TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

**KHOA CÔNG NGHỆ THÔNG TIN**

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



**BÁO CÁO** **CHUYÊN ĐỀ 1**

**MÔ HÌNH SVM TRONG PHÁT HIỆN SỚM UNG THƯ PHỔI QUA ẢNH CHỤP CT**

**Sinh viên thực hiện:** **LỤC TẤN ANH**

**Lớp/Khóa: CNTT K19K**

**Giảng viên hướng dẫn: TS.TRẦN QUANG QUÝ**

**Thái Nguyên – 2025**

# **MỤC LỤC**

[MỤC LỤC 2](#_Toc194587868)

[MỤC LỤC ẢNH 5](#_Toc194587869)

[LỜI MỞ ĐẦU 6](#_Toc194587870)

[Chương 1: TỔNG QUAN 7](#_Toc194587871)

[1.1.Phát biểu bài toán 7](#_Toc194587872)

[1.2.Phân tích bài toán 7](#_Toc194587873)

[1.3.Phương pháp tiếp cận và xây dựng mô hình 8](#_Toc194587874)

[1.4.Huấn luyện, phân tích và đánh giá kết quả 9](#_Toc194587875)

[1.5.Thách thức và giải pháp ứng dụng 9](#_Toc194587876)

[Chương 2: MÔ TẢ TẬP DỮ LIỆU 12](#_Toc194587877)

[2.1.Thu thập và phần tích dữ liệu 12](#_Toc194587878)

[2.1.1.Tổng quan về tập dữ liệu 12](#_Toc194587879)

[2.1.2.Thu thập dữ liệu 12](#_Toc194587880)

[2.1.3.Phân tích dữ liệu 12](#_Toc194587881)

[2.1.4.Thống kê dữ liệu 13](#_Toc194587882)

[2.2.Tiền xử lý dữ liệu 15](#_Toc194587883)

[2.2.1.Xử lý ảnh cơ bản 15](#_Toc194587884)

[2.2.1.1.Kiểm tra cấu trúc thư mục 15](#_Toc194587885)

[2.2.1.2.Đọc ảnh và tiền xử lý gốc 16](#_Toc194587886)

[2.2.1.3.Tăng cường dữ liệu ( Data Augmentation ) 16](#_Toc194587887)

[2.2.2.Kiểm tra chất lượng ảnh 19](#_Toc194587888)

[Chương 3: MÔ HÌNH 21](#_Toc194587889)

[3.1.Nghiên cứu mô hình SVM vận dụng cho bài toán 21](#_Toc194587890)

[3.1.1.Giới thiệu 21](#_Toc194587891)

[3.1.2.Thuật toán 22](#_Toc194587892)

[3.1.3. Ưu điểm và nhược điểm khi áp dụng SVM 22](#_Toc194587893)

[3.1.4.Ứng dụng vào bài toán 23](#_Toc194587894)

[3.2.Xây dựng và cài đặt mô hình SVM 24](#_Toc194587895)

[3.2.1.Môi trường phát triển 24](#_Toc194587896)

[3.2.2.Các thư viện và công cụ hỗ trợ 24](#_Toc194587897)

[3.2.3.Xây dựng các hàm phụ trợ 26](#_Toc194587898)

[3.2.4.Xây dượng hàm main 28](#_Toc194587899)

[3.2.5.Trích xuất và hợp nhất đặc trưng 29](#_Toc194587900)

[3.2.5.1.Trích xuất đặc trưng HOG 29](#_Toc194587901)

[3.2.5.2.Trích xuất đặc trưng LBP 33](#_Toc194587902)

[3.2.6.StandardScaler và PCA 38](#_Toc194587903)

[3.2.6.1.Chuẩn hóa dữ liệu với StandardScaler 38](#_Toc194587904)

[3.2.6.2.Giảm chiều không gian với PCA 40](#_Toc194587905)

[3.2.7.Xây dựng mô hình SVM 41](#_Toc194587906)

[3.2.8.Huấn luyện mô hình 42](#_Toc194587907)

[Chương 4: KẾT QUẢ THỰC NGHIỆM VÀ HƯỚNG PHÁT TRIỂN 45](#_Toc194587908)

[4.1.Trình bài các độ đo áp dụng trong bài toán 45](#_Toc194587909)

[4.1.1.Accuracy 45](#_Toc194587910)

[4.1.2.Precision 45](#_Toc194587911)

[4.1.3.Recall 46](#_Toc194587912)

[4.1.4.F1-Score 46](#_Toc194587913)

[4.1.5.Support 47](#_Toc194587914)

[4.1.6.Macro Average 47](#_Toc194587915)

[4.1.7.Weighted Average 47](#_Toc194587916)

[4.2.Đánh giá mô hình sau train 48](#_Toc194587917)

[4.3.Điều chỉnh mô hình cài thiện hiệu năng của mô hình 49](#_Toc194587918)

[4.3.1.Tối ưu hóa tham số với Grid Search 49](#_Toc194587919)

[4.3.2. Điều Chỉnh Trọng Số Lớp custom\_weights 50](#_Toc194587920)

[4.4.Đánh giá mô hình trên dữ liệu mới 51](#_Toc194587921)

[4.4.1.Thực hiện việc test trên ảnh mới với nhãn là normal 51](#_Toc194587922)

[4.4.2.Thực hiện việc test trên ảnh mới với nhãn là Cancer 52](#_Toc194587923)

[KẾT LUẬN 53](#_Toc194587924)

[TÀI LIỆU THAM KHẢO 54](#_Toc194587925)

# **MỤC LỤC ẢNH**

[Hình 1: Ảnh chụp phổi bình thường với nhãn “ Normal case ” 11](#_Toc194569386)

[Hình 2: Ảnh chụp phổi ung thư với nhãn “ Malignant case ” 12](#_Toc194569387)

[Hình 3: Maximum-margin hyperplane và margins cho một SVM được huấn luyện với các mẫu từ hai lớp 18](#_Toc194569388)

[Hình 4: Công thức biểu diễn của siêu phẳng 19](#_Toc194569389)

[Hình 5: Công thức tính kernel 19](#_Toc194569390)

[Hình 6: Công thứ tính Gradient 26](#_Toc194569391)

[Hình 7: Công thức tính biên độ Gradient 26](#_Toc194569392)

[Hình 8: Công thức tính hướng Gradient 27](#_Toc194569393)

[Hình 10: Công thức tính StandardScaler 35](#_Toc194569394)

[Hình 11 : Công thức tính Accuracy 42](#_Toc194569395)

[Hình 12 : Công thức tính Precision 43](#_Toc194569396)

[Hình 13 : Công thức tính Recall 43](#_Toc194569397)

[Hình 14 : Công thức tính F1-Score 43](#_Toc194569398)

[Hình 15 : Công thức tính Macro Average 44](#_Toc194569399)

[Hình 16 : Công thức tính Weighted Average 44](#_Toc194569400)

[Hình 17. Kết quả đánh giá tập test sau khi train 45](#_Toc194569401)

[Hình 18: test trên tập dữ liệu mới mẫu 1 48](#_Toc194569402)

[Hình 19: test trên dữ liệu mới mẫu 2 49](#_Toc194569403)

# 

# **LỜI MỞ ĐẦU**

Ung thư phổi là một trong những căn bệnh nguy hiểm hàng đầu, có tỷ lệ tử vong cao do thường được phát hiện ở giai đoạn muộn. Tuy nhiên, nếu có thể chẩn đoán sớm thông qua các phương pháp hiện đại, cơ hội điều trị thành công sẽ tăng lên đáng kể. Trong thời đại công nghệ phát triển mạnh mẽ, trí tuệ nhân tạo (AI) đã mang lại nhiều giải pháp hỗ trợ y học, đặc biệt là trong lĩnh vực chẩn đoán hình ảnh. Một trong những thuật toán được ứng dụng phổ biến trong phân loại hình ảnh y khoa là Support Vector Machine (SVM), với khả năng phân tách dữ liệu hiệu quả và đưa ra dự đoán chính xác.

Trong quá trình thực hiện nghiên cứu này, em đã nhận được sự hướng dẫn tận tình từ thầy/cô [...], cũng như sự hỗ trợ của các bạn. Những kiến thức về xử lý ảnh, học máy và mô hình SVM mà em tiếp thu được không chỉ giúp hoàn thành đề tài mà còn là nền tảng vững chắc cho việc nghiên cứu sâu hơn trong tương lai. Em

Cuối cùng, em xin gửi lời cảm ơn chân thành đến thầy/cô đã tận tình hướng dẫn và hỗ trợ trong suốt quá trình thực hiện báo cáo này. Những ý kiến đóng góp quý báu của thầy/cô không chỉ giúp em hoàn thiện kiến thức chuyên môn mà còn rèn luyện kỹ năng nghiên cứu khoa học một cách bài bản và chuyên sâu hơn.

Xin chân thành cảm ơn!

# 

# **Chương 1: TỔNG QUAN**

## **1.1.Phát biểu bài toán**

Trong lĩnh vực chẩn đoán y học, việc áp dụng mô hình máy học để phát hiện sớm ung thư phổi qua hình ảnh CT đã được để ý và phát triển mạnh mẽ. Mô hình SVM (Support Vector Machine) được lựa chọn vì khả năng phân loại hiệu quả trong trường hợp số lượng đặc trưng lớn và dữ liệu không tuyến tính. Quy trình xử lý bao gồm các bước tiền xử lý ảnh, trích xuất đặc trưng, chuẩn hóa dữ liệu và cuối cùng là huấn luyện mô hình SVM.

Việc áp dụng mô hình SVM trong bài toán này không chỉ giúp cải thiện quy trình chẩn đoán mà còn mở ra hướng đi mới trong ứng dụng công nghệ học máy cho y học lâm sàng, góp phần giảm thiểu tỷ lệ tử vong và nâng cao chất lượng điều trị bệnh nhân.

## **1.2.Phân tích bài toán**

Bài toán được hình thành dựa trên yêu cầu tự động phân biệt giữa các vùng có khả năng là ung thư và các vùng bình thường trong ảnh CT. Điều này đòi hỏi quá trình xử lý ảnh đầu vào gồm các bước tiền xử lý, trích xuất đặc trưng và cuối cùng là sử dụng SVM để phân loại. Nhận thấy rằng tính chính xác và độ tin cậy của hệ thống phụ thuộc nhiều vào chất lượng đặc trưng trích xuất từ ảnh CT, cũng như tham số được lựa chọn cho mô hình SVM.

Trong quá trình phân tích, xác định một số yêu cầu chính sau:

* **Tiền xử lý:**

Mỗi ảnh CT được xử lý qua các kỹ thuật lọc và điều chỉnh cường độ nhằm giảm nhiễu và làm nổi bật các chi tiết liên quan đến cấu trúc phổi. Qua đó, đảm bảo rằng các đặc trưng thu được không bị ảnh hưởng bởi những yếu tố không mong muốn.

* **Trích xuất đặc trưng:**

sử dụng các phương pháp phân tích kết cấu và biểu diễn hình ảnh như các bộ lọc Gabor, các hàm Fourier… để tách lấy các đặc trưng có ý nghĩa. Việc lựa chọn đặc trưng dựa trên khả năng phân biệt cao giữa vùng tổn thương và vùng bình thường. Quá trình này không chỉ giúp giảm chiều dữ liệu mà còn tăng hiệu quả cho bước phân loại sau đó.

* **Huấn luyện mô hình với SVM:**

Sau khi thu được đặc trưng từ các ảnh CT, chia tập dữ liệu thành tập huấn luyện và tập kiểm tra. Việc chọn lựa các tham số của SVM, chẳng hạn như loại kernel (linear, RBF…) được thực hiện thông qua quá trình thử nghiệm và đánh giá chéo (cross-validation). Kết quả cho thấy, SVM có khả năng phân loại tốt nhờ vào việc tìm ra siêu phẳng tối ưu phân tách các lớp dữ liệu ngay cả khi dữ liệu không tuyến tính.

* **Đánh giá và cải tiến:**

Sử dụng các độ đo chỉ số như accuracy, Recall và F1-Score để đánh giá hiệu quả của mô hình. Ngoài ra, các kỹ thuật như tăng cường dữ liệu và tinh chỉnh tham số cũng được áp dụng để cải thiện hiệu suất phân loại. Qua thử nghiệm, thấy rằng mô hình SVM có tiềm năng cao trong phát hiện sớm ung thư phổi, tuy nhiên cần có thêm những nghiên cứu sâu hơn về phương pháp trích xuất đặc trưng phù hợp và tối ưu hóa tham số cho từng bộ dữ liệu cụ thể.

## **1.3.Phương pháp tiếp cận và xây dựng mô hình**

Trong quá trình triển khai bài toán, bước đầu tiên cần chú trọng đến chất lượng dữ liệu đầu vào. Nhận định rằng ảnh CT chứa nhiều thông tin cấu trúc quan trọng của phổi, nhưng cũng có những nhiễu không cần thiết. Do vậy, bước tiền xử lý bao gồm việc chuyển đổi ảnh về dạng grayscale, điều chỉnh kích thước về cùng một chuẩn (ví dụ: 512×512 pixel) và áp dụng các kỹ thuật cân bằng histogram để làm rõ các đặc trưng cấu trúc của vùng phổi.

Tiếp theo, các đặc trưng của hình học và kết cấu được trích xuất thông qua sử dụng các phương pháp như HOG ( Histogram of Oriented Gradient ) và LBP ( Local Binary Pattern ). HOG cho phép nắm bắt các đường biên và hình dạng của vùng phổi, trong khi LBP hỗ trợ phân tích kết cấu và mức độ tương phản của mô phổi. Việc kết hợp các đặc trưng này tạo thành một vector đặc trưng tổng hợp, mang lại khả năng biểu diễn mạnh mẽ các đặc điểm hình ảnh cần thiết cho quá trình phân loại.

Sau khi thu thập các vector đặc trưng, dữ liệu được chuẩn hóa và giảm chiều thông qua một số phương pháp như StandardScaler và Principal Component Analysis (PCA) nhằm loại bỏ nhiễu và giảm thiểu tính đa chiều không cần thiết. Điều này giúp tối ưu hóa hiệu quả của mô hình phân loại khi đối mặt với dữ liệu có số lượng đặc trưng lớn.

