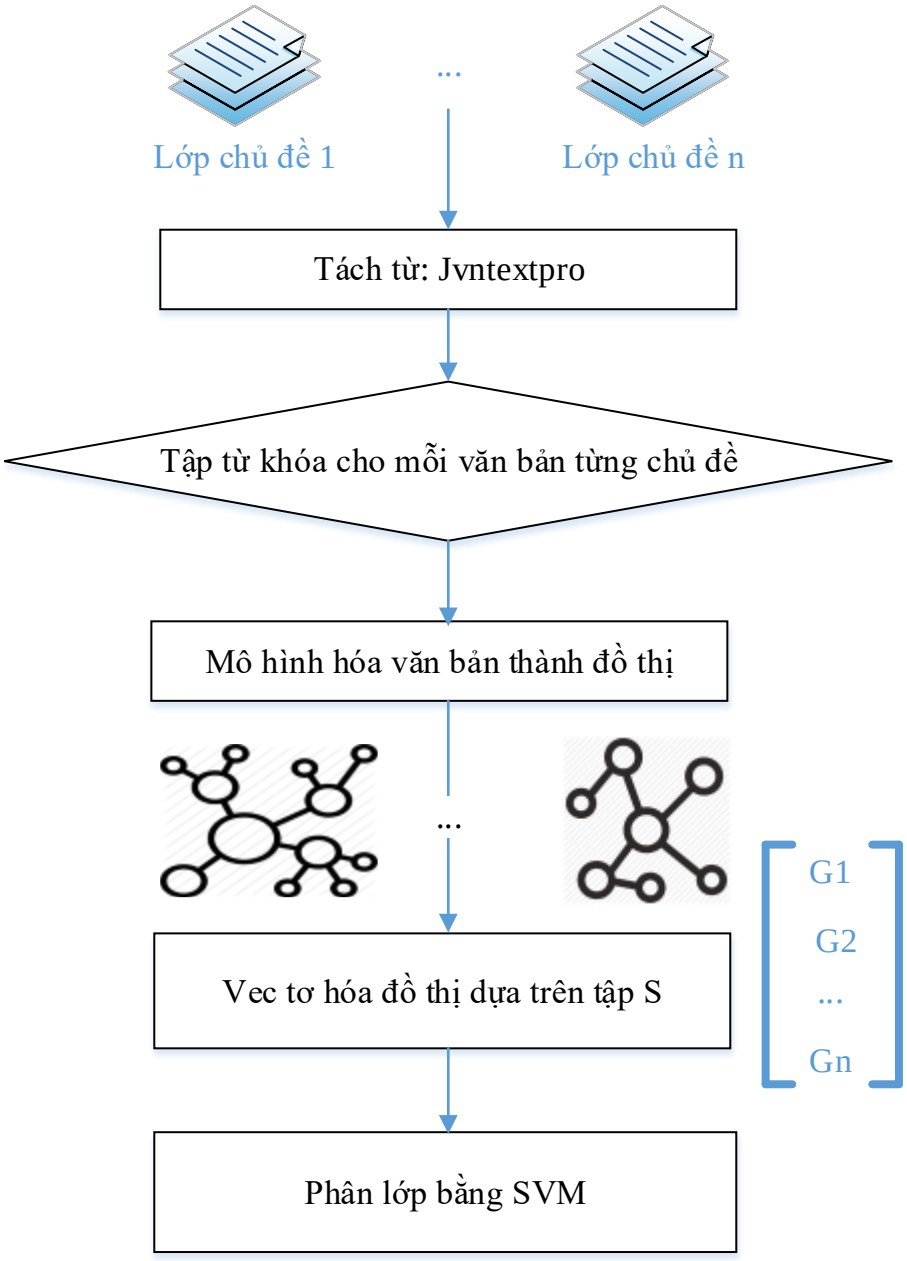


Hình 3.4 Vec tơ hóa đồ thị

+ **Bước 5:** "Huấn luyện phân lớp SVM": Sau khi có được các vec tơ đặc trưng của các văn bản trong từng chủ đề, tiến hành huấn luyện để phân lớp sử dụng các bộ phân lớp như: SVM, Naive Bayes, mạng nơron, cây quyết định... Trong luận văn, chúng tôi sử dụng SVM, bộ phân lớp rất phổ biến hiện nay.

Các vec tơ đặc trưng đầu vào sau khi qua bộ huấn luyện SVM sẽ cho ra các mô hình huấn luyện, sử dụng cho việc phân lớp văn bản sau này.

**Trong pha kiểm tra phân loại (TESTING):**




Hình 3.5 Phân loại văn bản dựa trên mô hình đồ thị

- Dữ liệu đầu vào là một hoặc tập các văn bản bất kỳ chưa được phân lớp. Quá trình kiểm tra phân loại văn bản trải qua các bước như sau:

+ **Bước 1:** Chúng tôi thực hiện công việc tách từ bằng bộ thư viện Jvntextpro.

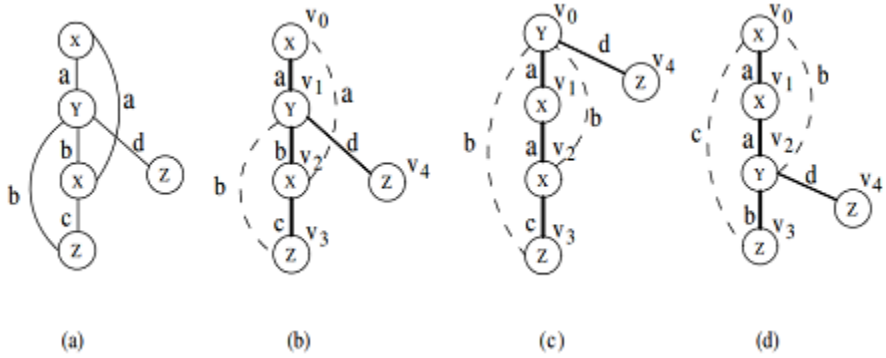
+ **Bước 1:** Sau bước tách từ việc mô hình hóa văn bản thành đồ thị, trích chọn đặc trưng của các đồ thị đã được mô hình hóa bằng cách chiếu lên không gian đặc trưng S (như ở pha huấn luyện) để nhận được các vec tơ đặc trưng tương ứng sẽ được thực hiện.

**+ Bước 3:** Khi có được tập vec tơ đăc trưng chúng ta đưa qua bộ phân lớp SVM đã được huấn luyện để phân lớp văn bản.

### III.2.5. Xây dựng pha khai phá đồ thị con phổ biến với thuật toán gSpan

**Thứ tự từ điển DFS**

+ Đánh chỉ số DFS: Khi thực hiện tìm kiếm theo chiều sâu [15] trong một đồ thị. Ta xây dựng một cây DFS. Một đồ thị có thể có nhiều cây DFS khác nhau. Ví dụ, các đồ thị trong hình 3.6 (b) - (d) là đẳng cấu với đồ thị trong hình 3.6 (a).




Hình 3.6 Cây DFS [10]

Việc làm dày các cạnh trong hình 3.6(b) - (d) biểu diễn cho ba cây DFS khác nhau cho đồ thị trong hình 3.6(a). Tìm kiếm theo chiều sâu phát hiện các đỉnh hình thành một trật tự tuyến tính. Sử dụng chỉ số dưới để gán nhãn cho thứ tự số lần phát hiện của chúng i <j nghĩa là $v_i$ được phát hiện trước $v_j$. Gọi $v_0$ nút gốc và vn nút ngoài cùng bên phải, đường nối từ $v_0$ đến $v_n$ là đường ngoài cùng bên phải. Trong hình 3.6(b) - (d) có ba chỉ số dưới được tạo ra cho các đồ thị trong hình 3.6(a). Hầu hết đường ngoài cùng bên phải là ($v_0$, $v_1$, $v_4$) trong hình 3.6(b), ($v_0$, $v_4$) trong hình 3.6(c) và ($v_0$, $v_1$, $v_2$, $v_4$) trong hình 3.6(d). Biểu diễn chỉ số dưới G là $G_T$.

+ Cạnh tiến và cạnh lùi:

Cho cạnh tiến $G_T$ là tập chứa tất cả các cạnh trong cây DFS và cạnh lùi là tập chứa tất cả các cạnh mà không có trong cây DFS.

Ví dụ: (i, j) là một cặp đỉnh có thứ tự để biểu diễn cho một cạnh. Nếu i <j thì nó là cạnh tiến và ngược lại là cạnh lùi.

Một thứ tự tuyến tính $\propto_T$ được xây dựng dựa vào tất cả các cạnh trong G theo các quy tắc sau đây:

Giả sử $e_1 = (i_1, j_1)$, $e_2 = (i_2, j_2)$:

(i) if $i_1 = i_2$ and $j_1 < j_2$ then $e_1 \propto_T e_2$;

(ii ) if $i_1 < j_1$ and $j_1 = i_2$ then $e_1 \propto_T e_2$

(iii) if $e_1 \propto_T e_2$ and $e_2 \propto_T e_3$ then $e_1 \propto_T e_3$.

* Định nghĩa 1 (Mã DFS)

Cho một cây DFST của một đồ thị G, một cạnh tuần tự ($e_i$) có thể được xây dựng dựa trên $\propto_T$, như là $e_i \propto_T e_{i+1}$, Trong đó i = 0,…, |E | - 1. ($e_i$) được gọi là một mã DFS, ký hiệu là mã (G, T).

Ví dụ: Một cạnh có thể được biểu diễn bởi một bộ gồm năm thành phần (i, j, $l_i$, $l_{(i,j)}$, $l_j$) trong đó $l_i$ và $l_j$ là nhãn của $v_i$ và $v_j$ và $l_{(i, j)}$ là nhãn cạnh giữa chúng.

Ví dụ: ($v_0$, $v_1$) trong hình 3.6(b) được biểu diễn bằng (0, 1, X, α, Y).

Bảng 3.1 cho thấy mã DFS tương ứng cho hình 3.6(b), 3.6(c), và 3.6(d).

Bảng 3.1: Mã DFS cho hình 3.6(b)-(d)

| Cạnh | (Hình 3.6b) α | (Hình 3.6c) β | (Hình 3.6d) γ |
|---|---|---|---|
| 0 | (0, 1, X, a, Y) | (0, 1, Y, a, X) | (0, 1, X, a, X) |
| 1 | (1, 2, Y, b, X) | (1, 2, X, a, X) | (1, 2, X, a, Y) |
| 2 | (2, 0, X, a, X) | (2, 0, X, b, Y) | (2, 0, Y, b, X) |
| 3 | (2, 3, X, c, Z) | (2, 3, X, c, Z) | (2, 3, Y, b, Z) |
| 4 | (3, 1, Z, b, Y) | (3, 0, Z, b, Y) | (3, 0, Z, c, X) |
| 5 | (1, 4, Y, d, Z) | (0, 4, Y, d, Z) | (2, 4, Y, d, Z) |

* Định nghĩa 2 (thứ tự từ điển DFS)

Giả sử Z = [code (G, T) | T là một cây DFS của G], Z là một tập hợp có chứa tất cả các mã DFS của tất cả các đồ thị có nhãn liên tục. Giả sử có một thứ tự tuyến tính $\propto_L$ trong tập nhãn (L) thì sự kết hợp theo thứ tự từ điển $\propto_T$ và $\propto_L$ là một thứ tự tuyến tính $\propto_e$ trên tập $E_T = L \times L \times L$. Để biết thêm chi tiết xem [10].

