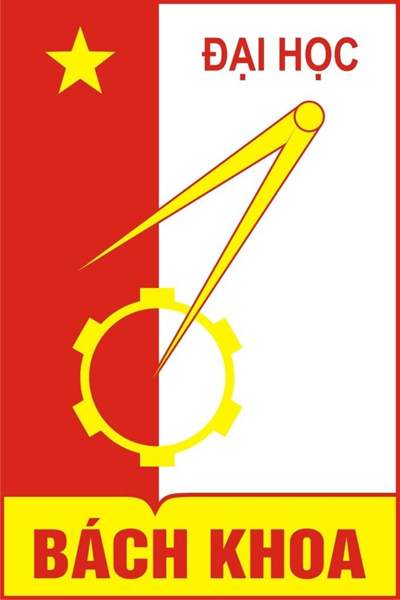
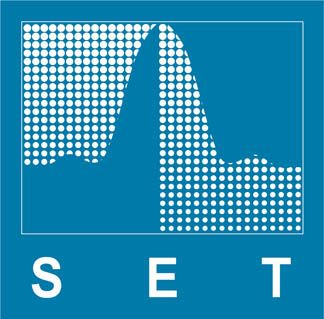
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Điện tử - Viễn thông

Bộ môn CNĐT & Kỹ thuật Y sinh

--------------------------------

**XỬ LÝ ẢNH Y TẾ**

ĐỀ TÀI: PHÂN LOẠI ẢNH UNG THƯ DẠ DÀY

SỬ DỤNG MÔ HÌNH TÚI TỪ TRỰC QUAN

*Giảng viên hướng dẫn:* **TS. Nguyễn Việt Dũng**

*Nhóm 10:  
Sinh viên thực hiện MSSV*

**Trịnh Anh Dương 20130759**

**Chử Văn Thành 20133502**

**Trần Hồng Giang 20131120**

**Nguyễn Đức Thiện 20133742**

***Hà Nội, 06/2017***

**MỤC LỤC**

[DANH MỤC HÌNH ẢNH 3](#_Toc486546384)

[LỜI MỞ ĐẦU 4](#_Toc486546385)

[NỘI DUNG 5](#_Toc486546386)

[CƠ SỞ LÝ THUYẾT 5](#_Toc486546387)

[I. Ảnh nội soi (Endoscope) 5](#_Toc486546388)

[1. Khái niệm 5](#_Toc486546389)

[2. Các kỹ thuật nội soi 5](#_Toc486546390)

[II. Giới thiệu về Machine Learning 9](#_Toc486546391)

[1. Khái niệm 9](#_Toc486546392)

[2. Một số phương pháp Machine Learning nổi tiếng 9](#_Toc486546393)

[III. Features Engineering 10](#_Toc486546394)

[1. Features Engineering là gì. 10](#_Toc486546395)

[2. Tầm quan trọng của Features Engineering 11](#_Toc486546396)

[3. Các bài toán con trong kĩ thuật feature engineering 12](#_Toc486546397)

[IV. Mô hình túi từ trực quan (Bag of visual words) 15](#_Toc486546398)

[1. Túi từ 15](#_Toc486546399)

[2. Túi từ trực quan 15](#_Toc486546400)

[V. Thuật toán SVM phân lớp dữ liệu 18](#_Toc486546401)

[1. Giới thiệu về SVM 18](#_Toc486546402)

[2. Cơ sở lý thuyết 19](#_Toc486546403)

[3. Bài toán phân 2 lớp với SVM 20](#_Toc486546404)

[4. Bài toán phân đa lớp với SVM 21](#_Toc486546405)

[5. Các bước chính của phương pháp SVM 22](#_Toc486546406)

[6. Ví dụ SVM trong phân loại hoa 22](#_Toc486546407)

[PHƯƠNG PHÁP THỰC HIỆN 25](#_Toc486546408)

[I. Hệ thống thực hiện 25](#_Toc486546409)

[II. Dataset 25](#_Toc486546410)

[III. Cross-validation và phương pháp Hout-out 25](#_Toc486546411)

[IV. Đánh giá độ chính xác (trung bình) của mô hình phân lớp 27](#_Toc486546412)

[V. Sơ đồ thực hiện 28](#_Toc486546413)

[VI. Các bước phân loại hình ảnh sử dụng Bag of features Words 28](#_Toc486546414)

[VII. Các yếu tố tác động đến kết quả phân loại 35](#_Toc486546415)

[1. Thay đổi kích thước hình ảnh 35](#_Toc486546416)

[2. Ảnh hưởng bởi tỷ lệ *N = trainingSets : testingSets* 37](#_Toc486546417)

[3. Kích thước Vocabulary 38](#_Toc486546418)

[4. Hệ số *StrongestFeatures* 40](#_Toc486546419)

[5. Tham số *PointSelection* 42](#_Toc486546420)

[6. Tham số *GridStep* 42](#_Toc486546421)

[7. Độ lệch chuẩn của bộ lọc Gaussian 43](#_Toc486546422)

[KẾT QUẢ VÀ ĐÁNH GIÁ 45](#_Toc486546423)

[TÀI LIỆU THAM KHẢO 49](#_Toc486546424)

[PHỤ LỤC 50](#_Toc486546425)

# DANH MỤC HÌNH ẢNH

[*Hình 1.1: Hình ảnh minh họa 7 công nghệ nội soi 8*](#_Toc486546306)

[*Hình 1.2: Bag-of-words chứa mặt người 17*](#_Toc486546307)

[*Hình 1.3: Bag-of-Words cho ảnh ô tô 17*](#_Toc486546308)

[*Hình 1.4: Visual Words cho ảnh xe máy 18*](#_Toc486546309)

[*Hình 1.5 : Tập dữ liệu được phân chia tuyến tính 20*](#_Toc486546310)

[*Hình 1.6: Tập dữ liệu phân chia tuyến tính nhưng có nhiễu 21*](#_Toc486546311)

[*Hình 1.7: Tập dữ liệu không thể phân chia tuyến tính 21*](#_Toc486546312)

[*Hình 1.8: Ba loài hoa trong tập dữ liệu Iris 23*](#_Toc486546313)

[*Hình 1.9: Biểu diễn tập dữ liệu bằng đồ thị 2D 23*](#_Toc486546314)

[*Hình 1.10: Đồ thị trên matlab ứng với các kernel khác nhau 24*](#_Toc486546315)

[*Hình 2.1: Hình ảnh mô tả các loại mô khác nhau từ đường tiêu hóa đại diện cho bộ dữ liệu 25*](#_Toc486546316)

[*Hình 2.2: Mô hình phương pháp Hold-out 26*](#_Toc486546317)

[*Hình 2.3: Sơ đồ thực hiện 28*](#_Toc486546318)

[*Hình 2.4: Tiền xử lý 29*](#_Toc486546319)

[*Hình 2.5: Ảnh* ***O*** *thu được từ ảnh* ***RBG*** *và* ***ROI*** *30*](#_Toc486546320)

[*Hình 2.6: Ảnh* ***O*** *trong quá trình trích xuất hình ảnh với phương pháp Grid 32*](#_Toc486546321)

[*Hình 2.7: Ma trận tính năng 32*](#_Toc486546322)

[*Hình 2.8: Minh họa phân cụm tập dữ liệu fisheriris trong Matlab 34*](#_Toc486546323)

[*Hình 2.9: Ma trận tính năng sau khi phân cụm 34*](#_Toc486546324)

[*Hình 2.10: Mô tả quá trình tạo “Túi từ trực quan” – BoW 35*](#_Toc486546325)

# LỜI MỞ ĐẦU

Ngày này ung thư trong đường tiêu hóa là một trong những căn bệnh nguy hiểm trên toàn thế giới, tuy nhiên loại ung thư này có ít triệu chứng ở giai đoạn đầu. Chính vì vậy, việc cung cấp các công cụ tốt hơn để chẩn đoán có thể giúp việc phát hiện sớm và điều trị là vô cùng cần thiết. Các công nghệ xử lý hình ảnh y tế để chẩn đoán ung thư đã và đang được phát triển trong thế kỷ qua, đặc biệt là nội soi. Chúng cho phép thu được hình ảnh của các mô trên đường tiêu hóa với độ phân giải tốt. Các kỹ thuật để phân tích những hình ảnh này đã được phát triển và đã được sử dụng bởi hệ thống Chẩn đoán bằng máy tính (Computer Aided Diagnosis -CAD).

Trong hệ tiêu hóa, các kỹ thuật của hệ thống CAD đã cho phép các bác sĩ phân tích các hình ảnh nội soi. Tuy nhiên, sự nhận biết ung thư trong đường tiêu hóa là một vấn đề hết sức phức tạp, trong đó chỉ có các bác sĩ được đào tạo có tỷ lệ thành công cao. Một số giải pháp nhận dạng mẫu bao gồm vấn đề này đã được phát hiện và ứng dụng trong quá khứ. Trong đề tài này, chúng em đã tạo một Bag of visual Words (BoW), từ BoW và dữ liệu ảnh tạo nên một bộ phân loại SVM tuyến tính 2 lớp.

Chúng em đã thực hiện qua bộ dữ liệu hình ảnh mà chúng em đã điều chỉnh tất cả các thông số có liên quan. Phương pháp của chúng em trình bày hiệu suất tối ưu của nó khi điều chỉnh: hình ảnh chuyển đổi sang màu xám, giảm kích thước 50%. Phương pháp thực hiện tốt hơn khi sử dụng “Máy dò tính tăng SURF” (*Speeded up robust features*), mang lại hiệu quả tốt hơn khi dùng phương pháp chia lưới lên ảnh. Chúng em đã đánh giá phương pháp và đưa ra được sai số trung bình khoảng 6 ± 1%.

Cuối cùng, chúng em xin chân thành cảm ơn thầy giáo **TS. Nguyễn Việt Dũng** đã có những bài giảng bổ ích trong quá trình học tập và hướng dẫn để chúng em có thể hoàn thành đề tài, chúng em xin chúc thầy hạnh phúc và thành đạt trong cuộc sống.

# CƠ SỞ LÝ THUYẾT

1. **Ảnh nội soi (Endoscope)**
2. **Khái niệm**

Nội soi (Endoscopy) là một kỹ thuật [y học](https://vi.wikipedia.org/wiki/Y_học) hiện đại được ứng dụng trong việc [khám bệnh](https://vi.wikipedia.org/w/index.php?title=Khám_bệnh&action=edit&redlink=1), chẩn đoán bệnh bằng việc sử dụng các loại dụng cụ chuyên biệt để quan sát trực tiếp bên trong các cơ quan của [cơ thể](https://vi.wikipedia.org/w/index.php?title=Cơ_thể&action=edit&redlink=1). Với kỹ thuật nội soi, người ta có thể [quay phim](https://vi.wikipedia.org/w/index.php?title=Quay_phim&action=edit&redlink=1), chụp hình bên trong các cơ quan, lấy dị vật, [sinh thiết](https://vi.wikipedia.org/w/index.php?title=Sinh_thiết&action=edit&redlink=1) và thậm chí là thực hiện hiện phẫu thuật nội soi. Nội soi hiện nay được sử dụng trong hầu hết các chuyên khoa: [tai mũi họng](https://vi.wikipedia.org/w/index.php?title=Tai_mũi_họng&action=edit&redlink=1), [tiêu hóa](https://vi.wikipedia.org/wiki/Tiêu_hóa) ([thực quản](https://vi.wikipedia.org/wiki/Thực_quản), [dạ dày](https://vi.wikipedia.org/wiki/Dạ_dày), ruột già…) sản, [ngoại, tiết niệu](https://vi.wikipedia.org/w/index.php?title=Ngoại,_tiết_niệu&action=edit&redlink=1), xương khớp, [thần kinh](https://vi.wikipedia.org/w/index.php?title=Thần_kinh&action=edit&redlink=1), thẩm mỹ…

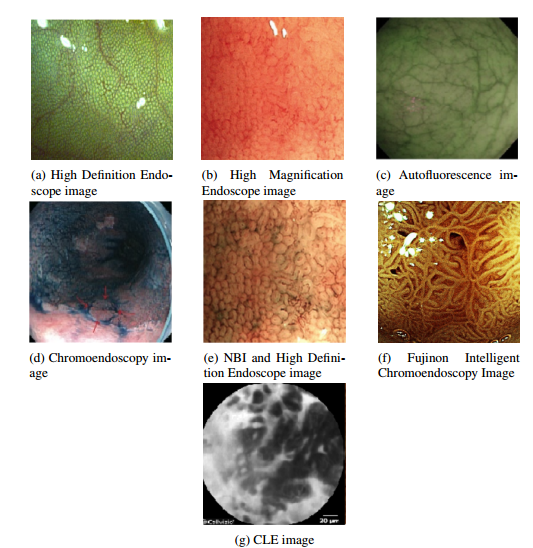
Thiết bị nội soi là một thiết bị y tế có khả năng thu được hai hình ảnh đồng thời và thậm chí các mẫu mô từ đường tiêu hóa. Mặc dù có rất nhiều cấu hình nhưng các loại nội soi đều có cùng một số đặc điểm:

* Bao gồm một ống mềm dẻo
* Có khả năng chiếu sáng và thu thập hình ảnh được kết hợp
* Có một kênh phụ, có thể chụp lấy mẫu sinh thiết.

