VIỆN DỮ LIỆU LỚN VINGROUP

HỌC MÁY CƠ BẢN

ĐỒ ÁN CUỐI KHÓA

PHÂN LOẠI VIDEO

ÁP DỤNG MỘT SỐ KỸ THUẬT HỌC MÁY

CHƯƠNG TRÌNH ĐÀO TẠO KỸ SƯ AI

HÀ NỘI, 12/2020

VIỆN DỮ LIỆU LỚN VINGROUP

VÕ MINH TÂM – 3658402

TRẦN BẢO SAM – 3658429

ĐỖ QUỐC CƯỜNG – 3658411

KHỔNG HỮU CƯỜNG – 3658423

HỌC MÁY CƠ BẢN

ĐỒ ÁN CUỐI KHÓA

PHÂN LOẠI VIDEO

ÁP DỤNG MỘT SỐ KỸ THUẬT HỌC MÁY

CHƯƠNG TRÌNH ĐÀO TẠO KỸ SƯ AI

GIẢNG VIÊN HƯỚNG DẪN

TS. PHAN HẢI HỒNG

HỌC VIỆN AI VIỆT NAM

HÀ NỘI, 12/2020

MỤC LỤC

Chương 1: TỔNG QUAN ĐỀ TÀI

Chương này gồm các nội dung giới thiệu, mô tả bài toán; đặt ra mục tiêu cho đồ án; chỉ rõ phạm vi, đối tượng nghiên cứu và khảo sát một số nghiên cứu liên quan.

1.1. Giới thiệu bài toán

Phát hiện và nhận dạng hành động bạo lực trong video là bài toán thú vị và đầy tính thách thức của lĩnh vực thị giác máy tính và nhận dạng mẫu. Bài toán này dần được xem như một đề tài nghiên cứu chuyên sâu vì có tính ứng dụng cao trong thực tế [1]. Điển hình là phát hiện hành vi đánh nhau từ các camera an ninh ở những nơi công cộng, nhà tù, trường học hay các nội dung bạo lực trên các phương tiện truyền thông, youtube. Phát hiện sớm các hành vi bạo lực giúp lực lượng an ninh đề ra phương án giải quyết, giảm thiểu thiệt hại về người và của, cảnh báo và ngăn chặn trẻ em truy cập vào những nội dung không phù hợp.

Qua tìm hiểu và trao đổi, nhóm chúng tôi nhận thấy bài toán phát hiện bất thường trong video rất thú vị và đáng thực hiện. Tuy nhiên, do giới hạn về thời gian cũng như kiến thức khi cả bốn thành viên đều chưa có kinh nghiệm làm việc với video, nhóm xin phép giới hạn lại bài toán cho đồ án cuối khóa sau khi nhận được ý kiến và sự hướng dẫn của giảng viên. Cụ thể, đồ án này sẽ thực hiện việc phân loại một video ngắn xem đây có phải là video có hành vi đánh nhau hay không. Bài toán mà nhóm thực hiện lấy cảm hứng từ bài báo *Vision-based Fight Detection from Surveillance Cameras* của Akti và các cộng sự [1].

1.2. Mô tả bài toán:

Mô tả cụ thể cho bài toán như sau,:

- Đầu vào: một video ngắn (từ 2 đến 4 giây).
- Đầu ra: nhãn là “1” nếu video có hành động đánh nhau, nhãn là “0” nếu video không có hành động đánh nhau.

1.3. Mục tiêu đồ án:

- Tìm hiểu, khảo sát và phân tích bài toán cùng bộ dữ liệu liên quan.

- Khảo sát một số phương pháp hiện đại đang được áp dụng cho bài toán này.
- Áp dụng một số phương pháp học máy giải quyết bài toán.
- Khảo sát vấn đề mất cân bằng dữ liệu, tìm hiểu phương pháp giải quyết.
- Xây dựng chương trình demo.

1.4. Phạm vi nghiên cứu:

Đồ án này tập trung vào các video có hành động đánh nhau hoặc không đánh nhau được cung cấp bởi tác giả [1].

