**A black background with blue lines

Description automatically generatedĐẠI HỌC QUỐC GIA HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

**A blue logo with a flower and a planet

Description automatically generated**

**BÁO CÁO MÔN HỌC**

**NHẬP MÔN THỊ GIÁC MÁY TÍNH**

**ĐỀ TÀI**

**PHÂN LOẠI GÀ VỊT VÀ GIA CẦM KHÁC**

**Giảng viên hướng dẫn : TS. Mai Tiến Dũng**

**Sinh viên thực hiện:**

Nguyễn Minh Thiện 22521391

Đinh Quốc Thịnh 22521402

**TP. Hồ Chí Minh, ngày 13 tháng 1 năm 2024**

# **Mục Lục**

[**LỜI MỞ ĐẦU 5**](#_Toc187869938)

[**THÔNG TIN THÀNH VIÊN 6**](#_Toc187869939)

[**Chương 1 : Tổng Quan 7**](#_Toc187869940)

[**1.1. Mô tả bài toán 7**](#_Toc187869941)

[**1.2. Lí do thực hiện bài toán 7**](#_Toc187869942)

[**1.3. Phát biểu bài toán 7**](#_Toc187869943)

[**Chương 2: Phương pháp thực hiện bài toán 8**](#_Toc187869944)

[**2.1. Tiền xử lý dữ liệu 8**](#_Toc187869945)

[**2.1.1. Loại bỏ nền ảnh ( Thư viện rembg) 8**](#_Toc187869946)

[**2.1.1.1. Giới thiệu 8**](#_Toc187869947)

[**2.1.1.2. Cài đặt 8**](#_Toc187869948)

[**2.1.1.3. Nhận xét 9**](#_Toc187869949)

[**2.2. Trích xuất dữ liệu 10**](#_Toc187869950)

[**2.2.1. Đặc trưng SIFT 10**](#_Toc187869951)

[**2.2.1.1. Định nghĩa 10**](#_Toc187869952)

[**2.2.1.2. Nguyên lý hoạt động 10**](#_Toc187869953)

[**2.2.1.2.1. Phát hiện điểm đặc trưng (Keypoints) 10**](#_Toc187869954)

[**2.2.1.2.2. Gán hướng cho điểm đặc trưng 10**](#_Toc187869955)

[**2.2.1.2.3. Tạo mô tả đặc trưng (Descriptor) 11**](#_Toc187869956)

[**2.2.1.2.4. Xếp hạng và chọn lọc 11**](#_Toc187869957)

[**2.2.1.3. Cài đặt 11**](#_Toc187869958)

[**2.2.1.4. Nhận xét 12**](#_Toc187869959)

[**2.2.2. Đặc trưng HOG 13**](#_Toc187869960)

[**2.2.2.1. Định nghĩa 13**](#_Toc187869961)

[**2.2.2.2. Nguyên lý hoạt động 13**](#_Toc187869962)

[**2.2.2.2.1. Tính Gradient của hình ảnh 13**](#_Toc187869963)

[**2.2.2.2.2. Phân vùng thành các ô ( cell) 13**](#_Toc187869964)

[**2.2.2.2.3. Chuẩn hóa các khối (block) 13**](#_Toc187869965)

[**2.2.2.2.4. Kết hợp và tạo vector đặc trưng 13**](#_Toc187869966)

[**2.2.2.3. Cài đặt 14**](#_Toc187869967)

[**2.2.2.4. Nhận xét 14**](#_Toc187869968)

[**2.3. Thuật toán máy học 15**](#_Toc187869969)

[**2.3.1. Thuật toán Support Vector Machine ( SVM) 15**](#_Toc187869970)

[**2.3.1.1. Lịch sử ra đời 15**](#_Toc187869971)

[**2.3.1.2. Nguyên lý hoạt động 15**](#_Toc187869972)

[**2.3.1.2.1. Tìm siêu phẳng tối ưu 15**](#_Toc187869973)

[**2.3.1.2.2. Hàm mục tiêu trong SVM 15**](#_Toc187869974)

[**2.3.1.2.3. Dự đoán nhãn 16**](#_Toc187869975)

[**2.3.1.3. Cài đặt trong bài toán 16**](#_Toc187869976)

[**2.3.2. Thuật toán Softmax Regression 17**](#_Toc187869977)

[**2.3.2.1. Lịch sử ra đời 17**](#_Toc187869978)

[**2.3.2.2. Nguyên lý hoạt động 18**](#_Toc187869979)

[**2.3.2.2.1. Hàm Softmax và Ý nghĩa 18**](#_Toc187869980)

[**2.3.2.2.2. Hàm mục tiêu trong Softmax Regression 18**](#_Toc187869981)

[**2.3.2.2.3. Dự đoán nhãn 19**](#_Toc187869982)

[**2.3.2.3. Cài đặt trong bài toán 19**](#_Toc187869983)

[**2.4. Phương pháp GridSearchCV 21**](#_Toc187869984)

[**2.4.1. Nguyên lý hoạt động 21**](#_Toc187869985)

[**2.4.2. Nhận xét về GridsearchCV 21**](#_Toc187869986)

[**2.4.3. GridsearchCV cho SVM 21**](#_Toc187869987)

[**2.4.4. GridsearchCV cho Softmax Regression 22**](#_Toc187869988)

[**Chương 3: Thực Nghiệm và Kết Quả 23**](#_Toc187869989)

[**3.1. Bộ dữ liệu 23**](#_Toc187869990)

[**3.1.1. Tổng quan 23**](#_Toc187869991)

[**3.1.2. Cấu trúc bộ dữ liệu 23**](#_Toc187869992)

[**3.2. Độ đo 23**](#_Toc187869993)

[**3.2.1. Accuracy 23**](#_Toc187869994)

[**3.2.2. Precision 24**](#_Toc187869995)

[**3.2.3. Recall 24**](#_Toc187869996)

[**3.2.4. F1-score 24**](#_Toc187869997)

[**3.3. Kết quả thực nghiệm 24**](#_Toc187869998)

[**3.3.1. SVM 24**](#_Toc187869999)

[**3.3.1.1. SVM kết hợp HOG 24**](#_Toc187870000)