1. **Các kỹ thuật nội soi**
2. *Chụp nội soi tiêu chuẩn độ nét cao*: Ống nội soi định nghĩa tiêu chuẩn được trang bị chip CCD, thu tín hiệu hình ảnh với 100-400 pixel và hình ảnh với tỷ lệ co 4: 3. Nội soi độ nét cao có một cấu hình tương tự nhưng cung cấp hình ảnh có độ phân giải cao hơn. Do đó, bác sĩ có thể phát hiện những thay đổi tinh vi trong niêm mạc dễ dàng hơn. Hình ảnh của nó có độ phân giải cao gấp 10 lần, hai tỷ lệ khung hình khác nhau và màn hình của nó có thể hiển thị hình ảnh chất lượng cao trong khi đang tiến hành. Tốc độ khung hình 60 lần / giây làm giảm lượng nhiễu và cho phép chụp nhanh các chuyển động nhanh
3. *Chụp nội soi phóng đại cao (HME) :* Những tiến bộ công nghệ đã dẫn tới sự phát triển của HME. Công nghệ này sử dụng các ống kính chuyển động điện tử cho phép hình ảnh hóa hình ảnh niêm mạc thời gian thực chi tiết hơn. Hình ảnh nội soi được phóng to đến 150 lần, trong khi vẫn giữ được chi tiết và độ phân giải cao. Đây là một cải tiến lớn khi so sánh với kỹ thuật số thu phóng thông thường hoặc các hệ thống phóng đại điện tử. Trong HME, hình ảnh được di chuyển gần màn hình, giảm số lượng quan sát được và độ phân giải hình ảnh. Hầu hết các nội soi thông thường đều có khả năng phóng đại điện tử 1,5 lần đến 2 lần nhưng lại cần một bộ xử lý tương thích.
4. *Hình ảnh phát huỳnh quang tự động (AFI):* AFI phát hiện huỳnh quang tự nhiên của mô, được phát ra bởi các phân tử kích thích ánh sáng cụ thể. Có thể nắm bắt sự khác biệt màu sắc trong phát xạ huỳnh quang trong thời gian thực, bởi vì quang phổ huỳnh quang đặc biệt của chúng. Kỹ thuật này cho phép mô tả mô và có thể được sử dụng để phát hiện một số lượng đáng kể các bệnh nhân ung thư sớm ở độ tuổi cao.
5. *Chromoendoscopy :* Kỹ thuật này dựa trên việc tăng cường bề mặt của niêm mạc bằng cách áp dụng các thuốc nhuộm khác nhau. Tùy thuộc vào các giao thức tiếp theo và các thuốc nhuộm khác nhau được sử dụng, làm cho các cấu trúc giải phẫu khác nhau dễ được quan sát hơn.
6. *Chụp quang tuyến Laser (CLE)*: công nghệ này cung cấp hình ảnh cực nhỏ độ phân giải cao ở độ phân giải tế bào nhỏ, luồn vào các lớp sâu nhất của niêm mạc dạ dày-ruột. Tuy nhiên phương pháp kết hợp này đòi hỏi người dùng có kinh nghiệm để thao tác chính xác nó và do đó đạt được kết quả tốt.
7. *I-scan và FICE* : Phương pháp này bao gồm ba loại thuật toán: Tăng cường bề mặt, Tăng cường tương phản và Tăng cường Tone. Tăng cường bề mặt làm tăng độ tương phản sáng-tối cho hình ảnh, bằng cách lấy dữ liệu cường độ luminance cho mỗi pixel. Sau đó, các thuật toán để quan sát chi tiết cấu trúc bề mặt niêm mạc được sử dụng. Tương phản tăng cường kỹ thuật số thêm màu xanh ở các vùng tương đối tối: dữ liệu cường độ luminance cho mỗi điểm ảnh thu được và những bất thường tinh vi xung quanh bề mặt được tăng cường tính toán. Cả hai chức năng tăng cường hoạt động trong thời gian thực mà không làm suy giảm màu sắc ban đầu của cơ quan. Ngoài ra, cả hai đều thích hợp để sàng lọc nội soi để phát hiện các khối u dạ dày, ruột ở giai đoạn sớm. Ngoài ra nó còn có khả năng chẩn đoán và điều trị bệnh đường tiêu hóa.
8. *Phẫu thuật nội soi viên không dây (WCE):* Việc kiểm tra ruột non là một công việc khó khăn do hình dạng dài và phức tạp. WCE được thiết kế để vượt qua giới hạn này và làm cho quy trình nội soi an toàn, ít xâm lấn hơn và thoải mái hơn cho bệnh nhân. Trong trường hợp này, nội soi không phải là ống linh hoạt mà là một viên nang mà bệnh nhân nuốt. Viên nang nhỏ được trang bị nguồn sáng, ống kính, máy ảnh, máy phát tín hiệu radio và pin. Được vận chuyển bởi peristalsis, viên nang đi qua hệ thống tiêu hóa trong khoảng 8 giờ và tự động chụp được hơn 50 000 hình ảnh. Chúng được truyền qua không dây tới máy ghi bên trong cơ thể. Hiện tại, WCE không chỉ muốn kiểm tra ruột non mà còn các cơ quan khác như đại tràng hoặc thực quản. Những hạn chế chính của WCE bao gồm việc thiếu khả năng lấy mẫu sinh thiết, trái với các kỹ thuật nội soi khác

BẢNG TÓM TẮT CÁC CÔNG NGHỆ NÔI SOI

|  |  |  |
| --- | --- | --- |
| CÔNG NGHỆ NỘI SOI | ỨNG DỤNG | MỤC ĐÍCH |
| Tiêu chuẩn và HD | Phát hiện những thay đổi tinh vi trong niêm mạc dễ dàng hơn | Tăng bề mặt nội soi và giảm các thành phần lạ trong ảnh |
| HME ( Phóng đại cao) | Xác định hiện tượng tân mạch | Quan sát tốt chi tiết bề mặt nội soi và mạch máu. |
| AFI ( Hình ảnh phát huỳnh quang tự động) | Xác định hoặc phát hiện sớm ung thư | Mô tả mô và phát hiện ung thư |
| NBI ( Chụp hình với biên độ hẹp) | Xác định hiện tượng tân mạch kết đặc biệt kết hợp với nội soi phóng đại. | Tăng cường mô hình vi mô |
| I-scan và FICE | Xác định hiện tượng tân mạch kết đặc biệt kết hợp với nội soi phóng đại | Phát hiện, chẩn đoán và điều trị bệnh dạ dày ruột dễ dàng hơn |
| CLE (Chụp quang tuyến Laser) | Xác định hiện tượng tân mạch | Cung cấp hình ảnh cực nhỏ có độ phân giải cao |



Hình 1.1: Hình ảnh minh họa 7 công nghệ nội soi

1. Hình ảnh nội soi có độ phân giải cao của bề mặt niêm mạc bình thường
2. Phóng đại cao của niêm mạc ruột bình thường
3. AFI trong ruột kết bình thường
4. Hình ảnh Chromoendoscope
5. Hình ảnh NBI và HD
6. Hình ảnh nội soi màu nâu nhạt của một tổn thương niêm mạc
7. Hình ảnh mô bình thường sử dụng đầu dò CLE
8. **Giới thiệu về Machine Learning**
9. **Khái niệm**

Machine Learning là một lĩnh vực của [trí tuệ nhân tạo](https://vi.wikipedia.org/wiki/Trí_tuệ_nhân_tạo) liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống "học" tự động từ dữ liệu để giải quyết những vấn đề cụ thể. Ví dụ như các máy có thể "học" cách phân loại [thư điện tử](https://vi.wikipedia.org/wiki/Thư_điện_tử) xem có phải [thư rác (spam)](https://vi.wikipedia.org/wiki/Thư_rác) hay không và tự động xếp thư vào thư mục tương ứng. Học máy rất gần với [suy diễn thống kê](https://vi.wikipedia.org/w/index.php?title=Suy_diễn_thống_kê&action=edit&redlink=1) (statistical inference) tuy có khác nhau về thuật ngữ.

Học máy có liên quan lớn đến [thống kê](https://vi.wikipedia.org/wiki/Khoa_học_Thống_kê), vì cả hai lĩnh vực đều nghiên cứu việc phân tích dữ liệu, nhưng khác với thống kê, học máy tập trung vào sự phức tạp của các giải thuật trong việc thực thi tính toán. Nhiều bài toán suy luận được xếp vào loại bài toán [NP-khó](https://vi.wikipedia.org/wiki/NP-khó) (nondeterministic polynomial time), vì thế một phần của học máy là nghiên cứu sự phát triển các giải thuật suy luận xấp xỉ mà có thể xử lý được.

Học máy hiện nay được áp dụng rộng rãi bao gồm [máy truy tìm dữ liệu](https://vi.wikipedia.org/wiki/Máy_truy_tìm_dữ_liệu), [chẩn đoán y khoa](https://vi.wikipedia.org/wiki/Chẩn_đoán), phát hiện [thẻ tín dụng giả](https://vi.wikipedia.org/w/index.php?title=Thẻ_tín_dụng_giả&action=edit&redlink=1), phân tích [thị trường chứng khoán](https://vi.wikipedia.org/wiki/Thị_trường_chứng_khoán), phân loại các [chuỗi DNA](https://vi.wikipedia.org/w/index.php?title=Chuỗi_DNA&action=edit&redlink=1), [nhận dạng tiếng nói](https://vi.wikipedia.org/wiki/Nhận_dạng_tiếng_nói) và [chữ viết](https://vi.wikipedia.org/w/index.php?title=Nhận_dạng_chữ_viết_tay&action=edit&redlink=1), [dịch tự động](https://vi.wikipedia.org/wiki/Dịch_tự_động), [chơi trò chơi](https://vi.wikipedia.org/w/index.php?title=Trò_chơi_chiến_lược&action=edit&redlink=1) và [cử động rô-bốt](https://vi.wikipedia.org/w/index.php?title=Cử_động_rô-bốt&action=edit&redlink=1) (robot locomotion).

1. **Một số phương pháp Machine Learning nổi tiếng**

Hai phương pháp của Machine Learning được chấp nhận rộng rãi chính là **supervised learning** (học có giám sát) và **unsupervised learning** (học không giám sát) nhưng cũng có những phương pháp khác như **semisupervised learning**(học bán giám sát)**, reinforcement learning**(học tăng cường)

Dưới đây là khái niệm chung về 2 phương pháp phổ biến nhất:

1. ***Supervised Learning (SL)***

Là một kĩ thuật học máy để học tập từ tập dữ liệu được gán nhãn cho trước. Tập dữ liệu cho trước sẽ chứa nhiều bộ dữ liệu. Mỗi bộ dữ liệu có cấu trúc theo cặp {x, y} với x được xem là dữ liệu thô (raw data) và y là nhãn của dữ liệu đó. Nhiệm vụ của SL là dự đoán đầu ra mong muốn dựa vào giá trị đầu vào. Dễ nhận ra, học có GIÁM SÁT tức là máy học dựa vào sự trợ giúp của con người, hay nói cách khác con người dạy cho máy học và giá trị đầu ra mong muốn được định trước bởi con người. Tập dữ liệu huấn luyện hoàn toàn được gán nhãn dựa vào con người. Tập càng nhỏ thì máy tính học càng ít.

SL cũng được áp dụng cho 2 nhóm bài toán chính là bài toán dự đoán (regression problem) và bài toán phân lớp (classification problem).

Kỹ thuật SL thực chất là để xây dựng một hàm có thể xuất ra giá trị đầu ra tương ứng với tập dữ liệu. Ta gọi hàm này là hàm h(x) và mong muốn hàm này xuất ra đúng giá trị y với một hoặc nhiều tập dữ liệu mới khác với dữ liệu được học. Hàm h(x) cần các loại tham số học khác nhau tùy thuộc với nhiều bài toán khác nhau. Việc học từ tập dữ liệu (training) cũng chính là tìm ra bộ tham số học cho hàm h(x).

1. ***Unsupervised learning (UL)***

Là một kĩ thuật của máy học nhằm tìm ra một mô hình hay cấu trúc bị ẩn bởi tập dữ liệu KHÔNG được gán nhãn cho trước. UL khác với SL là không thể xác định trước đầu ra từ tập dữ liệu huấn luyện được. Tùy thuộc vào tập huấn luyện kết quả output sẽ khác nhau. Trái ngược với SL, tập dữ liệu huấn luyện của UL không do con người gán nhãn, máy tính sẽ phải tự học hoàn toàn. Có thể nói, học KHÔNG GIÁM SÁT thì giá trị đầu ra sẽ phụ thuộc vào thuật toán UL.

*Ứng dụng:* Ứng dụng phổ biến nhất của học không giám sát là gom cụm (cluster). Ứng dụng này dễ nhận ra nhất là Google và Facebook. Google có thể gom nhóm các bài báo có nội dung gần nhau, hoặc Facebook có thể gợi ý kết bạn có nhiều bạn chung cho bạn. Các bài báo có cùng nội dung sẽ được gom lại thành một nhóm (cluster) phân biệt với các nhóm khác. Dữ liệu huấn luyện là các bài báo từ quá khứ tới hiện tại và tăng dần theo thời gian. Dễ nhận ra rằng dữ liệu không thể gán nhãn bởi con người. Khi một bài báo mới được cho vào input, nó sẽ tìm cụm (cluster) gần nhất với bài báo đó và gợi ý những bài liên quan.

1. **Features Engineering**
2. **Features Engineering là gì.**

Feature Engineering là quá trình chuyển đổi tập dữ liệu thô ban đầu thành tập các thuộc tính (features) có thể giúp biểu diễn tập dữ liệu ban đầu tốt hơn, tạo điều kiện để giải quyết các bài toán dễ dàng hơn, giúp tương thích với từng mô hình dự đoán cụ thể, cũng như cải thiện độ chính xác của mô hình dự đoán hiện tại.