Cuối cùng, mô hình SVM được huấn luyện trên dữ liệu đã chuẩn bị. Các tham số của SVM được chỉnh sửa thông qua các kỹ thuật tối ưu hóa như Grid Search để đạt được kết quả phân loại tối ưu nhất. Kết quả từ mô hình cho thấy khả năng phân biệt giữa các vùng phổi bình thường và các dấu hiệu ung thư với độ chính xác cao, góp phần nâng cao hiệu quả trong công tác chẩn đoán sớm ung thư phổi từ hình ảnh CT.

Mô hình SVM ( Support Vector Machine ) được chọn làm công cụ phân loại vì khả năng xử lý ảnh hiệu quả những trường hợp dữ liệu không tuyến tính và đảm bảo biên phân chia rõ ràng giữa các lớp. Trong quá trình huấn luyện, các tham số của SVM được điều chỉnh cẩn thận, sử dụng các kỹ thuật tối ưu như Grid Search, để tìm ra tập hợp tham số phù hợp nhất nhằm đạt hiệu suất phân loại cao.

## **1.4.Huấn luyện, phân tích và đánh giá kết quả**

Khi so sánh kết quả qua các phép đánh giá, có thể thấy rằng việc lựa chọn các đặc trưng phù hợp đóng vai trò then chốt trong hiệu quả của mô hình. Một số yếu tố cần được lưu ý bao gồm:

* **Chất lượng dữ liệu:**

Kết quả huấn luyện phụ thuộc rất nhiều vào chất lượng của ảnh CT ban đầu. Các ảnh có nhiễu cao hoặc không được chuẩn hóa tốt có thể gây ra hiện tượng overfitting hoặc underfitting.

* **Lựa chọn tham số:**

Quá trình tối ưu hóa tham số qua grid search đã cho thấy rằng một số giá trị của C và gamma có tác động lớn đến khả năng phân loại. Việc lựa chọn sai tham số có thể dẫn đến mất cân bằng trong việc phát hiện đúng các trường hợp ung thư và giảm thiểu sai số.

* **Tăng cường dữ liệu:**

Trong tập dữ liệu, số lượng mẫu của vùng ung thư thường ít hơn so với vùng bình thường. Do đó, đã áp dụng kỹ thuật tăng cường dữ liệu nhằm làm giàu tập huấn luyện, giúp học được nhiều đặc trưng giữa các ảnh hơn từ các mẫu dương tính.

* **Khả năng tổng quát hóa:**

Mặc dù kết quả trên tập kiểm tra cho thấy mô hình có hiệu quả cao, vẫn cần thực hiện thêm các thử nghiệm trên các bộ dữ liệu từ nhiều nguồn khác nhau để đánh giá khả năng tổng quát hóa và ứng dụng lâm sàng của hệ thống.

## **1.5.Thách thức và giải pháp ứng dụng**

Trong quá trình triển khai mô hình SVM, sẽ gặp một số khó khăn đáng kể mà cần được giải quyết để đảm bảo hệ thống có thể phát hiện chính xác các dấu hiệu sớm của ung thư phổi:

* **Vấn đề về chất lượng ảnh và tiền xử lý**

Ảnh CT thường có nhiễu, độ tương phản không đồng đều và các biến đổi về cường độ, làm cho việc nhận diện đặc trưng trở nên phức tạp.

Giải pháp thực hiện:

* Áp dụng các bộ lọc như trung vị và kỹ thuật cân bằng độ sáng để giảm bớt nhiễu và làm rõ các chi tiết cấu trúc.
* Việc chuẩn hóa cường độ ảnh được thực hiện đồng bộ trên toàn bộ dữ liệu, giúp cho quá trình trích xuất đặc trưng trở nên hiệu quả hơn.
* **Lựa chọn và trích xuất đặc trưng chính xác**

Một trong những bước quan trọng là xác định được các đặc trưng phân biệt rõ giữa các vùng tổn thương và vùng bình thường. Nếu các đặc trưng không được trích xuất tốt, mô hình SVM sẽ gặp khó khăn trong việc tìm ra siêu phẳng phân chia.

Giải pháp thực hiện:

* Kết hợp nhiều kỹ thuật trích xuất đặc trưng, bao gồm các bộ lọc Gabor, biến đổi Fourier và phân tích kết cấu, nhằm khai thác đầy đủ thông tin từ ảnh CT.
* Qua các thí nghiệm ban đầu, đã tinh chỉnh lựa chọn đặc trưng để đảm bảo chỉ giữ lại những thông tin hữu ích, giảm thiểu sự dư thừa dữ liệu không cần thiết.
* **Mất cân bằng giữa các lớp**

Số lượng ảnh biểu hiện dấu hiệu ung thư phổi thường ít hơn ảnh bình thường, gây ra tình trạng lệch tỉ lệ học, khiến mô hình dễ bị thiên về lớp có nhiều mẫu.

Giải pháp thực hiện:

* Áp dụng các kỹ thuật tăng cường dữ liệu như xoay, phóng to và lật ảnh và tăng giảm tường phản để làm phong phú thêm tập dữ liệu các mẫu dương tính.
* Cùng với đó, điều chỉnh trọng số của từng lớp trong mô hình SVM nhằm đảm bảo cân bằng khi huấn luyện.
* **Tối ưu tham số và khả năng tổng quát hóa của mô hình**

Việc lựa chọn các tham số như C và gamma đóng vai trò quan trọng trong hiệu năng của SVM. Tham số không phù hợp có thể dẫn đến overfitting hoặc underfitting, ảnh hưởng đến khả năng ứng dụng của mô hình trong dữ liệu thực tế.

Giải pháp thực hiện:

* Triển khai phương pháp tìm kiếm lưới (grid search) kết hợp với k-fold cross-validation để tìm ra bộ tham số tối ưu, giúp cho mô hình vừa học tốt dữ liệu dùng để huấn luyện vừa có khả năng đánh giá tổng quát hóa trên dữ liệu mới.
* Hơn nữa, thực hiện các bài thử nghiệm trên nhiều nguồn dữ liệu khác nhau để đánh giá khả năng thích ứng của mô hình trong môi trường thực tế.

# **Chương 2: MÔ TẢ TẬP DỮ LIỆU**

## **2.1.Thu thập và phần tích dữ liệu**

### **2.1.1.Tổng quan về tập dữ liệu**

Toàn bộ ảnh CT được ẩn danh trước khi tiến hành phân tích. Quá trình xét duyệt cũng đã được Hội đồng của các trung tâm y tế liên quan thông qua, và không yêu cầu ký văn bản đồng ý riêng. Việc đánh dấu (label) cho các lát cắt CT do các bác sĩ chuyên khoa ung bướu và X-quang tại hai cơ sở trên thực hiện, đảm bảo tính chính xác khi phân loại theo 2 nhóm: bình thường và ác tính.

Dữ liệu thu thập được sử dụng để huấn luyện và đánh giá mô hình SVM, trong đó mỗi ảnh CT thể hiện cấu trúc phổi ở một lát cắt nhất định. Mục tiêu chính là trích xuất các đặc trưng quan trọng liên quan đến hình thái và kết cấu của mô phổi, sau đó áp dụng mô hình phân loại nhằm dự đoán xem ảnh thuộc nhóm bình thường hay có dấu hiệu ung thư. Sự phong phú về số lượng lát cắt và tính đa dạng của ca bệnh trong bộ dữ liệu góp phần hỗ trợ quá trình phát triển, giúp tăng khả năng tổng quát và độ tin cậy trong kết quả phân loại.

* Gồm 2 nhãn chính:

- Ảnh chụp phổi ung thư với nhãn “ Malignant case ”

- Ảnh chụp phổi bình thường với nhãn “ Normal case ”

### **2.1.2.Thu thập dữ liệu**

Đây là bộ dữ liệu được tìm và thu thập tại Cancer Imaging Archive. Bộ dữ liệu chứa 804 ảnh, tất cả ảnh CT ban đầu được lưu ở định dạng DICOM, sử dụng máy quét SOMATOM để chụp chiếu.

Quá trình tìm tòi và thu thập bộ dữ liệu, là quá trình thực hiệc việc tìm, xem xét và tải về các hình ảnh CT dưới dạng DICOM sau đó sẽ đưa toàn bộ vào App “ MicroDicom DICOM Viewer ” do Microsft phát triển để chuyển từ định dạng DICOM sang JPG để phù hợp với bài toán.

### **2.1.3.Phân tích dữ liệu**

Bộ dữ liệu có 2 thư mục gồm 2 nhãn chính:

* Normal cases : ảnh chụp CT phổi bình thường.
* Malignant cases : ảnh chụp CT phổi ung thư.

Tất cả ảnh CT ban đầu được lưu ở định dạng DICOM, sử dụng máy quét SOMATOM của hãng Siemens. Thông số quét bao gồm:

* Mức điện thế 120 kV
* Độ dày lát cắt 1 mm
* Cửa sổ quan sát (window width) trong khoảng 350 đến 1200 HU
* Tâm cửa sổ (window center) từ 50 đến 600.

Kỹ thuật chụp đòi hỏi người tham gia giữ hơi thở ở thời điểm hít sâu để giảm thiểu biến dạng hình ảnh.

Đặc điểm nhận diện của các ca bệnh rất đa dạng, bao gồm nhiều độ tuổi, giới tính, ngành nghề và địa phương sinh sống, chủ yếu tập trung ở các tỉnh thuộc khu vực trung tâm của Iraq như Baghdad, Wasit, Diyala, Salahuddin và Babylon.

### **2.1.4.Thống kê dữ liệu**

Bộ dữ liệu bao gồm 804 ảnh CT thu thập từ 110 ca bệnh, trong đó phân thành 2 nhóm chính: bình thường (normal) và ác tính (malignant). Cụ thể, có 40 ca được chẩn đoán ác tính, và 55 ca bình thường. Mỗi ca bệnh được chụp với nhiều lát cắt khác nhau, số lượng có thể dao động từ 80 đến 200 lát, giúp cung cấp góc nhìn đa dạng về cấu trúc phổi

Bộ dữ liệu được chia làm 2 thư mục chứ các ảnh được đánh nhãn để phần biệt đâu là ảnh chụp phổi bình thường và đâu là ảnh chụp phổi bị ung thư là Malignant cases và Normal cases

* **Thư mục “ Normal cases ”**

Thư mục chứa các ảnh chụp CT phổi của người bình thường, chứ 402 ảnh của nhiều bệnh nhân khác nhau với nhiều lớp cắt để có thể nhìn và hiểu được ảnh chụp CT phổi bình thường

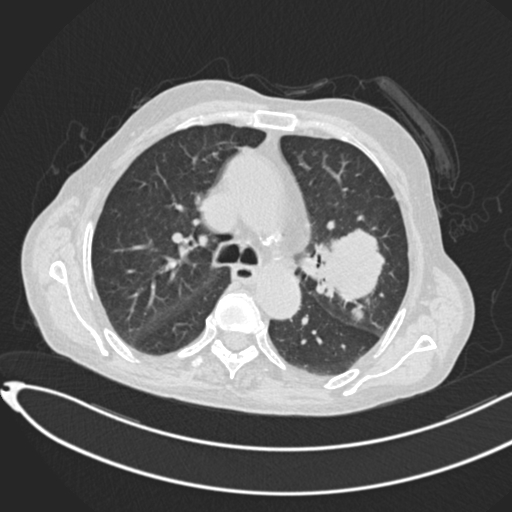


Hình 1: Ảnh chụp phổi bình thường với nhãn “ Normal case ”

Hình ảnh nhãn Normal case: Lát cắt CT của một trường hợp bình thường, không xuất hiện nốt bất thường hoặc khối u rõ rệt. Vùng nhu mô phổi có độ đậm tương đối đồng nhất, ranh giới giữa các khoang khí được thể hiện rõ ràng.

* **Thư mục “ Normal cases ”**

Thư mục chứa các ảnh chụp CT phổi của người bị ung thư, chứ 402 ảnh của nhiều bệnh nhân khác nhau với nhiều lớp cắt để có thể nhìn và hiểu được ảnh chụp CT phổi bị ung thư



Hình 2: Ảnh chụp phổi ung thư với nhãn “ Malignant case ”

Hình ảnh nhãn Malignant case: Lát cắt CT của một trường hợp ung thư phổi ác tính, có thể quan sát được sự khác biệt về số lượng mô tại vùng tổn thương. Khu vực bất thường này tương phản khá rõ so với phần nhu mô phổi xung quanh, gợi ý sự hiện diện của khối u.

## **2.2.Tiền xử lý dữ liệu**

### **2.2.1.Xử lý ảnh cơ bản**

Quá trình tiền xử lý dữ liệu được thiết kế để đảm bảo chất lượng đầu vào và tăng hiệu quả huấn luyện mô hình. Các bước chính bao gồm xử lý ảnh thô, tăng cường dữ liệu, kiểm tra tính nhất quán và chuẩn hóa đặc trưng.

Hàm load\_dataset được xây dựng nhằm mục đích xử lý dữ liệu hình ảnh CT dạng grayscale, kèm theo các bước tiền xử lý khác và tăng cường dữ liệu.

#### **2.2.1.1.Kiểm tra cấu trúc thư mục**

Dữ liệu đầu vào phải được phân loại thành hai thư mục con: normal (ảnh không bệnh lý) và cancer (ảnh có dấu hiệu ung thư). Hàm tự động kiểm tra tính hợp lệ của đường dẫn để tránh lỗi khi thiếu dữ liệu:

Code trích dẫn:

**for class\_name in ["normal", "cancer"]:**

**class\_dir = os.path.join(data\_dir, class\_name)**

**if not os.path.exists(class\_dir):**

**raise ValueError(f"Thiếu thư mục: {class\_dir}")**

Giải Thích:

* Lặp qua 2 lớp dữ liệu cần xử lý. Mỗi vòng lặp ứng với một lớp: class\_name = "normal": Dữ liệu ảnh phổi bình thường và class\_name = "cancer": Dữ liệu ảnh phổi có khối u
* Ghép đường dẫn thư mục gốc data\_dir với tên lớp để tạo đường dẫn đầy đủ.
* Kiểm tra xem thư mục của từng lớp có tồn tại không. Nếu thiếu một trong hai thư mục, chương trình sẽ dừng ngay lập tức và thông báo lỗi chi tiết.