[**3.3.1.2. SVM kết hợp SIFT 25**](#_Toc187870001)

[**3.3.2. Softmax regression 26**](#_Toc187870002)

[**3.3.2.1. Softmax regression kết hợp HOG 27**](#_Toc187870003)

[**3.3.2.2. Softmax regression kết hợp SIFT 28**](#_Toc187870004)

[**Tài liệu tham khảo 30**](#_Toc187870005)

# **LỜI MỞ ĐẦU**

Trong bối cảnh ngành chăn nuôi tại Việt Nam ngày càng phát triển, việc phân loại và quản lý gia cầm đóng vai trò quan trọng trong chuỗi cung ứng nông nghiệp, đặc biệt khi nhu cầu tiêu thụ thực phẩm sạch và truy xuất nguồn gốc ngày càng gia tăng. Tuy nhiên, phương pháp phân loại thủ công truyền thống dựa trên quan sát của con người thường không đảm bảo độ chính xác, đặc biệt trong các điều kiện bất lợi như ánh sáng yếu, mật độ gia cầm đông đúc, hoặc khi có sự tương đồng cao giữa các loài. Điều này đặt ra nhu cầu cấp thiết về các giải pháp tự động hóa, giúp phân loại gia cầm nhanh chóng, chính xác và hiệu quả.

Gia cầm như gà, vịt và các loài khác thường có sự đa dạng lớn về hình dáng, màu sắc, và đặc điểm bề ngoài, tạo nên nhiều thách thức trong việc nhận diện và phân loại. Các yếu tố môi trường như ánh sáng, góc nhìn, hoặc chuyển động của gia cầm cũng làm giảm hiệu quả của các mô hình nhận diện. Do đó, việc kết hợp các kỹ thuật trích xuất đặc trưng tiên tiến như HOG (Histogram of Oriented Gradients) và SIFT (Oriented FAST and Rotated BRIEF) với các thuật toán học máy như Support Vector Machine (SVM) và Softmax Regression sẽ là giải pháp khả thi. Những phương pháp này không chỉ giúp trích xuất đặc trưng hình ảnh hiệu quả mà còn tối ưu hóa quá trình phân loại, đảm bảo độ chính xác cao ngay cả trong môi trường phức tạp.

Nghiên cứu này không chỉ hướng đến việc phân loại gia cầm trong môi trường nuôi nhốt mà còn có tiềm năng áp dụng trong các dây chuyền tự động tại các cơ sở chế biến thực phẩm. Với dữ liệu đầu vào là hình ảnh thu thập từ thực tế, hệ thống được thiết kế nhằm xử lý hiệu quả các tình huống phức tạp như gia cầm di chuyển nhanh hoặc có sự giao thoa giữa các cá thể. Thuật toán SVM hỗ trợ phân tách các lớp dữ liệu phức tạp nhờ vào biên phân tách tối ưu, trong khi Softmax Regression đảm bảo việc phân loại đa lớp một cách chính xác và hiệu quả.

Hơn nữa, nghiên cứu còn đặt nền móng cho các ứng dụng công nghệ cao trong nông nghiệp thông minh, như giám sát sức khỏe gia cầm, dự đoán năng suất đàn, và tối ưu hóa chuỗi cung ứng. Đây chính là một bước tiến quan trọng, không chỉ đáp ứng yêu cầu sản xuất hiện đại mà còn góp phần thúc đẩy ngành nông nghiệp Việt Nam hội nhập sâu rộng vào nền kinh tế số toàn cầu.

**THÔNG TIN THÀNH VIÊN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Họ tên** | **MSSV** | **Phân công** | **Mức độ hoàn thành** |
| Nguyễn Minh Thiện | 22521391 | Tiền xử lý dữ liệu(Thư viện rembg) và trích xuất đặc trưng(SIFT), phát triển và đánh giá thuật toán (SVM) | 100% |
| Đinh Quốc Thịnh | 22521402 | Trích xuất đặc trưng(HOG), phát triển và đánh giá thuật toán (Soft max Regressioon) | 100% |

# **Chương 1 : Tổng Quan**

* 1. **Mô tả bài toán**

Trong lĩnh vực nông nghiệp, việc phân loại gia cầm như gà, vịt và các loại khác dựa trên đặc điểm hình dáng, kích thước, màu sắc và tư thế là một thách thức lớn do sự đa dạng của chúng. Số lượng lớn các cá thể trong một trang trại và sự biến đổi trong môi trường như ánh sáng, góc nhìn và chất lượng hình ảnh càng làm tăng độ phức tạp của bài toán. Trong đồ án này, dữ liệu được thu thập chủ yếu từ các hình ảnh thực tế tại trang trại, từ nhiều góc độ khác nhau. Mục tiêu là xây dựng một hệ thống tự động, có khả năng nhận diện và phân loại chính xác gia cầm thành các nhóm chính: gà, vịt và các loại gia cầm khác.

* 1. **Lí do thực hiện bài toán**

Phân loại gia cầm tự động là một vấn đề quan trọng trong ngành nông nghiệp, giúp nâng cao hiệu quả quản lý và chăm sóc gia cầm. Việc phân loại chính xác gà, vịt và các loại gia cầm khác sẽ hỗ trợ nông dân trong việc theo dõi sức khỏe, điều kiện sống và sản lượng của từng loài, từ đó cải thiện năng suất và chất lượng sản phẩm. Đây là một ứng dụng thiết thực tại Việt Nam, nơi nông nghiệp chiếm vai trò quan trọng trong nền kinh tế. Hệ thống phân loại này không chỉ giúp giảm thiểu công sức lao động thủ công mà còn tạo tiền đề cho các công nghệ tiên tiến như giám sát tự động, phân tích dữ liệu nông nghiệp, và hỗ trợ quản lý trang trại thông minh. Trong tương lai, hệ thống này sẽ đóng vai trò quan trọng trong việc phát triển nông nghiệp 4.0 và nâng cao hiệu quả sản xuất nông sản.