Feature engineering cố gắng biểu diễn tốt nhất tập dữ liệu ban đầu sao cho tương thích với mô hình dự đoán bạn đang sử dụng. Xét bài toán dự đoán sinh viên này có khả năng bỏ học là bao nhiêu phần trăm. Thông thường, ta sẽ sử dụng tất cả các thuộc tính liên quan đến sinh viên đó để áp dụng cho bài toán phân lớp, mà các thuộc tính này thường rất nhiều từ 20-50 cột thuộc tính. Tuy nhiên, khi đưa toàn bộ thuộc tính này vào mô hình phân lớp của mình (ví dụ mô hình cây quyết định), thời gian để máy traing rất lâu, đồng thời kết qủa dự đoán có độ chính xác thấp. Thay vì làm như vậy, ta sử dụng kĩ thuật feature engineering để chọn ra một số thuộc tính phù hợp hơn như có vay mượn để đóng học phí không, số điểm đầu vào là bao nhiêu, quá trình tiến bộ trong học tập là như thế nào,… Ngoài ra, ta có thể thu thập thêm các thuộc tính cần thiết khác để bổ sung vào tập dữ liệu ban đầu như có tham gia nhiều hoạt động ngoại khóa hay không, có được hỗ trợ vào kí túc xá hay không,… Thì số thuộc tính đưa vào mô hình phân lớp được giảm đi đáng kể giúp tốc độ để máy training nhanh hơn. Hơn nữa, nhờ biểu diễn tốt tập dữ liệu ban đầu mà độ chính xác mô hình phân lớp được cải thiện đáng kể.

Cũng như lập trình là một nghệ thuật, giao tiếp là một nghệ thuật, hay khám chữa bệnh là một nghệ thuật thì feature engineering cũng là một nghệ thuật. Do tập dữ liệu trên thực tế phức tạp hơn rất nhiều so với các giả định trong nghiên cứu, thêm vào đó các bài toán trong thực tế luôn luôn biến đổi và đòi hỏi những nhà khoa học dữ liệu phải biết thích nghi trong từng trường hợp để đưa ra cách xây dựng mô hình phù hợp nhất. Vì vậy, việc thêm bớt, chỉnh sửa các thuộc tính cho tập dữ liệu ban đầu sao cho có thể cải thiện tốc độ tính toán cũng như nâng cao độ chính xác của mô hình là một nghệ thuật. Để đạt được trình độ này, đòi hỏi chúng ta phải va chạm nhiều các bài toán thực tế, đồng thời học hỏi từ cộng đồng để nâng cao kiến thức cũng như kinh nghiệm cho bản thân.

1. **Tầm quan trọng của Features Engineering**

Các thuộc tính trong tập dữ liệu ảnh hưởng trực tiếp đến mô hình dự đoán, do đó ta cần xác định tốt cấu trúc của các thuộc tính sao cho diễn đạt hiệu quả nhất bản chất của tập dữ liệu.

Mặc dù ta không chọn được mô hình dự đoán tốt nhất nhưng ta vẫn có thể đạt được kết quả dự đoán cao. Hầu hết các thuật toán sẽ tự điều chỉnh các thông số phù hợp theo cấu trúc các thuộc tính hiện tại. Tuy nhiên, việc có được tập thuộc tính tinh gọn sẽ góp phần làm đơn giản hoá độ phức tạp tính toán của mô hình nhờ vậy mà tính toán được nhanh hơn và dễ dàng để diễn giải cho người dùng. Ví dụ, khi sử dụng mô hình cây quyết định, nếu ta sử dụng quá nhiều thuộc tính vào quá trình dự đoán, mặc dù cho kết quả rất tốt tuy nhiên, người dùng sẽ rất khó quan sát và diễn giải kết quả dự đoán.

Trong quá trình tối ưu hoá tham số, mặc dù không đạt được mục tiêu này nhưng với tập thuộc tính được thiết kế tốt, ta vẫn có thể đạt được kết quả dự đoán cao. Ta không cần phải cực lực tìm kiếm mô hình nào phù hợp nhất cũng như bộ trọng số được tối ưu nhất. Chỉ với tập thuộc tính được thiết kế tốt, ta đã mô tả được tập dữ liệu hiện có cũng như tiếp cận với bài toán thực tế dễ dàng và rõ ràng hơn rất nhiều.

1. **Các bài toán con trong kĩ thuật feature engineering**
   1. **Cần xác định đâu là features, đâu là attributes**

Đối với dữ liệu dạng bảng tính, thuật ngữ observations (các mẫu quan sát) hay instance (các mẫu thể hiện) là các dòng dữ liệu trong bảng tính. Mỗi dòng dữ liệu nào gồm có nhiều variables (biến) hay attributes (thuộc tính) là các cột của bảng tính.

Feature khác với attribute mặc dù cũng mang nghĩa thuộc tính nhưng feature mang nhiều ngữ nghĩa hơn. Ví dụ trong thị giác máy tính (computer vision), một bức ảnh là một observation gồm nhiều attributes là các điểm pixel. Tuy nhiên, feature ở đây có thể là các đường biên cạnh hay dáng dấp hình học trong bức ảnh. Trong xử lý ngôn ngữ tự nhiên (natural language processing), một văn bản hay một tweet sẽ là một observation gồm nhiều attributes là các từ trong đó. Thế nhưng, feature ở đây có thể là một ngữ (phrase) hay số lượng từ phân bố trong đó. Trong nhận dạng tiếng nói (speech recognition), một người phát ra âm thành là observation, attributes sẽ là các sóng âm dưới dạng tần số, còn feature sẽ là các nhóm âm thanh được gom lại để phân tích.

* 1. **Đánh giá độ hữu dụng của các features**

Ta có thể thực hiện việc đánh giá độ hữu dụng của các feature để tiền xử lý dữ liệu trước khi đưa vào xây dựng mô hình dự đoán. Các feature này sau khi được ước lượng và đánh giá sẽ có một điểm số để xếp hạng. Những feature có điểm xếp hạng cao sẽ được chọn ra để đưa vào training, còn lại những feature có điểm xếp hạng thấp sẽ bị lược bỏ.

Điểm xếp hạng của các feature còn được dùng để rút trích hay tạo ra những feature mới. Những mô hình khi được training có đưa ra được chỉ số xếp hạng cho các feature như MARS, [Random Forest](http://en.wikipedia.org/wiki/Random_forest#Variable_importance) và Gradient Boosted Machines.

* 1. **Bài toán trích xuất tính năng (feature extraction)**

Trên thực tế, các mẫu dữ liệu như hình ảnh, audio, hay văn bản mang trong mình rất nhiều thuộc tính. Mà các thuộc tính này khi đưa trực tiếp vào các mô hình dự đoán sẽ làm cho tốc độ training của các mô hình bị giảm đi rõ rệt.

[Feature extraction](http://en.wikipedia.org/wiki/Feature_extraction) là tiến trình tự động hoá được dùng để giảm số chiều dữ liệu sao cho dữ liệu ban đầu được chuyển đổi sang dạng đơn giản và nhỏ hơn, trước khi đưa vào mô hình dự đoán.

Đối với dữ liệu dạng bảng tính, ta có thể sử dụng các phương pháp như phân tích thành phần chính (Principal Component Analysis) hay phương pháp gom nhóm (unsupervised clustering). Đối với dữ liệu hình ảnh, ta có thể sử dụng phương pháp rút trích biên cạnh. Tuỳ thuộc vào từng lĩnh vực, hình ảnh, video, hay audio mà ta có các phương pháp rút trích đặc trưng khác nhau.

* 1. **Bài toán lựa chọn đặc trưng (feature selection)**

Các feature có tầm quan trọng khác nhau, tùy theo từng bài toán mà có thuộc tính này hữu ích hơn các thuộc tính kia và cần loại bỏ bớt các thuộc tính không liên quan đến bài toán. Có một số feature cần thiết để góp phần vào cải thiện độ chính xác của thuật toán cũng có các feature dư thừa không phù hợp với bài toán hiện tại.

[Feature selection](http://en.wikipedia.org/wiki/Feature_selection) giải quyết các vấn đề trên bằng các tự động hoá lựa chọn tập con trong số các feature ban đầu sao cho các feature được lựa chọn này phù hợp với bài toán hiện tại.

Các thuật toán feature selection có thể sử dụng phương pháp ranking để lựa chọn feature bằng cách dựa vào mối tương quan giữa các thuộc tính để lọc ra được các thuộc tính độc lập với nhau. Một số phương pháp nâng cao khác sẽ tìm tập con feature bằng cách thử và sai, xây dựng và ước lượng mô hình một cách tự động thông qua từng tập con feature tìm được.

* 1. **Bài toán xây dựng đặc trưng mới (feature construction)**

Đây là công việc đòi hỏi nhiều sự sáng tạo, và thời gian của nhà khoa học dữ liệu. Để xây dựng các đặc trưng mới, ta cần thao tác bằng sức người thay vì tự động hoá bằng các giải thuật như feature selection. Do đó, ta mới gọi feature engineering là một nghệ thuật.

Feature engineering sẽ chiếm nhiều thời gian xây dựng và thực nghiệm. Ta phải đào sâu suy nghĩ về bài toán hiện tại để có thể đưa ra cấu trúc đặc trưng phù hợp nhất. Nhờ vậy mà các mô hình dự đoán có thể cải thiện được độ chính xác của mình.

Đối với dữ liệu dạng bảng tính, ta thường gom nhóm hay kết hợp các feature hiện tại để tạo ra các feature mới, hay phân rã, tách thành các feature đơn lẻ để tạo thành các feature mới. Đối với dữ liệu dạng văn bản, ta cần dựa vào bối cảnh để đưa ra các chỉ số cụ thể cho bài toán của mình. Còn với dữ liệu hình ảnh, ta thường dành nhiều thời gian thực nghiệm các filter để chọn ra được một bộ lọc tốt nhất cho mô hình của mình.

* 1. **Bài toán xác định feature thông qua training dữ liệu**

Làm thế nào ta có thể tránh được việc xây dựng và rút trích đặc trưng một cách thủ công? Representation learning hay còn gọi là [feature learning](http://en.wikipedia.org/wiki/Feature_learning) là hướng tiếp cận để tự động hóa quá trình này.

Các phương pháp deep learning hiện đại đã có nhiều thành công trong lĩnh vực này ví dụ như phương pháp autoencoders và restricted Boltzmann machines. Các phương pháp này hoàn toàn tự động hóa quá trình feature engineering bằng cách unsupervised hay semi-supervised, nhờ vậy mà ta có thể biểu diễn các feature một cách trừu tượng nhất (dạng thu gọn). Đây là hướng tiếp cận tiên tiến nhất và đã có nhiều kết qủa đáng kể trong speech recognition, image classification, object recognition và các lĩnh vực khác.

Tuy nhiên, điểm hạn chế của hướng tiếp cận này đó là ta không hiểu được tại sao các feature này được trích chọn. Vì nó là black-box nên ta không thể biết được chuyện gì đã xảy ra bên trong để có thể hiểu được.

1. **Mô hình túi từ trực quan (Bag of visual words)**
2. **Túi từ**

Trong cuộc sống có rất nhiều trường hợp ta phải sử dụng Túi từ (Bag of words-BoW). Ví dụ như là bài toán phân loại tin rác. Ta thấy rằng nếu một tin có chứa các từ khuyến mại, giảm giá, trúng thưởng, miễn phí, quà tặng, tri ân, … thì nhiều khả năng đó là một tin nhắn rác. Vậy phương pháp đơn giản nhất là đếm xem trong tin đó có bao nhiêu từ thuộc vào các từ trên, nếu nhiều hơn 1 ngưỡng nào đó thì ta quyết định đó là tin rác. (Tất nhiên bài toán thực tế phức tạp hơn nhiều khi các từ có thể được viết dưới dạng không dấu, hoặc bị cố tình viết sai chính tả, hoặc dùng ngôn ngữ teen). Với các loại văn bản khác nhau thì lượng từ liên quan tới từng chủ đề cũng khác nhau. Từ đó có thể dựa vào số lượng các từ trong từng loại để làm các vector đặc trưng cho từng văn bản.

Sau đây là ví dụ về cách tạo ra vector đặc trưng cho mỗi văn bản dựa trên BoW. Giả sử ta có 2 văn bản đơn giản:

* (1) John likes to watch movies. Mary likes movies too.
* (2) John also likes to watch football games.