#### **2.2.1.2.Đọc ảnh và tiền xử lý gốc**

Đầu vào, ảnh được chuyển sang thang độ xám để giảm độ phức tạp tính toán, đồng thời tập trung vào cấu trúc hình ảnh thay vì thông tin màu sắc:

Code trích dẫn:

**img = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)**

**validate\_image(img)**

**img = cv2.resize(img, img\_size)**

**img = img / 255.0**

Mỗi ảnh được xử lý qua các bước:

* Đọc ảnh grayscale: Giảm kích thước dữ liệu so với ảnh màu RGB, phù hợp với đặc thù hình ảnh X-quang.
* Resize về kích thước chuẩn: Đảm bảo tất cả ảnh đầu vào có cùng kích thước (mặc định 512x512 pixel), phù hợp hơn với đầu vào của mô hình.
* Chuẩn hóa giá trị pixel: Chuyển giá trị pixel từ [0, 255] về [0, 1] để tăng tốc độ hội tụ của thuật toán gradient descent.

#### **2.2.1.3.Tăng cường dữ liệu ( Data Augmentation )**

Khi tham số augment được kích hoạt, ba phương pháp biến đổi dữ liệu được áp dụng để mở rộng tập huấn luyện, cụ thể như sau:

* **Lật Anh theo trục dọc**

Phương pháp này tạo ra ảnh phản chiếu của ảnh gốc, mô phỏng góc chụp đối xứng thường gặp trong hình ảnh y tế. Đoạn mã thực hiện tăng cường dữ liệu (data augmentation) bằng cách tạo phiên bản ảnh bị lật ngang (gương) để mở rộng tập dữ liệu huấn luyện, giúp cải thiện khả năng tổng quát hóa của mô hình.

Code trích dẫn:

**flipped = cv2.flip(img, 1)**

**images.append(flipped)**

Giải thích:

* Hàm flip từ thư viện OpenCV dùng để lật ảnh. Tham số 1 quy định hướng lật theo trục dọc (trái ↔ phải).
* Thêm ảnh đã lật vào danh sách images chứa dữ liệu huấn luyện. Tăng số lượng mẫu huấn luyện mà không cần thu thập thêm dữ liệu thật.
* **Xoay ảnh với góc nhỏ**

Thực hiện tăng cường dữ liệu bằng cách tạo ra các phiên bản ảnh xoay nhẹ từ ảnh gốc, giúp cải thiện chất lượng huấn luyện mô hình. Ảnh được xoay quanh tâm với hai góc nghiêng (±5 độ) để mô phỏng sai lệch do vị trí đặt máy hoặc cử động của bệnh nhân.

Code trích dẫn:

**(h, w) = img.shape**

**center = (w // 2, h // 2)**

**for angle in [-5, 5]:**

**M = cv2.getRotationMatrix2D(center, angle, 1.0)**

**rotated = cv2.warpAffine(img, M, (w, h))**

**images.append(rotated)**

**labels.append(0 if class\_name == "normal" else 1)**

Giải thích:

* Dòng (h, w) = img.shape lấy chiều cao (h) và chiều rộng (w) của ảnh đầu vào. Tiếp theo, center = (w // 2, h // 2) tính toán tọa độ trung tâm của ảnh làm điểm xoay. Ví dụ: ảnh có kích thước 512x512 pixel sẽ có tâm tại (256, 256).
* Vòng lặp for angle in [-5, 5] thực hiện xoay ảnh theo hai góc: -5 độ (xoay ngược chiều kim đồng hồ) và 5 độ (xoay cùng chiều kim đồng hồ). Việc chọn góc nhỏ (±5°) giúp:

+ Mô phỏng sai lệch tự nhiên khi chụp CT (bệnh nhân cử động nhẹ)

+ Bảo toàn cấu trúc giải phẫu phổi, tránh biến dạng ảnh quá mức

+ Tăng gấp đôi số lượng mẫu huấn luyện từ một ảnh gốc

* Hàm cv2.getRotationMatrix2D(center, angle, 1.0) sinh ma trận biến đổi affine với:

+ center:Điểm xoay cố định ở trung tâm ảnh

+ angle: Góc xoay được chọn từ vòng lặp,

+ 1.0: Tỷ lệ co giãn (giữ nguyên kích thước ảnh)

* Dòng rotated = cv2.warpAffine(img, M, (w, h)) thực hiện biến đổi ảnh gốc bằng ma trận M, đảm bảo ảnh xoay giữ nguyên kích thước ban đầu (w, h)
* Sau khi xoay, ảnh mới (rotated) được thêm vào danh sách images cùng nhãn tương ứng qua labels.append(). Ví dụ: Một ảnh "cancer" gốc sẽ sinh ra hai ảnh xoay cùng nhãn "cancer", giúp cân bằng dữ liệu khi lớp "cancer" thường ít mẫu hơn lớp "normal".
* **Điều Chỉnh Độ Tương Phản và Sáng (Window Width/Level)**

Trong quá trình xử lý ảnh, việc mô phỏng các điều kiện chụp ảnh khác nhau giúp mô hình học máy thích nghi tốt hơn với dữ liệu thực tế. Kỹ thuật mô phỏng thay đổi thiết lập cửa sổ tương phản (WW/WL) trên máy CT, giúp mô hình thích nghi với các thiết bị khác nhau. Đoạn mã dưới đây minh họa cách tạo biến thể ảnh bằng cách điều chỉnh độ tương phản (contrast) và độ sáng (brightness), hai tham số quan trọng trong kỹ thuật Window Width/Level (W/L) thường dùng để phân tích ảnh CT.

Code trích dẫn:

**for \_ in range(2):**

**alpha = np.random.uniform(0.5, 1.5)**

**beta = np.random.uniform(-0.3, 0.3)**

**augmented\_img = img \* alpha + beta**

**augmented\_img = np.clip(augmented\_img, 0, 1)**

**images.append(augmented\_img)**

**labels.append(0 if class\_name == "normal" else 1)**

Giải thích:

* Vòng lặp tạo biến thể, tạo 2 biến thể đạt cân bằng giữa hiệu quả và thời gian xử lý. Tạo quá nhiều biến thể có thể gây nhiễu dữ liệu.
* alpha (0.5–1.5): Giảm/ tăng độ chênh lệch giữa pixel tối và sáng. Khi α < 1, ảnh tối hơn; α > 1 làm nổi bật chi tiết.
* beta (-0.3–0.3): Dịch chuyển toàn bộ histogram pixel sang trái (âm) hoặc phải (dương), mô phỏng thay đổi độ phơi sáng.
* np.clip: Đảm bảo giá trị pixel sau biến đổi nằm trong khoảng [0, 1], tránh hiện tượng "cắt cụt" thông tin do vượt ngưỡng.
* Quản lý nhãn sau Augmentation: mỗi ảnh tăng cường được gán nhãn giống ảnh gốc, duy trì tính nhất quán giữa dữ liệu và nhãn

### **2.2.2.Kiểm tra chất lượng ảnh**

Hàm validate\_image đóng vai trò quan trọng trong việc đảm bảo tính toàn vẹn của dữ liệu trước khi xử lý. Quy trình kiểm tra bao gồm hai bước chính:

* **Phát hiện ảnh đọc lỗi**

Trường hợp ảnh không tồn tại hoặc định dạng không hỗ trợ sẽ được xử lý ngay để tránh làm gián đoạn quá trình tiền xử lý.

Code trích dẫn:

**if img is None:**

**raise ValueError("Không thể đọc ảnh")**

Giải thích:

* img is None: Hàm cv2.imread() trả về None nếu không đọc được ảnh. Câu lệnh này kiểm tra trạng thái đó.
* raise ValueError(): Dừng chương trình và hiển thị thông báo lỗi có ý nghĩa thay vì để lỗi phát sinh ở bước xử lý tiếp theo (ví dụ: trích xuất đặc trưng từ ảnh rỗng).
* **Đánh giá độ tương phản**

Độ lệch chuẩn (standard deviation) của giá trị pixel được sử dụng để phát hiện ảnh có độ tương phản thấp, thường gặp trong ảnh hỏng hoặc ảnh thuần màu.

Code trích dấn:

**if img.std() < 5:**

**raise ValueError("Ảnh có độ tương phản quá thấp")**

Giải thích:

* Độ lệch chuẩn đo mức độ biến thiên của pixel. Giá trị càng thấp, sự khác biệt giữa pixel sáng/tối càng nhỏ.
* Ngưỡng 5: Được xác định thực nghiệm từ tập dữ liệu ảnh, loại bỏ các ảnh có histogram co cụm quanh một giá trị (ví dụ: ảnh đen thuần hoặc trắng thuần).

# 

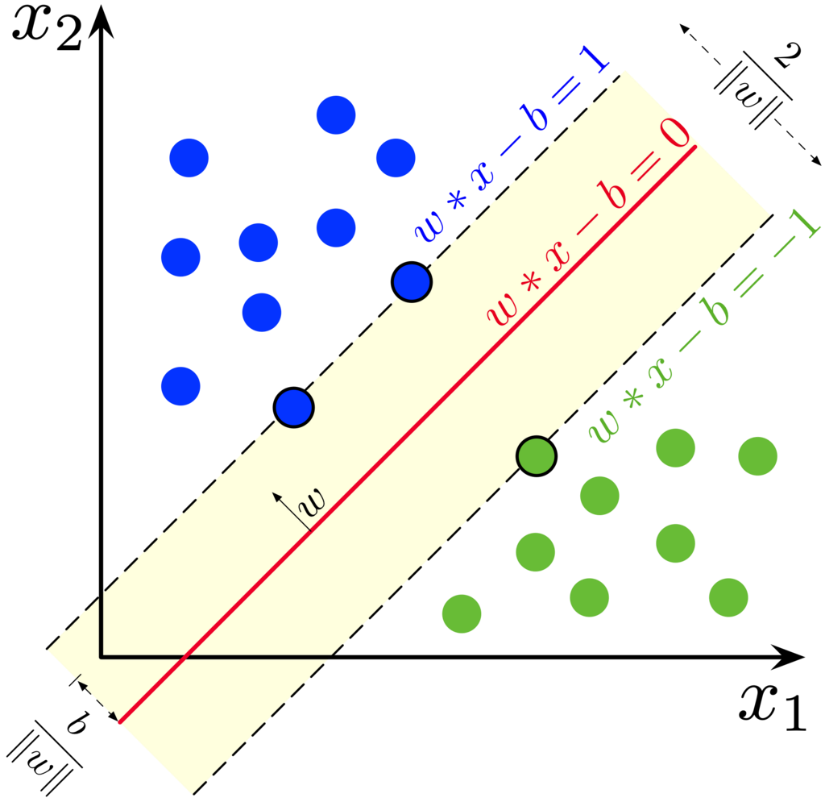
# **Chương 3: MÔ HÌNH**

## **3.1.Nghiên cứu mô hình SVM vận dụng cho bài toán**

### **3.1.1.Giới thiệu**

Máy hỗ trợ vectơ (SVM) thực hiện việc tạo ra một hay nhiều siêu phẳng trong không gian có nhiều chiều (hoặc thậm chí vô hạn chiều) để giải quyết các bài toán phân loại, hồi quy và các nhiệm vụ khác. Ý tưởng chính là càng đẩy các siêu phẳng xa các điểm dữ liệu từ các lớp khác nhau (điều này gọi là "biên" càng rộng), thì khả năng tổng quát hóa của thuật toán sẽ càng tốt, tức là sai số giảm.

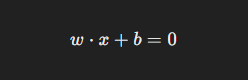
Trong nhiều trường hợp, việc phân chia tuyến tính các lớp dữ liệu trong không gian ban đầu là không thể. Do đó, thường cần ánh xạ các điểm dữ liệu từ không gian ban đầu vào một không gian mới có số chiều cao hơn để dễ dàng phân tách. Để đảm bảo tính hiệu quả trong quá trình tính toán, ánh xạ này được thiết kế sao cho việc tính tích vô hướng giữa các vectơ ở không gian mới có thể thực hiện nhanh chóng chỉ dựa trên tọa độ ban đầu.



Hình 3: Maximum-margin hyperplane và margins cho một SVM được huấn luyện với các mẫu từ hai lớp

### **3.1.2.Thuật toán**

Để thực hiện việc phân loại, SVM tìm cách xác định một siêu phẳng tối ưu sao cho khoảng cách từ các điểm dữ liệu đến siêu phẳng này đạt mức lớn nhất. Công thức biểu diễn của siêu phẳng có thể được trình bày như sau:



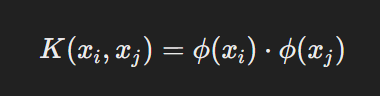
Hình 4: Công thức biểu diễn của siêu phẳng

Trong đó:

* **w** là vector trọng số của siêu phẳng.
* **x** là vector đặc trưng của điểm dữ liệu.
* **b** là độ dịch chuyển của siêu phẳng.

Để tối ưu hóa siêu phẳng (hyperplane), SVM sử dụng thuật toán tối thiểu hóa hàm mất mát, với mục tiêu tìm ra một giá trị w và b sao cho khoảng cách giữa các điểm dữ liệu gần siêu phẳng nhất.

Khi dữ liệu không thể phân loại tuyến tính, SVM sẽ áp dụng các hàm kernel như Radial Basis Function (RBF) hay Polynomial để chuyển đổi không gian dữ liệu và tìm siêu phẳng phân tách trong không gian mới. Mô hình này có thể biểu diễn như sau:



Hình 5: Công thức tính kernel

Trong đó:

* **K(xi,xj)** là hàm kernel
* **ϕ(xi)** là hàm ánh xạ dữ liệu vào không gian cao chiều.

### **3.1.3. Ưu điểm và nhược điểm khi áp dụng SVM**

Trong suốt quá trình nghiên cứu, nhận diện được một số ưu điểm cũng như nhược điểm của mô hình SVM trong ứng dụng bài toán:

* **Ưu điểm:**

SVM hoạt động trong không gian đặc trưng có kích thước lớn tốt, nhờ vào khả năng giảm thiểu ảnh hưởng của nhiễu.

Sử dụng các kernel giúp linh hoạt trong việc xử lý các mối quan hệ phi tuyến, rất phù hợp với dữ liệu hình ảnh phức tạp.

Khả năng tối ưu tham số thông qua các kỹ thuật tìm kiếm lưới giúp mô hình đạt hiệu năng tốt trên tập dữ liệu huấn luyện và kiểm tra.

* **Nhược điểm:**

SVM đòi hỏi quá trình lựa chọn tham số cẩn thận, nếu không sẽ dẫn đến overfitting hoặc underfitting, đặc biệt khi dữ liệu huấn luyện không đủ đa dạng.

Việc xử lý dữ liệu ảnh CT đòi hỏi quá trình trích xuất đặc trưng và tiền xử lý ảnh khá phức tạp, do đó yêu cầu thời gian và công sức nghiên cứu ban đầu cao.