* 1. **Phát biểu bài toán** 
     + **Input:** Tập dữ liệu gồm các ảnh và nhãn tưởng với ảnh , trong đo với là đặc trưng của ảnh và là nhãn của ảnh với là tập nhãn.
     + **Output:** Nhãn dự đoán trong đó là mô hỉnh phân loại, dự đoán nhãn của ảnh .
     + Mô hình phân loại sẽ học tập từ dữ liệu D để phân loại các ảnh vào ba loại: Gà, vịt và gia cầm khác.

# **Chương 2: Phương pháp thực hiện bài toán**

* 1. **Tiền xử lý dữ liệu**
     1. **Loại bỏ nền ảnh ( Thư viện rembg)**
        1. **Giới thiệu**

Trong lĩnh vực xử lý ảnh và ứng dụng trí tuệ nhân tạo, việc tách nền (background) ra khỏi hình ảnh là một nhiệm vụ quan trọng, đặc biệt trong các ứng dụng như thương mại điện tử, thiết kế đồ họa, và tiền xử lý dữ liệu cho học máy. Thư viện rembg ra đời như một công cụ mạnh mẽ và dễ sử dụng, cho phép tự động loại bỏ nền khỏi hình ảnh một cách nhanh chóng và chính xác.

rembg được phát triển dựa trên các mô hình học sâu hiện đại như MODNet và U2Net, giúp xử lý hiệu quả ngay cả các hình ảnh phức tạp với nền không đồng nhất. Với giao diện đơn giản và khả năng tích hợp linh hoạt, rembg không chỉ phù hợp cho các nhà phát triển mà còn dành cho những người không chuyên về lập trình.

Các tính năng nổi bật của thư viện bao gồm:

* Tự động tách nền chính xác cao mà không cần sự can thiệp thủ công.
* Hỗ trợ nhiều định dạng ảnh phổ biến như JPEG, PNG.
* Khả năng xử lý hàng loạt, tiết kiệm thời gian cho các dự án lớn.
* Dễ dàng cài đặt và sử dụng chỉ với vài dòng lệnh đơn giản.

Thư viện này đóng vai trò quan trọng trong việc tối ưu hóa quy trình xử lý ảnh, đặc biệt trong các bài toán nhận diện và phân loại đối tượng, tạo điều kiện thuận lợi cho các ứng dụng thực tế như phân loại sản phẩm, nhận diện động vật hoặc phân loại gia cầm trong ngành chăn nuôi.

Với sức mạnh và tính ứng dụng rộng rãi, rembg đang trở thành một công cụ không thể thiếu trong lĩnh vực xử lý ảnh hiện đại**.**

* + - 1. **Cài đặt**

Đoạn mã Python dưới đây minh họa cách sử dụng thư viện **rembg** để loại bỏ nền (background) của hình ảnh:

A computer screen with text and images

Description automatically generated

**Hình ảnh minh họa:**

A collage of a baby chick

Description automatically generated

* + - 1. **Nhận xét**
* **Ưu điểm :**
  + Dễ sử dụng: Giao diện đơn giản, phù hợp cả với người không chuyên.
  + Tự động hóa: Loại bỏ nền tự động, nhanh chóng và chính xác.
* **Nhược điểm:**
  + Tốn tài nguyên
  + Phụ thuộc mô hình**:** Đòi hỏi cấu hình máy đủ mạnh để chạy nhanh.
  1. **Trích xuất dữ liệu**
     1. **Đặc trưng SIFT**
        1. **Định nghĩa**

SIFT (Scale-Invariant Feature Transform) là một thuật toán trích xuất và mô tả đặc trưng trong lĩnh vực xử lý ảnh và thị giác máy tính. Nó được phát triển như một giải pháp phát hiện và mô tả các đặc trưng cục bộ trong hình ảnh. SIFT có khả năng bất biến (invariant) với các thay đổi về tỷ lệ, xoay, dịch chuyển, và tương đối ổn định trước sự thay đổi về ánh sáng và nhiễu.

* + - 1. **Nguyên lý hoạt động**
         1. **Xây dựng không gian tỷ lệ (Scale-Space Construction)**
* SIFT phát hiện các điểm đặc trưng bằng cách tìm kiếm các **điểm cực trị (extrema)** trong không gian tỷ lệ (scale-space).
* Không gian tỷ lệ được tạo ra bằng cách áp dụng **Gaussian Blur** lên hình ảnh gốc với các mức tỷ lệ (scale) khác nhau.
* Sau đó, SIFT tính toán sự khác biệt giữa hai ảnh Gaussian liền kề (Difference of Gaussians - DoG)
* Mục tiêu là tìm ra các điểm cực trị trong DoG, tức các điểm có cường độ sáng lớn nhất hoặc nhỏ nhất so với các điểm xung quanh ở cả không gian và tỷ lệ.
  + - * 1. **Xác định và lọc keypoints (Keypoint Localization)**
* Các điểm cực trị được kiểm tra để loại bỏ:
* **Điểm biên (Edge-like points):** Các điểm có độ cong cao nhưng không đặc trưng.
* **Điểm nhiễu:** Các điểm không đủ ý nghĩa (low contrast).
* Phương pháp toán học: SIFT sử dụng ma trận Hessian để phân tích độ cong và loại bỏ các điểm không phù hợp.
  + - * 1. **Định hướng keypoints (Orientation Assignment)**
* Mỗi keypoint được gán một hoặc nhiều hướng dựa trên gradient cường độ của vùng xung quanh nó.
* Tính toán gradient
* Hướng gradient chủ đạo (dominant direction) được xác định dựa trên histogram của các hướng gradient.
  + - * 1. **Mô tả đặc trưng (Keypoint Descriptor)**
* SIFT tạo một vector đặc trưng 128 chiều để mô tả mỗi keypoint. Vector này dựa trên gradient trong một vùng lân cận (patch) của keypoint, chia nhỏ thành các ô vuông 4x4.
* Mỗi ô vuông có histogram 8 hướng gradient, dẫn đến 4×4×8=128 giá trị.
  + - 1. **Cài đặt**

Đoạn mã Python dưới đây minh họa trích xuất đặc trưng SIFT:

A screenshot of a computer program

Description automatically generated

Hình ảnh minh họa:

A close-up of a duck

Description automatically generated

* + - 1. **Nhận xét**
* **Ưu điểm:**
  + **Tốc độ cao:**
    - SIFT nhanh hơn so với SIFT và SURF nhờ việc sử dụng thuật toán FAST để phát hiện keypoints và BRIEF để tạo descriptor.
    - Thích hợp cho các ứng dụng thời gian thực như SLAM (Simultaneous Localization and Mapping).
  + **Khả năng chống xoay và nhiễu:**
    - SIFT đảm bảo tính xoay vòng (rotation-invariant) nhờ gán hướng cho keypoints.
    - Các mô tả nhị phân (binary descriptors) giúp tăng khả năng chống lại nhiễu trong ảnh.
  + **Không bị ràng buộc bản quyền:**
    - SIFT là mã nguồn mở, miễn phí, dễ dàng sử dụng trong các dự án cá nhân và thương mại.
  + **Hiệu quả trong việc ghép nối đặc trưng (Feature Matching):**
    - Descriptor nhị phân của SIFT phù hợp với các phương pháp so khớp nhanh như Hamming Distance, giúp tăng tốc độ xử lý.
* **Nhược điểm:**
  + Không ổn định trong biến đổi tỷ lệ:
    - Dù tốt ở tính xoay vòng, SIFT kém hơn khi xử lý các biến đổi tỷ lệ (scale-invariant) so với SIFT và SURF.
  + Hạn chế với cảnh phức tạp:
    - SIFT không hoạt động tốt trong các cảnh có độ phức tạp cao hoặc nhiều chi tiết, dẫn đến mất mát thông tin quan trọng.
  + Descriptor nhị phân ít thông tin hơn:
    - Mặc dù nhanh, descriptor nhị phân của SIFT có thể thiếu chính xác trong việc phân biệt các đặc trưng phức tạp.
  + Hiệu suất giảm trong ánh sáng yếu:
    - SIFT phụ thuộc vào cường độ pixel, vì vậy nó có thể gặp khó khăn trong các điều kiện ánh sáng yếu hoặc ảnh bị thiếu sáng.
    1. **Đặc trưng HOG**
       1. **Định nghĩa**

**HOG (Histogram of Oriented Gradients)** là một thuật toán trích xuất đặc trưng được sử dụng phổ biến trong xử lý ảnh và thị giác máy tính. HOG biểu diễn sự phân bố cục bộ của gradient hướng và cường độ trong một ảnh hoặc một phần ảnh, giúp mô tả hình dạng và cấu trúc của đối tượng.

HOG được thiết kế đặc biệt hiệu quả trong việc nhận diện vật thể (như nhận diện con người) nhờ khả năng biểu diễn đặc trưng dựa trên gradient, không bị ảnh hưởng nhiều bởi ánh sáng hoặc màu sắc.

* + - 1. **Nguyên lý hoạt động**
         1. **Tính Gradient của hình ảnh**
* Gradient được tính bằng cách áp dụng bộ lọc Sobel hoặc các phương pháp tương tự trên ảnh.
* Gradient cho biết hướng thay đổi mạnh nhất của cường độ pixel, giúp xác định cạnh và cấu trúc trong ảnh.
* Kết quả là hai ma trận:
* Ma trận độ lớn (Magnitude).
* Ma trận hướng (Orientation).
  + - * 1. **Phân vùng thành các ô ( cell)**
* Ảnh được chia thành các ô nhỏ (cells), ví dụ: 8x8 pixel.
* Trong mỗi ô, một histogram của hướng gradient được tính toán, với các hướng được chia thành các khoảng, ví dụ: 0-180° hoặc 0-360°.
* Mỗi khoảng hướng nhận một giá trị tích lũy dựa trên độ lớn gradient của các pixel thuộc hướng đó.
  + - * 1. **Chuẩn hóa các khối (block)**
* Nhiều ô được nhóm lại thành khối (block), ví dụ: 2x2 ô.
* Các histogram của các ô trong một khối được chuẩn hóa để giảm ảnh hưởng của sự thay đổi ánh sáng hoặc độ tương phản.
* Phép chuẩn hóa thường sử dụng **L2-norm** hoặc **L1-norm**, giúp tăng tính ổn định và khả năng chống nhiễu.
  + - * 1. **Kết hợp và tạo vector đặc trưng**
  + Các vector đặc trưng từ tất cả các khối trong ảnh được ghép nối thành một vector đặc trưng duy nhất.
  + Vector đặc trưng này biểu diễn toàn bộ ảnh hoặc vùng quan tâm, dùng để huấn luyện hoặc so sánh trong các bài toán nhận diện.
    - 1. **Cài đặt**

HOG thường được cài đặt thông qua các thư viện xử lý ảnh như OpenCV, scikit-image trong Python:

A computer screen with white text

Description automatically generated

Hình ảnh minh họa

A duck and hog image

Description automatically generated

* + - 1. **Nhận xét**
* **Ưu điểm:**
  + HOG có hiệu quả cao trong việc mô tả hình dạng và cấu trúc đối tượng, đặc biệt trong nhận diện con người.
  + Khả năng chống nhiễu tốt nhờ chuẩn hóa histogram.
  + Không bị ảnh hưởng bởi sự thay đổi ánh sáng hoặc màu sắc của ảnh.
* **Nhược điểm:**
  + Yêu cầu nhiều tài nguyên tính toán khi xử lý ảnh kích thước lớn.
  + HOG không tốt trong việc xử lý các biến dạng phức tạp như thay đổi góc nhìn hoặc tỷ lệ.
  + Phụ thuộc nhiều vào các tham số như kích thước ô, khối và khoảng hướng, đòi hỏi tinh chỉnh kỹ lưỡng.
  1. **Thuật toán máy học**
     1. **Thuật toán Support Vector Machine ( SVM)**
        1. **Lịch sử ra đời**
  + SVM được giới thiệu lần đầu tiên bởi **Vladimir Vapnik** và **Alexey Chervonenkis** vào năm 1963 trong lý thuyết học thống kê.
  + Phiên bản SVM hiện đại, sử dụng **kernel trick**, được phát triển vào năm 1992.
  + Thuật toán đã trở thành một trong những phương pháp mạnh mẽ và phổ biến nhất trong học máy nhờ khả năng giải quyết bài toán với dữ liệu phi tuyến tính.
    - 1. **Nguyên lý hoạt động**
         1. **Tìm siêu phẳng tối ưu**
* Siêu phẳng tối ưu (**Optimal Hyperplane**) là đường (hoặc mặt phẳng) phân tách dữ liệu hai lớp với khoảng cách lớn nhất từ siêu phẳng đến các điểm dữ liệu gần nhất (gọi là margin).
* Các điểm dữ liệu nằm gần siêu phẳng nhất gọi là **support vectors**, và chúng quyết định vị trí của siêu phẳng.
  + - * 1. **Hàm mục tiêu trong SVM**
* Hàm mục tiêu của SVM là tối ưu hóa **margin** giữa các lớp dữ liệu, thường được biểu diễn qua phương trình:
  + - Với ràng buộc:
    - : nhãn của dữ liệu
    - **:** vector trọng số của siêu phẳng.
    - : vector đặc trưng của dữ liệu
  + Để xử lý dữ liệu không tuyến tính, SVM sử dụng **kernel trick** để ánh xạ dữ liệu sang không gian đặc trưng cao hơn, nơi các lớp dữ liệu có thể phân tách tuyến tính.
    - * 1. **Dự đoán nhãn**
* Với một điểm dữ liệu , dự đoán nhãn dựa vào hàm:
  + Nếu Dự đoán nhãn +1.
  + Nếu Dự đoán nhãn -1.
    - 1. **Cài đặt trong bài toán**