Dựa trên hai văn bản này, ta có danh sách các từ được sử dụng, được gọi là từ điển với 10 từ như sau:

* “John”, “likes”, “to”, “watch”, “movies”, “also”, “football”, “games”, “Mary”, “too”

Với mỗi văn bản, ta sẽ tạo ra một vector đặc trưng có số chiều bằng 10, mỗi phần tử đại diện cho số từ tương ứng xuất hiện trong văn bản đó. Với hai văn bản trên, ta sẽ có hai vector đặc trưng là:

* (1) [1, 2, 1, 1, 2, 0, 0, 0, 1, 1]
* (2) [1, 1, 1, 1, 0, 1, 1, 1, 0, 0]

1. **Túi từ trực quan**

Đối với dữ liệu dạng văn bản ta sử dụng mô hình Túi từ, nhưng với dạng hình ảnh ta cần sử dụng mô hình Túi từ trực quan. Ta xét 2 ví dụ như sau:

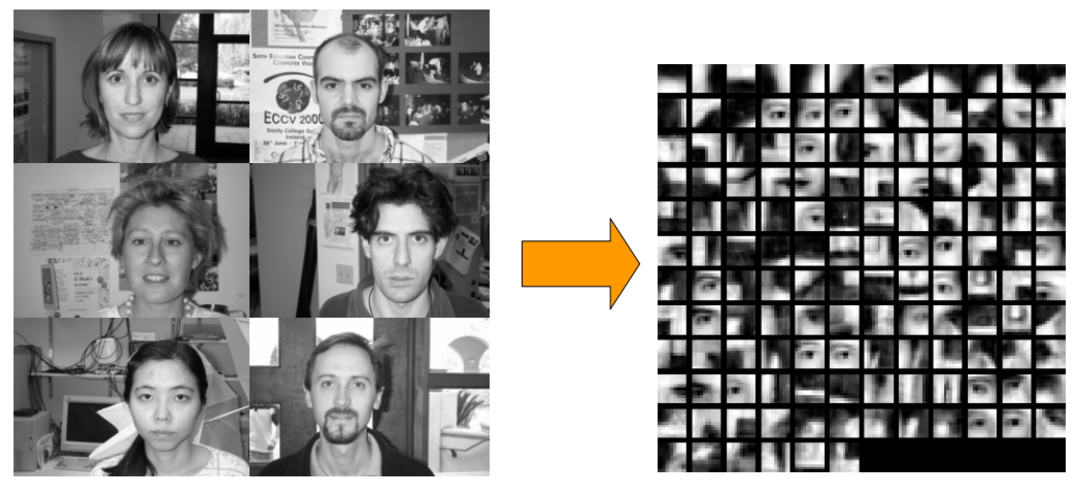
*Ví dụ 1:* Có hai class ảnh, một class là ảnh các khu rừng, một class là ảnh các sa mạc. Phân loại một bức ảnh là rừng hay sa mạc (giả sử ta biết rằng nó thuộc một trong hai loại này) một cách trực quan nhất là dựa vào màu sắc. Màu xanh nhiều là rừng, màu đỏ và vàng nhiều là sa mạc. Vậy chúng ta có thể có một mô hình đơn giản để trích chọn đặc trưng như sau:

* Với một bức ảnh, chuẩn bị một vector X có số chiều bằng 3, đại diện cho 3 màu xanh ( *x1* ), đỏ ( *x2* ), và vàng ( *x3* ).
* Với mỗi điểm ảnh trong bức ảnh đó, xem nó gần với màu xanh, đỏ hay vàng nhất dựa trên giá trị của pixel đó. Nếu nó gần điểm xanh nhất, tăng *x1* lên 1; gần đỏ nhất, tăng *x2* lên 1; gần vàng nhất, tăng *x3* lên 1.
* Sau khi xem xét tất cả các điểm ảnh, dù cho bức ảnh có kích thước thế nào, ta vẫn thu được một vector có độ dài bằng 3, mỗi phần tử thể hiện việc có bao nhiêu pixel trong bức ảnh có màu tương ứng. Vector cuối này còn được gọi là vector histogram của bức ảnh tương ứng với ba màu xanh, đỏ, vàng. Dựa vào vector này, ta có thể quyết định bức ảnh đó là ảnh rừng hay sa mạc.

*Ví dụ 2:* Trên thực tế, các bài toán xử lý ảnh không đơn giản như ví dụ 1 trên đây. Mắt người thực ra nhạy với các đường nét, hình dáng hơn là màu sắc. Một cái (ảnh) cây dù không có màu vẫn là một cái (ảnh) cây! Vì vậy, xem xét giá trị từng điểm ảnh một không mang lại kết quả khả quan vì lượng thông tin bị mất quá nhiều.

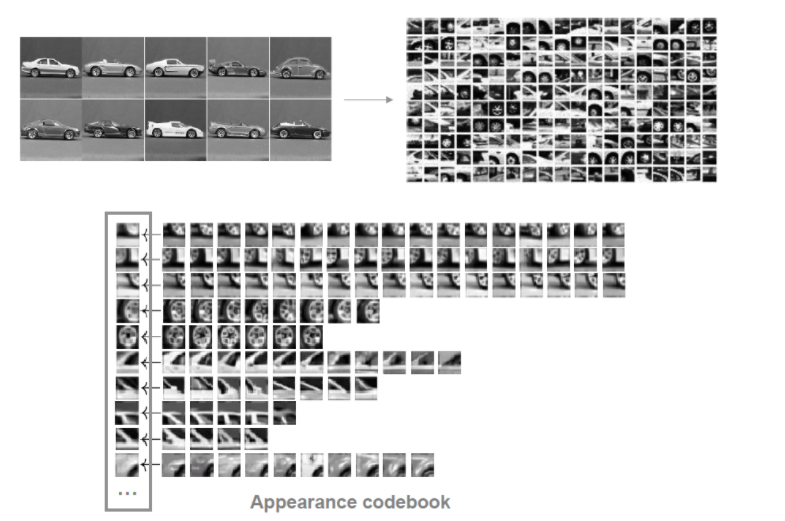
Có một cách khắc phục là thay vì xem xét một điểm ảnh, ta xem xét một cửa sổ nhỏ trong ảnh (trong Computer Vision, cửa sổ này được gọi là patch) là một hình chữ nhật chứa nhiều điểm ảnh gần nhau. Cửa sổ này đủ lớn để có thể chứa được các bộ phận có thể mô tả được vật thể trong ảnh.

Ví dụ với mặt người, các patch nên đủ lớn để chứa được các phần của khuôn mặt như mắt, mũi, miệng như hình dưới đây.

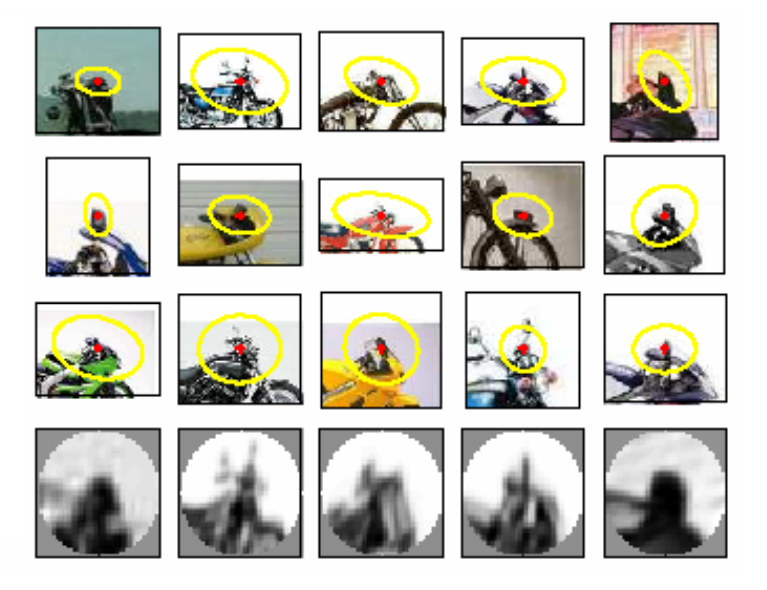


Hình 1.2: Bag-of-words chứa mặt người (tài liệu tham khảo [12])

Tương tự, với ảnh là ô tô, các patch thu được có thể là bánh xe, khung xe, cửa xe,… như trong hình dưới đây



Hình 1.3: Bag-of-Words cho ảnh ô tô (tài liệu tham khảo [12])



Hình 1.4: Visual Words cho ảnh xe máy (tài liệu tham khảo [12])

1. **Thuật toán SVM phân lớp dữ liệu**
2. **Giới thiệu về SVM**

**Máy vectơ hỗ trợ** (**SVM** - **support vector machine**) là một khái niệm trong [thống kê](https://vi.wikipedia.org/wiki/Khoa_học_Thống_kê) và [khoa học máy tính](https://vi.wikipedia.org/wiki/Khoa_học_máy_tính) cho một tập hợp các phương pháp [học có giám sát](https://vi.wikipedia.org/wiki/Học_có_giám_sát) liên quan đến nhau để [phân loại](https://vi.wikipedia.org/w/index.php?title=Phân_loại_(học_máy)&action=edit&redlink=1) và [phân tích hồi quy](https://vi.wikipedia.org/wiki/Phân_tích_hồi_quy). SVM dạng chuẩn nhận dữ liệu vào và phân loại chúng vào hai lớp khác nhau. Do đó SVM là một [thuật toán phân loại](https://vi.wikipedia.org/w/index.php?title=Thuật_toán_phân_loại&action=edit&redlink=1) [nhị phân](https://vi.wikipedia.org/wiki/Phân_loại_nhị_phân). Với một bộ các ví dụ luyện tập thuộc hai thể loại cho trước, thuật toán luyện tập SVM xây dựng một mô hình SVM để phân loại các ví dụ khác vào hai thể loại đó. Một mô hình SVM là một cách biểu diễn các điểm trong không gian và lựa chọn ranh giới giữa hai thể loại sao cho khoảng cách từ các ví dụ luyện tập tới ranh giới là xa nhất có thể. Các ví dụ mới cũng được biểu diễn trong cùng một không gian và được thuật toán dự đoán thuộc một trong hai thể loại tùy vào ví dụ đó nằm ở phía nào của ranh giới. [13]

Một máy vectơ hỗ trợ xây dựng một [*siêu phẳng*](https://vi.wikipedia.org/wiki/Siêu_phẳng) hoặc một tập hợp các siêu phẳng trong một không gian nhiều chiều hoặc vô hạn chiều, có thể được sử dụng cho phân loại, hồi quy, hoặc các nhiệm vụ khác. Một cách trực giác, để phân loại tốt nhất thì các siêu phẳng nằm ở càng xa các điểm dữ liệu của tất cả các lớp (gọi là hàm lề) càng tốt, vì nói chung lề càng lớn thì [*sai số tổng quát hóa*](https://vi.wikipedia.org/w/index.php?title=Sai_số_tổng_quát_hóa&action=edit&redlink=1) của thuật toán phân loại càng bé.

Trong nhiều trường hợp, không thể phân chia các lớp dữ liệu một cách tuyến tính trong một không gian ban đầu được dùng để mô tả một vấn đề. Vì vậy, nhiều khi cần phải [ánh xạ](https://vi.wikipedia.org/wiki/Ánh_xạ) các điểm dữ liệu trong không gian ban đầu vào một không gian mới nhiều chiều hơn, để việc phân tách chúng trở nên dễ dàng hơn trong không gian mới. Để việc tính toán được hiệu quả, ánh xạ sử dụng trong thuật toán SVM chỉ đòi hỏi [tích vô hướng](https://vi.wikipedia.org/wiki/Tích_vô_hướng) của các vectơ dữ liệu trong không gian mới có thể được tính dễ dàng từ các tọa độ trong không gian cũ. Tích vô hướng này được xác định bằng một hàm hạt nhân *K*(*x*,*y*) phù hợp. Một siêu phẳng trong không gian mới được định nghĩa là tập hợp các điểm có tích vô hướng với một vectơ cố định trong không gian đó là một hằng số. Vectơ xác định một siêu phẳng sử dụng trong SVM là một tổ hợp tuyến tính của các vectơ dữ liệu luyện tập trong không gian mới với các hệ số *αi*. Với siêu phẳng lựa chọn như trên, các điểm *x* trong không gian đặc trưng được ánh xạ vào một siêu mặt phẳng là các điểm thỏa mãn:

= hằng số

Chú ý rằng nếu *K*(*x*,*y*) nhận giá trị ngày càng nhỏ khi *y* xa dần khỏi *x* thì mỗi số hạng của tổng trên được dùng để đo độ tương tự giữa *x* với điểm *xi* tương ứng trong dữ liệu luyện tập. Như vậy, tác dụng của tổng trên chính là so sánh khoảng cách giữa điểm cần dự đoán với các điểm dữ liệu đã biết. Lưu ý là tập hợp các điểm *x* được ánh xạ vào một siêu phẳng có thể có độ phức tạp tùy ý trong không gian ban đầu, nên có thể phân tách các tập hợp thậm chí không lồi trong không gian ban đầu..

1. **Cơ sở lý thuyết**

SVM thực chất là một bài toán tối ưu, mục tiêu của thuật toán này là tìm được một không gian F và siêu phẳng quyết định f trên F sao cho sai số phân loại là thấp nhất.

Cho tập mẫu *D =* {*(xi, yi)| xi* ϵ **R***n , yi* ϵ {-1, 1}}(-1 biểu thị lớp I, 1 biểu thị lớp II). Ta có phương trình siêu phẳng chứa vector trong không gian:

+ *b = 0*

Đặt *f(* ) = *sign(* + *b) =*

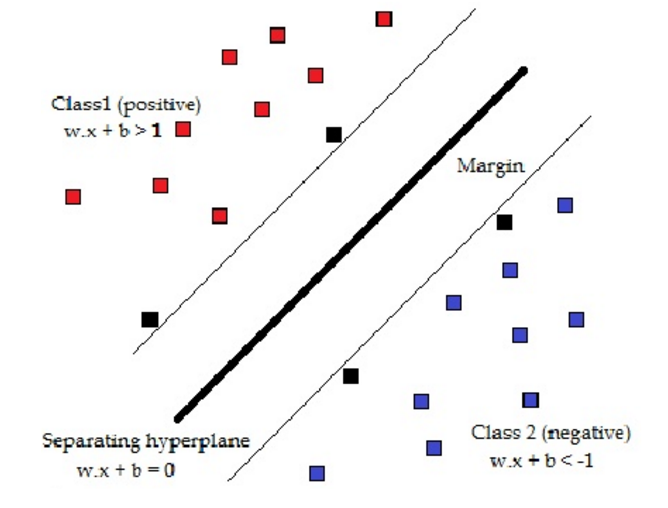
Như vậy, *f(* ) biểu diễn sự phân lớp của vào 2 lớp như đã nêu. Ta nói *yi* = +1 nếu thuộc lớp I và *yi* = -1 nếu thuộc lớp II

1. **Bài toán phân 2 lớp với SVM**

Bài toán đặt ra là: Xác định hàm phân lớp để phân lớp các mẫu trong tương lai, nghĩa là với một mẫu dữ liệu mới *x­I* thì cần phải xác định *x­I* được phân vào lớp +1 hay -1.