Khi số lượng mẫu dữ liệu lớn, thời gian huấn luyện SVM có thể tăng lên đáng kể, do đó cần có biện pháp xử lý tối ưu về mặt tính toán.

### **3.1.4.Ứng dụng vào bài toán**

Bắt đầu bằng việc xây dựng một quy trình từ khâu chuẩn bị dữ liệu cho đến triển khai dự đoán trên ảnh CT. Quy trình này gồm các bước tiền xử lý, trích xuất đặc trưng và huấn luyện mô hình. Sau đây là cách áp dụng SVM vào bài toán:

* **Tiền xử lý ảnh và trích xuất đặc trưng:**

Ảnh CT ban đầu thường có nhiều biến đổi về chất lượng như nhiễu và độ tương phản không đồng nhất. Thực hiện các bước tiền xử lý như thay đổi kích thước, chuẩn hóa và tăng cường dữ liệu nhằm đảm bảo đầu vào đồng nhất. Sau đó, trích xuất các đặc trưng nổi bật từ ảnh bằng phương pháp HOG và LBP. Những đặc trưng này giúp mô hình SVM có thể phân biệt được những đặc điểm nhỏ của tổn thương, đặc biệt khi dấu hiệu ung thư xuất hiện ở giai đoạn sớm.

* **Huấn luyện và tối ưu SVM:**

Dựa trên vector đặc trưng thu được, chia dữ liệu thành tập huấn luyện và tập kiểm tra, sau đó áp dụng phương pháp tìm kiếm lưới (grid search) kết hợp với cross-validation để lựa chọn tham số tối ưu cho SVM (như C và gamma). Quy trình này đảm bảo rằng mô hình không chỉ đạt được độ chính xác cao trên tập huấn luyện mà còn có khả năng tổng quát tốt trên dữ liệu mới.

Thử nghiệm với nhiều kernel khác nhau và nhận thấy kernel RBF cho kết quả ổn định nhất đối với dữ liệu ảnh CT có đặc trưng phi tuyến.

* **Triển khai dự đoán và hỗ trợ chẩn đoán:**

Sau khi huấn luyện, mô hình SVM được tích hợp vào một ứng dụng dự đoán tự động. Ứng dụng này cho phép người dùng tải lên ảnh CT để nhận kết quả dự đoán ngay lập tức. Mỗi ảnh sẽ được xử lý qua các bước tiền xử lý, trích xuất đặc trưng, chuẩn hóa và giảm chiều dữ liệu qua PCA trước khi SVM dự đoán nhãn "Normal" hay "Cancer".

Việc ứng dụng mô hình SVM vào quy trình chẩn đoán không nhằm thay thế hoàn toàn bác sĩ, mà đóng vai trò hỗ trợ, cung cấp thêm thông tin dự báo nhanh chóng giúp rút ngắn thời gian chẩn đoán ban đầu. Điều này có ý nghĩa đặc biệt trong bối cảnh ung thư phổi, khi phát hiện sớm góp phần cải thiện đáng kể hiệu quả điều trị.

* **Đánh giá và cải tiến liên tục:**

Tiến hành điều chỉnh với các tham số và cải tiến thuật toán tiền xử lý nhằm nâng cao độ tin cậy và khả năng nhận dạng của mô hình. Các bài thử nghiệm trên dữ liệu thực tế cho thấy mô hình SVM có khả năng phát hiện những dấu hiệu sớm của ung thư phổi với độ chính xác khá cao, mặc dù vẫn cần bổ sung thêm các mẫu dữ liệu đa dạng để mở rộng khả năng tổng quát của hệ thống.

Qua việc áp dụng SVM vào bài toán, nhận thấy rằng việc kết hợp các bước tiền xử lý tỉ mỉ và trích xuất đặc trưng hiệu quả đã tạo nền tảng vững chắc cho quá trình huấn luyện mô hình. Hệ thống hỗ trợ chẩn đoán dựa trên SVM không chỉ mang lại tốc độ dự đoán nhanh mà còn giúp làm rõ những khu vực có khả năng bất thường trong ảnh CT, từ đó hỗ trợ đưa ra chuẩn đoán lâm sàng kịp thời.

## **3.2.Xây dựng và cài đặt mô hình SVM**

### **3.2.1.Môi trường phát triển**

Sử dụng ngôn ngữ lập trình Python cho toàn bộ dự án vì tính linh hoạt và cộng đồng hỗ trợ mạnh mẽ. Python cho phép tích hợp dễ dàng các thư viện chuyên dụng cho xử lý ảnh và học máy, qua đó giúp xây dựng hệ thống hiệu quả. Ngoài ra, việc sử dụng các công cụ dòng lệnh (Command Line Interface) thông qua thư viện argparse giúp tùy biến cấu hình của ứng dụng một cách linh động.

### **3.2.2.Các thư viện và công cụ hỗ trợ**

* **OpenCV (opencv-python)**

Thư viện này là nền tảng trong việc đọc, xử lý và biến đổi ảnh. Sử dụng OpenCV để chuyển ảnh màu sang ảnh grayscale, thay đổi kích thước, cũng như thực hiện các thao tác làm mịn và chuẩn hóa dữ liệu ảnh. Việc chuẩn bị dữ liệu qua OpenCV giúp đảm bảo rằng các ảnh đầu vào có định dạng và kích thước nhất quán, qua đó tạo điều kiện cho các bước xử lý sau.

* **scikit-learn**

Đây là thư viện cốt lõi cho việc xây dựng mô hình học máy. Sử dụng scikit-learn để cài đặt mô hình SVM (SVC), cũng như các công cụ tiền xử lý như StandardScaler và PCA. Ngoài ra, scikit-learn còn cung cấp các phương pháp chia dữ liệu (train\_test\_split, GroupShuffleSplit) và công cụ tối ưu tham số (GridSearchCV). Việc này không chỉ đảm bảo mô hình được huấn luyện hiệu quả mà còn giúp kiểm soát hiện tượng overfitting qua quá trình cross-validation.

* **scikit-image**

Để trích xuất đặc trưng từ ảnh, dựa vào scikit-image với các hàm như hog và local\_binary\_pattern. Các đặc trưng này đóng vai trò quan trọng trong việc chuyển đổi thông tin trực quan của ảnh CT thành các vector số học, tạo nên cơ sở cho mô hình SVM phân loại vùng ung thư và vùng bình thường. Đã tinh chỉnh các tham số của hàm hog (như số lượng orientations, kích thước cell, block) để nắm bắt được các đặc trưng vi mô của ảnh.

* **joblib**

Sau khi huấn luyện, việc lưu trữ và tái sử dụng mô hình là bước không thể thiếu. Sử dụng joblib để lưu mô hình SVM, các đối tượng tiền xử lý như StandardScaler và PCA dưới dạng file nhị phân. Điều này cho phép sử dụng lại các đối tượng này trong các phiên dự đoán mới mà không phải huấn luyện lại từ đầu, giúp tiết kiệm thời gian và đảm bảo tính nhất quán trong kết quả.

* **tqdm**

Trong quá trình xử lý dữ liệu, đặc biệt là khi duyệt qua số lượng lớn các ảnh, sử dụng tqdm để hiển thị progress bar. Điều này giúp theo dõi tiến trình thực thi của các vòng lặp một cách trực quan, từ đó kịp thời phát hiện và xử lý các vấn đề phát sinh.

* **argparse**

Thư viện này được dùng để xử lý tham số dòng lệnh, tạo điều kiện cho việc cấu hình các tùy chọn chạy như bật tối ưu tham số, tăng cường dữ liệu hay thực hiện dự đoán trên ảnh mới. Nhờ vào argparse, có thể chạy chương trình với các cấu hình khác nhau mà không cần thay đổi mã nguồn.

* **matplotlib**

Để trực quan hóa kết quả xử lý và kiểm tra chất lượng dữ liệu, dùng matplotlib. Thư viện này giúp hiển thị mẫu ảnh, cũng như các đặc trưng trích xuất như bản đồ HOG và LBP, qua đó cung cấp cái nhìn trực quan về quá trình tiền xử lý và hỗ trợ đánh giá hiệu quả của mô hình.

* **imagehash**

Trong dự án, việc kiểm tra dữ liệu trùng lặp là rất cần thiết để đảm bảo chất lượng tập dữ liệu. Sử dụng imagehash để tính toán giá trị băm của mỗi ảnh, từ đó phát hiện ra các ảnh lặp lại, giúp dọn dẹp dữ liệu và đảm bảo tính đa dạng trong tập huấn luyện.

* **cài đặt tất cả các gói cần thiết:**

pip install opencv-python scikit-learn scikit-image joblib tqdm argparse matplotlib imagehash

Nhờ vào việc tích hợp chặt chẽ các thư viện này, có thể đảm bảo quy trình xử lý dữ liệu từ đầu đến cuối diễn ra mượt mà, ổn định và dễ bảo trì. Hơn nữa, môi trường phát triển với Python và các thư viện mở rộng đã giúp tối ưu hóa hiệu quả của mô hình SVM, tạo nền tảng vững chắc cho việc hỗ trợ chẩn đoán lâm sàng ung thư phổi qua ảnh CT

### **3.2.3.Xây dựng các hàm phụ trợ**

Việc xây dụng các hàm phụ trợ qua được thực hiện qua 2 thành phần quan trọng trong hệ thống gồm quản lý log và xử lý tham số dòng lệnh được triển khai như sau:

* **Thiết lập hệ thống Logging**

Để theo dõi quá trình hoạt động của chương trình, ghi lại các sự kiện quan trọng, và hỗ trợ việc gỡ lỗi khi cần thiết, đã triển khai hàm setup\_logging(). Hàm này cấu hình hệ thống logging của Python, cho phép chương trình ghi lại thông tin ra console và vào file log.

Hiển thị log thời gian thực trên terminal để debug. Ví dụ: Khi chạy huấn luyện mô hình, log sẽ hiển thị tiến trình xử lý từng batch dữ liệu, cảnh báo lỗi nếu có.

Lưu trữ lịch sử log phục vụ phân tích lỗi sau này. File app.log ghi lại toàn bộ thông tin từ lúc khởi động đến khi kết thúc chương trình, hỗ trợ kiểm tra nguyên nhân crash hoặc hiệu suất.

Code trích dẫn:

**def setup\_logging():**

**logging.basicConfig(**

**level=logging.INFO,**

**format="%(asctime)s - %(levelname)s - %(message)s", handlers=[**

**logging.StreamHandler(),**

**logging.FileHandler("app.log", encoding="utf-8")**

**]**

**)**

Giải thích:

* logging.basicConfig hàm chính để cấu hình logging.
* Thiết lập mức độ logging là INFO. Điều này có nghĩa là chương trình sẽ ghi lại các thông tin ở mức INFO và các mức độ quan trọng hơn như WARNING, ERROR, và CRITICAL. Các thông tin ở mức DEBUG sẽ bị bỏ qua.
* Định dạng cho mỗi dòng log

+ “ %(asctime)s ” sẽ hiển thị thời gian ghi log,

+ “ %(levelname)s ” là mức độ log

+ “ %(message)s ” là nội dung thông báo log.

+ Dấu "-" được sử dụng để phân tách các thành phần này, giúp log dễ đọc hơn.

* handlers=[...]: Xác định các handlers để xử lý các bản ghi log. Trong trường hợp này, sử dụng hai handlers:

+ logging.StreamHandler(): Handler này in thông tin log ra console (standard output). Điều này hữu ích để theo dõi hoạt động của chương trình trực tiếp khi chạy.

+ logging.FileHandler("app.log", encoding="utf-8"): Handler này ghi thông tin log vào file có tên "app.log". encoding="utf-8" đảm bảo rằng file log hỗ trợ các ký tự Unicode, tránh lỗi khi ghi log bằng tiếng Việt hoặc các ngôn ngữ khác.

Với cấu hình logging này, khi chương trình chạy, các thông tin quan trọng như bắt đầu xử lý dữ liệu, huấn luyện mô hình, đánh giá mô hình, và các thông báo lỗi (nếu có) sẽ được ghi lại đồng thời trên console và trong file "app.log". Điều này giúp dễ dàng theo dõi tiến trình và kiểm tra lại lịch sử hoạt động của chương trình sau này.

* **Xử lý tham số dòng lệnh**

Để chương trình trở nên linh hoạt và dễ dàng tùy chỉnh khi chạy, sử dụng thư viện argparse để xây dựng hàm parse\_arguments(). Hàm này cho phép người dùng truyền các tham số cấu hình cho chương trình thông qua dòng lệnh khi khởi chạy.

parser : Tạo một đối tượng ArgumentParser. description được sử dụng để hiển thị mô tả ngắn gọn về chương trình khi người dùng chạy chương trình với --help hoặc -h.

Sau đó, sử dụng parser.add\_argument(...) để định nghĩa các tham số dòng lệnh mà chương trình chấp nhận. Mỗi add\_argument định nghĩa một tham số:

* grid\_search: Kích hoạt tối ưu tham số SVM .
* visualize\_features : Hiển thị HOG & LBP features.
* augment : Bật tăng cường dữ liệu (Data Augmentation).
* custom\_weights : Sử dụng class weights tùy chỉnh cho SVM.
* group\_split : Phân chia dữ liệu theo patient bằng GroupShuffleSplit.
* detailed : Sử dụng chế độ trích xuất đặc trưng chi tiết hơn.
* predict : Đường dẫn ảnh hoặc thư mục cần dự đoán.

return parser.parse\_args(): Hàm này phân tích các tham số dòng lệnh mà người dùng đã nhập và trả về một đối tượng Namespace chứa các tham số và giá trị tương ứng. Đối tượng này sau đó được gán cho biến args trong hàm main() để sử dụng các tham số cấu hình trong suốt chương trình.

### **3.2.4.Xây dượng hàm main**

* **Trong trường hợp có tham số (--predict)**

Chương trình sẽ chuyển sang chế độ inference: các file model, scaler và PCA đã được lưu được tải về, sau đó tiến hành dự đoán trên ảnh đơn lẻ hoặc toàn bộ ảnh trong thư mục.

* **Nếu không có tham số (--predict)**

Chương trình sẽ tiến hành xử lý dữ liệu huấn luyện. Dữ liệu được tải từ thư mục dataset với các lớp “normal” và “cancer”. Quá trình tiền xử lý bao gồm việc đọc ảnh ở chế độ grayscale, resize về kích thước chuẩn (512×512) và chuẩn hóa giá trị pixel. Tùy chọn tăng cường dữ liệu (augmentation) cũng được cung cấp để mở rộng tập dữ liệu qua các biến đổi như lật ảnh, xoay và điều chỉnh độ sáng/tương phản.