+ Thứ tự từ điển DFS là một thứ tự tuyến tính được định nghĩa như sau: Nếu $a =$ $m\tilde{a}\ (G_a, T_a) = (a_0, a_1, ..., a_m)$ và $\beta =$ $m\tilde{a}\ (G_\beta, T_\beta) = (b_0, b_1, ..., b_n), a, \beta \in Z$, khi đó $a \le \beta$ khi và chỉ khi hai điều sau đây là đúng:

(i) $\exists t, 0 \le t \le \min(m, n), a_k = b_k \ với i \ k < t, a_t \propto_e b_t$

(ii) $a_k = b_k \ với i \ 0 \le k \le m, và \ n > m$

Đối với đồ thị trong hình 3.6(a), có 10 mã DFS tồn tại khác nhau. Ba trong số chúng được xây dựng dựa trên cây DFS trong hình 3.6(b) - (d) được liệt kê trong Bảng 3.1. Theo thứ tự từ điển DFS, $\gamma \propto a \propto \beta$

Định nghĩa 3 (Mã DFS tối thiểu)

Cho một đồ thị G, Z (G) = {code (G, T) | T là một cây DFS của G}. Dựa trên thứ tự từ điển DFS, một tối thiểu min (Z (G)) được gọi là mã DFS tối thiểu của G, cũng được gọi là một nhãn chính tắc của G.

Định lý 3.1: Cho hai đồ thị G và G ', G là đẳng cấu với G ' nếu và chỉ nếu Min (G) =Min(G'). Do đó, vấn đề của khai phá các đồ thị con liên thông thường xuyên tương đương với việc khai phá mã DFS tối thiểu tương ứng của chúng.

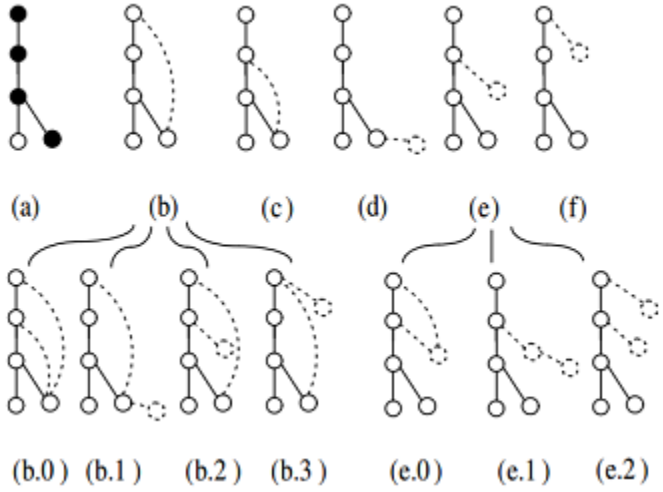
Cho một mã DFS α = ($a_0$, $a_1$, …, $a_m$) và mã DFS có giá trị bất kỳ β = ($a_0$, $a_1$, …, $a_m$, b) được gọi là con của α và α được gọi là cha của β.

Để xây dựng một giá trị mã DFS thì b phải có một cạnh mà chỉ phát triển từ các đỉnh ngoài cùng bên phải.

Trong hình 3.7 đồ thị thể hiện trong hình 3.7 (a) có nhiều tiềm năng con với tốc độ phát triển cạnh được thể hiện trong hình 3.7 (b) - (f) (giả sử các nút con tạo thành đường ngoài cùng bên phải). Trong số đó hình 3.7 (b), 3.7 (c), và 3.7 (d) tăng từ nút ngoài cùng

bên phải, trong khi 3.7 (e) và 3.7 (f) tăng từ nút con khác trên đường ngoài cùng bên phải 3.7 (b.0) - (b.3) là con của 3.7 (b) và 3.7 (e.0) - (e.2) là con của 3.7 (e). cạnh sau chỉ có thể phát triển từ nút ngoài cùng bên phải trong khi cạnh trước có thể phát triển từ các nút ngoài cùng bên phải.

Hạn chế này cũng tương tự như phần mở rộng  lớp, tương đương của TreeMinerV [16] và mở rộng bìa phải FREQT [14] trong phát hiện cây thường xuyên. Trình tự đếm những con được phát triển bằng thứ tự tự từ điển DFS tức là, nó phải ở trong thứ tự của 3.7 (b), 3.7 (c), 3.7 (d), 3.7 (e) và 3.7 (f).




Hình 3.7 Mã DFS/Phát triển đồ thị [10]

**Định nghĩa 4 (Cây mã DFS)**

Trong cây mã DFS, mỗi nút biểu diễn cho một mã DFS, mối quan hệ giữa nút cha và nút con tuân theo luật:

Với mối quan hệ cha-con được mô tả phần trên. Mối quan hệ anh - em là bao gồm thứ tự tự từ điển DFS. Đó là tìm kiếm theo thứ tự của cây mã DFS theo thứ tự tự từ điển DFS.

Cho một tập nhãn hợp L, một cây mã DFS chứa vô hạn số đồ thị. Chỉ xét các đồ thị con phổ biến trong một cơ sở dữ liệu vô hạn. Kích thước của một cây DFS là vô hạn. Hình 3.6 cho thấy một cây mã DFS, các nút mức $n_{th}$ chứa mã DFS của các đồ thị (n - 1) – Cạnh. Qua việc tìm kiếm theo chiều sâu của cây mã, tất cả các mã DFS tối thiểu của đồ thị con

phổ biến có thể được phát hiện. Đó là, tất cả các đồ thị con phổ biến có thể được phát hiện bằng cách này. Nếu trong hình 3.6 các nút Darken chứa đồ thị tương tự nhưng mã DFS khác nhau, sau đó là 'không phải là mã tối thiểu (đã chứng minh  trong [10]). Vì vậy, toàn bộ nhánh phụ của s có thể được cắt tỉa vì nó sẽ không chứa bất kỳ mã DFS tối thiểu.

**Thuật toán gSpan:**

Ý tưởng: Duyệt qua cây DFS code với tập các nhãn cho trước, tiến hành cắt tỉa dựa trên độ hỗ trợ và các mã cực tiểu.

Đầu vào của thuật toán: CSDL đồ thị giao tác D, độ hỗ trợ tối thiểu minsup.

Đầu ra: tập đồ thị con phổ biến S.

Thuật toán thực hiện như sau:

Bước 1: Tìm tất cả các đồ thị con phổ biến có một cạnh trong D (sử dụng mã DFS). Thêm chúng vào tập S.

Bước 2: Sắp xếp S theo thứ tự từ điển.

Bước 3: Gán N = S (S lúc này đã được sắp xếp lại theo từ điển).

Bước 4: Duyệt qua các mã DFS (đồ  thị con) trong N. Với mỗi đồ thị con n của N, tiến hành gọi hàm gSpan_extend(D, n, minsup, S).

Bước 5: Xóa n khỏi tất cả các đồ thị trong D

Thủ tuc gSpan_extend: sử dụng để phát triển các mã DFS và cắt tỉa ứng cử.

Đầu vào: tập đồ thị giao tác D, độ hỗ trợ tối thiểu minsup, mã DFS n.

Đầu ra: tập đồ thị con phổ biến S.

Thủ tục gSpan_extend thực hiện như sau:

1) Nếu n không phải mã cực tiểu thì kết thúc.

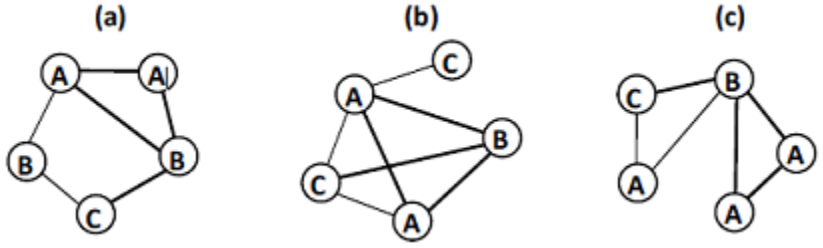
2) Nếu không:

- Thêm n vào S.

- Với mỗi cạnh đơn e được phát triển về phía bên phải của n:

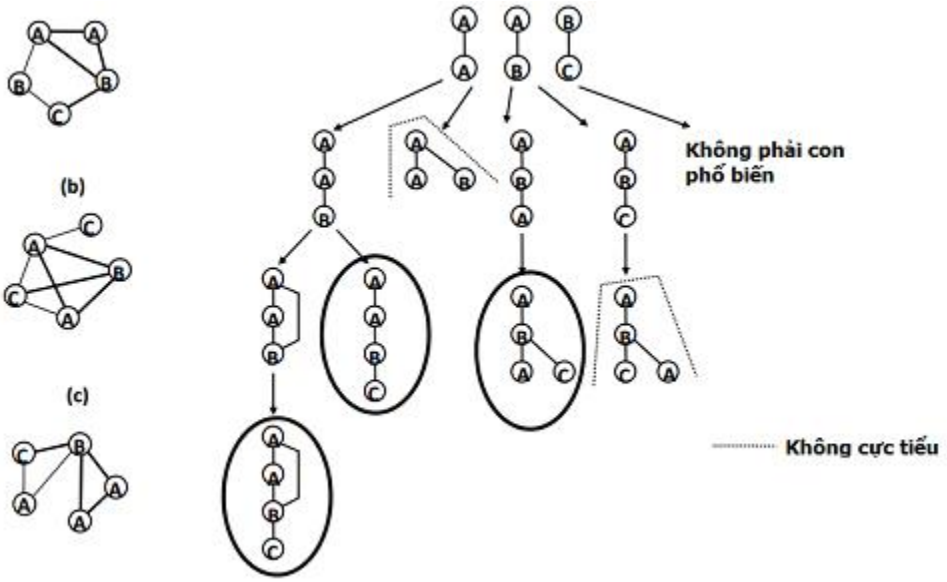
Nếu độ hỗ  trợ của e lớn hơn hoặc bằng ngưỡng minsup ta tiến hành gọi đệ quy thủ tục trên: gSpan_extend(D, e, minsup, S).

Ví dụ: Đầu vào là tập 3 đồ thị, độ hỗ trợ minsup = 3.




Hình 3.8 Tập đồ thị đầu vào

Áp dụng thuật toán gSpan ta có đồ thị con phổ biến như sau:

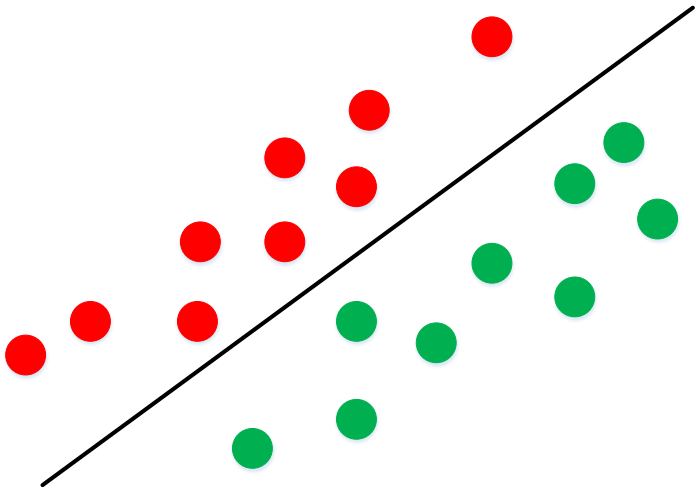



Hình 3.9  Minh họa khai phá đồ thị con phổ biến với gSpan

### III.2.6 Xây dựng pha phân loại văn bản với thuật toán SVM

SVM là một phương pháp phân lớp dựa trên lý thuyết học thống kê, được đề xuất bởi Vapnik (1995). Để đơn giản ta sẽ xét bài toán phân lớp nhị phân, sau đó sẽ mở rộng vấn đề ra cho bài toán phân nhiều lớp.