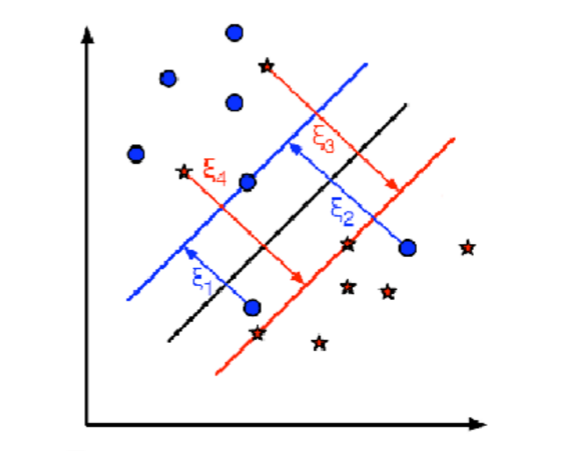
Ta xét 3 trường hợp, mỗi trường hợp sẽ có 1 bài toán tối ưu, giải được bài toán tối ưu đó ta sẽ tìm được siêu phẳng cần tìm.

* *Trường hợp 1:* Tập D có thể phân chia tuyến tính được mà không có nhiễu (tất cả các điểm được gán nhãn +1 thuộc về phía dương của siêu phẳng, tất cả các điểm được gán nhãn -1 thuộc về phía âm của siêu phẳng)



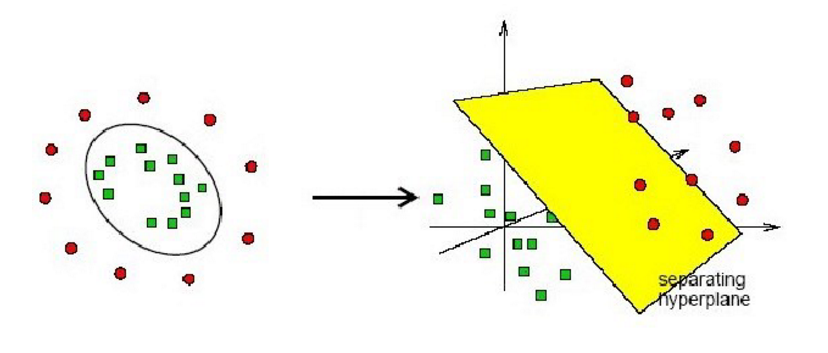
Hình 1.5 : Tập dữ liệu được phân chia tuyến tính

* *Trường hợp 2:* Tập dữ liệu có thể phân chia tuyến tính được nhưng có nhiễu. Trong trường hợp này, hầu hết các điểm đều được phân chia đúng bởi siêu phẳng. Tuy nhiên có một số điểm bị nhiễu, nghĩa là: Điểm có nhãn dương nhưng lại thuộc phía âm của siêu phẳng, và ngược lại, điểm có nhãn âm nhưng lại thuộc phía dương của siêu phẳng.



Hình 1.6: Tập dữ liệu phân chia tuyến tính nhưng có nhiễu

* *Trường hợp 3:* Tập dữ liệu D không thể phân chia tuyến tính được, ta sẽ ánh xạ các vector dữ liệu từ không gian *n* chiều vào một không gian *m* chiều (*m>n*), sao cho trong không gian *m* chiều, D có thể phân chia tuyến tính được.



Hình 1.7: Tập dữ liệu không thể phân chia tuyến tính

1. **Bài toán phân đa lớp với SVM**

Để phân đa lớp thì kĩ thuật SVM sẽ chia không gian dữ liệu thành 2 phần và tiếp tục với không gian đã được phân chia. Khi đó hàm quyết định phân dữ liệu vào lớp thứ *i* sẽ là:

Những phần tử *x* là support vector nếu thỏa mãn điều kiện:

Giả sử bài toán phân loại *k* lớp (*k* > 2), ta sẽ tiến hành *k*(*k* – 1)/2 lần phân lớp nhị phân sử dụng phương pháp SVM. Mỗi lớp sẽ tiến hành phân tách với *k*-1 lớp còn lại để xác định *k*-1 hàm phân tách.

Kỹ thuật phân đa lớp bằng phương pháp SVM hiện vẫn đang được tiếp tục nghiên cứu và phát triển.

1. **Các bước chính của phương pháp SVM**

* Tiền xử lí dữ liệu: Phương pháp SVM yêu cầu dữ liệu được diễn tả như các vector của các số thực. Như vậy nếu đầu vào chưa phải là số thực thì ta cần phải tìm cách chuyển chúng về dạng số của SVM. Tránh các số quá lớn, thường nên co giãn dữ liệu để chuyển về đoạn [-1, 1] hoặc [0, 1].
* Chọn hàm hạt nhân: Cần chọn hàm hạt nhân phù hợp tương ứng cho từng bài toán cụ thể để đạt được độ chính xác cao trong quá trình phân lớp.
* Thực hiện việc kiểm tra chéo để xác định các tham số cho ứng dụng
* Sử dụng các tham số cho việc huấn luyện tập mẫu
* Kiểm thử tập dữ liệu test.

1. **Ví dụ SVM trong phân loại hoa**

Con người qua quá trình lớn lên đã tự tích lũy cho mình những đặc điểm rất riêng của mỗi loài hoa. Ví dụ hoa loa kèn có hình giống cái kèn, hoa huệ màu trắng có mùi thơm đặc trưng, hoa hồng màu đỏ và có gai… Vậy tức là trong đầu ta đã có một *tập dữ liệu* về các loài hoa và nó chính là căn cứ để chúng ta nhận dạng một loài hoa khi chúng ta gặp phải. Nguyên tắc học của máy tính cũng tương tự như vậy. Ta cần phải cung cấp cho nó một *tập dữ liệu* và *huấn luyện* cho nó bằng một cách nào đó để nó có thể căn cứ vào đó mà đoán được một bông hoa mới thuộc vào loài hoa nào. Trong ví dụ này ta sẽ sử dụng tập dữ liệu hoa là *tập dữ liệu Iris*

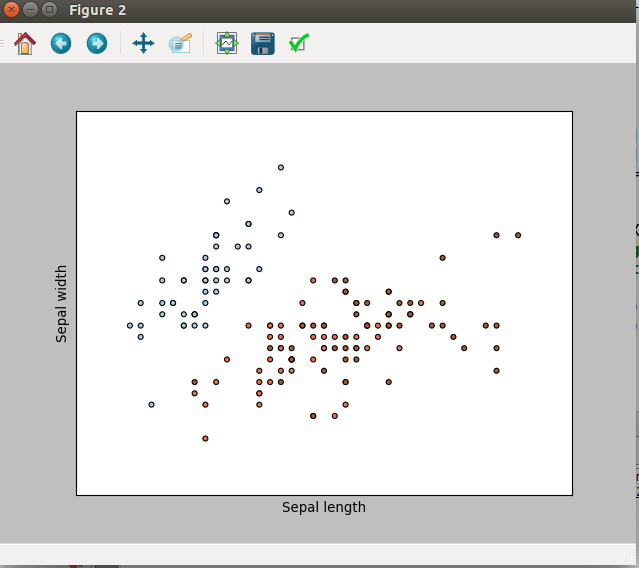
*Tập dữ liệu Iris* do [Ronald Fisher](https://en.wikipedia.org/wiki/Ronald_Fisher) thu thập và tổng hợp. Tập dữ liệu này gồm 50 mẫu về 3 loài hoa khác nhau của họ Iris là *(Iris setosa, Iris virginica và Iris versicolor)*



Hình 1.8: Ba loài hoa trong tập dữ liệu Iris

Với mỗi một mẫu hoa này Ronald Fisher thu thập bốn thuộc tính là *chiều dài*và *chiều rộng* của *đài hoa* và *cánh hoa* với đơn vị centimet. Để có thể sử dụng tập dữ liệu này chúng ta sẽ sử dụng thư viện *datasets*trong *sklearn.*

Trước tiên, ta sẽ biểu diễn tập dữ liệu trong đồ thị 2 chiều với 2 thuộc tính là *độ rộng* và *chiều cao* của đài hoa. Qua hàm xử lí vẽ đồ thị 2D trên matlab ta thu được kết quả như hình vẽ:

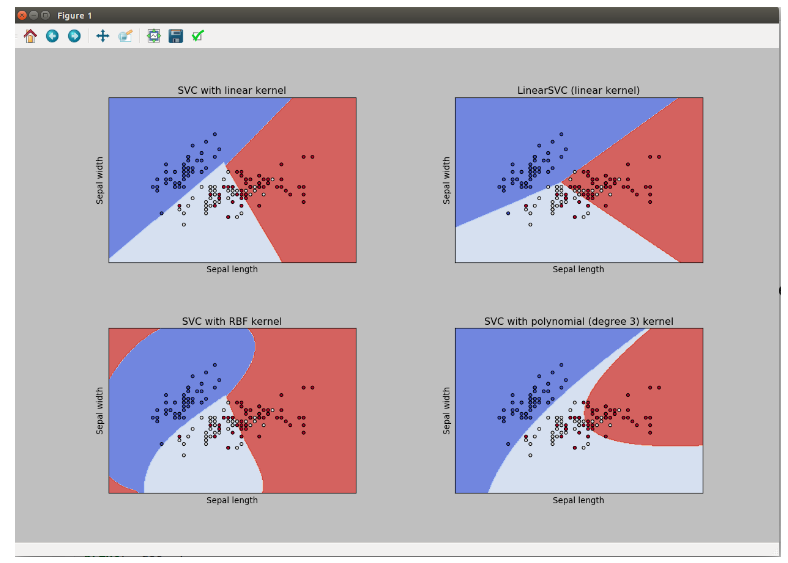


Hình 1.9: Biểu diễn tập dữ liệu bằng đồ thị 2D

Với tập dữ liệu **Iris** chúng ta cần phân loại các bông hoa thành 3 lớp dữ liệu. Sử dụng SVM với các phương pháp khác nhau sẽ cho hiệu quả phân lớp khác nhau. Cũng tương tự như trên, ta chỉ xem xét đến 2 thuộc tính đầu tiên của tập dữ liệu, tức là phân lớp trong không gian 2 chiều. Chúng ta sử dụng các Kernel khác nhau bao gồm:

* SVC with linear kernel
* LinearSVC (linear kernel)
* SVC with RBF kernel
* SVC with polynomial (degree 3) kernel

Sau khi huấn luyện dữ liệu và đẩy các mô hình thu được vào đồ thị ta thu được kết quả như hình dưới đây.



Hình 1.10: Đồ thị trên matlab ứng với các kernel khác nhau

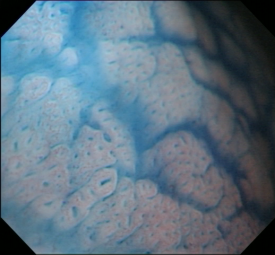
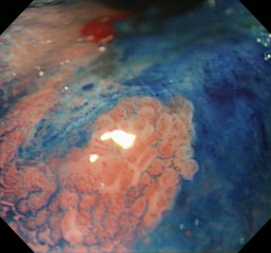
# PHƯƠNG PHÁP THỰC HIỆN

1. **Hệ thống thực hiện**

Quá trình được thực hiện trên phần mềm Matlab 2016a, cho phép chế độ xử lý song song với *Parallel Computing Toolbox Support.* Đồng thời, sử dụng hệ thống máy tính laptop với cấu hình CPU Intel Core i5-3437U và 4Gb Ram.

1. **Dataset**

Sử dụng tập dữ liệu hình ảnh nội soi “*Chromoendoscopy*” gồm 176 ảnh, mỗi ảnh có kích thước 518x481, trong đó có 56 ảnh mô bình thường, và 120 ảnh mô bất thường (96 ảnh mắc chứng metaplasia và 24 ảnh thuộc triệu chứng dysplasia) cùng với vùng ảnh quan tâm (Region Of Interest) tương ứng với mỗi ảnh, được xác định bởi bác sĩ.

a) Mô bình thường b) Mô bị nhiễm dysplasia c) Mô bị nhiễm metaplasia

Hình 2.1: Hình ảnh mô tả các loại mô khác nhau từ đường tiêu hóa đại diện cho bộ dữ liệu

Hình ảnh *Chromoendoscope* thu được bằng cách sử dụng một máy nội soi Olympus GIF-H180 tại Viện Ung thư Bồ Đào Nha (IPO) Porto, Bồ Đào Nha trong các nghiên cứu lâm sàng thông thường [4], [6].Ta chia tập dữ liệu thành 2 lớp ảnh là bình thường (Normal, gồm 56 ảnh) và bất bình thường (Abnormal, gồm 120 ảnh) để thực hiện quá trình phân loại

1. **Cross-validation và phương pháp Hout-out**

Cross-validation là một kỹ thuật kiểm tra độ chính xác của 1 máy học (Machine Learning) dựa trên một tập dữ liệu cho trước. Thay vì chỉ dùng một phần dữ liệu làm tập dữ liệu học thì cross-validation dùng toàn bộ dữ liệu để dạy cho máy. Cross-validation có 3 phương pháp phổ biến:

1. *Hold-out:* Phương pháp đơn giản nhất. Dữ liệu được chia một cách ngẫu nhiên thành một tập dữ liệu học và một tập dữ liệu kiểm tra. Dùng tập đầu tiên để dạy máy rồi dùng ngay tập còn lại để kiểm tra.
2. *K-fold:* đây là phương pháp nâng cấp của hold-out. Toàn bộ dữ liệu được chia thành K tập con. Quá trình học của máy có K lần. Trong mỗi lần, một tập con được dùng để kiểm tra và K-1 tập còn lại dùng để dạy.
3. *Leave-one-out:* Tương tự như K-Fold nhưng tối đa hóa số tập con (K = số dữ liệu).

Trong nghiên cứu này chúng em sử dụng phương pháp Hold-out. Mô hình của phương pháp cụ thể như sau:

Chia Dataset

Đào tạo bộ phân loại (Classifier)

Kiểm định bộ phân loại

Tính độ chính xác trung bình

Hình 2.2: Mô hình phương pháp Hold-out

Tập dữ liệu gốc được chia ngẫu thành 2 tập dữ liệu theo tỷ lệ *N = trainingSets : testingSets* , 1 tập là *trainingSets* được dùng để đào tạo “Bộ phân loại” – Classifier, và 1 tập là *testingSets* để kiểm định “Bộ phân loại”; thông thường tập *trainingSets* lớn hơn tập *testingSets* . Trong nghiên cứu, chúng em còn thực hiện kiểm định trên tập *trainingSets*.