SVM có thể dễ dàng được cài đặt bằng các thư viện học máy phổ biến như scikit-learn trong Python:

A computer screen shot of a program code

Description automatically generated

Giải thích các tham số trong mô hình SVM :

**1. C: Tham số điều chỉnh mức phạt lỗi phân loại**

* Kiểm soát mức độ chấp nhận lỗi trong quá trình phân loại.
* Giá trị nhỏ: Cho phép một số lỗi phân loại, tăng khả năng tổng quát hóa.
* Giá trị lớn: Yêu cầu phân loại chính xác hơn, giảm lỗi nhưng dễ dẫn đến overfitting.

**2. degree: Bậc của đa thức trong kernel 'poly'**

* Xác định bậc của đa thức được sử dụng trong kernel đa thức.
* Quyết định độ phức tạp của biên phân cách khi sử dụng kernel 'poly'

**3. gamma: Hệ số kiểm soát ảnh hưởng của điểm dữ liệu trong kernel**

* Quyết định mức độ ảnh hưởng của từng điểm dữ liệu đến ranh giới phân cách.
* Giá trị nhỏ: Ảnh hưởng rộng, tổng quát hóa tốt hơn.
* Giá trị lớn: Ảnh hưởng hẹp, tập trung vào các điểm gần, dễ dẫn đến overfitting.

**4. kernel: Hàm kernel dùng để ánh xạ dữ liệu**

* 'poly': Sử dụng kernel đa thức, phù hợp với biên phân tách phi tuyến phức tạp.
* 'linear': Kernel tuyến tính, phù hợp với dữ liệu tuyến tính.
* 'rbf': Kernel Gaussian, xử lý tốt các biên phân tách phi tuyến mượt.
* 'sigmoid': Kernel sigmoid, giống hàm kích hoạt trong mạng nơ-ron.
  + 1. **Thuật toán Softmax Regression**
       1. **Lịch sử ra đời**
* Softmax Regression, hay còn gọi là Multinomial Logistic Regression, là một mở rộng của thuật toán Logistic Regression nhằm giải quyết các bài toán phân loại với nhiều lớp. Ban đầu, Logistic Regression được phát triển để giải quyết các bài toán phân loại nhị phân (hai lớp), và được sử dụng rộng rãi trong các bài toán như phân loại email (spam hay không spam), dự đoán kết quả của các sự kiện nhị phân trong các lĩnh vực y tế, tài chính, và khoa học.
* Tuy nhiên, khi các bài toán phân loại có nhiều lớp (như phân loại các loài hoa trong tập dữ liệu Iris), Logistic Regression không thể áp dụng trực tiếp vì không có cách thức để phân loại nhiều lớp. Để giải quyết vấn đề này, Softmax Regression đã được phát triển, là một kỹ thuật mở rộng của Logistic Regression.
* 1972: Logistic Regression được phát triển bởi David R. Cox trong bối cảnh thống kê, sử dụng hàm sigmoid để dự đoán xác suất thuộc về một lớp trong bài toán phân loại nhị phân. Sau đó, mô hình này được ứng dụng trong các bài toán phân loại và phân tích hồi quy.
* 1980s: Khi các bài toán phân loại bắt đầu có nhiều hơn hai lớp, thuật toán Multinomial Logistic Regression đã được phát triển để giải quyết bài toán này bằng cách sử dụng hàm Softmax. Softmax là một hàm toán học dùng để chuẩn hóa các giá trị thành xác suất trong các bài toán phân loại đa lớp.
* 1989: Softmax được sử dụng trong các bài toán học máy và học sâu (deep learning), đặc biệt trong mạng nơ-ron đa lớp. Sự phát triển này giúp các mô hình học máy có thể phân loại được nhiều nhãn trong một lần dự đoán.
* 1990s – 2000s: Thuật toán Softmax Regression trở nên phổ biến trong các bài toán học máy thực tế, nhất là trong các ứng dụng như nhận dạng văn bản, nhận dạng hình ảnh, và phân tích dữ liệu lớn. Nó được áp dụng rộng rãi nhờ vào tính toán đơn giản và khả năng mở rộng tốt với dữ liệu lớn.
  + - 1. **Nguyên lý hoạt động**
         1. **Hàm Softmax và Ý nghĩa**

Hàm Softmax là một hàm toán học dùng để chuyển đổi đầu ra của mô hình (các giá trị logits) thành xác suất. Softmax giúp mô hình có thể đưa ra dự đoán phân loại cho bài toán có nhiều lớp (multi-class).

* Công thức của hàm Softmax cho một lớp kkk trong một mô hình phân loại có k lớp là:
* là giá trị đầu ra của mô hình cho lớp k,

Điều này đảm bảo rằng tổng các xác suất của tất cả các lớp sẽ bằng 1, tức là mỗi lớp có xác suất giữa 0 và 1.