Xét một ví dụ của bài toán phân lớp như hình vẽ; ở đó ta phải tìm một đường thẳng sao cho bên trái nó toàn là các điểm đỏ, bên phải nó toàn là các điểm xanh. Bài toán mà dùng đường thẳng để phân chia này được gọi là phân lớp tuyến tính (linear classification).



Hình 3.10 Phân lớp tuyến tính

Hàm tuyến tính phân biệt hai lớp như sau:

$$y(x) = \mathrm{w}^T \phi(x) + b \qquad (1)$$

Trong đó:

$\mathrm{w} \in R^m$ là vector trọng số hay vector chuẩn của siêu phẳng phân cách, T là kí hiệu chuyển vị.

$b \in R$ là độ lệch

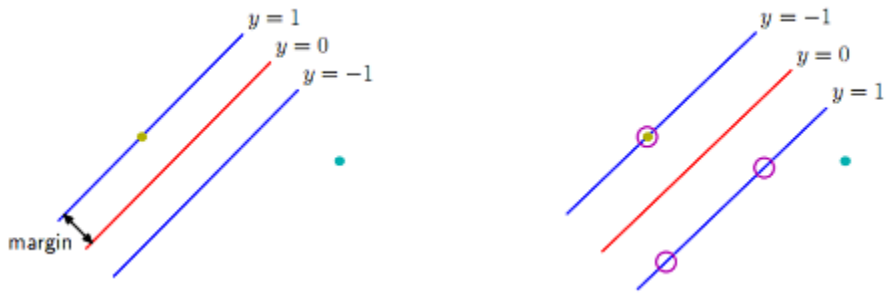
$\phi(x) \in R$ là véc tơ đặc trưng, $\phi$ làm hàm ánh xạ từ không gian đầu vào sang không gian đặc trưng.

Tập dữ liệu đầu vào gồm N mẫu input vector $\{x_1, x_2,...,x_N\}$, với các giá trị nhãn tương ứng là $\{t_1,\ldots,t_N\}$ trong đó $t_n \in \{-1,1\}$

Lưu ý cách dùng từ ở đây: điểm dữ liệu, mẫu… đều được hiểu là input vector $x_i$; nếu là không gian 2 chiều thì đường phân cách là đường thẳng, nhưng trong không gian đa chiều thì gọi đó là siêu phẳng.

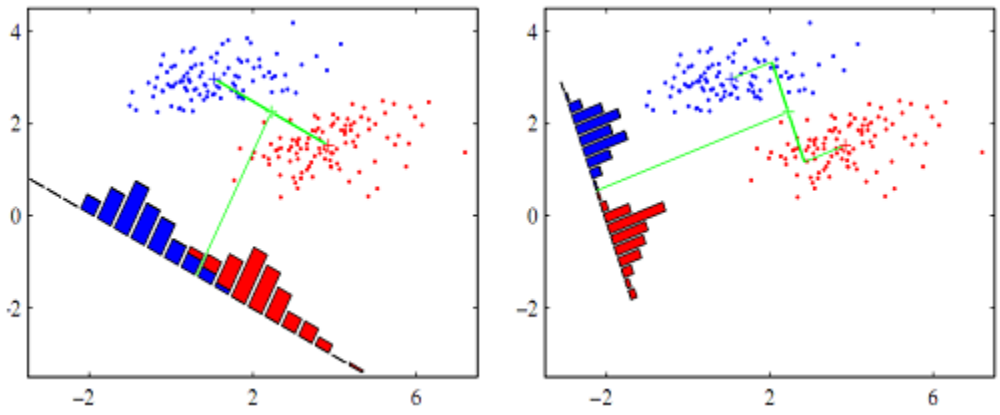
Giả sử tập dữ liệu của ta có thể phân tách tuyến tính hoàn toàn (các mẫu đều được phân đúng lớp) trong không gian đặc trưng (feature space), do đó sẽ tồn tại giá trị tham số w và b theo (1) thỏa $y(x_n) > 0$ cho những điểm có nhãn $t_n = +1$ và $y(x_n) < 0$ cho những điểm có $t_n = -1$, vì thế mà $t_n y(x_n) > 0$ cho mọi điểm dữ liệu huấn luyện.

SVM tiếp cận giải quyết vấn đề này thông qua khái niệm gọi là lề, đường biên… (margin). Lề được chọn là khoảng cách nhỏ nhất từ đường phân cách đến mọi điểm dữ liệu hay là khoảng cách từ đường phân cách đến những điểm gần nhất.




Hình 3.11 Minh họa lề trong thuật toán SVM

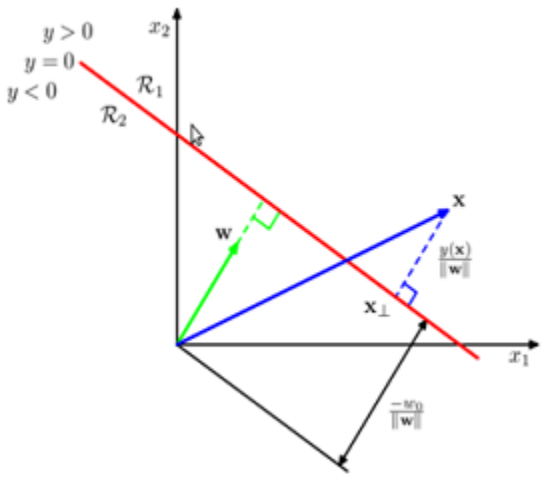
Trong SVM, đường phân lớp tốt nhất chính là đường có khoảng cách margin lớn nhất (tức là sẽ tồn tại rất nhiều đường phân cách xoay theo các phương khác nhau, và ta chọn ra đường phân cách mà có khoảng cách margin là lớn nhất).




Hình 3.12 Phân lớp SVM bằng cách sử dụng lề

Ta có công thức tính khoảng cách từ điểm dữ liệu đến mặt phân cách như sau:

$$\frac{|y(x)|}{\|w\|}$$




Hình 3.13 Minh họa khoảng cách từ điểm dữ liệu đến mặt phân cách

Do ta đang xét trong trường hợp các điểm dữ liệu đều được phân lớp đúng nên $t_n y(x_n) > 0$ cho mọi n. Vì thế khoảng cách từ điểm $x_n$ đến mặt phân cách được viết lại như sau:

$$\frac{t_n y(x_n)}{\|w\|} = \frac{t_n\left(w^T \phi(x_n) + b\right)}{\|w\|} \qquad (2)$$

Lề là khoảng cách vuông góc đến điểm dữ liệu gần nhất xn từ tập dữ liệu, và chúng ta muốn tìm giá trị tối ưu của w và b bằng cách cực đại khoảng cách này. Vấn đề cần giải quyết sẽ được viết lại dưới dạng công thức sau:

$$\arg\max_{w,b}\left\{\frac{1}{\|w\|}\min_n\left[t_n\left(w^T\phi(x_n)+b\right)\right]\right\} \qquad (3)$$

Chúng ta có thể đem nhân tử $\frac{1}{\|w\|}$ ra ngoài bởi vì w không phụ thuộc n. Giải quyết vấn đề này một cách trực tiếp sẽ rất phức tạp, do đó ta sẽ chuyển nó về một vấn đề tương đương dễ giải quyết hơn. Ta sẽ scale $w \to \vartheta w$ và $b \to \vartheta b$ cho mọi điểm dữ liệu, từ đây khoảng cách lề trở thành 1, việc biến đổi này không làm thay đổi bản chất vấn đề.

$$t_n\left(w^T\phi(x_n)+b\right) = 1 \qquad (4)$$

Từ bây giờ, các điểm dữ liệu sẽ thỏa ràng buộc:

$$t_n\left(\mathrm{w}^T\phi\left(x_n\right)+b\right)\geq 1,\ \mathrm{n}=1,...,\mathrm{N} \qquad (5)$$

Vấn đề tối ưu yêu cầu ta cực đại $\|\mathrm{w}\|^{-1}$ được chuyển thành cực tiểu $\|\mathrm{w}\|^2$, ta viết lại công thức:

$$\arg\min_{\mathrm{w},b}\frac{1}{2}\|\mathrm{w}\|^2 \qquad (6)$$

Việc nhân hệ số ½ sẽ giúp thuận lợi cho lấy đạo hàm về sau.

Lý thuyết Nhân tử Lagrange:

Vấn đề cực đại hàm f(x) thỏa điều kiện $g\left(x\right)\geq 0$ sẽ được viết lại dưới dạng tối ưu của hàm Lagrange như sau:

$$L\left(x,\lambda\right)\equiv f\left(x\right)+\lambda g\left(x\right)$$

Trong đó x và λ phải thỏa điều kiện Karush-Kuhn-Tucker (KKT) như sau:

$g\left(x\right)\geq 0$

$\lambda\geq 0$

$\lambda g\left(x\right)=0$

Nếu là cực tiểu hàm f(x) thì hàm Lagrange sẽ là

$$L\left(x,\lambda\right)\equiv f\left(x\right)-\lambda g\left(x\right)$$

Để giải quyết bài toán trên, ta viết lại theo hàm Lagrange như sau:

$$L(\mathrm{w},b,a)=\frac{1}{2}\|\mathrm{w}\|^2-\sum_{n=1}^{N}a_n\left\{t_n\left(\mathrm{w}^T\phi\left(x_n\right)+b\right)-1\right\} \qquad (7)$$

Trong đó $a=\left(a_1,a_2,...,a_N\right)^T$ là nhân tử Lagrange.

Lưu ý dấu (–) trong hàm Lagrange, bởi vì ta cực tiểu theo biến w và b, và là cực đại theo biến a.

Lấy đạo hàm L(w,b,a) theo w và b ta có:

$$\frac{\partial L}{\partial \mathrm{w}} = 0 \rightarrow \mathrm{w} = \sum_{n=1}^{N} a_n t_n \phi(x_n) \qquad (8)$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow 0 = \sum_{n=1}^{N} a_n t_n \qquad (9)$$

Loại bỏ w và b ra khỏi L(w,b,a) bằng cách thế (8), (9) vào. Điều này sẽ dẫn ta đến vấn đề tối ưu:

$$\tilde{L}(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(x_n, x_m) \qquad (10)$$

Thỏa các ràng buộc:

$$a_n \geq 0, n = 1, ..., N \qquad (11)$$

$$\sum_{n=1}^{N} a_n t_n = 0 \qquad (12)$$

Ở đây hàm nhân (kernel function) được định nghĩa là $k(x_n, x_m) = \phi(x_n)^T \phi(x_m)$

Vấn đề tạm thời gác lại ở đây, ta sẽ thảo luận kỹ thuật giải quyết (10) thỏa (11), (12) này sau.

Để phân lớp cho 1 điểm dữ liệu mới dùng mô hình đã huấn luyện, ta tính dấu của y(x) theo công thức (1), nhưng thế w trong (8) vào:

$$y(x) = \sum_{n=1}^{N} a_n t_n k(x, x_n) + b \qquad (13)$$

Thỏa các điều kiện KKT sau:

$$a_n \geq 0 \qquad (14)$$

$$t_n y(x_n) - 1 \geq 0 \qquad (15)$$

$$a_n \{t_n y(x_n) - 1\} = 0 \qquad (16)$$

Vì thế với mọi điểm dữ liệu, hoặc là $a_n = 0$ hoặc là $t_n y(x_n) = 1$. Những điểm dữ liệu mà có $a_n = 0$ sẽ không xuất hiện trong (13) và do đó mà không đóng góp trong việc dự đoán điểm dữ liệu mới.

Những điểm dữ liệu còn lại $(a_n \neq 0)$ được gọi là support vector, chúng thỏa $t_n y(x_n) = 1$, đó là những điểm nằm trên lề của siêu phẳng trong không gian đặc trưng.