1.5. Đối tượng nghiên cứu:

- Các hành động đánh nhau hoặc không đánh nhau.
- Các phương pháp xử lý dữ liệu.
- Mô hình trích xuất đặc trưng từ video.
- Một số mô hình phân loại cơ bản như KNN, Naïve Bayes, SVM…

1.6. Nghiên cứu liên quan

Aktı và các cộng sự [1] sử dụng VGG16 và Xception để trích xuất các đặc trưng từ video, sau đó sử dụng Bi-LSTM và cơ chế attention để phân loại. Kết quả đạt được trên bộ dữ liệu Surveillance Camera Fight Dataset là 72.

Theo Simonyan và cộng sự [2], phương pháp học sâu thường dùng trong nhận dạng hành động là two-stream convolutional networks. Trong đó, hai CNNs được sử dụng: một để trích xuất đặc trưng không gian, học các hành động và một để trích xuất đặc trưng thời gian, học từ các vectơ luồng quang học (optical flow) của nhiều khung hình. Sau đó, đầu ra của hai mạng được kết hợp ở cuối.

Swathikiran [3] đề xuất mô hình bao gồm một mạng CNN để trích xuất các đặc trưng theo từng frame, sau đó áp dụng convLSTM để lấy các thông tin trong miền thời gian và cho kết quả rất tốt. Ngoài ra, các tác giả còn cho biết mạng convLSTM cho kết quả tốt hơn LSTM thông thường.

Chương 2: BỘ DỮ LIỆU

Bộ dữ liệu **Surveillance Camera Fight Dataset** được thu thập cho mục đích nghiên cứu của nhóm tác giả [1]. Mặc dù đã có một số bộ dữ liệu cụ thể cho bài toán phân loại video có chứa nội dung đánh nhau hay không, các mẫu dữ liệu này chủ yếu được lấy từ phim hoặc trò chơi khúc côn cầu, tương ứng với các loại cảnh khác nhau. Các tập dữ liệu này có thể giúp tự học các hành động, nhưng chúng không hoàn toàn phù hợp với nhiệm vụ giám sát an ninh.

Trong các ứng dụng giám sát, con người trong các cảnh quay luôn khác nhau và nền của cảnh quay cũng khác đối với mỗi camera. Trong phim và trò chơi khúc côn cầu, nền chuyển động do các kỹ thuật quay phim như phóng to, thu nhỏ. Mặt khác, camera giám sát chủ yếu là tĩnh và nền trong các bản ghi ổn định hơn.

Bộ dữ liệu **Surveillance Camera Fight Dataset** có tổng cộng 300 video gồm 150 cảnh đánh nhau và 150 cảnh không đánh nhau. Phần lớn các đoạn video giám sát được thu thập từ YouTube và một số bộ dữ liệu camera giám sát như CamNet [4] và bộ dữ liệu Synopsis [5], [6] được sử dụng để trích xuất các đoạn video không đánh nhau.

Các video có kích thước và số lượng khung hình khác nhau. Do đó, các khung hình sẽ được thay đổi về cùng kích thước trước khi vào CNN. Sau đó, lấy mẫu đồng nhất (uniform sampling) được áp dụng bằng cách tính đến tổng số khung hình của các video.

Có nhiều loại "đánh nhau" như đá, đấm, đánh với vật thể và đấu vật. Vì các camera an ninh có các điều kiện ánh sáng, màu sắc, địa điểm khác nhau, các biến thể này cũng được xem xét để tăng tính đa dạng trong tập dữ liệu hơn nữa. Bộ dữ liệu này được sử dụng trong bài báo [1] và có thể được truy cập thông qua đường link https://github.com/sayibet/fight-detection-survdataset.

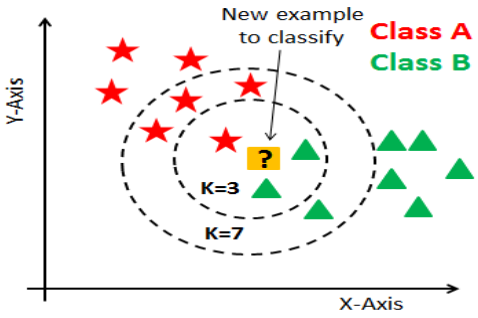
Chương 3: CƠ SỞ LÝ THUYẾT

Chương trình trình bày một số thuật toán phân loại và mô hình trích xuất đặc trưng được sử dụng trong đồ án.