### **3.2.5.Trích xuất và hợp nhất đặc trưng**

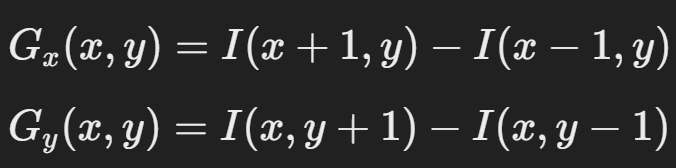
#### **3.2.5.1.Trích xuất đặc trưng HOG**

* **Giới thiệu**

Histogram of Oriented Gradients (HOG) là một phương pháp trích xuất đặc trưng phổ biến trong thị giác máy tính, đặc biệt hiệu quả trong các bài toán nhận dạng đối tượng như người, xe cộ. Khác với các kỹ thuật dựa trên texture hay màu sắc, HOG tập trung vào phân tích hình dạng thông qua sự thay đổi cường độ pixel, mô phỏng cách não người cảm nhận biên và hướng vật thể.

* **Thuật toán**

Tính Gradient của Ảnh: Để tính gradient của ảnh, sử dụng bộ lọc Sobel. Công thức tính cho mỗi điểm ảnh (x, y) được xác định như sau:



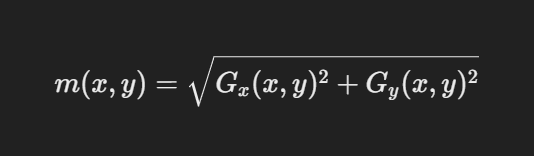
Hình 6: Công thứ tính Gradient

Trong đó,

* I(x,y) là giá trị cường độ của điểm ảnh tại vị trí (x, y).

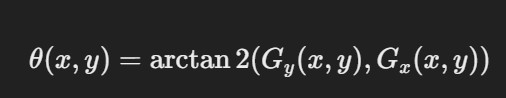
Tính Biên Độ và Hướng Gradient: Sau khi có các thành phần Gx và Gy, biên độ và hướng gradient được tính như sau:

* Biên Độ:



Hình 7: Công thức tính biên độ Gradient

* Hướng:



Hình 8: Công thức tính hướng Gradient

Thông thường, hướng được chuyển về khoảng 0 độ đến 180 độ để biểu diễn dưới dạng không dấu (unsigned).

* **Ưu điểm và nhược điểm**

Phần dưới đây trình bày chi tiết về những điểm mạnh và hạn chế của phương pháp Histogram of Oriented Gradients (HOG) trong bối cảnh xử lý ảnh và trích xuất đặc trưng, được ghi nhận từ quá trình nghiên cứu và thực nghiệm.

Ưu điểm của HOG:

* Phương pháp HOG tập trung vào việc mô tả sự phân bố của các hướng gradient trong ảnh, giúp nắm bắt được các đặc trưng hình học và biên dạng của đối tượng. Điều này cho phép nhận dạng đối tượng có hiệu quả, đặc biệt khi đối tượng có hình dạng rõ ràng.
* Từ góc nhìn ứng dụng, HOG không quá phụ thuộc vào giá trị cường độ điểm ảnh tuyệt đối. Nhờ đó, hiệu quả của nó không bị ảnh hưởng nhiều khi có biến đổi về ánh sáng hay tương phản. Quá trình tính toán dựa trên các gradient cục bộ giúp giữ được tính ổn định của đặc trưng, góp phần vào khả năng phát hiện đối tượng trong các điều kiện ánh sáng khác nhau.
* Do bản chất của các phép tính dựa trên hướng gradient và tần số biểu diễn, HOG có thể áp dụng cho nhiều bài toán nhận dạng với hiệu năng tương đối cao, đặc biệt khi đối tượng cần nhận dạng có kích thước và hình dạng ổn định.
* Hơn nữa, tính đơn giản của phương pháp làm cho việc triển khai và tối ưu hóa dễ dàng hơn so với các phương pháp trích xuất đặc trưng phức tạp khác. Việc tính toán các histogram của các hướng gradient thường không đòi hỏi tài nguyên tính toán quá lớn, giúp giảm thời gian xử lý trong nhiều trường hợp.

Nhược điểm của HOG:

* Một trong những hạn chế nổi bật của HOG là khả năng biểu diễn thông tin vi mô. Trong những trường hợp đối tượng có biến đổi hình dạng phức tạp hoặc chứa nhiều chi tiết nhỏ, HOG có thể bỏ sót những đặc trưng tinh tế đó, dẫn đến hiệu năng nhận dạng không được tối ưu.
* Trong bối cảnh ảnh có độ phân giải cao, việc tính toán hướng gradient và thống kê các histogram có thể trở nên tốn thời gian và tài nguyên. Điều này đặc biệt rõ ràng khi áp dụng trên tập dữ liệu lớn, làm chậm quá trình xử lý tổng thể của hệ thống.
* HOG thường hoạt động tốt với những ảnh có đối tượng rõ ràng, nhưng đối với các bối cảnh có nhiều nhiễu hoặc biến đổi góc nhìn, các đặc trưng được trích xuất có thể không đủ mạnh để phân biệt các lớp đối tượng. Điều này làm giảm khả năng khái quát hóa của mô hình khi gặp phải dữ liệu mới có sự biến đổi lớn về hình dạng hoặc cấu trúc.
* Ngoài ra, khi đối tượng trong ảnh có sự thay đổi về tỷ lệ hoặc vị trí, HOG không tích hợp trực tiếp cơ chế để khắc phục vấn đề này, nên cần phải kết hợp với các kỹ thuật xử lý tiền đề khác để đảm bảo hiệu quả nhận dạng ổn định.

Nhìn chung, HOG là một công cụ mạnh mẽ trong việc trích xuất đặc trưng dựa trên hình dạng tổng quát, song cũng cần kết hợp với các phương pháp bổ trợ khác hoặc thực hiện các biện pháp xử lý tiên tiến để khắc phục những hạn chế khi áp dụng vào các bối cảnh phức tạp hoặc yêu cầu độ chi tiết cao.

* **Ứng dụng vào bào toán**

Hàm extract\_hog\_features\_enhanced, được thiết kế để trích xuất đặc trưng HOG (Histogram of Oriented Gradients) với hai chế độ khác nhau, phù hợp cho các bài toán phân tích ảnh y tế đòi hỏi độ chi tiết cao hoặc xử lý nhanh.

Hàm nhận đầu vào là danh sách ảnh đã tiền xử lý và tham số detailed để điều chỉnh mức độ chi tiết. Trích xuất thông tin hình dạng và cấu trúc dựa trên hướng gradient của các vùng ảnh nhỏ. Tham số chính gồm 16 hướng gradient, kích thước ô 16×16 pixel.

Code trích dẫn:

**def extract\_hog\_features\_enhanced(images, detailed=False):**

**hog\_features = []**

**for img in tqdm(images, desc="Trích xuất HOG"):**

**if detailed:**

**features = hog(**

**img,**

**orientations=32,**

**pixels\_per\_cell=(8, 8),**

**cells\_per\_block=(2, 2),**

**channel\_axis=None**

**)**

**else:**

**features = hog(**

**img,**

**orientations=16,**

**pixels\_per\_cell=(16, 16),**

**cells\_per\_block=(3, 3),**

**channel\_axis=None**

**)**

**hog\_features.append(features)**

**return np.array(hog\_features)**

Giải thích:

tqdm: Thư viện hiển thị thanh tiến trình, giúp ước lượng thời gian trích xuất đặc trưng khi xử lý tập dữ liệu lớn.

Chế độ trích xuất chi tiết (detailed=True), khi kích hoạt, hàm sử dụng các tham số tối ưu để bắt các đặc trưng vi mô như cấu trúc mạch máu hoặc viền khối u:

* orientations=32: Chia 360° thành 32 hướng, tăng khả năng phân biệt các cấu trúc hình học phức tạp.
* pixels\_per\_cell=(8,8): Mỗi ô 8x8 pixel giúp phát hiện biến đổi gradient ở mức độ hạt.
* cells\_per\_block=(2,2): Chuẩn hóa histogram theo khối 2x2 ô, giảm ảnh hưởng của ánh sáng không đều.
* Ứng dụng: Phù hợp cho ảnh CT độ phân giải cao (≥512px), nơi các chi tiết nhỏ như vôi hóa hạch lympho có ý nghĩa chẩn đoán.

Chế Độ Cơ Bản (detailed=False), chế độ mặc định tập trung vào tốc độ xử lý, phù hợp cho ảnh độ phân giải thấp hoặc hệ thống tài nguyên hạn chế:

* orientations=16: Chia 360° thành 16 hướng, tăng khả năng phân biệt các cấu trúc hình học phức tạp.
* pixels\_per\_cell=(16,16): Mỗi ô 16x16 pixel giúp phát hiện biến đổi gradient ở mức độ hạt.
* cells\_per\_block=(3,3): Chuẩn hóa histogram theo khối 3x3 ô, giảm ảnh hưởng của ánh sáng không đều.
* Ứng dụng: Phân loại sơ bộ ảnh độ phân giải thấp

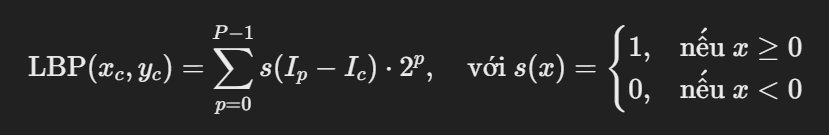
#### **3.2.5.2.Trích xuất đặc trưng LBP**

* **Giới thiệu**

LBP là một kỹ thuật đơn giản nhưng rất hiệu quả trong việc mô tả cấu trúc cục bộ của ảnh. Ý tưởng cốt lõi của LBP là chuyển đổi mỗi điểm ảnh thành một chuỗi nhị phân dựa trên so sánh giá trị của các điểm lân cận với điểm trung tâm. Những mẫu nhị phân này sau đó được gom lại thành histogram, từ đó tạo nên một vector đặc trưng mô tả kết cấu và hình dạng của ảnh. Nhờ tính bất biến với độ sáng và khả năng nắm bắt được các chi tiết vi mô, LBP được ứng dụng rộng rãi trong các bài toán nhận dạng khuôn mặt, phân loại kết cấu và phát hiện bất thường trong ảnh.

* **Thuật toán**

Trong chương trình này, sử dụng công thức sau để tính LBP cho mỗi điểm ảnh. Việc sử dụng công thức này giúp chuyển đổi thông tin cường độ của ảnh thành một dạng biểu diễn dựa trên tần suất các mẫu nhị phân, từ đó làm nổi bật các đặc trưng kết cấu của ảnh bất kể sự thay đổi về độ sáng.

  
Hình 9: Công thức tính LBP

Trong đó:

* Ic là giá trị cường độ của điểm ảnh trung tâm tại (xc,yc).
* Ip là giá trị cường độ của điểm ảnh lân cận thứ p.
* P là số điểm lân cận (thường lấy 8).
* Giá trị LBP được tính cho mỗi điểm ảnh, và toàn bộ ảnh sau đó được biểu diễn thông qua histogram các giá trị LBP.
* **Ưu điểm và nhược điểm**

Dưới đây là phần mô tả chi tiết về những ưu điểm và nhược điểm của phương pháp Local Binary Patterns (LBP):

Ưu điểm của LBP:

* Phương pháp LBP được đánh giá cao nhờ tính đơn giản trong việc cài đặt và thực hiện. Cơ chế so sánh giá trị điểm ảnh giữa trung tâm và các điểm lân cận cho phép chuyển đổi thông tin kết cấu cục bộ thành một mã nhị phân, qua đó giảm bớt sự phức tạp của dữ liệu.
* Hiệu quả trong việc phát hiện các đặc trưng kết cấu nhỏ và vi mô của ảnh được cải thiện rõ rệt. Nhờ cách mã hóa các mối quan hệ cục bộ, LBP có khả năng tách biệt các đặc trưng như vân, bề mặt và cạnh của đối tượng.
* Do không phụ thuộc trực tiếp vào cường độ ánh sáng tuyệt đối mà chỉ so sánh tương đối giữa các điểm, LBP có khả năng chịu đựng tốt với sự biến đổi về ánh sáng và tương phản. Điều này làm cho phương pháp trở nên ổn định khi xử lý các tập dữ liệu có điều kiện chiếu sáng thay đổi.
* Với việc tính toán đơn giản, phương pháp LBP có thể được triển khai trong các ứng dụng thời gian thực mà không đòi hỏi phần cứng hay tài nguyên tính toán quá cao.

Nhược điểm của LBP:

* LBP có tính nhạy cảm cao đối với nhiễu. Khi ảnh đầu vào chứa nhiều nhiễu, quá trình so sánh giữa các điểm có thể bị ảnh hưởng, dẫn đến việc tạo ra các mã nhị phân không phản ánh chính xác cấu trúc kết cấu của ảnh.
* Mặc dù có khả năng mô tả các đặc trưng kết cấu cục bộ, LBP không cung cấp đủ thông tin về mối quan hệ không gian tổng thể trong ảnh. Điều này có thể làm giảm hiệu quả trong việc nhận diện các đối tượng có hình dạng phức tạp, khi thông tin về bố cục tổng quát lại trở nên quan trọng.
* Kết quả trích xuất đặc trưng của LBP phụ thuộc vào các tham số như bán kính và số điểm lấy mẫu. Việc lựa chọn tham số không phù hợp có thể dẫn đến mất mát thông tin cần thiết hoặc tạo ra các đặc trưng không ổn định, ảnh hưởng đến hiệu quả của mô hình khi áp dụng vào các tập dữ liệu đa dạng.
* Trong các trường hợp đối tượng trong ảnh có sự biến đổi về góc nhìn hoặc tỷ lệ, LBP cần được kết hợp với các kỹ thuật bổ trợ (ví dụ: các biến thể của LBP hay các phương pháp trích xuất đặc trưng khác) để đảm bảo độ chính xác cao hơn trong việc phân loại hoặc nhận dạng.

Qua đó, LBP mang lại lợi thế lớn trong việc xử lý kết cấu cục bộ của ảnh với chi phí tính toán thấp, nhưng đồng thời cũng tồn tại những hạn chế nhất định khi phải đối phó với nhiễu và những biến đổi phức tạp của ảnh.