Support vector chính là cái mà ta quan tâm trong quá trình huấn luyện của SVM. Việc phân lớp cho một điểm dữ liệu mới sẽ chỉ phụ thuộc vào các support vector.

Giả sử rằng ta đã giải quyết được vấn đề (10) và tìm được giá trị nhân tử a, bây giờ ta cần xác định tham số b dựa vào các support vector $x_n$ có $t_n y(x_n) = 1$. Thế (13) vào:

$$t_n\left(\sum_{m\in S} a_m t_m k(x_n, x_m) + b\right) = 1 \qquad (17)$$

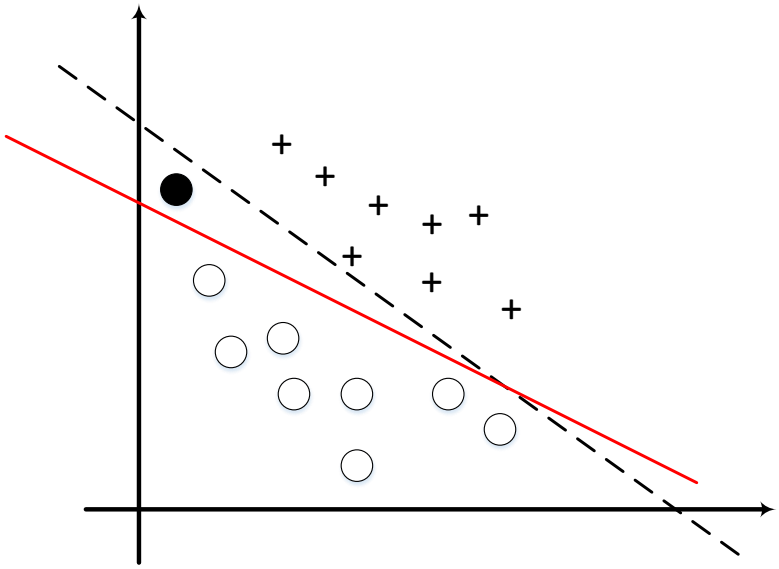
Trong đó S là tập các support vector. Mặc dù ta chỉ cần thế một điểm support vector $x_n$ vào là có thể tìm ra b, nhưng để đảm bảo tính ổn định của b ta sẽ tính b theo cách lấy giá trị trung bình dựa trên các support vector.

Đầu tiên ta nhân $t_n$ vào (17) (lưu ý $t_n^2 = 1$, và giá trị b sẽ là:

$$b = \frac{1}{N_S}\sum_{n\in S}\left(t_n - \sum_{m\in S} a_m t_m k(x_n, x_m)\right) \qquad (18)$$

Trong đó $N_s$ là tổng số support vector.

Ban đầu để dễ trình bày thuật toán ta đã giả sử là các điểm dữ liệu có thể phân tách hoàn toàn trong không gian đặc trưng $\phi(x)$. Nhưng việc phân tách hoàn toàn này có thể dẫn đến khả năng tổng quát hóa kém, vì thực tế một số mẫu trong quá trình thu thập dữ liệu có thể bị gán nhãn sai, nếu ta cố tình phân tách hoàn toàn sẽ làm cho mô hình dự đoán quá khớp.



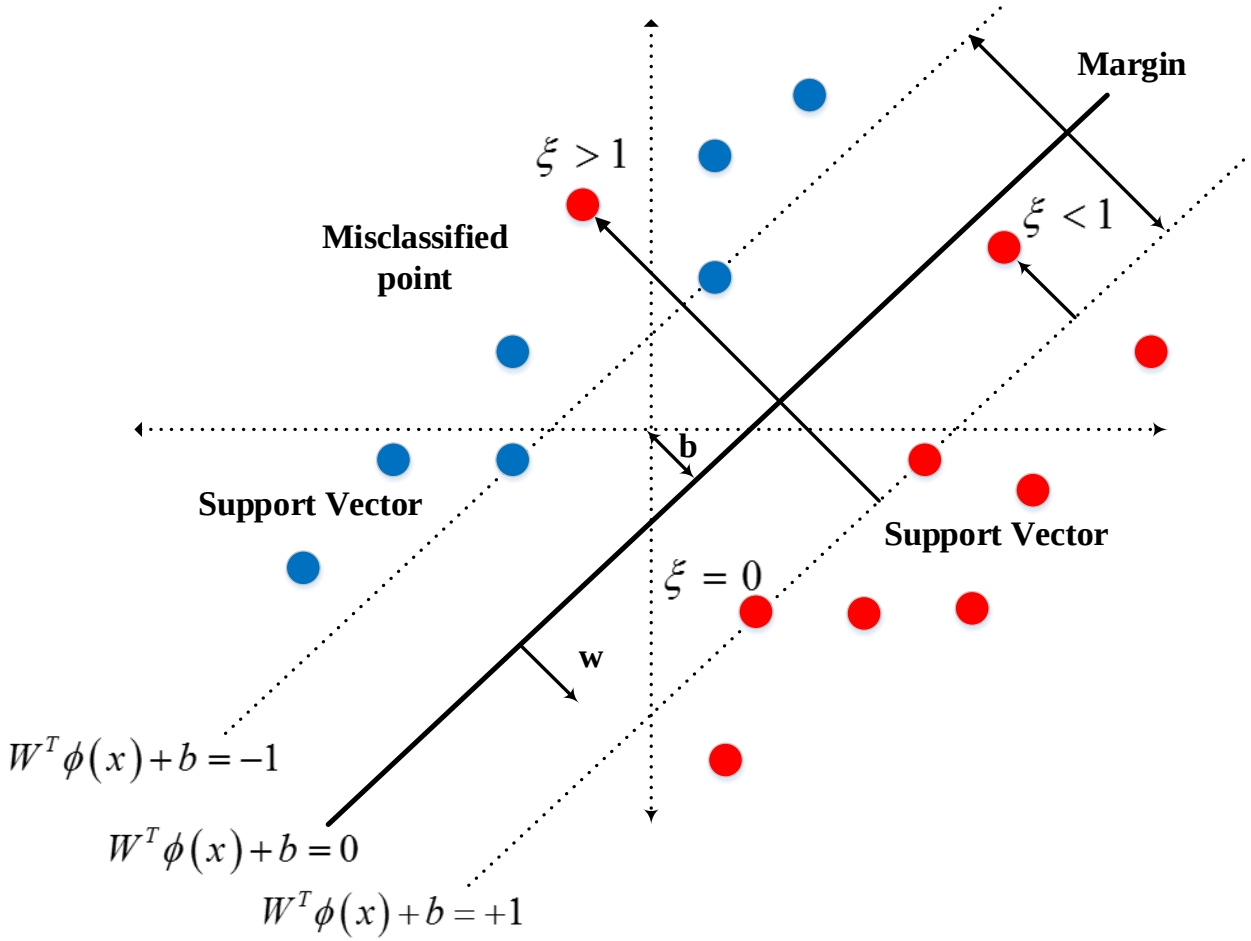
Hình 3.14 Mô hình dự đoán không khớp hoàn toàn

Để chống lại sự quá khớp, chúng ta chấp nhận cho một vài điểm bị phân lớp sai. Để làm điều này, ta dùng các biến slack variables $\xi_n \geq 0$, với n = 1, …, N cho mọi điểm dữ liệu.

$\xi_n = 0$ cho những điểm nằm trên lề hoặc phía trong của lề

$\xi_n = \|t_n - y(x_n)\|$ (19) cho những điểm còn lại.

Do đó những điểm nằm trên đường phân cách $y(x_n) = 0$ sẽ có $\xi_n = 1$. Còn những điểm phân lớp sai sẽ có $\xi_n > 1$




Hình 3.15 Phân lớp với một số điểm bị phân lớp sai

Công thức (5) sẽ viết lại như sau:

$$t_n y(x_n) \geq 1-\xi_n, n=1,...,N \qquad (20)$$

Mục tiêu của ta bây giờ là cực đại khoảng cách lề, nhưng đồng thời cũng đảm bảo tính mềm mỏng cho những điểm bị phân lớp sai. Ta viết lại vấn đề cần cực tiểu:

$$C\sum_{n=1}^{N}\xi_n + \frac{1}{2}\|w\|^2 \qquad (21)$$

Trong đó C > 0 đóng vai trò quyết định đặt tầm quan trọng vào biến $\xi_n$ hay là lề.

Bây giờ chúng ta cần cực tiểu (21) thỏa ràng buộc (20) và $\xi_n \geq 0$ . Theo Lagrange ta viết lại:

$$L(w,b,a) = \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}a_n\{t_n y(x_n)-1+\xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n \qquad (22)$$

Trong đó $\{a_n \geq 0\}$ và $\{\mu_n \geq 0\}$ là các nhân tử Lagrange.

Các điều kiện KKT cần thỏa là:

$$a_n \geq 0 \quad (23)$$
$$\mathrm{t}_n\, y(x_n) - 1 + \xi_n \geq 0 \quad (24)$$
$$\mathrm{a}_n\left(\mathrm{t}_n\, y(x_n) - 1 + \xi_n\right) = 0 \quad (25)$$
$$\mu_n \geq 0 \quad (26)$$
$$\xi_n \geq 0 \quad (27)$$
$$\mu_n \xi_n = 0 \quad (28)$$

Với n = 1,…,N

Lấy đạo hàm (22) theo w, b và $\{\xi_n\}$:

$$\frac{\partial L}{\partial \mathrm{w}} = 0 \rightarrow \mathrm{w} = \sum_{n=1}^{N} a_n t_n \phi(x_n) \quad (29)$$
$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{n=1}^{N} a_n t_n = 0 \quad (30)$$
$$\frac{\partial L}{\partial \xi_n} = 0 \rightarrow a_n = C - \mu_n \quad (31)$$

Thế (29), (30), (31) vào (22) ta được:

$$\tilde{L}(a) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{n=1}^{N} a_m t_n t_m k(x_n, x_m) \quad (32)$$

Từ (23), (26) và (31) ta có: $a_n \leq C$

Vấn đề cần tối ưu giống hệt với trường hợp phân tách hoàn toàn, chỉ có điều kiện ràng buộc khác biệt như sau:

$$0 \leq a_n \leq C \quad (33)$$
$$\sum_{n=1}^{N} a_n t_n = 0 \quad (34)$$

Thế (29) vào (1), ta sẽ thấy để dự đoán cho một điểm dữ liệu mới tương tự như (13)

Như trước đó, tập các điểm có $a_n = 0$ không có đóng góp gì cho việc dự đoán điểm dữ liệu mới.

Những điểm còn lại tạo thành các support vector. Những điểm có $a_n > 0$ và theo (25) thỏa:

$$t_n y(x_n) = 1 - \xi_n \qquad (35)$$

Nếu $a_n < C$ theo (31) có $\mu_n > 0$, từ (28) suy ra $\xi_n = 0$ và đó là những điểm nằm trên lề.