Chúng em lặp lại quá trình trên 10 lần, và lấy giá trị độ chính xác cũng như tỷ lệ lỗi trung bình trên 10 lần lặp. Dựa vào đó đánh giá chất lượng cũng như độ chính xác “Bộ phân loại” đã được đào tạo.

1. **Đánh giá độ chính xác (trung bình) của mô hình phân lớp**

Đầu tiên ta làm quen với ma trận nhầm lẫn (Confusion matrix):

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Lớp dự đoán | |
| Abnormal | Normal |
| Lớp thực tế | Abnormal | a | b |
| Normal | c | d |

Ta có:

* a:TP(true positive) – mẫu mang nhãn Abnormal được phân lớp **đúng** vào lớp **Abnormal**.
* b:FN (false negative) – mẫu mang nhãn Abnormal bị phân lớp **sai** vào lớp **Normal**.
* c:FP (false positive) – mẫu mang nhãn Normal bị phân lớp **sai** vào lớp **Abnormal**.
* d:TN (true negative) – mẫu mang nhãn Normal được phân lớp **đúng** vào lớp **Normal**.

Từ đây, độ chính xác của mô hình được tính như sau:

Tỷ lệ lỗi của mô hình:

Error\_rate = 1 – Acc

Trong nghiên cứu này, độ chính xác (hoặc tỷ lệ lỗi) được tính trung bình của 10 lần lặp.

1. **Sơ đồ thực hiện**

Tiền xử lý

Tạo túi từ trực quan (BoW)

Đào tạo

“Bộ phân loại”

Kiểm định

“Bộ phân loại”

Tính độ chính xác trung bình

Hình 2.3: Sơ đồ thực hiện

1. **Các bước phân loại hình ảnh sử dụng Bag of features Words**

***Bước 1: Tiền xử lý***

Dữ liệu thô ban đầu (Dataset gốc) có chứa nhiễu do ánh sáng, sự chuyển động của đầu ống nội soi… cần phải loại bỏ và đồng thời cần phải phân vùng hình ảnh dựa vào vùng quan tâm ROI mà bác sĩ đã xác định.

Sơ đồ thuật toán

TIỀN XỬ LÝ

Đọc ảnh RGB

Đọc ảnh ROI

RGB sang ảnh xám

Đảo ngược bit nhị phân

Lọc thông thấp Gaussian (m)

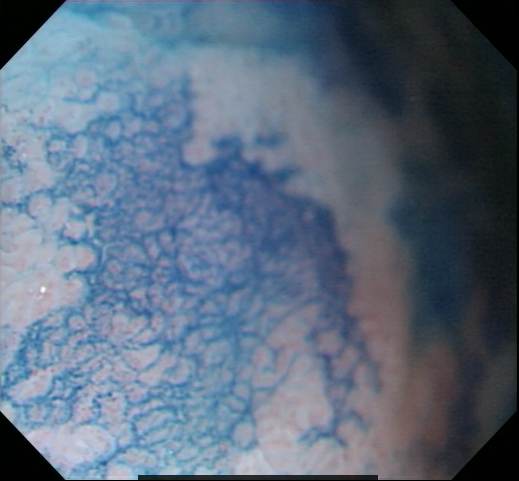
Ảnh nhị phân sang ảnh xám (n)

O = m - n

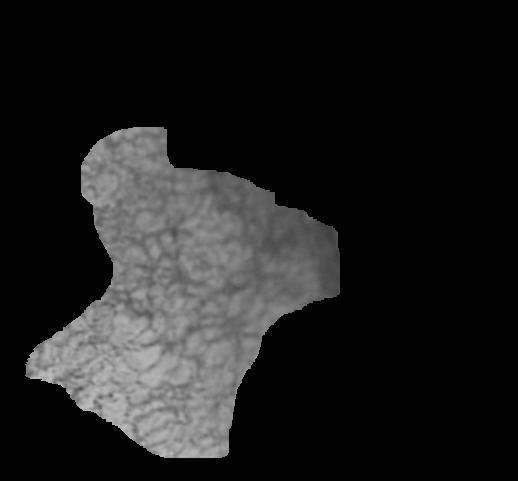
Ảnh đầu ra (O)

Hình 2.4: Tiền xử lý

Sau khi thực hiện theo sơ đồ trên ta được ảnh xám O cùng kích thước với ảnh đầu vào và đã được loại bỏ nhiễu.

Ảnh RBG Ảnh ROI



Ảnh O

Hình 2.5: Ảnh O thu được từ ảnh RBG và ROI

Sau khi xử lý trên từng cặp ảnh (RBG,ROI) ta thu được tập data mới chứa 120 ảnh bất thường và 56 ảnh bình thường.

Cân bằng số lượng ảnh mỗi lớp, nghĩa là mỗi lớp bình thường và bất thường sẽ có 56 ảnh, 56 ảnh bất thường sẽ được chọn ngẫu nhiên từ 120 ảnh. Và tập data mới sẽ có 112 ảnh gồm 2 lớp bình thường và bất thường.

Chia tập data mới thành 2 tập ảnh *trainingSets* và *testingSets* với tỷ lệ chia là   
N = 7:3, tương ứng với 2 lớp.

***Bước 2: Tạo túi từ trực quan (Túi từ tính năng – BoW)***

Đối tượng Bag of Features hay Bag of Visual Words (BoW) xác định các tính năng, hoặc các từ trực quan bằng cách sử dụng thuật toán phân cụm K-means trên các bộ mô tả tính năng được trích ra từ trainingSets. Thuật toán lặp đi lặp lại, nhóm các bộ mô tả thành các cụm riêng biệt. Các cụm kết quả nhỏ gọn và được phân tách bằng các đặc tính tương tự. Mỗi trung tâm cụm đại diện cho một tính năng, hoặc hình ảnh từ.

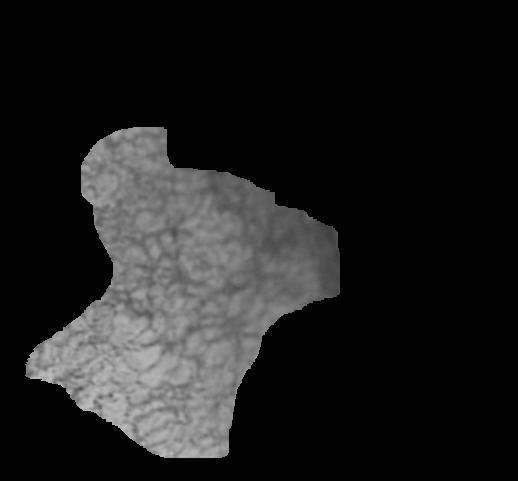
Đầu tiên, từ tập ảnh *trainingSets,* ta trích xuất tính năng của từng ảnh trong *trainingSets,* quá trình này gọi là Features extraction.

Mỗi tính năng sẽ được trích xuất dưới dạng vector tính năng (features vector), tập hợp các tính năng ta được một bảng tính năng tạo thành một ma trận có kích thước m x n , trong đó m gọi là số quan sát (Observations) và n là số chiều hay còn gọi là thuộc tính của 1 vector tính năng.

Các vector tính năng này đều có số chiều bằng nhau và bằng n. Để trích xuất được các vector tính năng, thì trước hết cần phải phát hiện các tính năng, quá trình này gọi là Features detection. Chúng em sử dụng 2 phương pháp để phát hiện tính năng gồm phương pháp ‘Grid’ và phương pháp sử dụng “Máy dò tính năng SURF” (*Speeded up robust features*).

Tăng tốc các tính năng mạnh mẽ *Speeded up robust features* (SURF) là một máy dò và mô tả tính năng địa phương được cấp bằng sáng chế. Nó có thể được sử dụng cho các tác vụ như nhận dạng đối tượng, đăng ký hình ảnh, phân loại hoặc tái thiết 3D. Khi sử dụng phương pháp này các tính năng sẽ tự động được trích xuất từ các điểm SURF (SURFPoints) với hệ thống tự nhận dạng tích hợp.

Đối với phương pháp ‘Grid’, các Features vector cũng được trích xuất từ các điểm SURF (SURFPoints) tuy nhiên vị trí điểm ta sẽ xác định bằng các điểm lưới ngẫu nhiên. Từ kích thước hình ảnh, ta xây dựng một lưới dùng để xác định các điểm trích xuất tính năng, gọi là *Extract point*.



GridStep

Hình 2.6: Ảnh **O** trong quá trình trích xuất hình ảnh với phương pháp Grid

Khoảng cách giữa các điểm được xác định bằng tham số là ‘*GridStep*’ có đơn vị là pixel. Giá trị GridStep càng nhỏ thì số điểm trích xuất càng nhiều, và ngược lại.

Tại mỗi điểm, tính năng sẽ được lấy ra theo các khối hình vuông gọi là các “*Patch*” với các kích thước khác nhau hay độ rộng khác nhau (32 pixel, 64 pixel, 96 pixel hoặc 128 pixel). Tất cả quá trình lựa chọn *Extract point* hay *Patch* đều được thực hiện ngẫu nhiên.

Sau khi thu thập các tính năng ta thu được một ma trận tính năng m x n hay còn gọi là bộ tính năng, mỗi hàng là một *Feature vector*.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | … | … | n |
| X1 | 1 | A1 | A2 | A3 | A4 | A5 | A6 | … | …. | An |
| X2 | 2 |  |  |  |  |  |  |  |  |  |
| X3 | 3 |  |  |  |  |  |  |  |  |  |
| X4 | 4 |  |  |  |  |  |  |  |  |  |
|  | …. | … | … | … |  |  |  | … | … |  |
| Xm | m |  |  |  |  |  |  | … | …. |  |

Hình 2.7: Ma trận tính năng

Các tính năng được trích xuất từ tập ảnh với số lượng rất lớn, lên đến vài trăm nghìn *Features vector*. Việc xử lý và tính toán với số lượng lớn tính năng như vậy, yêu cầu bộ nhớ và tốc độ xử lý cao, tùy nhiên thời gian càng dài tương ứng với số lượng tính năng càng lớn; hơn thế nữa độ chính xác không hẳn là cao với tập tính năng lớn. . Vì vậy, việc làm giảm số lượng *Features vector* là công việc cần thiết.

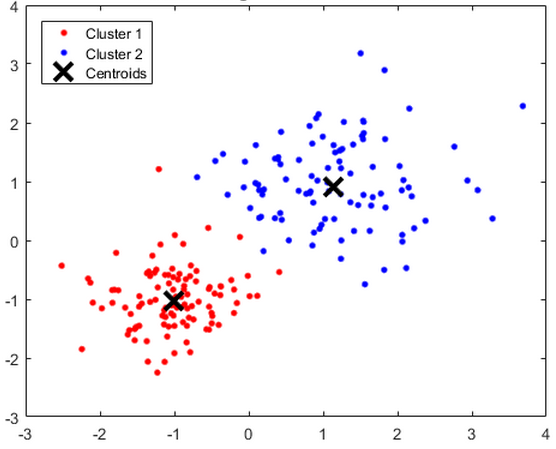
Đầu tiên, ta giảm số lượng tính năng bằng việc giảm số điểm trích xuất tính năng (SURFPoints) bằng cách sử dụng hàm “*selectStrongest*” trong Matlab. Hàm này sẽ trả về các điểm trích xuất có số liệu mạnh mẽ và đặc trưng nhất của ảnh. Trong thuật toán, ta có hệ số *StrongestFeatures* dùng để điều chỉnh điều này.

Hệ số *StrongestFeatures* là tỷ lệ giữa số tính năng đặc trưng, mạnh mẽ nhất với tổng số tính năng được trích xuất ra từ bộ dữ liệu. Khoảng giá trị nằm trong khoảng [0 1].

Tiếp tục giảm các tính năng trùng lặp để tăng độ chính xác, đồng thời làm giảm số lượng và thời gian tính toán bằng phương pháp phân cụm K-means tạo ra “Túi từ trực quan” (BoW).

Phân cụm K-means trong nghiên cứu này sử dụng thuật toán “Khoảng cách Euclide bình phương”, mỗi centroid là trung bình của các điểm trong cụm đó. Nghĩa là biến đổi ma trận tính năng thành một ma trận mới với số hàng mới là K, với K << m và số thuộc tính hay số chiều (n) giữ nguyên. Đồng thời, mỗi cụm sẽ có giá trị mới gọi là giá trị trọng tâm (centroid), đại diện cho các phần tử trong cụm.

Ví dụ: Ta có 3 *Feature vector* (X1,X3,X5) có tính chất tương tự nhau sẽ được phân vào cùng một cụm mới là Y1. Giá trị vector Y1 sẽ đặc trưng cho 3 *Feature vector* trên.

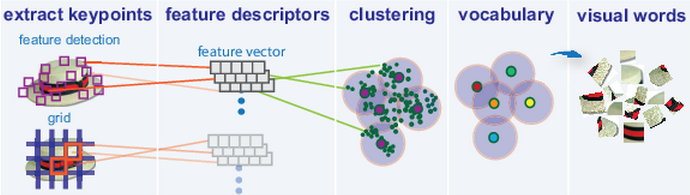


Hình 2.8: Minh họa phân cụm tập dữ liệu fisheriris trong Matlab

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | … | … | n |
| Y1 | 1 | B1 | B2 | B3 | B4 | B5 | B6 | … | …. | Bn |
| Y2 | 2 |  |  |  |  |  |  |  |  |  |
| Y3 | 3 |  |  |  |  |  |  |  |  |  |
| Y4 | 4 |  |  |  |  |  |  |  |  |  |
|  | …. | … | … | … |  |  |  | … | … |  |
| Yk | k |  |  |  |  |  |  | … | …. |  |

Hình 2.9: Ma trận tính năng sau khi phân cụm

Từ đây, ta có *k* tính năng tạo thành “Túi tính năng” hay “Túi từ trực quan”. Mỗi tính năng sẽ được gán nhãn “Abnormal” (bất thường) hoặc “Normal” (bình thường) dựa vào nguồn trích xuất là ảnh thuộc lớp bất thường hay bình thường.