* **Ý nghĩa:** Hàm Softmax giúp mô hình biến các giá trị vô hướng (logits) thành các xác suất có ý nghĩa, thể hiện khả năng thuộc về từng lớp cụ thể.
  + - * 1. **Hàm mục tiêu trong Softmax Regression**

Hàm mục tiêu trong Softmax Regression là Cross-Entropy Loss (mất mát chéo), được sử dụng để đánh giá mức độ phù hợp giữa nhãn thực tế và nhãn dự đoán của mô hình. Cross-Entropy Loss giúp tối ưu hóa mô hình bằng cách giảm thiểu sự khác biệt giữa xác suất dự đoán và nhãn thực tế.

Công thức của Cross-Entropy Loss cho một bài toán phân loại nhiều lớp là:

Trong đó:

* là nhãn thực tế cho lớp k,
* là xác suất dự đoán lớp k ( do Softmax tính toán).
  + - * 1. **Dự đoán nhãn**

Sau khi huấn luyện mô hình, khi đưa một mẫu mới vào, mô hình sẽ tính toán các giá trị logits cho từng lớp. Sau đó, hàm Softmax sẽ chuyển đổi các giá trị này thành các xác suất.

Dự đoán nhãn: Nhãn có xác suất cao nhất sẽ được chọn làm dự đoán của mô hình. Điều này có nghĩa là mô hình sẽ chọn lớp có xác suất lớn nhất trong số tất cả các lớp.

Ví dụ: Nếu mô hình dự đoán các xác suất sau:

* Lớp 1: 0.2
* Lớp 2: 0.7
* Lớp 3: 0.1

Mô hình sẽ chọn lớp 2, vì xác suất của lớp này là cao nhất (0.7).

* + - 1. **Cài đặt trong bài toán**

Cài đặt Softmax Regression trong bài toán phân loại đa lớp (multi-classclassification) sử dụng thư viện sklearn:

A computer screen shot of a program

Description automatically generated

Giải thích các tham số:

* **solver**: Tham số này quyết định thuật toán tối ưu hóa được sử dụng để tìm ra các trọng số trong mô hình.
* **'lbfgs'**: Đây là một thuật toán tối ưu hóa hàm log-likelihood (tối ưu theo lý thuyết Newton-Raphson). Được sử dụng phổ biến cho bài toán phân loại nhiều lớp với mô hình Logistic Regression.
* **'newton-cg'**: Thuật toán Newton’s method dùng để tối ưu hóa.
* **'liblinear'**: Phù hợp cho các bài toán phân loại nhị phân, không được tối ưu cho bài toán đa lớp.
* **'saga'**: Một thuật toán dựa trên stochastic gradient descent, thích hợp khi có số lượng đặc trưng lớn.
* **multi\_class**: Tham số này xác định cách thức mô hình xử lý các bài toán phân loại đa lớp.
* **'multinomial'**: Sử dụng **Softmax** để phân loại nhiều lớp. Đây là cách mô hình Logistic Regression mở rộng để xử lý các bài toán phân loại với nhiều lớp thay vì chỉ hai lớp (như Logistic Regression truyền thống).
* **'ovr' (One-vs-Rest)**: Phương pháp phân loại nhị phân cho mỗi lớp, và kết quả sẽ là lớp có xác suất cao nhất.
* **max\_iter**: Tham số này xác định số vòng lặp tối đa mà thuật toán tối ưu hóa được phép thực hiện để hội tụ.
* **default=100**: Giá trị mặc định là 100, nhưng bạn có thể điều chỉnh giá trị này nếu mô hình không hội tụ trong số vòng lặp mặc định.
* **Lưu ý**: Nếu số vòng lặp ít quá, thuật toán có thể không hội tụ và cần tăng số vòng lặp.
* **C**: Tham số này quyết định mức độ điều chỉnh (regularization) của mô hình.
* **C** là nghịch đảo của tham số regularization. Giá trị **C** lớn tương ứng với ít điều chỉnh (mô hình sẽ tập trung vào việc giảm thiểu sai số huấn luyện nhiều hơn).
* Giá trị nhỏ của **C** giúp mô hình điều chỉnh mạnh hơn, giảm overfitting nhưng có thể làm mô hình mất độ chính xác trong huấn luyện.
* **penalty**: Tham số này xác định loại hình phạt regularization được áp dụng.
* **'l2'**: Phạt theo chuẩn L2, giảm thiểu sự thay đổi quá lớn trong trọng số (thường được dùng trong Logistic Regression).
* **'l1'**: Phạt theo chuẩn L1, có thể giúp giảm bớt các trọng số không quan trọng và có tác dụng chọn lọc tính năng.
* **tol**: Tham số này xác định độ chính xác của thuật toán khi hội tụ. Nếu sự thay đổi trong giá trị hàm mục tiêu nhỏ hơn giá trị tol, thuật toán sẽ dừng lại.
* **Giá trị mặc định = 1e-4**.
* **max\_iter**: Số vòng lặp tối đa để thuật toán hội tụ.
  1. **Phương pháp GridSearchCV**
     1. **Nguyên lý hoạt động**

GridSearchCV là một phương pháp tìm kiếm tham số tốt nhất (hyperparameters) cho mô hình bằng cách thử tất cả các kết hợp có thể của các tham số đã cho trong không gian tìm kiếm. Phương pháp này giúp tìm ra các giá trị tối ưu cho các tham số, nhằm cải thiện hiệu suất của mô hình.