Những điểm có $a_n = C$ có thể là những điểm phân lớp đúng nằm giữa lề và đường phân cách nếu $\xi_n \le 1$ hoặc có thể là phân lớp sai nếu $\xi_n > 1$

Để xác định tham số b trong (1) ta sẽ dùng những support vector mà $0 < a_n < C$ có $\xi_n = 0$ vì thế : $t_n y(x_n) = 1$:

$$t_n\left(\sum_{m\in S} a_m t_m k(x_n, x_m) + b\right) = 1 \qquad (36)$$

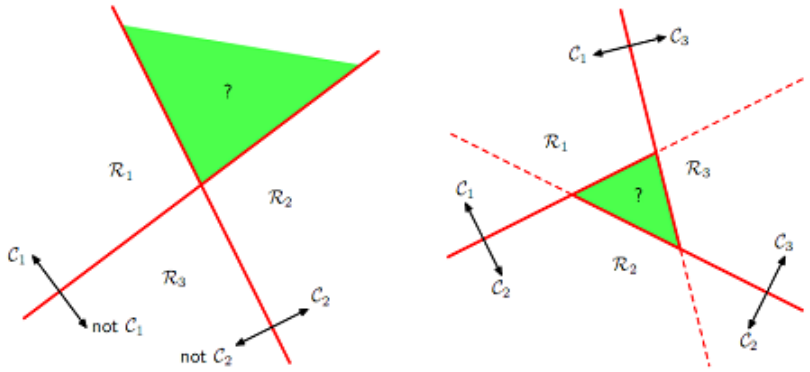
Lần nữa, để đảm bảo tính ổn định của b ta tính theo trung bình:

$$b = \frac{1}{N_M}\sum_{n\in M}\left(t_n - \sum_{m\in S} a_m t_m k(x_n, x_m)\right) \qquad (37)$$

Trong đó M là tập các điểm có $0 < a_n < C$

Để giải quyết (10) và (32) ta dùng thuật toán Sequential Minimal Optimization (SMO) do Platt đưa ra vào 1999.

MultiClass SVMs:




Hình 3.16 Phân lớp đa lớp với SVM

Bây giờ xét đến trường hợp phân nhiều lớp K > 2. Chúng ta có thể xây dựng việc phân K-lớp dựa trên việc kết hợp một số đường phân 2 lớp. Một hướng tiếp cận khác do Wu (2004) đề xuất phương pháp ước lượng xác suất cho việc phân m lớp.

# CHƯƠNG IV: THỰC NGHIỆM

## IV.1 Thực nghiệm giảm số lượng đồ thị con phổ biến thông qua TF - IDF

Thông qua việc xác định trọng số từ TF-IDF cho từng văn bản trong mỗi chủ đề của tập dữ liệu huấn luyện giúp ta có thể loại bỏ đi một số lượng lớn các từ dư thừa không đại diện cho văn bản trước khi chuyển đổi thành đồ thị, mỗi chủ đề sẽ được xác định một ngưỡng (threshold) TF-IDF nhất định khác nhau nhằm để loại bỏ bớt các từ dư thừa nhưng cũng không làm mất đi các từ, cụm từ đặc trưng quan trọng – làm đại diện cho chủ đề đó. Ngoài ra ngưỡng TF-IDF xác định cho từng chủ đề cũng phụ thuộc rất nhiều vào số lượng văn bản huấn luyện và độ lớn của từng văn bản ở mỗi chủ đề.

Một phương pháp xác định ngưỡng TF-IDF được áp dụng trong luận văn là chúng tôi sẽ lấy trung bình cộng trọng số TF-IDF của tất cả các từ trong văn bản đó để làm ngưỡng chung cho một văn bản.

Ví dụ: ta có văn bản $d = \{t_1, t_2, t_3 \dots t_n\}$ → ta sẽ xác định ngưỡng threshold cho văn bản này bằng cách:

$$threshold_d = \frac{\sum_n^1 w_{TF-IDF}(t)}{|d|}$$

Thực nghiệm cho thấy thì việc xác định ngưỡng TF-IDF để loại bỏ các từ không quan trọng trong văn bản huấn luyện làm giảm thiểu kích thước của đồ thị đi rất nhiều cũng như số lượng tập đồ thị phổ biến (frequent graph) được rút trích. Chúng tôi tiến hành thực nghiệm với số lượng tập văn bản đầu vào khác nhau (tăng dần về số lượng) lần lượt ở các ngưỡng minSup 20, 30 và 40 như bảng 4.1 dưới đây:

Bảng 4.1: So sánh số lượng đồ thị con phổ biến

| **Chủ đề** | **Số văn bản** | **Số lượng đồ thị phổ biến (FreqGraph)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | **minSup = 20%** | | | **minSup = 30%** | | | **minSup = 40%** | | |
| | | Không TF-IDF | Có TF-IDF | % | Không TF-IDF | Có TF-IDF | % | Không TF-IDF | Có TF-IDF | % |
| **Chính trị** | 200 | 281 | 68 | 24.20 | 214 | 47 | 21.96 | 172 | 37 | 21.51 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **& xã hội** | 300 | 402 | 94 | 23.38 | 294 | 75 | 25.51 | 235 | 59 | 25.11 |
| | 400 | 613 | 140 | 22.84 | 476 | 96 | 20.17 | 380 | 68 | 17.89 |
| | Trung bình (%) | 23.47 | | | 22.55 | | | 21.50 | | |
| **Sức khỏe** | 200 | 187 | 58 | 31.02 | 134 | 46 | 34.33 | 107 | 36 | 33.64 |
| | 300 | 354 | 97 | 27.40 | 281 | 77 | 27.40 | 225 | 58 | 25.78 |
| | 400 | 567 | 113 | 19.93 | 432 | 102 | 23.61 | 346 | 97 | 28.03 |
| | Trung bình (%) | 26.12 | | | 28.45 | | | 29.15 | | |
| **Thể thao** | 200 | 234 | 79 | 33.76 | 192 | 63 | 32.81 | 157 | 48 | 30.57 |
| | 300 | 456 | 85 | 18.64 | 378 | 68 | 17.99 | 297 | 52 | 17.51 |
| | 400 | 546 | 156 | 28.57 | 436 | 98 | 22.48 | 348 | 72 | 20.69 |
| | Trung bình (%) | 26.99 | | | 24.43 | | | 22.92 | | |

### IV.2 Thực nghiệm mức độ chính xác của phân lớp

Để đánh giá mức độ chính xác của mô hình được huấn luyện chúng tôi tiến hành chạy thực nghiệm trên tập dữ liệu như sau:

Dữ liệu đầu vào của quá trình huấn luyện được cho trong bảng 4.2

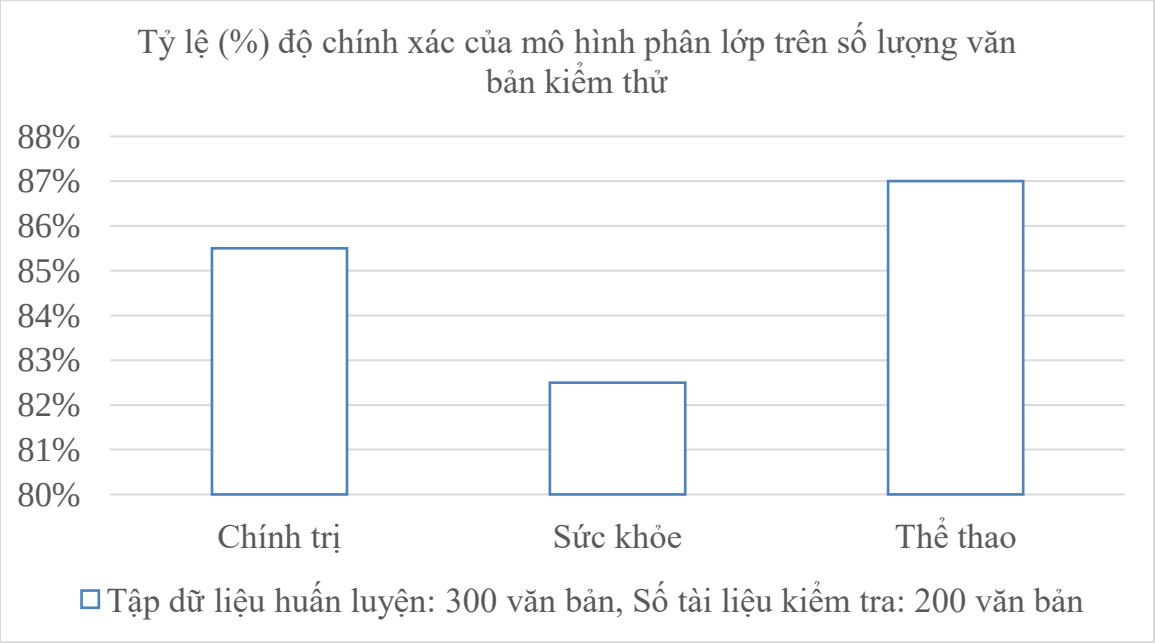
Bảng 4.2: Dữ liệu đầu vào của quá trình huấn luyện phân lớp (300 văn bản)

| **Chủ đề** | **Số lượng văn bản đầu vào được chọn lọc** | **Số đồ thị con phổ biến (FreqGraph) minSup=25%** |
|---|---|---|
| **Chính trị & xã hội** | 300 | 88 |
| **Sức khỏe** | 300 | 71 |
| **Thể thao** | 300 | 66 |

Sau khi hoàn tất quá trình huấn luyện chúng tôi tiến hành thu thập một số lượng lớn bài báo thuộc ba chủ đề trên từ các nguồn tin tức điện tử http://vnexpress.net/, http://dantri.com.vn/, http://tuoitre.vn/, quá trình kiểm nghiệm được tiến hành như sau: lần lượt một số lượng nhất định các văn bản của mỗi chủ đề: chúng tôi lấy 200 văn bản cho mỗi chủ đề sẽ được đưa vào để thử nghiệm phân lớp – sau đó chúng tôi đếm đếm số lượng bài báo được phân lớp chính xác vào chủ đề đó để làm kết quả so sánh và đưa ra bảng kết quả trong bảng 4.3 dưới đây:

Bảng 4.3: Kết quả phân lớp với dữ liệu huấn luyện 300 văn bản

| **Chủ đề** | **Số lượng tài liệu** | | **Tỷ lệ chính xác (%)** |
|---|---|---|---|
| | **Đầu vào** | **Phân loại chính xác** | |
| **Chính trị xã hội** | 200 | 171 | 86% |
| **Sức khỏe** | 200 | 165 | 83% |
| **Thể thao** | 200 | 174 | 87% |




Hình 4.1: Kết quả phân lớp với dữ liệu huấn luyện 300 văn bản

Để đánh giá mức độ chính xác hơn nữa mô hình phân loại văn bản của chúng tôi. Trong bước thực nghiệm tiếp theo chúng tôi tăng dữ liệu đầu vào của quá trình huấn luyện

lên trong mỗi chủ đề. Với mỗi chủ đè chúng tôi đã tăng số lượng văn bản huấn luyện lên thành 500 văn bản cho mỗi chủ đề.