3.1. Thuật toán KNN

K-nearest neighbor hay KNN là một trong những thuật toán supervised-learning đơn giản nhất. Thuật toán này không học một điều gì từ dữ liệu huấn luyện. Chỉ khi nào cần dự đoán kết quả của dữ liệu mới thì các tính toán mới được thực thi.

Trong bài toán phân loại, nhãn của một điểm dữ liệu mới được suy ra trực tiếp từ K điểm dữ liệu gần nhất trong tập dữ liệu huấn luyện. Nhãn của một mẫu kiểm thử có thể được quyết định bằng major voting (bầu chọn theo số phiếu) giữa các điểm gần nhất.



Hình 1: Minh họa thuật toán KNN
(link ảnh: https://www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html).

KNN có các ưu điểm:

- Độ phức tạp tính toán của quá trình huấn luyện bằng không.
- Việc dự đoán kết quả của dữ liệu mới rất đơn giản.
- Không cần giả sử gì về phân phối của các lớp.

Bên cạnh đó, KNN cũng có một số nhược điểm:

- KNN rất nhạy cảm với nhiễu khi K nhỏ.

- Việc tính khoảng cách tới từng điểm dữ liệu trong tập dữ liệu huấn luyện sẽ tốn rất nhiều thời gian, đặc biệt với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu.
- Giá trị K càng lớn thì độ phức tạp càng tăng.
- Tốn bộ nhớ lưu trữ cả bộ dữ liệu. [7]

3.2. Thuật toán Naïve Bayes

Naive Bayes là một thuật toán sử dụng cho phân loại nhị phân (hai lớp) và đa lớp, lấy cảm hứng từ định lý Bayes trong xác suất thống kê.

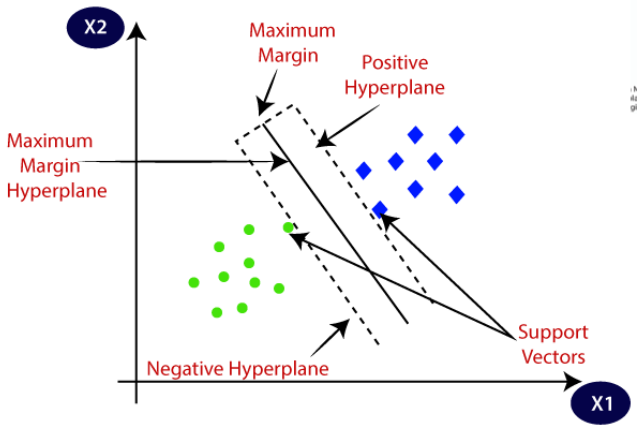
$$P(y|X) = \frac{P(X|y).P(y)}{P(X)} \quad (1)$$

Thuật toán này tính xác suất cho các nhãn, sau đó chọn kết quả với xác suất cao nhất. Tuy nhiên, điều cần lưu ý là thuật toán Naive Bayes giả định các đặc trưng đầu vào là độc lập với nhau. Naïve Bayes là một thuật toán mạnh mẽ cho phân loại văn bản, lọc thư rác, hệ thống khuyến nghị…

Một số kiểu mô hình Naïve Bayes: Bernoulli, Multinomial và Gaussian.

3.3. Support Vector Machine

Support Vector Machine hay SVM là một thuật toán học máy có giám sát mạnh mẽ nên được sử dụng rất phổ biến trong các bài toán phân lớp hay hồi qui.

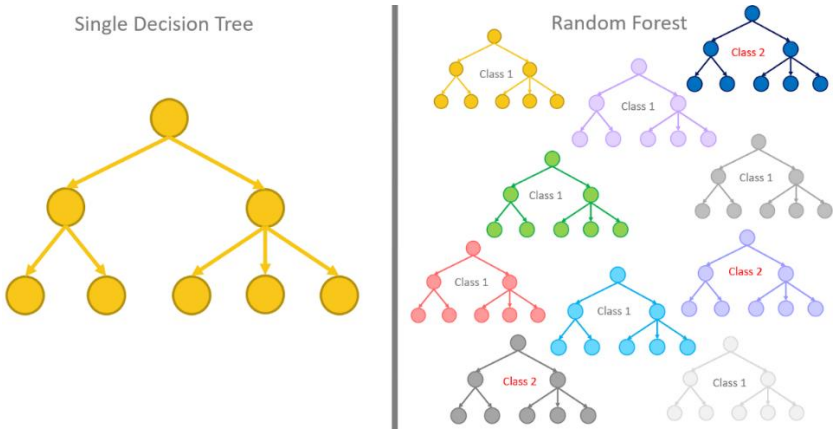


Hình 2: Minh họa thuật toán SVM
(link ảnh: https://in.pinterest.com/pin/735564551631159408/).