Hình 2.10: Mô tả quá trình tạo “Túi từ trực quan” – BoW

*(tài liệu tham khảo [10])*

***Bước 3: Đào tạo “Bộ phân loại”***

Từ “Túi từ trực quan” và *trainingSets* , chúng em đào tạo ra một “Bộ phân loại” – Classifier, bằng cách sử dụng hàm “*trainImageCategoryClassifier*” trong Matlab. Hàm này trả về một “Bộ phân loại” (SVM – Support Vector Machine) mà chúng em đặt tên là “*categoryClassifier*”

***Bước 4: Kiểm định “Bộ phân loại” và tính tỷ lệ lỗi***

Đánh giá “Bộ phân loại” – Classifier sử dụng 2 tập ảnh *trainingSets* và *testingSets* , dựa vào “Ma trận nhầm lẫn” (Confusion matrix) (được đề cập trong phần IV). Từ “Ma trận nhầm lẫn” ta tính được độ chính xác trung bình (Average Accuracy) và tỷ lệ sai số trung bình.

Ta lấy giá trị trung bình trong 10 lần lặp, theo phương pháp Hold-out mà chúng em đã đề cập trong phần III.

1. **Các yếu tố tác động đến kết quả phân loại**

Dưới đây là các yếu tố tác động đến kết quả phân loại. Ta sử dụng các giá trị mặc định của các hàm Matlab để thực hiện với yếu tố đầu tiên là “kích thước hình ảnh”, các yếu tố tiếp theo được sử dụng giá trị tối ưu của các yếu tố trước để đánh giá.

1. **Thay đổi kích thước hình ảnh**

* Dataset không thay đổi kích thước

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal Normal  ----------------------------------  Abnormal | 0.97 0.03  Normal | 0.72 0.28  \* Độ chính xác trung bình: 0.63 | PREDICTED  KNOWN | Abnormal Normal  ----------------------------------  Abnormal | 0.94 0.06  Normal | 0.82 0.18  \* Độ chính xác trung bình: 0.56 |

* Dataset với kích thước ảnh giảm 25%

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_75 Normal\_75  -------------------------------------------  Abnormal\_75 | 0.97 0.03  Normal\_75 | 0.56 0.44  \* Độ chính xác trung bình: 0.71. | PREDICTED  KNOWN | Abnormal\_75 Normal\_75  -------------------------------------------  Abnormal\_75 | 1.00 0.00  Normal\_75 | 0.65 0.35  \* Độ chính xác trung bình: 0.68. |

* Dataset với kích thước ảnh giảm 50%

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.41 0.59  \* Độ chính xác trung bình: 0.79. | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.94 0.06  Normal\_50 | 0.41 0.59  \* Độ chính xác trung bình: 0.76 |

* Dataset với kích thước ảnh giảm 75%

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_25 Normal\_25  -------------------------------------------  Abnormal\_25 | 1.00 0.00  Normal\_25 | 0.21 0.79  \* Độ chính xác trung bình: 0.9 | PREDICTED  KNOWN | Abnormal\_25 Normal\_25  -------------------------------------------  Abnormal\_25 | 1.00 0.00  Normal\_25 | 0.47 0.53  \* Độ chính xác trung bình: 0.76 |

Từ những kết quả trên ta thấy:

* Đối với tập *trainingSets* độ chính xác trung bình cao nhất là 0.9 khi giảm kích thước ảnh đi 75% , và thấp nhất là 0.63 khi không giảm kích thước ảnh
* Đối với tập *testingSets* độ chính xác trung bình cao nhất là 0.76 khi giảm kích thước đi 50% hoặc 75% và thấp nhất là 0.56 khi không giảm kích thước ảnh
* Khi giảm kích thước ảnh ta có sai số thấp hơn khi so sánh với hình ảnh có kích thước đầy đủ

Như vậy, thông thường khi ta sử dụng *trainingSets* để đào tạo một “*Bộ phân loại*” - Classifier để phân loại thì tỷ lệ dự đoán chính xác tập *trainingSets* cao hơn *testingSets* . Độ chính xác càng cao chứng tỏ rằng mô hình của chúng ta sử dụng càng phù hợp với tập dataset sử dụng.

Thay đổi kích thước hình ảnh tới 50% hoặc thậm chí 25% kích thước của nó làm giảm nhiễu và giảm lỗi trung bình một cách đáng kể. Mặt khác, khi phân loại hình ảnh thay đổi kích cỡ bằng 25% kích thước ban đầu của nó, chúng em nhận được một lỗi của cùng một phạm vi (cao hơn một chút), tuy nhiên với phương pháp thống kê khác nhau, cho thấy quy mô này thay đổi kích thước là có hại: mặc dù hiện vật hình ảnh được làm mịn và chi phí tính toán giảm nhưng các thông tin liên quan trong các hình ảnh cũng sẽ bị mất, do đó ảnh hưởng đến việc phân loại.

1. **Ảnh hưởng bởi tỷ lệ *N = trainingSets : testingSets***

Chọn dataset với kích thước ảnh giảm 50% để đào tạo “*Bộ phân loại*” – Classifier, khi thay đổi tỷ lệ *N* = { 5:5, 6:4, 7:3, 8:2 } thì kết quả đạt được như sau:

* N = 5:5

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.96 0.04  Normal\_50 | 0.39 0.61  \* Độ chính xác trung bình: 0.79 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.43 0.57  \* Độ chính xác trung bình: 0.79 |

* N = 6:4

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.97 0.03  Normal\_50 | 0.44 0.56  \* Độ chính xác trung bình: 0.76 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.36 0.64  \* Độ chính xác trung bình: 0.82 |

* N = 7:3

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.41 0.59  \* Độ chính xác trung bình: 0.79 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.94 0.06  Normal\_50 | 0.41 0.59  \* Độ chính xác trung bình: 0.76 |

* N = 8:2

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.38 0.62  \* Độ chính xác trung bình: 0.81 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.45 0.55  \* Độ chính xác trung bình: 0.77 |

Khi thay đổi tỷ lệ *N* = { 5:5, 6:4, 7:3, 8:2 } sai số đối với tập *trainingSets* là 222 % , còn đối với tập *testingSets* là 213% và sai số trong các trường hợp khác sai lệch đáng kể.

1. **Kích thước Vocabulary**

*VocabularySize* là một số nguyên ( 2), chỉ số từ trực quan có trong túi (BoW). Nó tương ứng với số cụm K trong thuật toán K-means dùng để định lượng số từ trực quan.

Chọn dataset với kích thước ảnh giảm 50% để đào tạo “*Bộ phân loại*” – Classifier, chọn tỷ lệ N = 7:3. Khi thay đổi *VocabularySize* với các giá trị 200, 300, 400, 500 và 700 ta thu được kết quả.

* *VocabularySize* = 200

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.98 0.02  Normal\_50 | 0.36 0.64  \* Độ chính xác trung bình: 0.81 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.97 0.03  Normal\_50 | 0.33 0.67  \* Độ chính xác trung bình: 0.82 |

* *VocabularySize* = 300

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.44 0.56  \* Độ chính xác trung bình: 0.78 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.99 0.01  Normal\_50 | 0.31 0.69  \* Độ chính xác trung bình: 0.84 |

* *VocabularySize* = 400

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.44 0.56  \* Độ chính xác trung bình: 0.78 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.34 0.66  \* Độ chính xác trung bình: 0.83 |

* *VocabularySize* = 500

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 1.00 0.00  Normal\_50 | 0.41 0.59  \* Độ chính xác trung bình: 0.79 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.94 0.06  Normal\_50 | 0.41 0.59  \* Độ chính xác trung bình: 0.76 |

* *VocabularySize* = 700

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.97 0.03  Normal\_50 | 0.47 0.53  \* Độ chính xác trung bình: 0.75 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.98 0.02  Normal\_50 | 0.40 0.60  \* Độ chính xác trung bình: 0.79 |

Khi kích thước Vocabulary thay đổi thì độ chính xác cũng thay đổi, các trường hợp có sai số khác nhau không đáng kể, hiệu quả nhất tại 2 giá trị là 200 và 300, tuy nhiên *VocabularySize* càng lớn thì thời gian tính toán, xử lý càng nhiều. Vì vậy, tối ưu nhất là chọn *VocabularySize* = 200. Sai số khi dự đoán hai tập *trainingSets* và *testingSets* dùng “*Bộ phân loại*” – Classifier là khoảng 20%.

1. **Hệ số *StrongestFeatures***

Hệ số *StrongestFeatures* là tỷ lệ giữa số tính năng đặc trưng, mạnh mẽ nhất với tổng số tính năng được trích xuất ra từ bộ dữ liệu. Khoảng giá trị nằm trong khoảng [0 1].

* *StrongestFeatures* = 0.5

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.95 0.05  Normal\_50 | 0.33 0.67  \* Độ chính xác trung bình: 0.81 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.90 0.10  Normal\_50 | 0.28 0.72  \* Độ chính xác trung bình: 0.81 |

* *StrongestFeatures* = 0.6

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.98 0.02  Normal\_50 | 0.32 0.68  \* Độ chính xác trung bình: 0.83 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.97 0.03  Normal\_50 | 0.37 0.63  \* Độ chính xác trung bình: 0.80 |

* *StrongestFeatures* = 0.7

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.97 0.03  Normal\_50 | 0.37 0.63  \* Độ chính xác trung bình: 0.83 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.98 0.02  Normal\_50 | 0.34 0.66  \* Độ chính xác trung bình: 0.82 |

* *StrongestFeatures* = 0.8

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.98 0.02  Normal\_50 | 0.38 0.62  \* Độ chính xác trung bình: 0.80 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.98 0.02  Normal\_50 | 0.34 0.66  \* Độ chính xác trung bình: 0.82 |

Hệ số *StrongestFeatures* càng lớn thì số lượng vector tính năng (features vector) để phân cụm K-means càng lớn, có nghĩa là thời gian của quá trình xử lý tăng lên. Tuy nhiên khi số lượng vector tính năng giảm thì đồng nghĩa có thể mất đi một số tính năng quan trọng, mô tả hình ảnh. Do vậy cần phải chọn hệ số *StrongestFeatures* phù hợp để cân bằng giữa thời gian xử lý, đồng thời không mất đi những tính năng đặc trưng của dữ liệu. Hệ số *StrongestFeatures* tối ưu mà chúng em chọn là 0.7.

Dựa vào kết quả, ta cũng có thể thấy sai số trung bình cho cả 2 tập dữ liệu vào khoảng dưới 20%.

1. **Tham số *PointSelection***

Tham số *PointSelection* có 2 giá trị là 'Grid' hoặc 'Detector'. Khi đặt *PointSelection* là 'Detector', các điểm đặc trưng được chọn bằng cách sử dụng “Máy dò tính năng SURF”. Nếu không, các điểm được chọn trên một lưới thông thường với khoảng cách được xác định bởi 'GridStep'. .

Các kết quả có được ở các phần trên là khi chúng em sử dụng giá trị mặc định ‘Grid’. Khi sử dụng giá trị ‘Detector’ với các lựa chọn tối ưu ở trên ta thu được ma trận nhầm lẫn:

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.92 0.08  Normal\_50 | 0.04 0.96  \* Độ chính xác trung bình: 0.93 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.94 0.06  Normal\_50 | 0.03 0.97  \* Độ chính xác trung bình: 0.95 |

Ta thấy kết quả mang lại hiệu suất rất tốt cho tập dữ liệu ảnh sử dụng, sai số trung bình dưới 10%. Đồng thời tốc độ thực hiện nhanh lên rất nhiều so với lựa chọn ‘Grid’ (khoảng 10 lần) .

1. **Tham số *GridStep***

Khi ta đặt một lưới lên ảnh để xác định các vị trí để trích xuất tính năng, thì khoảng cách giữa các điểm được chọn trên lưới được xác định bởi *GridStep* (đơn vị là pixel)

Trong trường hợp giá trị của *PointSelection* được đặt là ‘Grid’ thì ta có thể điều chỉnh giá trị *GridStep*. Khi thay đổi giá trị *GridStep* lần lượt là 5, 8 và 11 pixel thì sai số trung bình vào khoảng 20%. Thời gian xử lý tương đối dài.