* GridSearchCV thực hiện phép thử với tất cả các giá trị của các tham số đã định nghĩa và sau đó sử dụng Cross-Validation để đánh giá mô hình với mỗi bộ tham số.
* Mỗi lần thử nghiệm sẽ trả về một điểm số (score) từ việc sử dụng cross-validation.
* Sau khi thử tất cả các kết hợp tham số, GridSearchCV chọn tham số nào cho kết quả tốt nhất (thường là dựa trên điểm số chính xác cao nhất).
  + 1. **Nhận xét về GridsearchCV**
* Ưu điểm:
* Tìm kiếm toàn diện qua tất cả các giá trị tham số, giúp tìm ra bộ tham số tốt nhất.
* Có thể cải thiện đáng kể hiệu suất mô hình bằng cách tối ưu hóa các tham số.
* Tự động hóa quá trình tối ưu hóa tham số mà không cần phải thử nghiệm thủ công.
* Nhược điểm:
* Quá trình tìm kiếm có thể rất tốn kém về mặt thời gian và tài nguyên, đặc biệt khi không gian tìm kiếm có rất nhiều tham số và giá trị.
* Nếu không gian tìm kiếm quá rộng, việc chạy GridSearchCV có thể trở nên không khả thi.
  + 1. **GridsearchCV cho SVM**

Cài đặt kĩ thuật tìm kiếm tham số bằng phương pháp GridsearchCV:

A screenshot of a computer program

Description automatically generated

* + 1. **GridsearchCV cho Softmax Regression**

Cài đặt kĩ thuật tìm kiếm tham số bằng phương pháp GridsearchCV :

**A screen shot of a computer code

Description automatically generated**

**Chương 3: Thực Nghiệm và Kết Quả**

* 1. **Bộ dữ liệu**
     1. **Tổng quan**

Dữ liệu đóng vai trò cốt lõi trong việc xây dựng các mô hình học máy. Để đảm bảo hiệu quả và độ chính xác cao, bộ dữ liệu cần được xây dựng dựa trên các tiêu chí rõ ràng về đối tượng, nguồn thu thập và tính đa dạng.

Trong nghiên cứu này, chúng tôi tập trung thu thập hình ảnh các loài gia cầm như gà, vịt và một số đối tượng khác để xây dựng một bộ dữ liệu tiêu chuẩn. Quá trình thu thập được thực hiện thông qua hai nguồn chính:

1. Nguồn trực tuyến: Các hình ảnh từ các trang web công khai và kho dữ liệu mở: **Google** Với các từ khóa: Gà, Vịt, Ngỗng, Chim …
2. Nguồn thực tế: Tự chụp ảnh với thiết bị di động hoặc máy ảnh, đảm bảo tính đa dạng về bối cảnh và góc chụp.

Tổng cộng, bộ dữ liệu bao gồm **300** hình ảnh chỉ chứa **1 đối tượng cần phân loại** , được phân bố như sau:

* Gà: Hình ảnh với các loại gà khác nhau, từ toàn thân đến cận cảnh.
* Vịt: Hình ảnh của nhiều giống vịt trong các điều kiện ánh sáng và bối cảnh khác nhau.
* Khác: Các hình ảnh không thuộc hai nhóm trên, bổ trợ hoặc tạo tính đa dạng cho bộ dữ liệu( Như Ngỗng, Chim cút, vẹt, Quạ …)
  + 1. **Cấu trúc bộ dữ liệu**

Trong nghiên cứu này, bộ dữ liệu gồm tổng cộng 300 ảnh, được phân loại thành ba nhóm đối tượng: gà (chicken), vịt (duck), và các loài gia cầm khác (other), mỗi nhóm có 100 ảnh. Để xây dựng và kiểm tra mô hình, dữ liệu được chia thành hai tập chính: tập huấn luyện và tập kiểm tra theo tỷ lệ 80% - 20%. Cụ thể, 80% của tổng số ảnh được sử dụng cho việc huấn luyện mô hình, trong khi 20% còn lại được dành cho việc kiểm tra.

Bảng dưới đây thể hiện sự phân chia dữ liệu giữa tập huấn luyện và tập kiểm tra:

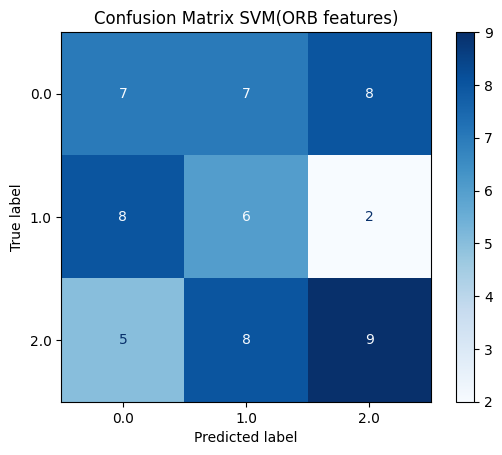
|  |  |  |  |
| --- | --- | --- | --- |
| Loại đối tượng | Tổng số ảnh | Tập huấn luyện (80%) | Tập kiểm tra (20%) |
| Gà (Chicken) | 100 | 84 | 16 |
| Vịt (Duck) | 100 | 78 | 22 |
| Khác (Other) | 100 | 78 | 22 |
| Tổng cộng | 300 | 240 | 60 |

* 1. **Độ đo**
     1. **Accuracy**
* **Định nghĩa**: Accuracy là tỷ lệ các dự đoán đúng trên tổng số mẫu trong bộ dữ liệu. Đây là chỉ số đơn giản nhất dùng để đo lường hiệu quả của mô hình.
* **Công thức** :
  + **TP (True Positive):** Số mẫu thực sự thuộc lớp dương và được mô hình dự đoán đúng là lớp dương.
  + **TN (True Negative):** Số mẫu thực sự thuộc lớp âm và được mô hình dự đoán đúng là lớp âm.
  + **FP (False Positive):** Số mẫu thực sự thuộc lớp âm nhưng được mô hình dự đoán sai là lớp dương.
  + **FN (False Negative):** Số mẫu thực sự thuộc lớp dương nhưng được mô hình dự đoán sai là lớp âm.
    1. **Precision**
* **Định nghĩa:** Precision là tỷ lệ giữa số mẫu thực sự thuộc lớp dương mà mô hình dự đoán đúng là lớp dương so với tổng số mẫu mà mô hình dự đoán là lớp dương.
* **Công thức:**
  + 1. **Recall**
* **Định nghĩa:** Recall là tỷ lệ giữa số mẫu thực sự thuộc lớp dương mà mô hình phân loại đúng là lớp dương so với tổng số mẫu thực sự thuộc lớp dương.
* **Công thức :**
  + 1. **F1-score**
* **Định nghĩa:** F1-score là chỉ số kết hợp giữa Precision và Recall, dùng để đánh giá tổng hợp hiệu suất của mô hình trong việc phân loại lớp dương.
* **Công thức :**
  1. **Kết quả thực nghiệm**
     1. **SVM**
        1. **SVM kết hợp HOG**
* Kết quả dự đoán Mô hình SVM với đặc trưng HOG:
  + Lớp Gà (Chicken): Dự đoán đúng 14 ảnh, 8 ảnh phân loại sai vào lớp Vịt, 0 ảnh phân loại sai vào lớp Khác.
  + Lớp Vịt (Duck): Dự đoán đúng 11 ảnh, 4 ảnh phân loại sai vào lớp Gà, 1 ảnh vào lớp Khác.
  + Lớp Khác (Other): Dự đoán đúng 13 ảnh, 4 ảnh phân loại sai vào lớp Gà, 5 ảnh phân loại sai vào lớp Vịt.