Ý tưởng của SVM là tìm một siêu phẳng (hyperlane) để phân tách các điểm dữ liệu, hình 2. Siêu phẳng này sẽ chia không gian thành các miền khác nhau, mỗi miền sẽ chứa một loại dữ liệu. Với loại dữ liệu phi tuyến thì không thể tìm được một siêu phẳng nào thỏa mãn các yêu cầu của thuật toán. Giải pháp là sử dụng "kernel trick" ánh xạ dữ liệu vào miền không gian khác có khả năng phân tách tốt hơn.

3.4. Cây quyết định

Cây quyết định (Decision Tree) là một cây phân cấp có cấu trúc được dùng để phân lớp các đối tượng dựa vào dãy các luật (Hình 3 bên trái). Các thuộc tính của đối tượng có thể thuộc các kiểu dữ liệu khác nhau như nhị phân, định danh, thứ tự, số lượng trong khi thuộc tính phân lớp phải có kiểu dữ liệu là nhị phân hoặc thứ tự.



Hình 3: Minh họa thuật toán cây quyết định và rừng ngẫu nhiên

(link ảnh: https://www.mygreatlearning.com/blog/decision-tree-algorithm/).

Thuật toán cây quyết định khá hiệu quả cho các bài toán ra quyết định, nhất là trong lĩnh vực y tế, các hệ thống gợi ý...

3.5. Rừng ngẫu nhiên

Rừng ngẫu nhiên (Random Forest) là một thuật toán học có giám sát thuộc loại "ensemble learning", sử dụng các cây quyết định làm nền tảng. Thuật toán này cũng tỏ ra hiệu quả với các dạng bài toán ra quyết định với các ưu điểm: có thể sử dụng

cho cả bài toán phân loại và hồi quy, làm việc được với dữ liệu thiếu giá trị, có thể tránh overfitting khi số lượng cây lớn, có thể tạo mô hình cho các giá trị phân loại, Hình 3 bên phải.

3.6. Stochastic Gradient Decent

Gradient Decent (GD) là thuật toán quan trọng và phổ biến trong lĩnh vực học sâu. Một biến thể của GD là Stochastic Gradient Decent (SGD).

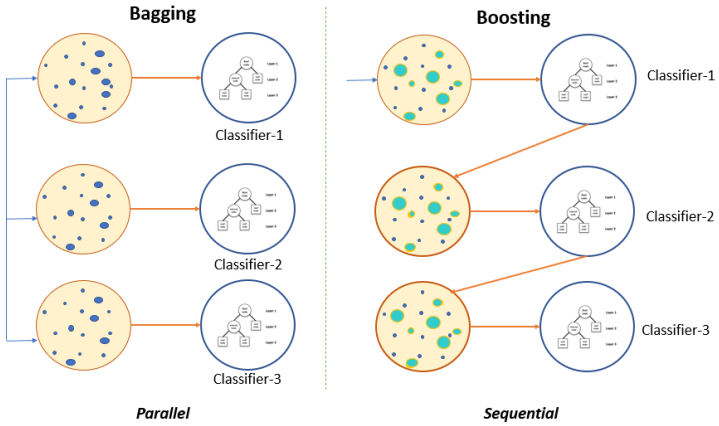
$$\theta = \theta - \eta \nabla_\theta J(\theta; x_i; y_i) \text{ (CT 2)}$$

Tại một thời điểm, SGD chỉ tính đạo hàm của hàm mất mát dựa trên chỉ một điểm dữ liệu x_i rồi cập nhật θ dựa trên đạo hàm này. Quá trình này được lặp lại cho đến khi hết các điểm dữ liệu huấn luyện (CT 2). Thuật toán dù rất đơn giản nhưng trên thực tế lại làm việc khá hiệu quả.