Dữ liệu đầu vào của quá trình huấn luyện được cho trong bảng 4.4

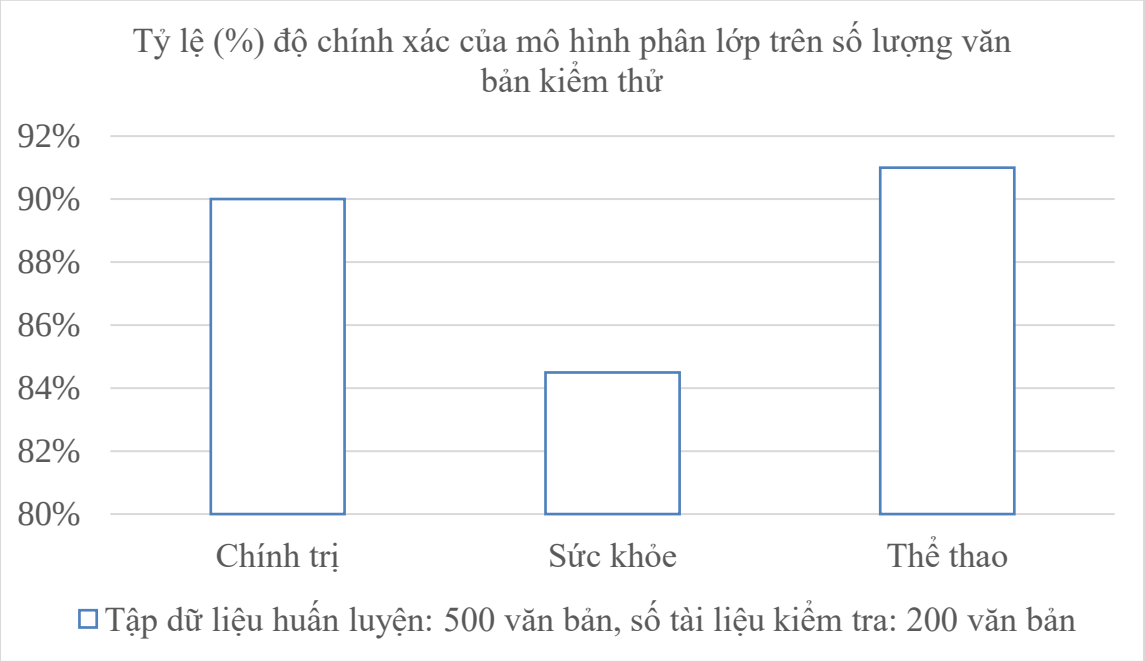
Bảng 4.4: Dữ liệu đầu vào của quá trình huấn luyện phân lớp (500 văn bản)

| **Chủ đề** | **Số lượng văn bản đầu vào được chọn lọc** | **Số đồ thị con phổ biến (FreqGraph) minSup=25%** |
|---|---|---|
| **Chính trị & xã hội** | 500 | 146 |
| **Sức khỏe** | 500 | 112 |
| **Thể thao** | 500 | 98 |

Sau khi hoàn tất quá trình huấn luyện chúng tôi tiến hành lấy 200 văn bản cho mỗi chủ đề đã có ở trên sẽ được đưa vào để thử nghiệm phân lớp – sau đó chúng tôi đếm đếm số lượng bài báo được phân lớp chính xác vào chủ đề đó để làm kết quả so sánh và đưa ra kết quả như trong bảng 4.5

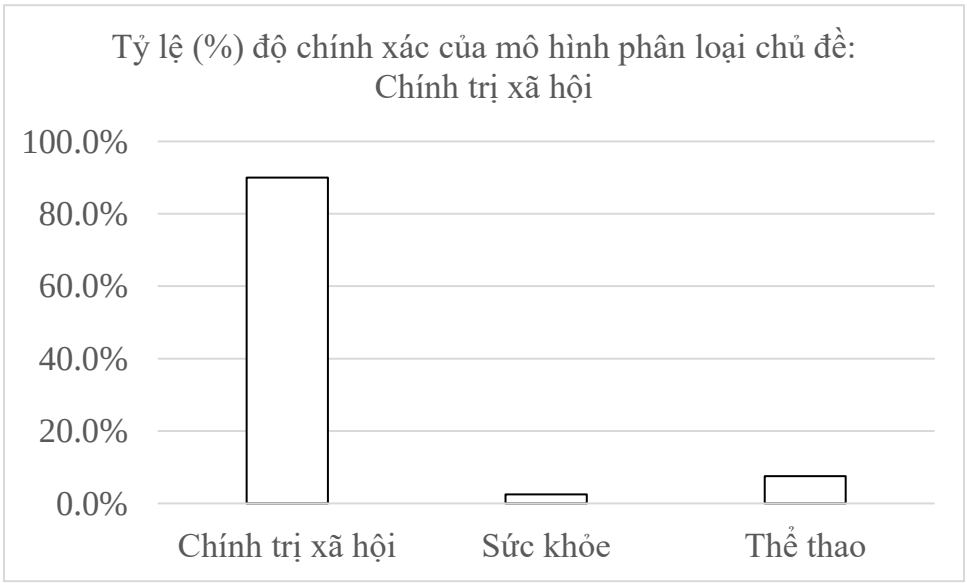
Bảng 4.5: Kết quả phân lớp với dữ liệu huấn luyện 500 văn bản

| **Chủ đề** | **Số lượng tài liệu** | | **Tỷ lệ chính xác (%)** |
|---|---|---|---|
| | **Đầu vào** | **Phân loại chính xác** | |
| **Chính trị xã hội** | 200 | 180 | 90% |
| **Sức khỏe** | 200 | 169 | 84,5% |
| **Thể thao** | 200 | 182 | 91% |

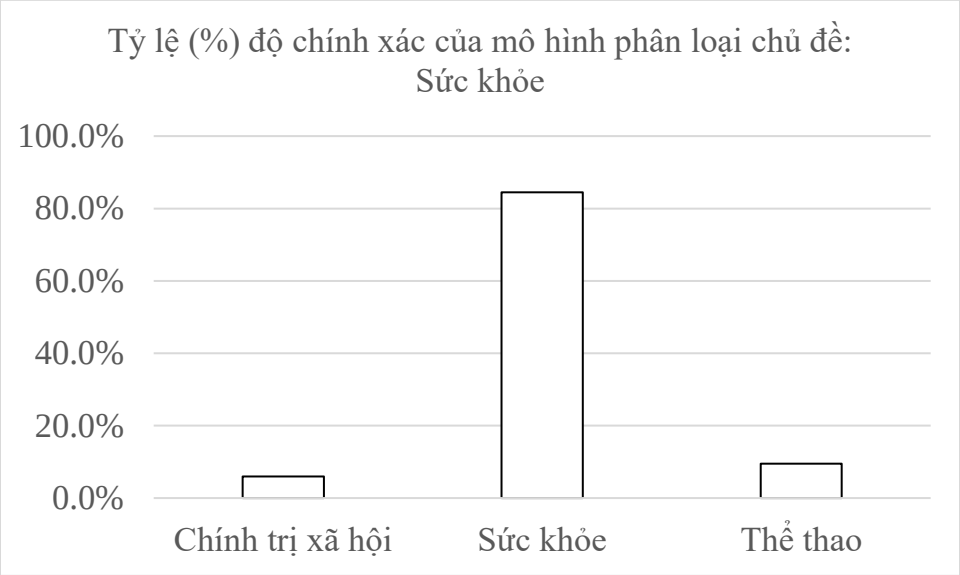



Hình 4.2: Kết quả phân lớp với dữ liệu huấn luyện 500 văn bản

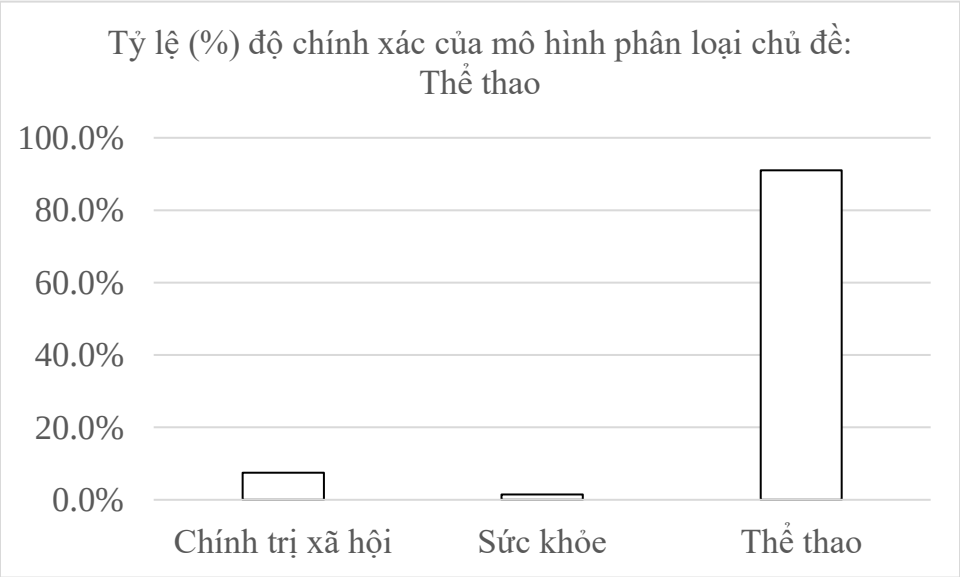
Tỷ lệ (%) độ chính xác của mô hình phân loại cho từng chủ đề được thể hiện ở hình 4.3 – 4.5




Hình 4.3: Kết quả phân lớp chủ đề Chính trị xã hội




Hình 4.4: Kết quả phân lớp chủ đề Sức khỏe




Hình 4.5: Kết quả phân lớp chủ đề Thể thao

Sau khi tiến hành phân lớp lại khi tăng bộ dữ liệu huấn luyện đầu vào. Chúng tôi nhận thấy rằng khi dữ liệu huấn luyện đầu vào càng tăng thì độ chính xác của phân lớp càng tăng. Điều này càng khẳng định tính đúng đắn của qúa trình học có giám sát.

Thực nghiệm cuối cùng chúng tôi làm là chúng tôi tiến hành gộp các văn bản của 3 chủ đề phân lớp chung lại và tiến hành phân lớp. Sau đó chúng tôi tiến hành đếm số lượng bài báo được phân lớp chính xác vào từng chủ đề để làm kết quả so sánh và đưa ra kết quả như trong bảng 4.6

Bảng 4.6: Kết quả phân lớp khi gộp các văn bản

| Đầu vào | Số lượng tài liệu, kết quả phân loại | | | Tỷ lệ chính xác (%) |
|---|---|---|---|---|
| | Chủ đề | Kết quả phân loại | Phân loại chính xác | |
| 600 | **Chính trị xã hội** | 207 | 180 | 90% |
| | **Sức khỏe** | 177 | 169 | 84,5% |
| | **Thể thao** | 216 | 182 | 91% |

Kết quả phân lớp của chúng tôi đạt độ chính xác như phân lớp riêng rẽ từng chủ đề.

## IV.3 Kết luận

Sau khi đã hoàn thành quá trình tổng kết toàn bộ các phần đã xây dựng, đặt ra các phương pháp kiểm thử và chạy thực nghiệm để đánh giá kết quả đạt được ở từng thành phần của hệ thống, chúng tôi đã xây dựng thử nghiệm hệ thống phân loại văn bản tiếng Việt dựa trên mô hình biểu diễn văn bản bằng đồ thị. Mô hình đồ thị cho phép lưu trữ các thông tin cấu trúc quan trọng của văn bản như vị trí, thứ tự của từ. Kết quả thử nghiệm cho thấy mô hình đồ thị cho kết quả phân lớp tương đối cao hầu như tất cả đều vượt ngưỡng trên 85%. Trong các chủ đề phân lớp chủ đề chính trị & xã hội và chủ đề thể thao đạt ngưỡng cao nhất với độ chính xác gần 90%. Chúng tôi rất hài lòng vì các kết quả đã đạt được tuy các kết quả chưa đạt được như dự kiến ban đầu. Để đánh giá chính xác hơn nữa, chúng tôi dự kiến sẽ thu thập và xây dựng bộ dữ liệu huấn luyện và thực nghiệm lớn hơn nữa. Chúng tôi hy vọng và tin rằng sẽ đạt được các kết quả thực nghiệm cao hơn cho các cải tiến và nâng cấp trong tương lai.

# CHƯƠNG V: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## V.1 Kết luận

Luận văn tập chung nghiên cứu các vấn đề liên quan đến phân loại văn bản dựa trên mô hình đồ thị, gồm có:

- Các khái niệm cơ bản của lý thuyết đồ thị: đồ thị được gán nhãn, đồ thị con, tính đẳng cấu của đồ thị, …

- Bài toán khai phá đồ thị con phổ biến, các thuật toán phổ biến sử dụng cho bài toán khai phá đồ thị con phổ biến như gSpan, Subdue, Gaston, FSG, ...