* **Ứng dụng vào bài toán**

Hàm extract\_lbp\_features được thiết kế để trích xuất đặc trưng LBP (Local Binary Pattern), tập trung vào phân tích kết cấu vi mô của ảnh CT phổi. Cấu hình tham số linh hoạt cho phép tối ưu hóa giữa độ chi tiết và tốc độ xử lý.Quy trình thực hiện LBP diễn ra theo các bước sau:

* Bước 1: Lấy các điểm lân cận
* Bước 2: So sánh giá trị với điểm trung tâm
* Bước 3: Mã hóa nhị phân thành giá trị số
* Bước 4: Tạo Histogram

Hàm nhận đầu vào là danh sách ảnh và điều chỉnh đặc trưng dựa trên tham số detailed. Phân tích texture cục bộ thông qua mã hóa nhị phân các điểm lân cận, sử dụng bán kính 3 pixel và 24 điểm tham chiếu.

Code trích dẫn:

**def extract\_lbp\_features(images, detailed=False, radius=3, n\_points=24):**

**lbp\_features = []**

**for img in tqdm(images, desc="Trích xuất LBP"):**

**if detailed:**

**lbp = local\_binary\_pattern(img, n\_points, 2, method="uniform")**

**bins = np.arange(0, n\_points + 3)**

**range\_val = (0, n\_points + 2)**

**else:**

**lbp = local\_binary\_pattern(img, n\_points, radius, method="uniform")**

**bins = np.arange(0, n\_points + 3)**

**range\_val = (0, n\_points + 2)**

**(hist, \_) = np.histogram(lbp.ravel(), bins=bins, range=range\_val)**

**hist = hist.astype("float")**

**hist /= (hist.sum() + 1e-7)**

**lbp\_features.append(hist)**

**return np.array(lbp\_features)**

Giải thích:

* radius: Bán kính vùng lân cận xung quanh pixel trung tâm.
* n\_points=24: Số điểm lấy mẫu trên vòng tròn bán kính radius.
* tqdm: Hiển thị thanh tiến trình trích xuất đặc trưng.

Chế độ chi tiết (detailed=True).Khi kích hoạt, hàm tập trung vào đặc trưng vi mô bằng cách giảm bán kính phân tích.Code trích dẫn:

**lbp = local\_binary\_pattern(img, n\_points, 2, method="uniform")**

Giải thích Tham số:

* radius=2: Giảm từ 3 xuống 2 để tập trung vào vùng nhỏ hơn (3x3 pixel → 5x5 pixel).
* method="uniform": Lọc các mẫu LBP không liên tục, giữ lại 58 mẫu hợp lệ từ 256 tổ hợp**.**
* Ứng dụng: Phát hiện thay đổi kết cấu nhỏ như vôi hóa hạch lympho hoặc xơ phổi.

Chế độ cơ bản (detailed=False).Chế độ mặc định sử dụng bán kính lớn hơn để phủ rộng vùng phân tích.Code trích dẫn:

**lbp = local\_binary\_pattern(img, n\_points, radius, method="uniform")**

Giải thích tham số:

* radius=3: Phân tích vùng 7x7 pixel (2\*radius + 1), phát hiện đặc trưng kích thước trung bình.
* n\_points=24: Tăng mật độ lấy mẫu so với giá trị mặc định (thường 8 điểm).

Đặc trưng LBP được chuyển đổi thành histogram để biểu diễn phân phối kết cấu. Code trích dẫn:

**bins = np.arange(0, n\_points + 3)**

**range\_val = (0, n\_points + 2)**

**(hist, \_) = np.histogram(lbp.ravel(), bins=bins, range=range\_val)**

**hist /= (hist.sum() + 1e-7) # Chuẩn hóa**

Giải thích:

* bins: Tạo 27 bin (0–26) để chứa 58 mẫu LBP "uniform" (từ 0 đến n\_points + 2).
* 1e-7: Tránh chia cho 0 trong trường hợp ảnh trơn (không có kết cấu).

Tính Histogram và Chuẩn Hóa, Đặc trưng LBP được chuyển đổi thành histogram để biểu diễn phân phối kết cấu.Code trích dẫn:

**bins = np.arange(0, n\_points + 3)**

**range\_val = (0, n\_points + 2)**

**(hist, \_) = np.histogram(lbp.ravel(), bins=bins, range=range\_val)**

**hist /= (hist.sum() + 1e-7) # Chuẩn hóa**

Giải thích:

* bins: Tạo 27 bin (0–26) để chứa 58 mẫu LBP "uniform" (từ 0 đến n\_points + 2).
* 1e-7: Tránh chia cho 0 trong trường hợp ảnh trơn (không có kết cấu).
* Ưu điểm: Histogram chuẩn hóa giúp mô hình ổn định với ảnh có độ sáng khác nhau.
* **Kết hợp HOG và LBP thành vector đặc trưng duy nhất**

Hàm extract\_combined\_features kết hợp hai loại đặc trưng hình ảnh (HOG và LBP) thành một vector đặc trưng duy nhất, tận dụng ưu điểm bổ sung của cả hai phương pháp để nâng cao hiệu suất mô hình phân loại. Đoạn mã thực hiện việc trích xuất và kết hợp đặc trưng qua ba bước chính:

**def extract\_combined\_features(images, detailed=False):**

**hog\_feats = extract\_hog\_features\_enhanced(images, detailed=detailed)**

**lbp\_feats = extract\_lbp\_features(images, detailed=detailed)**

**combined = np.hstack([hog\_feats, lbp\_feats])**

**return combined**

Giải thích:

* Trích Xuất HOG: Gọi hàm extract\_hog\_features\_enhanced đã định nghĩa trước đó. Tham số detailed điều khiển việc sử dụng cấu hình chi tiết (ô 8x8 pixel) hoặc cơ bản (ô 16x16 pixel).
* Trích Xuất LBP: Gọi hàm extract\_lbp\_features để thu được histogram kết cấu. Tham số detailed điều chỉnh bán kính phân tích (2 hoặc 3 pixel).
* Kết Hợp Đặc Trưng: Sử dụng np.hstack để nối vector HOG và LBP theo chiều ngang.
* Lý do Kết Hợp HOG và LBP bổ sung đặc trưng:

+ HOG: Tập trung vào hình dạng và cấu trúc lớn (ví dụ: viền khối u).

+ LBP: Phân tích kết cấu vi mô (ví dụ: độ thô/mịn của mô phổi).

### **3.2.6.StandardScaler và PCA**

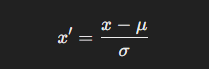
#### **3.2.6.1.Chuẩn hóa dữ liệu với StandardScaler**

* **Giới thiệu**

StandardScaler là một công cụ trong tiền xử lý dữ liệu, giúp “chuẩn hóa” các đặc trưng (features) của tập dữ liệu. Ý nghĩa cơ bản của StandardScaler là chuyển đổi dữ liệu sao cho các giá trị của mỗi đặc trưng có trung bình bằng 0 và độ lệch chuẩn bằng 1. Điều này giúp làm giảm sự chênh lệch giữa các thang đo khác nhau của dữ liệu, từ đó các mô hình học máy có thể hoạt động hiệu quả hơn.

* **Thuận toán**

Kỹ thuật này hoạt động dựa trên nguyên tắc chuẩn hóa từng đặc trưng riêng biệt theo công thức:



Hình 10: Công thức tính StandardScaler

Trong đó,

* x là giá trị ban đầu của đặc trưng,
* μ là giá trị trung bình của đặc trưng trên tập huấn luyện, và
* σ là độ lệch chuẩn. Quy trình này đảm bảo rằng sau khi chuẩn hóa, mỗi đặc trưng sẽ có trung bình bằng 0 và độ lệch chuẩn bằng 1.
* **Ứng dụng vào bài toán**

Trong chương trình này, StandardScaler được áp dụng sau khi trích xuất các vector đặc trưng từ ảnh (kết hợp HOG và LBP). Việc chuẩn hóa giúp các đặc trưng được đưa về cùng một quy mô, tránh trường hợp một đặc trưng có giá trị lớn có thể chi phối quá trình huấn luyện mô hình. Cụ thể, trong tập dữ liệu huấn luyện, StandardScaler tính toán trung bình và độ lệch chuẩn cho từng cột của ma trận đặc trưng, sau đó biến đổi dữ liệu theo công thức trên. Dữ liệu kiểm tra sau đó được biến đổi bằng cùng các tham số đã học từ tập huấn luyện, đảm bảo tính nhất quán giữa các tập.

Việc sử dụng StandardScaler không chỉ giúp giảm thiểu vấn đề về sự chênh lệch đơn vị đo giữa các đặc trưng mà còn góp phần cải thiện hiệu quả huấn luyện mô hình, đặc biệt là đối với các thuật toán nhạy cảm với thang đo như SVM. Điều này giúp mô hình học được các đặc trưng một cách ổn định và có khả năng tổng quát hóa tốt hơn trên dữ liệu mới.

Trong quá trình xử lý dữ liệu, đặc trưng được trích xuất từ ảnh (kết hợp HOG và LBP) có thể có các giá trị với thang đo khác nhau. Việc chuẩn hóa giúp các đặc trưng có phân phối thống nhất, loại bỏ ảnh hưởng của các đơn vị đo lường không đồng nhất. Để thực hiện điều này, đoạn code sử dụng StandardScaler từ thư viện scikit-learn. Cụ thể, trên tập huấn luyện các giá trị trung bình và độ lệch chuẩn được tính toán và áp dụng biến đổi cho cả tập huấn luyện và tập kiểm tra như sau:

**scaler = StandardScaler()**

**X\_train = scaler.fit\_transform(X\_train)**

**X\_test = scaler.transform(X\_test)**

Giải thích:

* scaler.fit\_transform(X\_train) tính toán các tham số (trung bình và độ lệch chuẩn) từ dữ liệu huấn luyện và chuyển đổi dữ liệu về dạng phân phối chuẩn (mean = 0, std = 1).
* scaler.transform(X\_test) sử dụng các tham số đã học để chuyển đổi dữ liệu kiểm tra, đảm bảo tính nhất quán giữa tập huấn luyện và tập kiểm tra.

#### **3.2.6.2.Giảm chiều không gian với PCA**

* **Giới thiệu**

Phân tích thành phần chính (PCA) là một kỹ thuật giảm chiều không gian rất phổ biến, giúp rút gọn số lượng biến số trong tập dữ liệu mà vẫn giữ lại phần lớn thông tin về biến thiên.

* **Thuật toán**

Chuẩn hóa dữ liệu: Trước tiên, cần đưa các đặc trưng về cùng một thang đo. Điều này được thực hiện bằng cách trừ giá trị trung bình của từng đặc trưng và chia cho độ lệch chuẩn. Việc chuẩn hóa giúp loại bỏ hiệu ứng của các đơn vị đo lường khác nhau và tạo điều kiện thuận lợi cho bước tính toán tiếp theo.

Tính toán ma trận hiệp phương sai: Sau khi dữ liệu được chuẩn hóa, ma trận hiệp phương sai được xây dựng. Ma trận này phản ánh mức độ liên quan giữa các đặc trưng, tức là đo lường cách các biến thay đổi cùng nhau. Mỗi phần tử trong ma trận cho biết sự tương quan giữa một cặp đặc trưng, giúp xác định những hướng mà dữ liệu có biến thiên mạnh.

Giải bài toán giá trị riêng: Tiếp theo, thực hiện giải bài toán giá trị riêng (eigenvalue problem) trên ma trận hiệp phương sai. Quá trình này cho ra các giá trị riêng (eigenvalues) cùng với các vector riêng (eigenvectors). Các vector riêng này chính là các trục chính (principal components) mà dữ liệu có thể được biểu diễn lại, và mỗi giá trị riêng biểu thị mức độ biến thiên dọc theo vector tương ứng.

Chọn lựa thành phần chính: Sau khi có các giá trị riêng và vector riêng, sắp xếp các giá trị riêng theo thứ tự giảm dần. Các thành phần chính được lựa chọn dựa trên các giá trị riêng lớn nhất, vì chúng giải thích phần lớn biến thiên của dữ liệu. Số lượng thành phần được giữ lại (trong trường hợp này là 100 thành phần) được xác định dựa trên mức độ thông tin cần duy trì và mục tiêu giảm chiều.

Chiếu dữ liệu vào không gian con: Cuối cùng, dữ liệu ban đầu được chuyển đổi sang không gian con được tạo thành từ các thành phần chính đã chọn. Quá trình này có nghĩa là mỗi mẫu dữ liệu ban đầu được biểu diễn dưới dạng tổ hợp tuyến tính của các vector riêng. Nhờ đó, số chiều của dữ liệu được giảm xuống nhưng vẫn giữ lại được phần lớn thông tin biến thiên.

Trong chương trình, PCA được áp dụng sau khi các vector đặc trưng được chuẩn hóa thông qua StandardScaler. Kết quả của PCA là tập dữ liệu có số chiều ít hơn, tập trung vào những đặc trưng mang nhiều biến thiên nhất. Điều này không chỉ giảm thiểu nhiễu mà còn giúp quá trình huấn luyện mô hình SVM được thực hiện nhanh chóng và chính xác hơn.

* **Ứng dụng vào bài toán**

Sau khi chuẩn hóa, số lượng đặc trưng ban đầu có thể khá lớn và chứa nhiều thông tin dư thừa. Để giảm nhiễu, giảm tính phức tạp và tăng hiệu quả của mô hình SVM, sử dụng phương pháp giảm chiều không gian là PCA (Principal Component Analysis). Việc giảm chiều không gian giúp giữ lại những thành phần chính mang nhiều biến thiên nhất trong dữ liệu, đồng thời loại bỏ những thành phần không quan trọng. Trong đoạn code, PCA được cấu hình với 100 thành phần chính như sau:

**pca = PCA(n\_components=100, random\_state=42)**

**X\_train = pca.fit\_transform(X\_train)**

**X\_test = pca.transform(X\_test)**

Giải thích:

* pca.fit\_transform(X\_train) tìm ra các thành phần chính từ tập huấn luyện và biến đổi dữ liệu theo các thành phần này, giữ lại những thông tin có biến thiên lớn nhất.
* pca.transform(X\_test) chuyển đổi tập kiểm tra dựa trên các thành phần đã tính toán từ tập huấn luyện, đảm bảo rằng dữ liệu kiểm tra được đưa vào cùng không gian đặc trưng đã rút gọn.