3.7. Một số phương pháp ensemble khác

3.7.1. Bagging

Phương pháp này xây dựng một lượng lớn các mô hình (thường là cùng loại) trên những tập con khác nhau lấy từ tập huấn luyện bằng phương pháp bootstrap. Những mô hình này sẽ được huấn luyện độc lập và song song với nhau nhưng đầu ra của chúng sẽ được trung bình cộng để cho ra kết quả cuối cùng, Hình 4 bên trái.



Hình 4: Minh họa thuật toán Bagging và Boosting

(link ảnh https://www.pluralsight.com/guides/ensemble-methods:-bagging-versus-boosting).

3.7.2. Adaboost

AdaBoost (Adaptive Boost) là một thuật toán học mạnh, được cải tiến dựa vào thuật toán Boosting (hình 4 bên phải) giúp đẩy nhan việc tạo ra một bộ phân loại mạnh bằng cách chọn các đặc trưng tốt trong một họ các bộ phân loại yếu (weak classifer - bộ phân loại yếu) và kết hợp chúng lại tuyến tính bằng cách sử dụng các trọng số. Điều này thật sự cải thiện dần độ chính xác nhờ áp dụng hiệu quả một chuỗi các bộ phân loại yếu.

3.7.3. XGBoost

XGBoost hay Extreme Gradient Boosting là thuật toán state-of-the-art nhằm giải quyết bài toán supervised learning cho độ chính xác khá cao được mở rộng từ Boosting (hình 4 bên phải). XGBoost có khả năng tránh overfitting khá hiệu quả, tính toán song song trên CPU/GPU, tính toán phân tán trên nhiều server, tính toán khi tài nguyên bị giới hạn và tối ưu bộ nhớ để tăng tốc huấn luyện. Bên cạnh đó, XGBoost có thể xử lý dữ liệu thiếu, tiếp tục huấn luyện bằng mô hình đã được xây dựng trước đó để tiết kiệm thời gian.

3.8. Mô hình trích xuất đặc trưng

Chúng tôi sử dụng mô hình trích xuất đặc trưng 3D ResNets [12]. Mô hình này được huấn luyện trên bộ dữ liệu HowTo100M (https://www.rocq.inria.fr/cluster-willow/amiech/howto100m/paper.pdf).

Mô hình pretrained: https://www.rocq.inria.fr/cluster-willow/amiech/howto100m/models/resnext101.pth

Source code: https://github.com/kenshohara/3D-ResNets-PyTorch

Chương 4: THÍ NGHIỆM VÀ KẾT QUẢ

Chương này trình kết các kết quả thí nghiệm và rút ra nhận xét về hiệu suất của các mô hình đã sử dụng.