A diagram of a diagram

Description automatically generated with medium confidence

* Nhận xét:
* Mô hình có xu hướng nhầm lẫn nhiều giữa các lớp Gà và Vịt, cho thấy đặc trưng HOG không đủ mạnh để tách biệt hai lớp này trong bộ dữ liệu hiện tại.
* Đối với lớp Khác, mức độ nhầm lẫn sang hai lớp Gà và Vịt cũng tương đối cao, chứng tỏ rằng mô hình chưa thực sự nắm bắt tốt sự khác biệt giữa các nhóm ảnh.
  + - 1. **SVM kết hợp SIFT**
* Kết quả dự đoán Mô hình SVM với đặc trưng SIFT:
  + Lớp Gà (Chicken): Dự đoán đúng 7 ảnh, 7 ảnh phân loại sai vào lớp Vịt, 8 ảnh phân loại sai vào lớp Khác.
  + Lớp Vịt (Duck): Dự đoán đúng 6 ảnh, 8 ảnh phân loại sai vào lớp Gà, 2 ảnh vào lớp Khác.
  + Lớp Khác (Other): Dự đoán đúng 9 ảnh, 8 ảnh phân loại sai vào lớp Gà, 5 ảnh phân loại sai vào lớp Vịt.



* Nhận xét:
* Kết quả với SIFT cho thấy hiệu suất dự đoán thấp hơn so với HOG. Cả ba lớp đều có tỉ lệ nhầm lẫn cao, đặc biệt giữa lớp Gà và lớp Vịt, chứng tỏ đặc trưng SIFT không đủ mạnh để tách biệt rõ ràng các lớp trong bài toán này. Đối với lớp Khác, mức độ nhầm lẫn sang hai lớp Gà và Vịt cũng tương đối cao, chứng tỏ rằng mô hình chưa thực sự nắm bắt tốt sự khác biệt giữa các nhóm ảnh.
* HOG hiện có hiệu suất tốt hơn SIFT, nhưng vẫn gặp nhầm lẫn giữa các lớp có đặc điểm tương đồng. Việc tăng cường dữ liệu, kết hợp thêm đặc trưng, tinh chỉnh tham số HOG, và thử nghiệm mô hình phức tạp hơn như CNN có thể giúp cải thiện hiệu quả phân loại đáng kể. Trong khi đó, SIFT có thể được giữ lại như một đặc trưng bổ sung để tăng cường hiệu suất thông qua mô hình ensemble.
  + 1. **Softmax regression**
       1. **Softmax regression kết hợp HOG**
* Kết quả dự đoán Mô hình Softmax với đặc trưng HOG:
  + Lớp Gà (Chicken): Dự đoán đúng 13 ảnh, 7 ảnh phân loại sai vào lớp Vịt, 2 ảnh phân loại sai vào lớp Khác.
  + Lớp Vịt (Duck): Dự đoán đúng 10 ảnh, 2 ảnh phân loại sai vào lớp Gà, 4 ảnh vào lớp Khác.
  + Lớp Khác (Other): Dự đoán đúng 16 ảnh, 2 ảnh phân loại sai vào lớp Gà, 4 ảnh phân loại sai vào lớp Vịt.

A blue squares with white text

Description automatically generated

* Nhận xét:
* So với các mô hình trước đó, Softmax với đặc trưng HOG có hiệu suất tốt hơn, đặc biệt là trong việc giảm nhầm lẫn giữa các lớp, nhất là giữa Gà và Vịt. Tuy nhiên, vẫn còn một số hạn chế:
* Sự nhầm lẫn giữa lớp Gà và lớp Vịt vẫn là vấn đề chính, mặc dù đã được cải thiện so với SVM.
* Nhầm lẫn giữa lớp Khác và hai lớp Gà, Vịt vẫn tồn tại, nhưng ở mức thấp hơn.
  + - 1. **Softmax regression kết hợp SIFT**
* Kết quả dự đoán Mô hình Softmax với đặc trưng SIFT:
  + Lớp Gà (Chicken): Dự đoán đúng 8 ảnh, 4 ảnh phân loại sai vào lớp Vịt, 10 ảnh phân loại sai vào lớp Khác.
  + Lớp Vịt (Duck): Dự đoán đúng 6 ảnh, 9 ảnh phân loại sai vào lớp Gà, 1 ảnh vào lớp Khác.
  + Lớp Khác (Other): Dự đoán đúng 8 ảnh, 8 ảnh phân loại sai vào lớp Gà, 8 ảnh phân loại sai vào lớp Vịt.

A blue squares with white text

Description automatically generated

* Nhận xét:
* Hiệu suất dự đoán của mô hình Softmax với đặc trưng SIFT là khá thấp ở cả ba lớp, với tỉ lệ nhầm lẫn cao.
* Mức độ nhầm lẫn giữa lớp Gà và lớp Vịt, cũng như giữa lớp Khác và hai lớp còn lại, cho thấy đặc trưng SIFT không phù hợp để phân loại trong bài toán này.
* So với đặc trưng HOG, SIFT có hiệu suất kém hơn do chỉ dựa vào các điểm đặc trưng cục bộ (keypoints), không tận dụng tốt các thông tin tổng thể về hình dạng và cấu trúc.

# **Tài liệu tham khảo**

<https://github.com/danielgatis/rembg>

<https://machinelearningcoban.com/2017/04/09/smv/>

https://vn.got-it.ai/blog/softmax-function-la-gi-tong-quan-ve-softmax-function