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.97 0.03  Normal\_50 | 0.37 0.63  \* Độ chính xác trung bình: 0.80 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.98 0.02  Normal\_50 | 0.38 0.62  \* Độ chính xác trung bình: 0.80 |

Bảng kiểm định với giá trị GridStep là 8 pixel

1. **Độ lệch chuẩn của bộ lọc Gaussian**

Bộ lọc Gaussian được chúng em sử dụng để giảm nhiễu, cũng như làm mịn dữ liệu ảnh đầu vào, khi thay đổi giá trị độ lệch chuẩn *sigma* với các giá trị {0,5; 1; 1,5; 2} thì sai số trung bình của 10 lần lặp thấp hơn 10%. Tuy nhiên khi giá trị sigma càng lớn thì có thể mất một số chi tiết quan trọng. Dựa trên các kết quả kiểm định ta lựa chọn giá trị tối ưu là 0.5

* Với *sigma* = 0.5

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.92 0.08  Normal\_50 | 0.04 0.96  \* Độ chính xác trung bình: 0.93 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.94 0.06  Normal\_50 | 0.03 0.97  \* Độ chính xác trung bình: 0.95 |

* Với *sigma* = 1

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.91 0.09  Normal\_50 | 0.03 0.97  \* Độ chính xác trung bình: 0.94 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.86 0.14  Normal\_50 | 0.02 0.98  \* Độ chính xác trung bình: 0.92 |

* Với *sigma* = 1.5

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.93 0.07  Normal\_50 | 0.03 0.97  \* Độ chính xác trung bình: 0.95 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.88 0.12  Normal\_50 | 0.04 0.96  \* Độ chính xác trung bình: 0.92 |

Với *sigma* = 2

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.91 0.09  Normal\_50 | 0.02 0.98  \* Độ chính xác trung bình: 0.94 | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.90 0.10  Normal\_50 | 0.05 0.95  \* Độ chính xác trung bình: 0.92 |

# KẾT QUẢ VÀ ĐÁNH GIÁ

Kết quả mô hình đã đem lại nhiều dấu hiệu tích cực trong nghiên cứu phân loại ảnh ung thư dạ dày. Đầu tiên là kết quả tóm lược với các tham số thay đổi:

* Kích thước ảnh

|  |  |  |
| --- | --- | --- |
| Giá trị | Độ chính xác trung bình (%) | |
| *trainingSets* | *testingSets* |
| Giữ nguyên kích thước | 63 | 56 |
| Giảm 25% | 71 | 68 |
| Giảm 50% | 79 | 76 |
| Giảm 75% | 90 | 76 |

* Tỷ lệ *N = trainingSets : testingSets*

|  |  |  |
| --- | --- | --- |
| Giá trị | Độ chính xác trung bình (%) | |
| *trainingSets* | *testingSets* |
| 5:5 | 79 | 79 |
| 6:4 | 76 | 82 |
| 7:3 | 79 | 76 |
| 8:2 | 81 | 77 |

* *VocabularySize*

|  |  |  |
| --- | --- | --- |
| Giá trị | Độ chính xác trung bình (%) | |
| *trainingSets* | *testingSets* |
| 200 | 81 | 82 |
| 300 | 78 | 84 |
| 400 | 78 | 83 |
| 500 | 79 | 76 |
| 700 | 75 | 79 |

* *StrongestFeatures*

|  |  |  |
| --- | --- | --- |
| Giá trị | Độ chính xác trung bình (%) | |
| *trainingSets* | *testingSets* |
| 0.5 | 81 | 81 |
| 0.6 | 83 | 80 |
| 0.7 | 83 | 82 |
| 0.8 | 80 | 82 |

* *PointSelection*

|  |  |  |
| --- | --- | --- |
| Giá trị | Độ chính xác trung bình (%) | |
| *trainingSets* | *testingSets* |
| ‘Detector’ | 93 | 95 |
| ‘Grid’  (với GirdStep = 8 pixel) | 80 | 80 |

* Độ lệch chuẩn sigma của bộ lọc Gaussian

|  |  |  |
| --- | --- | --- |
| Giá trị | Độ chính xác trung bình (%) | |
| *trainingSets* | *testingSets* |
| 0.5 | 93 | 95 |
| 1 | 94 | 92 |
| 1.5 | 95 | 92 |
| 2 | 94 | 92 |

Sau khi kiểm định với các tham số thay đổi, chúng em đã lựa chọn chế độ xử lý tối ưu:

* Giảm kích thước ảnh 50%
* Tỷ lệ *N = trainingSets : testingSets* = 7:3
* *VocabularySize* = 200
* *StrongestFeatures* = 0.7
* *PointSelection,* lựa chọn “Detector”
* Độ lệch chuẩn sigma của bộ lọc Gaussian, sigma = 0.5

Việc lựa chọn các tham số tối ưu phụ thuộc vào kết quả kiểm định 2 tập đó là *trainingSets* và *testingSets.*

Kiểm định *trainingSets* nhằm mục đích kiểm tra tính đúng đắn của việc đào tạo bộ phân loại, nghĩa là chúng ta sử dụng *trainingSets* để tạo nên “Túi từ trực quan” (Bow) và từ túi này ta đào tạo nên “Bộ phân loại” – Classifier. Như vậy khi sử dụng Classifier này kiểm tra lại tập *trainingSets* cho ra độ chính xác càng cao thì có nghĩa là quá trình đào tạo của ta là càng chuẩn xác.

Kiểm định *testingSets* nhằm mục đích kiểm tra độ chính xác của Classifier với tập dữ liệu mới, tức là các ảnh cần phân loại. Trong nhiều trường hợp chúng ta thấy, với cùng một “Bộ phân loại” thì độ chính xác đối với tập *trainingSets* là cao, nhưng đối với tập *testingSets* lại thấp và ngược lại. Chúng ta cần phải cân bằng 2 yếu tố này để vừa đảm bảo quá trình đào tạo là chuẩn xác và vừa đáp ứng được phân loại ảnh với độ chính xác cao.

Ngoài ra còn nhiều yếu tố khác để lựa chọn phương án tối ưu. Như dựa vào số lượng ảnh mà chúng ta có và cần phân loại. Trong thực tế, luôn có giới hạn về tốc độ xử lý cũng như bộ nhớ của máy tính nên số lượng ảnh trong *trainingSets* sẽ có một giới hạn nhất định thỏa mãn điều kiện trên, đồng thời đảm bảo thời gian tính toán. Như vậy ta luôn biết trước được số lượng ảnh trong *trainingSets* .

Thông thường, khi có một vị trí nghi ngờ ung thư thì bác sĩ thường chụp nhiều ảnh về vị trí đó với các góc độ, khoảng cách chụp… khác nhau, tạo nên tập dữ liệu ảnh về vị trí nghi ngờ (*testingSets* ).

Từ 2 điều trên ta luôn có thể tính toán được tỷ lệ N *= trainingSets : testingSets*, và số lượng ảnh trong *testingSets* luôn nhỏ hơn nhiều so với *trainingSets.* Dựa vào bảng kết quả độ chính xác khi thay đổi tỷ lệ N để chọn giá trị N sao cho phù hợp với bài toán thực tế.

Hơn nữa cũng cần phải cân bằng giữa tốc độ xử lý, và độ chính xác (tức là không làm mất đi quá nhiều chi tiết, tính năng đặc trưng trong ảnh).

Sau đây là kết quả kiểm định tập *trainingSets* và *testingSets* ở chế độ tối ưu:

|  |  |
| --- | --- |
| *trainingSets* | *testingSets* |
| PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.92 0.08  Normal\_50 | 0.06 0.94  \* Độ chính xác trung bình: 93% | PREDICTED  KNOWN | Abnormal\_50 Normal\_50  -------------------------------------------  Abnormal\_50 | 0.94 0.06  Normal\_50 | 0.03 0.97  \* Độ chính xác trung bình: 95% |

Từ những kết quả nghiên cứu, chúng ta đều có thể thấy phương pháp này đã đem lại hiệu suất cao với độ chính xác khoảng hơn 90% với tập dataset sử dụng, tương ứng với tỷ lệ dự đoán sai trong khoảng  (dưới 10%), kết quả tương tự và có phần tốt hơn với kết quả của các bài báo đã được công bố như “Recognition of Cancer using Random Forests as a Bag-of-Words Approach for Gastroenterology” [1], “Superpixel-based classification of gastric chromoendoscopy images” [2].

# TÀI LIỆU THAM KHẢO

[1] S. Francisco, M. T. Coimbra, and R. Gamelas Sousa, “Recognition of Cancer using a Bag-of-Words Random Forest for Gastroenterology,” 2015

[2] D. Boschetto and E. Grisan, “Superpixel-based classification of gastric chromoendoscopy images” , Conference: SPIE Conference on Medical Imaging, February 2017

[3] Farhan Riaz, Fernando Vilarino, Mario Dinis Ribeiro, and Miguel Coimbra, “Identifying Potentially Cancerous Tissues in Chromoendoscopy Images”, Conference: Pattern Recognition and Image Analysis - 5th Iberian Conference, IbPRIA 2011, Las Palmas de Gran Canaria, Spain, June 8-10, 2011. Proceedings

[4] Ribeiro, M.D.: Clinical, Endoscopic and Laboratorial Assessment of Patients with Associated Lesions to Gastric Adenocarcinoma. Faculdade de Medicina, Universidade do Porto, PhD Thesis (2005)

[5] Jing Zhang, Shi-Bin Guo, Zhi-Jun Duan, “Application of magnifying narrow-band imaging endoscopy for diagnosis of early gastric cancer and precancerous lesion”, in BMC Gastroenterology 11(1):135 · December 2011

[6] [https://aidasub-chromogastro.grand-challenge.org](https://aidasub-chromogastro.grand-challenge.org/)

[7] https://vi.wikipedia.org/wiki/Học\_máy

[8] <https://techtalk.vn/tat-tan-tat-moi-kien-thuc-co-ban-ve-machine-learning.html>

[9] <https://ongxuanhong.wordpress.com/2015/08/25/danh-gia-mo-hinh-model-evaluation/>

[10] <https://www.mathworks.com/help/vision/ug/image-classification-with-bag-of-visual-words.html?s_tid=srchtitle>

[11] <https://www.mathworks.com/help/vision/examples/image-category-classification-using-bag-of-features.html>

[12] [http://machinelearningcoban.com/general/2017/02/06/featureengineering/#](http://machinelearningcoban.com/general/2017/02/06/featureengineering/)

feature-scaling-and-normalization

[13][https://vi.wikipedia.org/wiki/M%C3%A1y\_vect%C6%A1\_h%E1%BB%97\_tr%E1%BB%A3](https://vi.wikipedia.org/wiki/Máy_vectơ_hỗ_trợ)

# PHỤ LỤC

Code tạo ảnh đầu ra O

function tien\_xu\_ly(hObject, eventdata, handles)

close all;

clear all;

clc;

n=28; % So luong anh

for count=1:n

% Doc anh goc

input=imread(['E:\Xu ly anh y te\PROJECT\Test\z\_Normal\CHGastro\_Normal\_',num2str(count),'.png']);

% Doc anh ROI

mask=imread(['E:\Xu ly anh y te\PROJECT\Test\z\_Normal\CHGastro\_Normal\_',num2str(count),'\_ROI.png']);

% Chuyen anh rgb sang anh xam

input = rgb2gray(input);

%-------------------------------

% Lay Vung ROI

%-------------------------------

% Loc thong thap gaussian

k=fspecial('gaussian',[3 3],0.5);

input=imfilter(input,k);

%-------------------------------

IM2 = imcomplement(mask); % Xu ly anh nhi phan ROI: dao nguoc bit

gray = 255\*uint8(IM2); % Chuyen anh nhi phan thanh anh xam

out=input-gray; % Lay vung anh can thiet

%------------------------------

out = imresize(out,0.5); % Giam kich thuoc anh

filename = (['E:\Xu ly anh y te\PROJECT\Data\_2\Normal\Anh\_',num2str(count),'.png']);

imwrite(out, filename ); % Luu anh dau ra

end

Code chương trình chính

function BagOfWords()

% So lan lap

k=10;

% Do chinh xac trung binh cua tap trainingSets

Average\_Accuracy\_train=0;

% Do chinh xac trung binh cua tap testingSets

Average\_Accuracy\_test=0;

tMatrix=zeros(2); % Ma tran nham lan trainingSets

vMatrix=zeros(2); % Ma tra nham lan testingSets

for i=1:k

%% Khoi tao data set

dataFolder = fullfile('E:\Xu ly anh y te\PROJECT', 'Data\_0.5');

imgSets = [ imageSet(fullfile(dataFolder, 'Abnormal\_50')), ...

imageSet(fullfile(dataFolder, 'Normal\_50')) ];

%% Chia data anh thanh cac tap bang nhau

% DataSet

% Xac dinh so luong anh it nhat trong 2 tap Abnormal va Normal

minSetCount = min([imgSets.Count]);

% Can bang so luong anh moi loai

imgSets = partition(imgSets, minSetCount, 'randomize');

% Hien thi so luong anh moi tap

[imgSets.Count]

%% Chuan bi 70% data anh de dao tao va 30% de kiem dinh

[trainingSets, testingSets] = partition(imgSets, 0.7, 'randomize');

%% Tao tui tinh nang

bag = bagOfFeatures(trainingSets,'VocabularySize',200,'StrongestFeatures',0.7,'Verbose',false,'PointSelection','Detector');

%% Dao tao bo phan loai - Classifier

categoryClassifier = trainImageCategoryClassifier(imgSets, bag);

%% Dung Classifier phan loai trainingSets

trainMatrix = evaluate(categoryClassifier, trainingSets);

%% Tinh do chinh xac trung binh khi phan loai trainingSets

Average\_Accuracy\_train=Average\_Accuracy\_train+mean(diag(trainMatrix));

tMatrix=tMatrix+trainMatrix;

%% Dung Classifier phan loai testingSets

testMatrix = evaluate(categoryClassifier, testingSets);

%% Tinh do chinh xac trung binh khi phan loai testingSets

Average\_Accuracy\_test=Average\_Accuracy\_test+mean(diag(testMatrix));

vMatrix=vMatrix+testMatrix;

end

% Tinh do chinh xac trung binh cua cac tap

Average\_Accuracy\_train=Average\_Accuracy\_train/k;

Average\_Accuracy\_test=Average\_Accuracy\_test/k;

% Tinh ma tran nham lan trung binh cua cac tap

tMatrix=tMatrix/k;

vMatrix=vMatrix/k;

% Luu ket qua vao file mat

save('Ketqua\_0.7\_0.5.mat','tMatrix');

save('Ketqua\_0.7\_0.5.mat','vMatrix','-append');

save('Ketqua\_0.7\_0.5.mat','Average\_Accuracy\_train','-append');

save('Ketqua\_0.7\_0.5.mat','Average\_Accuracy\_test','-append');