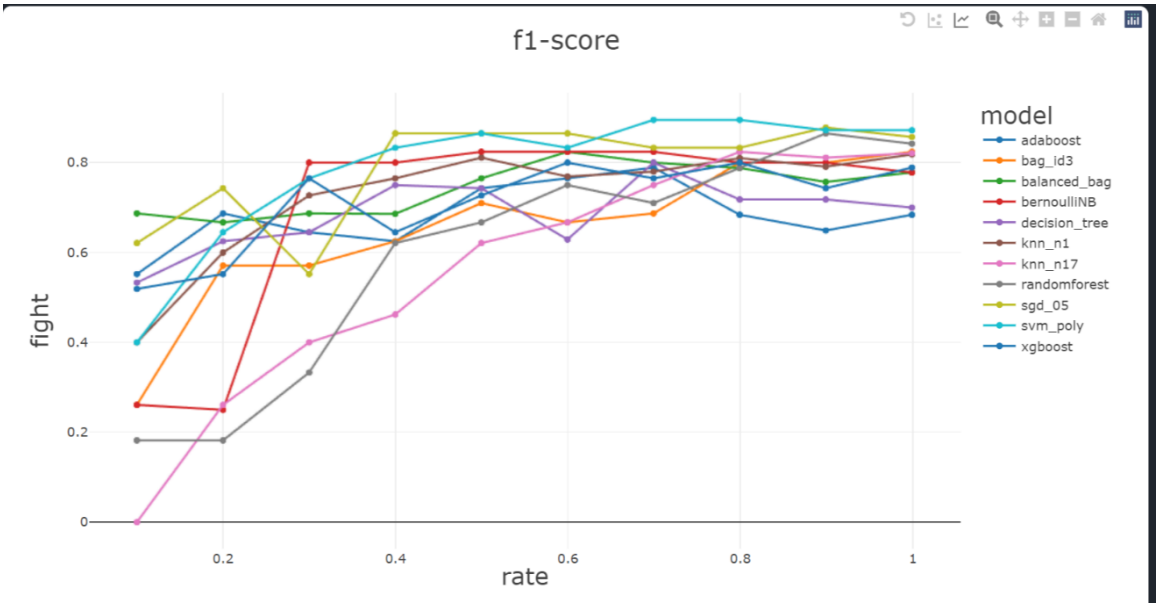
4.1. Độ đo đánh giá mô hình

Đồ án này sử dụng các độ đo **precision**, **recall**, **f1-score** để đánh giá hiệu suất mô hình.

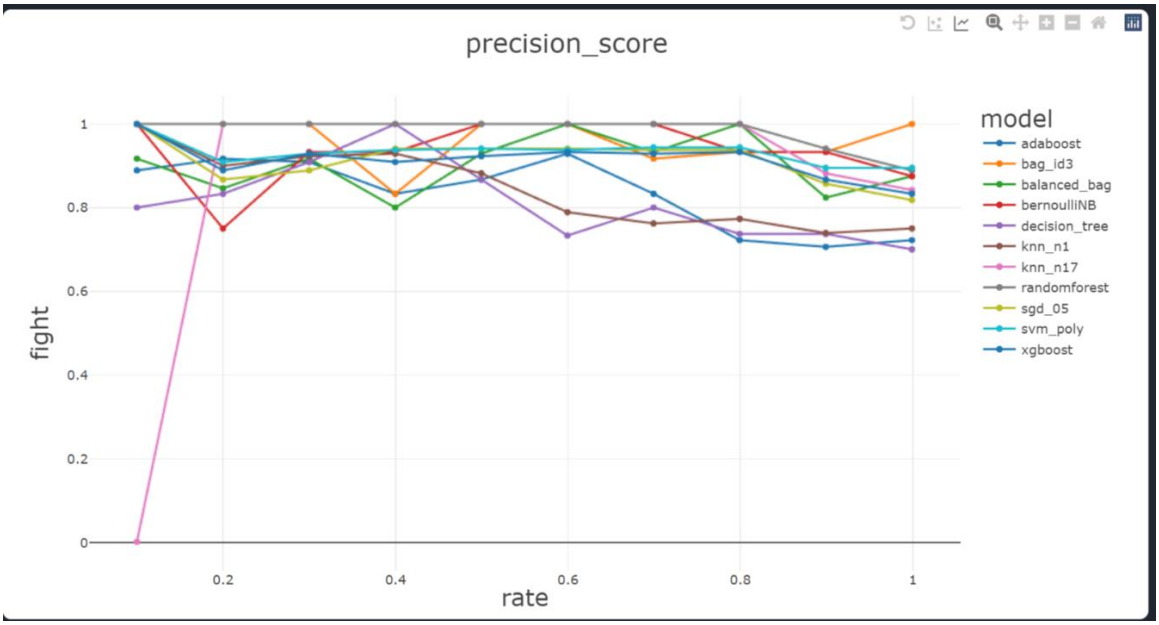
4.2. Kết quả thí nghiệm

Chúng tôi thực hiện lấy 20 videos mỗi loại để làm tập test cho tất cả các thí nghiệm. Số lượng 260 videos còn lại sẽ được sử dụng để huấn luyện mô hình với tỷ lệ từ 1:10 đến 10:10 tương ứng với lớp video có hành vi đánh nhau và không đánh nhau.