- Nghiên cứu thuật toán khai thác đồ thị con phổ biến gSpan và mô hình SVM. Áp dụng cho bài toán phân loại văn bản.

- Với các kiến thức nghiên cứu được, chúng tôi đã tiến hành xây dựng chương trình phân loại văn bản dựa trên mô hình đồ thị. Những công việc đó là: tách từ tiếng Việt, xác định trọng số từ TF-IDF, mô hình hóa văn bản thành đồ thị, cài đặt thuật toán gSpan để khai phá các đồ thị con phổ biến, tiến hành xây dựng các véc tơ đặc trưng cho các đồ thị và cài đặt bộ phân lớp các véc tơ đặc trưng dựa trên thuật toán SVM.

- Tiến hành kiểm chứng, thực nghiệm và đánh giá độ chính xác của mô hình đã xây dựng với bộ dữ liệu thực nghiệm là các bài báo tiếng việt được lấy từ các nguồn tin tức điện tử http://vnexpress.net/, http://dantri.com.vn/, http://tuoitre.vn/

- Kết quả thực nghiệm cho thấy mô hình phân loại này đạt độ chính xác tương đối cao trên 85%. Đặc biệt với chủ đề Chính trị xã hội và chủ đề Thể thao đạt độ chính xác hơn 90%, khẳng định hướng nghiên cứu là đúng đắn.

## V.2 Hướng phát triển

Trong thời gian tới, chúng tôi sẽ tiến hành nghiên cứu, phát triển thêm để khắc phục một số hạn chế của luận văn như:

- Xây dựng và đánh giá mô hình với nhiều bộ dữ liệu thực tế hơn nữa.

- So sánh mô hình xây dựng với các mô hình khác như: phân loại văn bản dựa vào mô hình không gian vectơ truyền thống, sử dụng cây quyết định, …

- Tăng tốc độ khai phá đồ thị con phổ biến bằng cách cài đặt song song hóa thuật toán gSpan.

- Đưa mô hình đồ thị có hướng vào trong bài toán xây dựng đồ thị văn bản.

- Áp dụng mô hình đề xuất cho các bài toán thực tế khác như: khai phá các mạng xã hội, phân loại hình ảnh dựa trên các điểm quan tâm, phân loại email, ...

# TÀI LIỆU THAM KHẢO


**Tiếng Việt**

[1] Nguyễn Hoàng Tú Anh, Nguyễn Trần Kim Chi, Nguyễn Hồng Phi (2009). "Mô hình biểu diễn văn bản thành đồ thị", Tạp chí Phát triển Khoa học và Công nghệ, ĐHQG-HCM, Tập 12, số 07/2009, pp.5-14.

[2] Nguyễn Hoàng Tú Anh, Hoàng Kiếm (2009), "Áp dụng kỹ thuật khai thác đồ thị vào bài toán phân loại văn bản", Tạp chí Tin học và Điều khiển học, T.25, S.1(2009), pp.43-52.

[3] Nguyễn Hoàng Tú Anh (2011), *Tiếp cận đồ thị biểu diễn, khai thác văn bản và ứng dụng*, Luận án tiến sĩ, Ngành Toán học, Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia TP.HCM.

[4] Nguyễn Linh Giang, Nguyễn Mạnh Hiển (2006). "Phân loại văn bản tiếng Việt với bộ phân loại vectơ hỗ trợ SVM", Đặc san Tạp chí BCVT & CNTT, số 7/2006.

[5] Nguyễn Hoàng Tú Anh, Hoàng Kiếm (2008), "Tóm tắt văn bản tiếng Việt dựa trên mô hình đồ thị", Đặc san Các công trình nghiên cứu khoa học, nghiên cứu triển khai Công nghệ thông tin và Truyền thông, Tạp chí Công nghệ thông tin và Truyền thông, kỳ 3, số 20, tháng 10 năm 2008, pp. 91-100.

**Tiếng Anh**

[6] Francois Rousseau; Emmanouil Kiagias; Michalis Vazirgiannis. Text Categorization as a Graph Classification Problem [online], viewed 26/03/2016, from:< http://www.aclweb.org/anthology/P/P15/P15-1164.pdf>.

[7] Roi Blanco and Christina Lioma. 2012. Graph-based term weighting for information retrieval. Informa-tion Retrieval, 15(1):54–92.

[8] Francois Rousseau and Michalis Vazirgiannis. 2015. Main Core Retention on Graph-of-words for Single-Document Keyword Extraction. In Proceedings of the 37th European Conference on Information Re-trieval, ECIR '15, pages 382–393.

[9] François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and TW-IDF:

New Approach to Ad Hoc IR. In Proceedings of the 22nd ACM inter-national conference on Information and knowledge management, CIKM ’13, pages 59–68.

[10] X. Y an and J. Han. gspan: Graph-based substructure pattern mining. Technical Report UIUCDCS-R-2002-2296, Department of Computer Science, University of Illinois at UrbanaChampaign, 2002.

[11] Thorsten Joachims. 1998. Text categorization with Support Vector Machines: Learning with many rel-evant features. In Proceedings of the 10th European Conference onMachine Learning, ECML ’98, pages 137–142.

[12] Thorsten Joachims. 2006. Training Linear SVMs in Linear Time. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery and Data mining, KDD ’06, pages 217–226.

[13] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraph in the presence of isomorphism. In Proceedings of the 2003 International Conference on Data Mining (ICDM'03), pages 549-552, 2003.

[14] T . Asai, K. Abe, S. Kawasoe, H. Arimura, H. Satamoto, and S. Arikawa. Efcient substructure discovery from large semistructured data. In SIAM SDM'02, April 2002.

[15] T . H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. Introduction to Algorithms. MIT Press, 2001, Second Edition.

[16] M. J. Zaki. Efficiently mining frequent trees in a forest. In KDD'02, July 2002.

[17] H. Hu, X. Yan, Y. Huang, J. Han, and X.J. Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. Bioinformatics, 21(1):213-221, 2005.

[18] X. Yan, F. Zhu, J. Han, and P.S. Yu. Searching substructures with superimposed distance. In Proceedings of the 22nd International Conference on Data Engineering (ICDE'06), page 88, 2006.

[19] Y. Chi, Y. Yang, Y. Xia, and R.R. Muntz. Canonical forms for labelled trees and their applications in frequent subtree mining. Journal of Knowledge and Information Systems, 8(2):203-234, 2005.

[20] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In VLDB'94, pages 487– 499, Sept. 1994.

[21] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In ICDM'01, pages 313– 320, Nov . 2001.

[22] Luc Dehaspe, Hannu Toivonen, and Ross D. King. Finding frequent substructures in chemical compounds. In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98), pages 30-36, New York, August 1998. AAI Press.

[23] A. Inokuchi, T . Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In PKDD'00, pages 13– 23, 2000.

[24] H. Hu, X. Yan, Y. Huang, J. Han, and X.J. Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. Bioinformatics, 21(1):213-221, 2005.

[25] R.C. Read and D.G. Corneil. The graph isomorph disease. Journal of Graph Theory, 1:339-363, 1977.

[29] Yang, Y. and Petersen, J. (1997). A comparative study on feature selection in text categorization. In International Conference on Machine learning(ICML)

[30] V.Vapnik, The Nature of Statistical Learning Theory. Springger, NewYork, 1995.

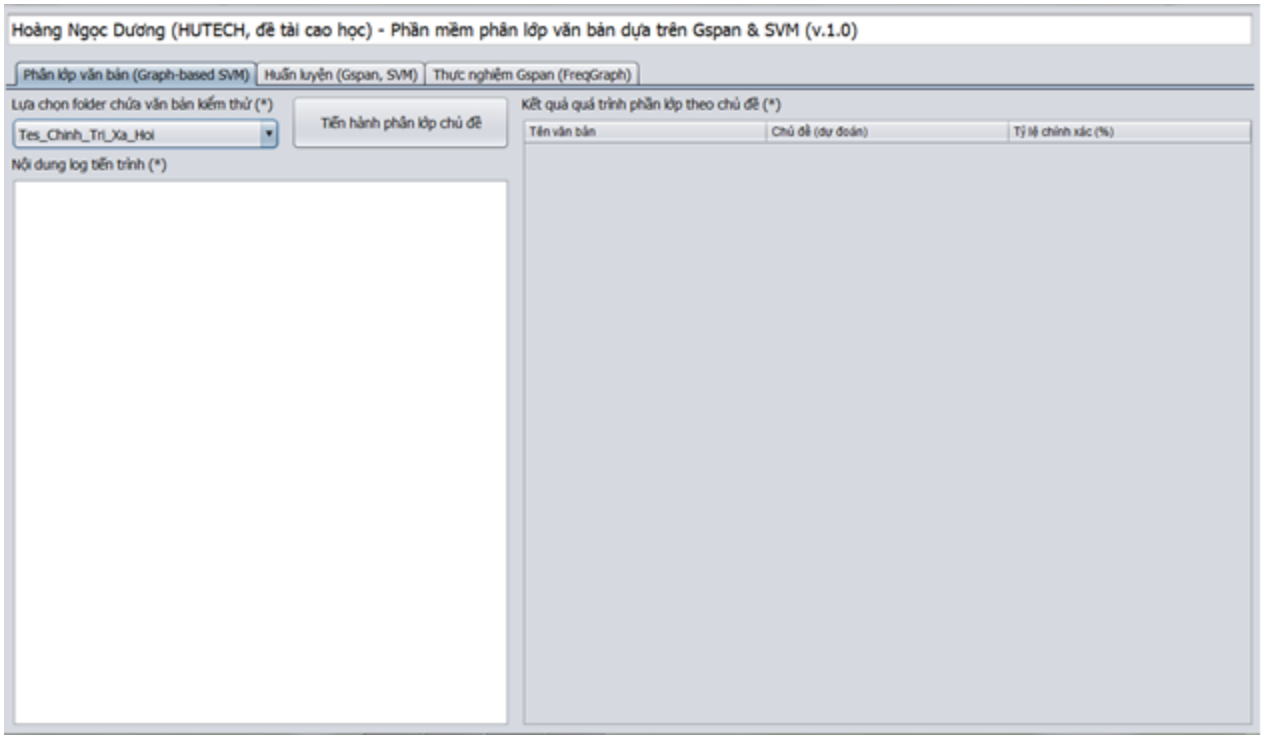
[31] Thorsten Joachims. Text Categorization with support Vector Machines: Learning with Many Relevant Features. In European Conference on Machine Learning (ECML), 1998

[32] J.Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. In Technical Report MST-TR-98-14. Microsoft Research, 1998