### **3.2.7.Xây dựng mô hình SVM**

Trong phần xây dựng mô hình, việc tập trung vào việc khai báo và huấn luyện một SVM với kernel RBF, thiết lập các tham số để phù hợp với bài toán phân loại ảnh CT thành hai nhóm: bình thường và ung thư. Sau khi đã tiền xử lý dữ liệu và trích xuất đặc trưng, tiến hành chuẩn hóa và giảm chiều (PCA) trước khi đưa vào mô hình. Nhằm hạn chế hiện tượng overfitting, sử dụng tham số class\_weight='balanced' để cân bằng số lượng mẫu giữa các lớp, đồng thời kích hoạt tính năng dự đoán xác suất qua probability=True.

Đoạn code dưới đây là phần cốt lõi để khởi tạo và huấn luyện mô hình SVM với cấu hình như sau:

**def train\_default\_model(X\_train, y\_train):**

**model = SVC(**

**kernel='rbf',**

**C=1.0,**

**gamma='scale',**

**class\_weight='balanced',**

**probability=True,**

**random\_state=42**

**)**

**model.fit(X\_train, y\_train)**

**return model**

Giải thích:

* kernel='rbf': Sử dụng hàm kernel Gaussian (RBF) cho khả năng phân chia phi tuyến
* C=1.0: Hệ số phạt, điều chỉnh sự cân bằng giữa độ phức tạp của mô hình và lỗi huấn luyện
* gamma='scale': Tham số gamma được tính theo tỉ lệ dữ liệu, giúp xác định phạm vi ảnh hưởng của mỗi điểm dữ liệu
* class\_weight='balanced': Cân bằng trọng số của các lớp, đặc biệt hữu ích khi dữ liệu có sự chênh lệch về số lượng mẫu giữa các lớp
* probability=True: Kích hoạt tính năng tính xác suất dự đoán cho mỗi mẫu
* random\_state=42: Đặt seed cố định để đảm bảo tính tái lập của quá trình huấn luyện

### **3.2.8.Huấn luyện mô hình**

Hàm train\_default\_model trong chương trình được xây dựng để thực hiện việc huấn luyện mô hình. Chương trình được xây dựng dựa trên liên kết giữa các bước tiền xử lý, trích xuất đặc trưng, huấn luyện mô hình. Toàn bộ quy trình huấn luyện được thực hiện theo trình tự các bước sau:

**1.Xác định tham số và khởi tạo môi trường:**

- Ban đầu, các tham số dòng lệnh được phân tích để xác định các tùy chọn mà chương trình sẽ thực hiện. Trong số đó, tùy chọn tăng cường dữ liệu (--augment) được kích hoạt, đồng nghĩa với việc áp dụng các phép biến đổi bổ sung trong quá trình tiền xử lý. Đồng thời, hệ thống ghi nhận log được thiết lập nhằm theo dõi các hoạt động và lưu lại thông tin về quá trình chạy của chương trình, đảm bảo tính minh bạch và khả năng kiểm soát sau này.

**2.Tải và tiền xử lý dữ liệu**

- Bộ dữ liệu bao gồm các hình ảnh thuộc hai nhãn phân loại (Normal và Cancer) được lấy từ các thư mục đã được sắp xếp sẵn. Mỗi ảnh được chuyển sang định dạng ảnh xám, sau đó được thay đổi kích thước về một chuẩn xác định (ví dụ: 512x512 pixel). Giá trị của các điểm ảnh được chuẩn hóa về khoảng từ 0 đến 1 nhằm đảm bảo tính đồng nhất của dữ liệu. Khi tham số tăng cường được bật, các phép biến đổi như lật ảnh, xoay ảnh theo các góc nhỏ và điều chỉnh độ sáng, độ tương phản được áp dụng cho từng ảnh gốc. Các biến đổi này tạo ra nhiều phiên bản khác nhau của cùng một ảnh, mở rộng tập dữ liệu huấn luyện và giúp mô hình học được các đặc trưng ổn định hơn dưới nhiều điều kiện khác nhau.

**3.Trích xuất đặc trưng**

- Sau khi các ảnh đã qua xử lý, các đặc trưng quan trọng được thu thập từ hình ảnh tăng cường. Phương pháp trích xuất kết hợp hai kỹ thuật phổ biến là Histogram of Oriented Gradients (HOG) và Local Binary Patterns (LBP). Mỗi ảnh được chuyển hóa thành một vector đặc trưng duy nhất, chứa đựng các thông tin về biên, kết cấu và hình dạng của đối tượng. Các vector đặc trưng này sẽ là đầu vào cho các bước xử lý tiếp theo của mô hình.

**4.Chia tách dữ liệu**

- Bộ dữ liệu đặc trưng sau khi trích xuất được chia thành tập huấn luyện và tập kiểm tra. Quy trình tách dữ liệu được thực hiện sao cho tỷ lệ giữa các lớp vẫn được cân bằng trong từng tập, đảm bảo rằng quá trình huấn luyện và đánh giá không bị thiên vị. Các tham số cố định như giá trị ngẫu nhiên được sử dụng để đảm bảo rằng việc chia tách có thể tái lập được trong các lần chạy khác nhau của chương trình.

**5.Chuẩn hóa và giảm chiều không gian**

- Trước khi đưa vào mô hình, các vector đặc trưng được chuẩn hóa để đưa về cùng một thang đo. Sau đó, kỹ thuật giảm chiều không gian thông qua Principal Component Analysis (PCA) được áp dụng nhằm rút gọn số lượng đặc trưng. Quá trình này giúp giữ lại những thành phần mang lại nhiều biến thiên nhất, từ đó loại bỏ những nhiễu không cần thiết. Việc giảm số chiều không chỉ tăng tốc độ tính toán mà còn giúp mô hình tập trung vào những đặc trưng quan trọng nhất của dữ liệu.

**6.Huấn luyện mô hình SVM**

- Dữ liệu đầu vào sau khi được xử lý (chuẩn hóa và giảm chiều) được sử dụng để huấn luyện mô hình SVM. Mô hình được cấu hình với kernel RBF cùng với các tham số được thiết lập nhằm giảm thiểu hiện tượng overfitting. Quá trình huấn luyện được tối ưu hóa dựa trên tập dữ liệu huấn luyện, tạo ra một mô hình có khả năng phân loại hai lớp một cách hiệu quả.

**7.Đánh giá và lưu trữ mô hình**

- Sau khi hoàn tất giai đoạn huấn luyện, mô hình được kiểm tra trên tập dữ liệu kiểm tra để đánh giá hiệu năng, sử dụng các chỉ số như độ chính xác và báo cáo phân loại. Kết quả đánh giá giúp xác nhận khả năng tổng quát của mô hình. Cuối cùng, mô hình đã huấn luyện, cùng với bộ chuẩn hóa và đối tượng PCA, được lưu trữ dưới dạng file để có thể sử dụng trong quá trình dự đoán dữ liệu mới.

**8.Tổng kết**

Những bước trên tạo nên một quy trình huấn luyện toàn diện, đảm bảo mở rộng dữ liệu đầu vào thông qua tăng cường, từ đó giúp cải thiện khả năng tổng quát hóa của mô hình trong việc phát hiện sớm ung thư phổi từ ảnh chụp CT.

# **Chương 4: KẾT QUẢ THỰC NGHIỆM VÀ HƯỚNG PHÁT TRIỂN**

## **4.1.Trình bài các độ đo áp dụng trong bài toán**

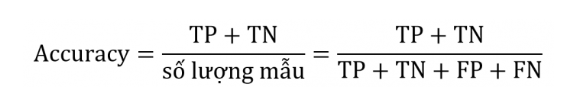
Trong quá trình xây dựng chương trình, việc đánh giá kết quả dự đoán của mô hình đóng vai trò then chốt để hiểu được khả năng nhận diện của nó. Sử dụng một số độ đo tiêu chuẩn như Accuracy, Precision, Recall, F1-Score, support, macro avg và weighted avg. Mỗi chỉ số cung cấp một góc nhìn khác nhau, từ đó giúp phát hiện và khắc phục những hạn chế của mô hình.

### **4.1.1.Accuracy**

Accuracy là tỷ lệ phần trăm mẫu mà mô hình dự đoán đúng trên tổng số mẫu. Đây là chỉ số cơ bản, giúp có cái nhìn tổng quan về khả năng của mô hình. Tuy nhiên, chỉ số này có thể gây hiểu nhầm khi dữ liệu bị mất cân bằng.

Ví dụ, nếu 90% mẫu thuộc lớp normal, một mô hình dự đoán tất cả đều là normal có thể đạt độ chính xác cao nhưng lại bỏ sót nhiều thông tin của lớp cancer.

Công thức:



Hình 11 : Công thức tính Accuracy

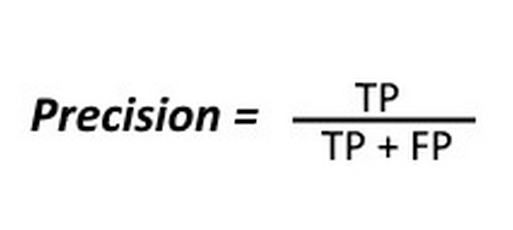
Trong đó:

* TP (True Positives): Số mẫu dương tính được dự đoán đúng.
* FP (False Positives): Số mẫu âm tính bị nhầm thành dương tính.
* FN (False Negatives): Số mẫu dương tính mà mô hình không phát hiện được.
* TN (True Negative): số lượng mẫu mà mô hình dự đoán là âm tính và thực tế cũng thuộc nhóm âm tính.

### **4.1.2.Precision**

Precision (độ chính xác dương tính) đo lường tỷ lệ các mẫu được dự đoán là dương tính mà thực sự là dương tính. Chỉ số này đặc biệt quan trọng khi việc gán nhãn sai (dự đoán dương tính giả) là cao. Nói cách khác, trong số các mẫu mà mô hình dự đoán là dương tính, bao nhiêu phần trăm là chính xác

Công thức:



Hình 12 : Công thức tính Precision

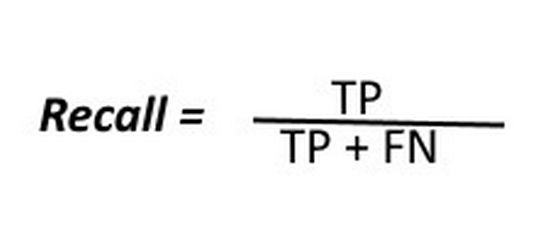
Trong đó:

* TP (True Positives): Số mẫu dương tính được dự đoán đúng.
* FP (False Positives): Số mẫu âm tính bị nhầm thành dương tính.

### **4.1.3.Recall**

Recall (độ nhạy) cho biết khả năng mô hình phát hiện đúng các mẫu dương tính trên tổng số mẫu dương tính có sẵn. Đây là chỉ số quan trọng khi việc bỏ sót mẫu dương tính (âm tính giả) có thể dẫn đến hậu quả nghiêm trọng.

Công thức:



Hình 13 : Công thức tính Recall

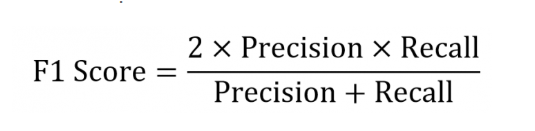
Trong đó:

* TP (True Positives): Số mẫu dương tính được dự đoán đúng.
* FN (False Negatives): Số mẫu dương tính mà mô hình không phát hiện được.

### **4.1.4.F1-Score**

F1-Score là trung bình điều hòa của Precision và Recall, cung cấp một thước đo tổng hợp về hiệu năng của mô hình. Chỉ số này rất hữu ích khi cần cân bằng giữa khả năng phát hiện mẫu dương tính và độ chính xác của các dự đoán dương tính.

Công thức:



Hình 14 : Công thức tính F1-Score

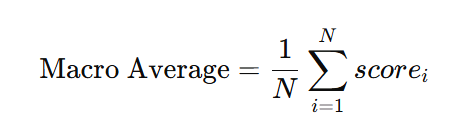
### **4.1.5.Support**

Support là số lượng mẫu thực tế thuộc mỗi lớp trong tập dữ liệu. Chỉ số này không phải là thước đo hiệu suất trực tiếp, nhưng nó rất quan trọng để hiểu được phân bố dữ liệu. Support giúp nhận biết nếu dữ liệu có sự mất cân bằng, từ đó điều chỉnh lại trọng số trong quá trình huấn luyện hoặc lựa chọn các chỉ số tổng hợp khác.

### **4.1.6.Macro Average**

Macro Average tính trung bình các giá trị của Precision, Recall và F1-Score trên tất cả các lớp mà không tính đến số lượng mẫu của từng lớp. Đây là cách đánh giá hiệu suất trung bình của mô hình trên mỗi lớp, tuy nhiên nếu dữ liệu không cân bằng, macro avg có thể không phản ánh đúng hiệu quả trên các lớp chiếm số lượng lớn.

Công thức:



Hình 15 : Công thức tính Macro Average

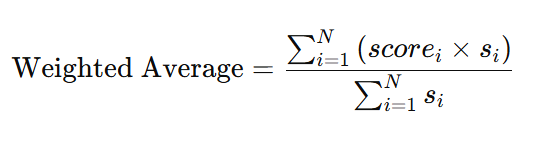
Trong đó:

* N: Số lượng các lớp trong bài toán phân loại.
* score i: Giá trị của chỉ số đánh giá (chẳng hạn như Precision, Recall hay F1-Score) đối với lớp thứ i. Mỗi lớp sẽ có một giá trị riêng cho từng chỉ số.

### **4.1.7.Weighted Average**

Weighted Average cũng tính trung bình các chỉ số nhưng trọng số của mỗi lớp được xác định bởi số mẫu (support) của lớp đó. Nhờ đó, chỉ số này thiên về các lớp có nhiều mẫu hơn, giúp tổng thể hiệu suất của mô hình được đánh giá một cách chính xác hơn khi có sự mất cân bằng dữ liệu.

Công thức:

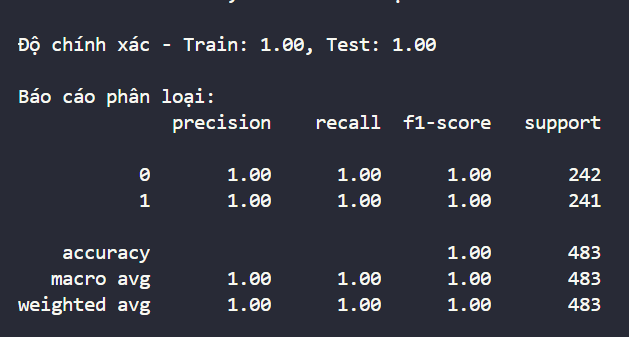


Hình 16 : Công thức tính Weighted Average

Trong đó:

* N: Số lượng các lớp trong bài toán phân loại.
* score i: Giá trị của chỉ số đánh giá (chẳng hạn như Precision, Recall hay F1-Score) đối với lớp thứ i. Mỗi lớp sẽ có một giá trị riêng cho từng chỉ số.
* Si (Support): Số lượng mẫu thực tế thuộc lớp thứ i. Đây là số liệu cho biết mức độ phổ biến của từng lớp trong tập dữ liệu.