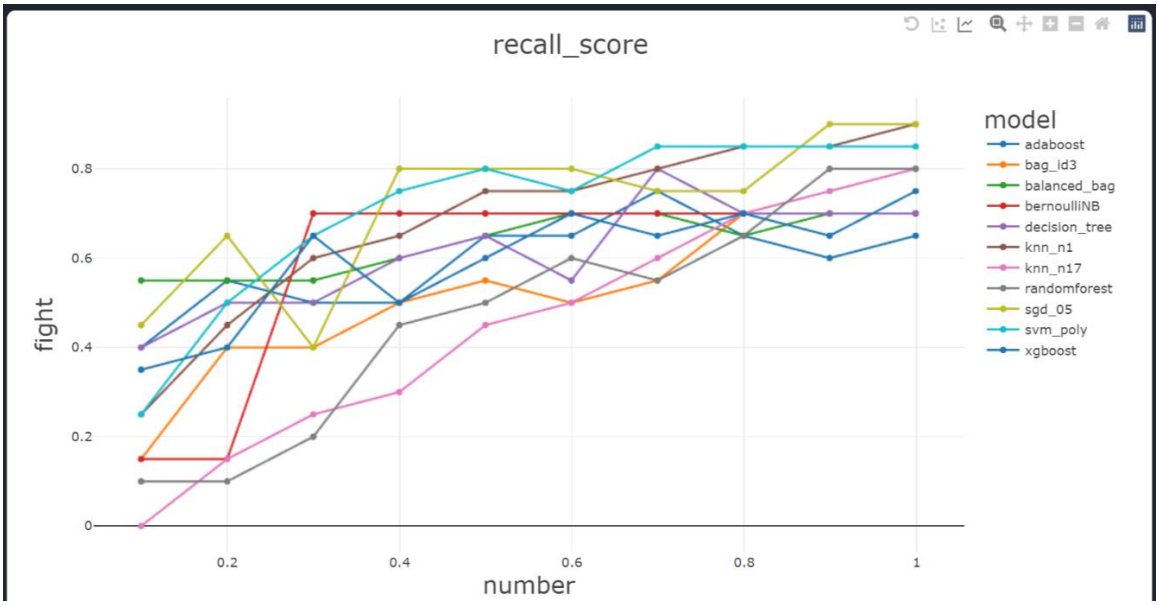
Đầu tiên, chúng tôi thực hiện huấn luyện và đánh giá mô hình khi chưa áp dụng thuật toán xử lý mất cân bằng dữ liệu.



Hình 5: Kết quả f1-score khi chưa xử lý mất cân bằng.



Hình 6: Kết quả precision khi chưa xử lý mất cân bằng.



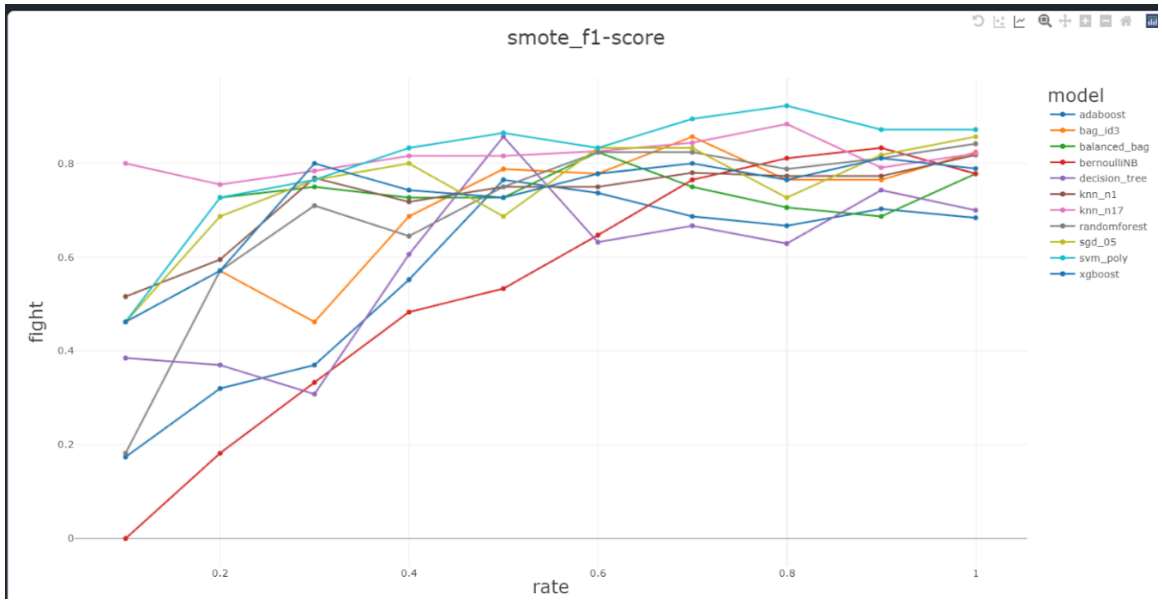
Hình 7: Kết quả recall khi chưa xử lý mất cân bằng.