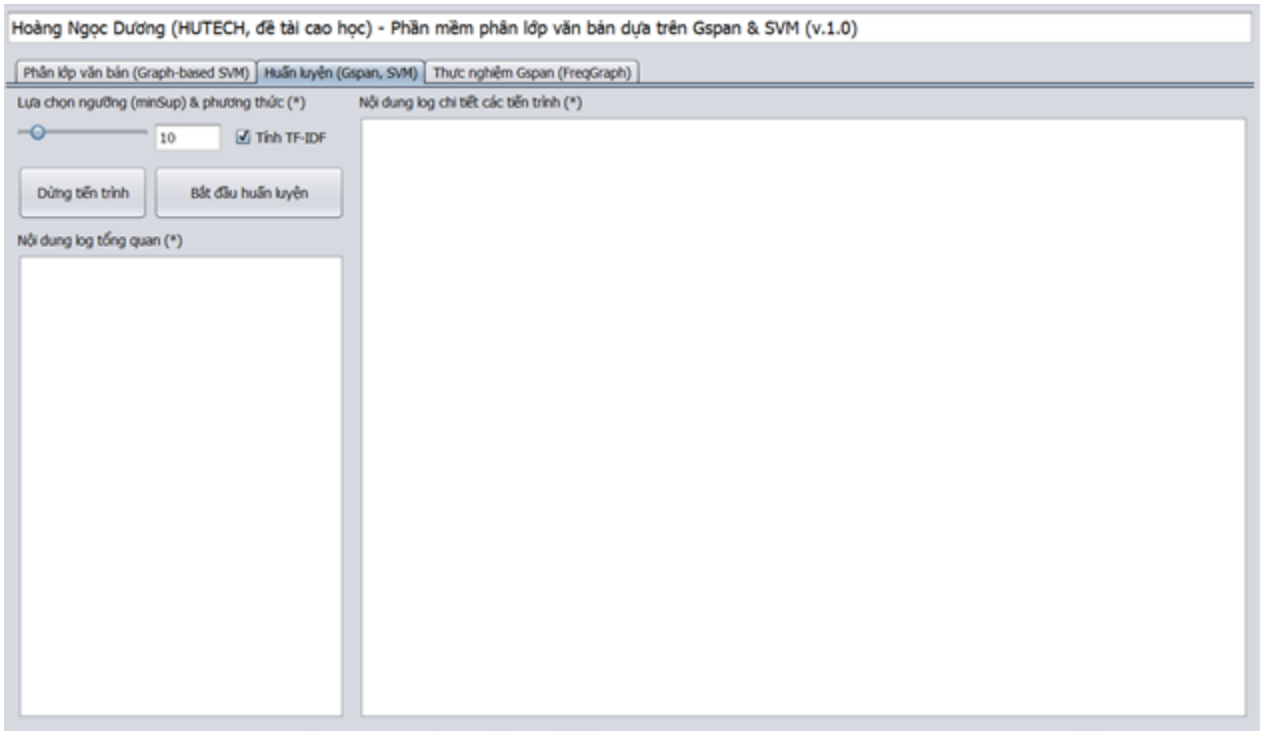
# PHỤ LỤC

## Một số hình ảnh về chương trình phân loại văn bản dựa trên mô hình đồ thị

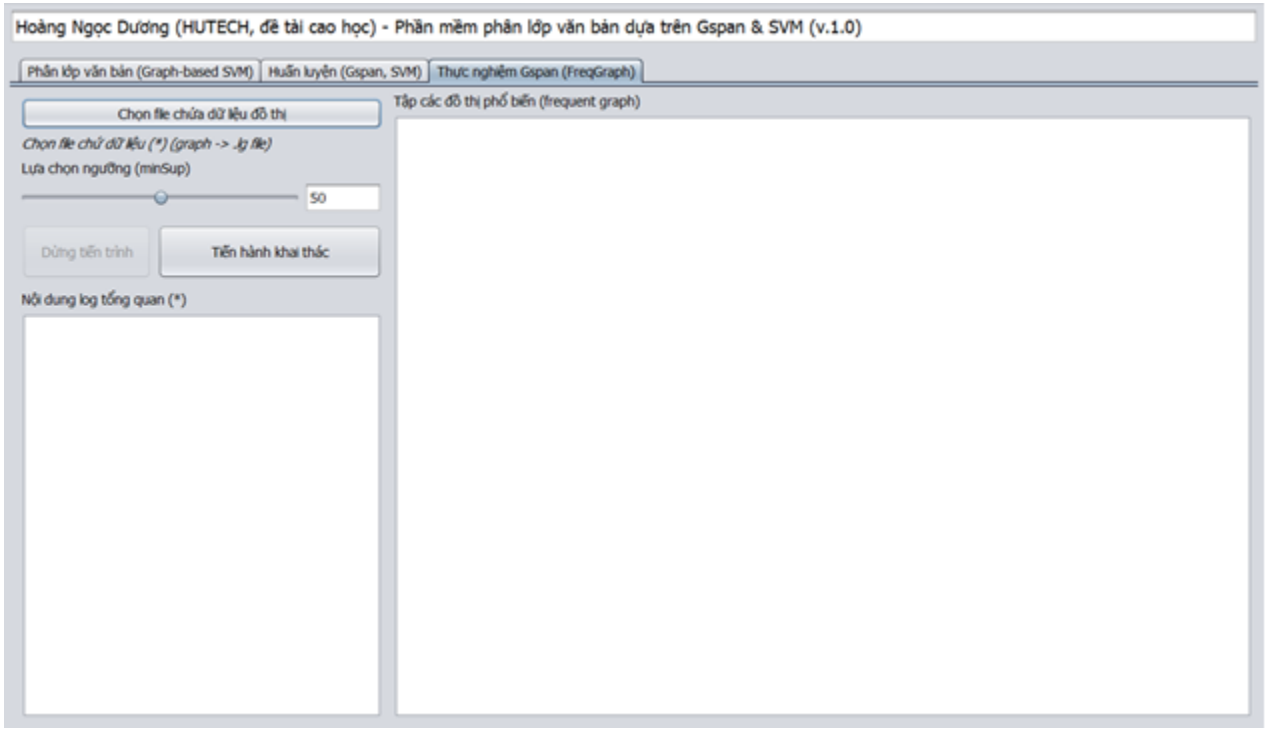
### Giao diện chương trình




### Chức năng huấn luyện phân lớp:

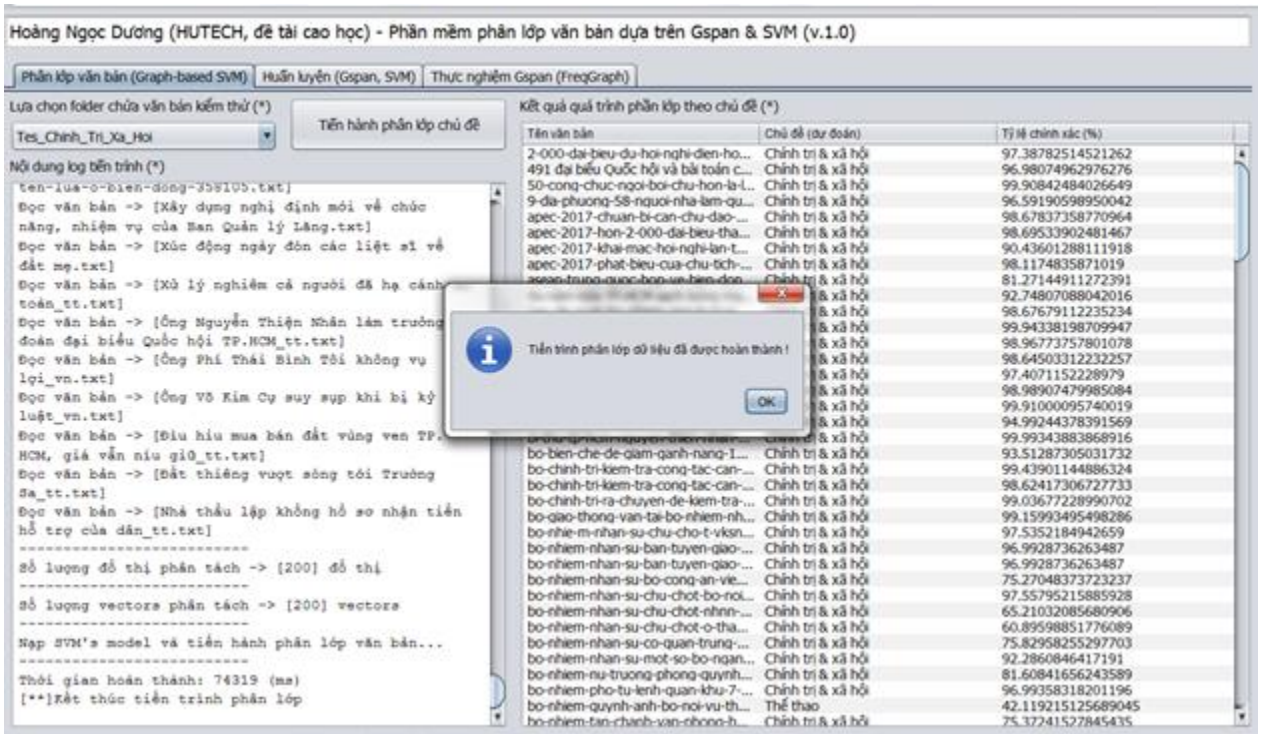


## Chức năng thực nghiệm gSpan:

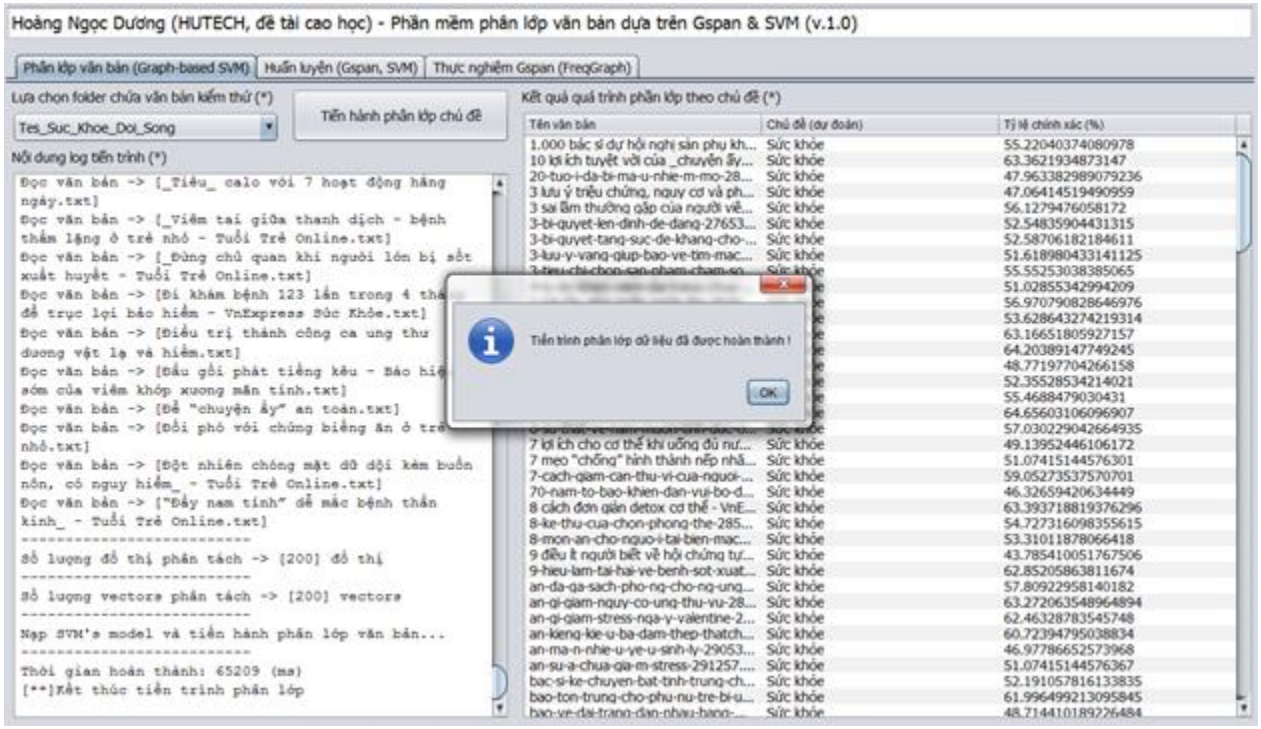



## Chức năng phân loại văn bản:

## + Phân loại chủ đề: Chính trị xã hội



**+ Phân loại chủ đề: Sức khỏe**




**+ Phân loại chủ đề: Thể thao**