## **4.2.Đánh giá mô hình sau train**



Hình 17. Kết quả đánh giá tập test sau khi train

Trong đó:

* Độ chính xác (Accuracy) 100%
  + Cho thấy mô hình dự đoán đúng khoảng 100% tổng số tweet trong tập kiểm thử.
* Chỉ số Precision, Recall, F1-score
  + Các lớp 0 ( Normal ), 1 ( Cancer ) đều có giá trị F1-score dao động trong 1.00, tương đối đồng đều.
  + Điều này thể hiện mô hình không bị thiên lệch quá nhiều về một lớp nào đó.
* Macro avg và Weighted avg
  + Macro avg F1 1.00 , Weighted avg F1 1.00 => Mô hình đạt hiệu năng khá cân bằng trên cả 2 lớp.

## **4.3.Điều chỉnh mô hình cài thiện hiệu năng của mô hình**

### **4.3.1.Tối ưu hóa tham số với Grid Search**

* Mục Đích:

Tùy chọn “**--grid\_search** ” cho phép tự động hóa quá trình tìm kiếm bộ tham số tối ưu cho mô hình SVM, giúp cải thiện hiệu suất dự đoán.

* Cơ Chế Hoạt Động:

Khi kích hoạt, hệ thống sử dụng GridSearchCV từ thư viện scikit-learn để duyệt qua một tập hợp các tham số định trước.

Quá trình tìm kiếm sử dụng 5-fold cross-validation và độ đo f1\_weighted để đánh giá, phù hợp với dữ liệu không cân bằng.

Lưới tham số được định nghĩa bao gồm các giá trị khác nhau cho siêu tham số:

**def train\_with\_grid\_search(X\_train, y\_train):**

**param\_grid = {**

**'C': [0.1, 1, 10, 100], # Độ mạnh regularization**

**'gamma': ['scale', 'auto', 0.001, 0.01], # Ảnh hưởng của từng mẫu**

**'kernel': ['rbf', 'linear', 'poly'] # Loại hàm nhân**

**}**

**grid\_search = GridSearchCV(**

**SVC(class\_weight='balanced', probability= True,random\_state=42),**

**param\_grid,**

**cv=5,**

**scoring='f1\_weighted',**

**verbose=2**

**)**

**grid\_search.fit(X\_train, y\_train)**

**return grid\_search.best\_estimator\_**

* Ưu Điểm:

Giảm thiểu rủi ro chọn tham số thủ công, tăng khả năng tìm ra mô hình có độ chính xác cao.

### **4.3.2. Điều Chỉnh Trọng Số Lớp custom\_weights**

* Mục Đích:

Giải quyết vấn đề mất cân bằng dữ liệu bằng cách gán trọng số ưu tiên cho lớp thiểu số (ung thư), giúp mô hình học đặc trưng tốt hơn.

* Chiến Lược Cân Bằng:

Trọng số được thiết lập thủ công theo tỷ lệ 1:2 giữa lớp "normal" (0) và "cancer" (1):

**class\_weights = {0: 1, 1: 2} # Tăng ảnh hưởng của lớp cancer**

Cơ chế này phạt nặng hơn khi mô hình phân loại sai lớp cancer, phù hợp với yêu cầu y tế cần giảm tỷ lệ âm tính giả.

**Triển Khai Code:**

**def train\_custom\_weight\_model(X\_train, y\_train):**

**model = SVC(**

**kernel='rbf',**

**class\_weight={0:1, 1:2}, # Trọng số tùy chỉnh**

**probability=True,**

**random\_state=42**

**)**

**model.fit(X\_train, y\_train)**

**return model**

* So Sánh với Mặc Định:

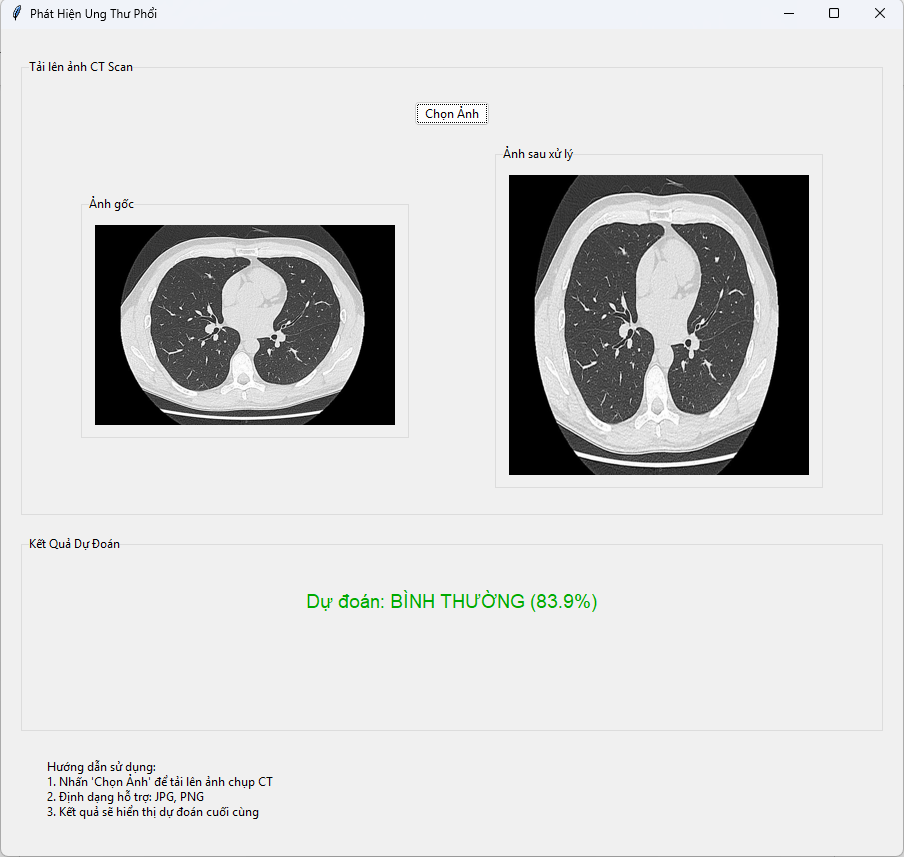
Khác với **class\_weight='balanced'** (tính trọng số tự động dựa trên tần suất lớp), phiên bản tùy chỉnh cho phép kiểm soát chính xác mức độ ưu tiên.

* Ứng Dụng Thực Tế:

Trong trường hợp dữ liệu có sự chênh lệch lớn (ví dụ: ít ảnh ung thư), việc điều chỉnh trọng số giúp cải thiện độ nhạy (recall) của lớp quan trọng.

## **4.4.Đánh giá mô hình trên dữ liệu mới**

### **4.4.1.Thực hiện việc test trên ảnh mới với nhãn là normal**



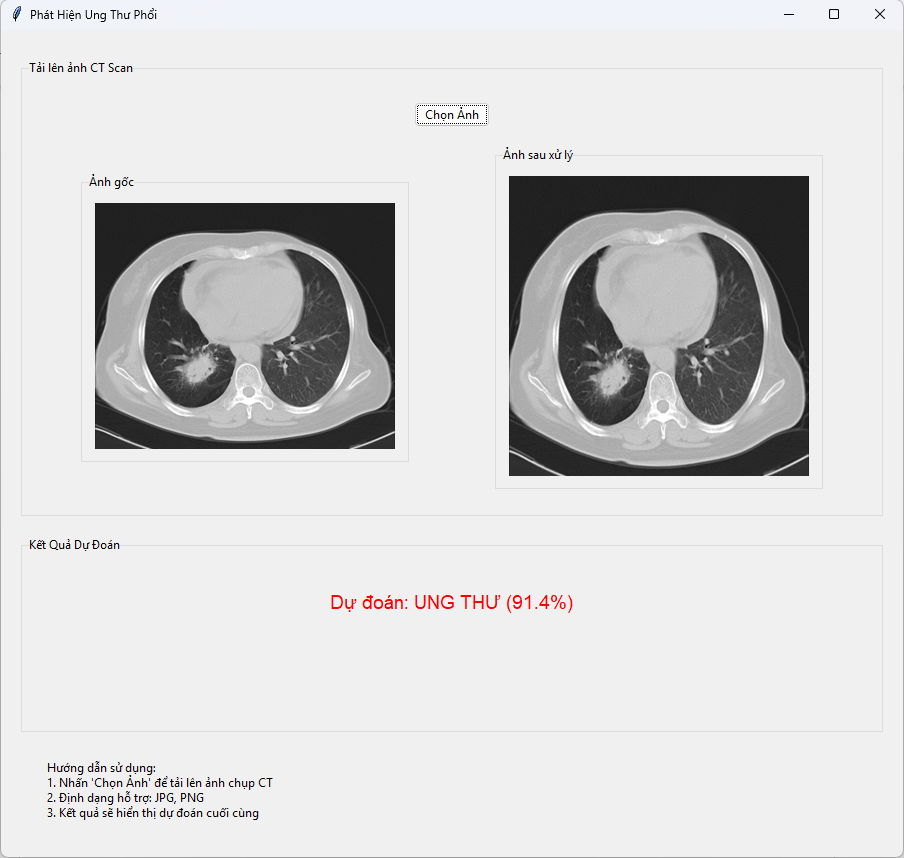
Hình 18: test trên tập dữ liệu mới mẫu 1

Giải thích:

Qua hình ảnh có thể thấy được đây là ảnh chụp CT phổi bình thường của 1 bênh bệnh nhân.Trong hình có thể thấy rõ khoang ngực, tim, các mạch máy xung quanh phổi và trái tim của bệnh nhân nhưng không thấy được dấu hiện xuất hiện ung thư .

Như vậy, có thể thấy mô hình đã có thể đánh giá khá chính xác trên dữ liệu mới đưa vào từ giao diện người dùng

### **4.4.2.Thực hiện việc test trên ảnh mới với nhãn là Cancer**



Hình 19: test trên dữ liệu mới mẫu 2

Giải thích:

Qua hình ảnh có thể thấy được đây là ảnh chụp CT phổi có dấu hiệu ung thư của 1 bênh bệnh nhân.Trong hình có thể thấy rõ khoang ngực, tim, các mạch máy xung quanh phổi và trái tim của bệnh nhân. Bên cạnh đó phía bên lá phổi trái bên dưới cùng xuất hiện một vùng trắng khác thường và đó là dấu hiệu của ung thư phổi.

Như vậy, có thể thấy mô hình đã có thể đánh giá khá chính xác trên dữ liệu mới đưa vào từ giao diện người dung

# **KẾT LUẬN**

Báo cáo đã trình bày quy trình xây dựng mô hình SVM (Support Vector Machine) để phát hiện sớm ung thư phổi thông qua ảnh chụp CT. Với sự kết hợp giữa 2 kỹ thuật trích xuất đặc trưng HOG (Histogram of Oriented Gradients) và LBP (Local Binary Pattern), cùng quá trình chuẩn hóa dữ liệu và giảm chiều bằng PCA, mô hình đã đạt hiệu suất ấn tượng trên cả tập huấn luyện và dữ liệu mới, thể hiện thông qua các độ đo:

* + Độ chính xác (Accuracy): 100%
  + Recall, Precision, F1-Score: 1.00 cho cả hai lớp
  + Kết quả nhất quán trên ảnh kiểm tra mới (Normal và Cancer).
* Ưu điểm nổi bật:

Khả năng phân loại chính xác: SVM với kernel RBF và tối ưu tham số đã cho thấy hiệu quả trong việc phân biệt các đặc trưng tinh vi giữa mô phổi bình thường và bất thường.

Quy trình xử lý chặt chẽ: Tiền xử lý ảnh (resize, cân bằng sáng), tăng cường dữ liệu (xoay, lật, điều chỉnh tương phản), và trích xuất đặc trưng đa dạng giúp mô hình học được các mẫu tổng quát.

Ứng dụng thực tiễn: Hệ thống có thể có tiềm năng hỗ trợ trong chẩn đoán sớm, giảm thiểu sai sót do yếu tố chủ quan.

* Hạn chế và hướng phát triển:

Nguy cơ overfitting: Kết quả hoàn hảo (100%) trên tập test có thể phản ánh việc dữ liệu huấn luyện chưa đủ đa dạng. Cần kiểm tra chéo (cross-validation) và mở rộng tập dữ liệu với nhiều trường hợp lâm sàng hơn nữa.

Tối ưu hiệu năng: Thời gian huấn luyện SVM tăng đáng kể với dữ liệu lớn. Có thể thay bằng ứng dụng các mô hình deep learning (CNN) để tự động trích xuất đặc trưng.

* Kết luận chung:

Kết quả đã chứng minh tính khả thi của SVM trong bài toán phân loại ảnh y tế, mở ra hướng tiếp cận mới cho lĩnh vực chẩn đoán hình ảnh. Dù còn nhiều thử thách về tính tổng quát và độ phức tạp, kết quả ban đầu cho thấy triển vọng ứng dụng trí tuệ nhân tạo vào y học, góp phần nâng cao chất lượng chăm sóc sức khỏe cộng đồng.

# **TÀI LIỆU THAM KHẢO**

1. Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." Signal Processing, IEEE Transactions on 45.11 (1997): 2673-2681.2. Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan

2. Salehinejad, Hojjat; Sankar, Sharan; Barfett, Joseph; Colak, Errol; Valaee, Shahrokh (2017). "Recent Advances in Recurrent Neural Networks". arXiv:1801.01078 [cs.NE].

3. Graves, Alex, Santiago Fernández, and Jürgen Schmidhuber. "Bidirectional LSTM networks for improved phoneme classification and recognition." Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005. Springer Berlin Heidelberg, 2005. 799-804.

4. Graves, Alan, Navdeep Jaitly, and Abdel-rahman Mohamed. "Hybrid speech recognition with deep bidirectional LSTM." Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. IEEE, 2013.

5. Sundermeyer, Martin, et al. "Translation modeling with bidirectional recurrent neural networks." Proceedings of the Conference on Empirical Methods on Natural Language Processing, October. 2014