Mô hình cho hiệu quả cao nhất SVM_Poly F1 score 0.88, recall score 0.85, precision score 0.90 khi xác định video là có đánh nhau. Kết quả này có thể chấp nhận được trong thực tiễn và có tính ứng dụng cao khi có thể xác định được trên 85% những trường hợp có đánh nhau trong video. Hơn nữa, với những video không đánh nhau, recall score là 0.9, f1 score là 0.88, và precision là 0.86. Thông qua

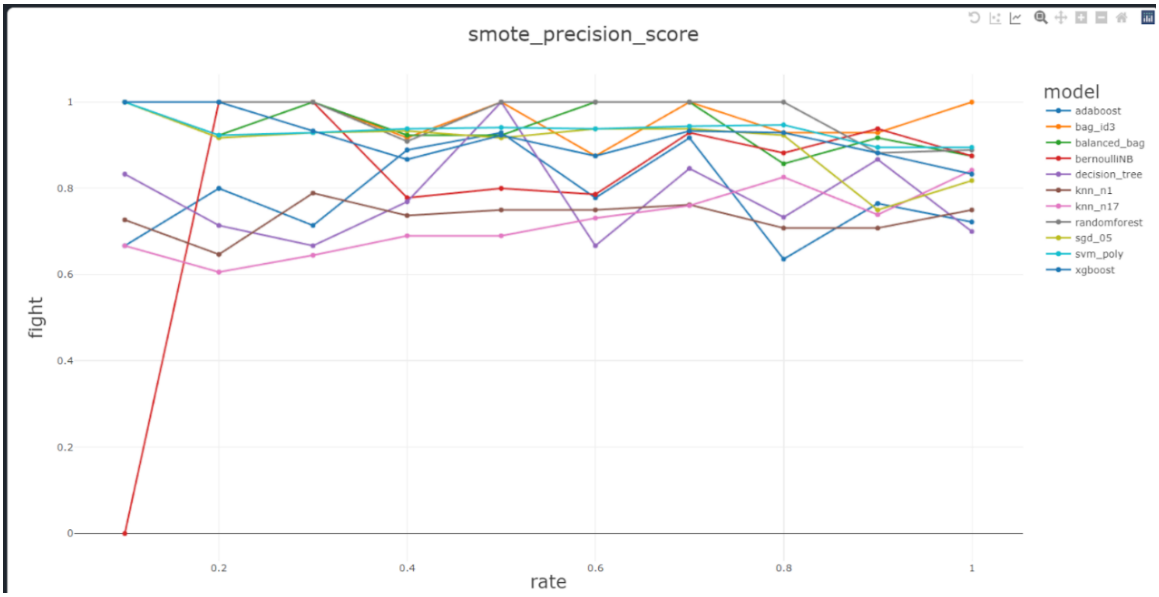
những dữ liệu này, ta có thể thấy SVM model mang tính ứng dụng cao nhất, vì nó không chỉ phán đoán đúng các video chứa đánh nhau, mà còn xác định video không đánh nhau một cách hiệu quả. Do đó giảm thiểu tối đa Type I error va Type II error trong bài toán.

Dựa trên khảo sát mất cân bằng dữ liệu giữa các model, ta có thể nhận thấy các model này không mang lại hiệu quả. Model cho chất lượng tốt nhất là Balanced Bag. Nhưng những số liệu đã chỉ ra rằng, hiệu quả của model này vẫn chỉ ở mức rất khiêm tốn. Ta có f1 score, precision score, recall score lần lượt là 0.7, 0.9, 0.55. Điều này có nghĩa là, ta chỉ có thể phát hiện được phân nửa số video là đánh nhau, phân nửa còn lại model không thể nhận ra. Model cho hiệu quả thấp nhất là KNN với f1 score, precision score và recall score đều là 0%. Model không thể dự đoán được bất cứ video nào.

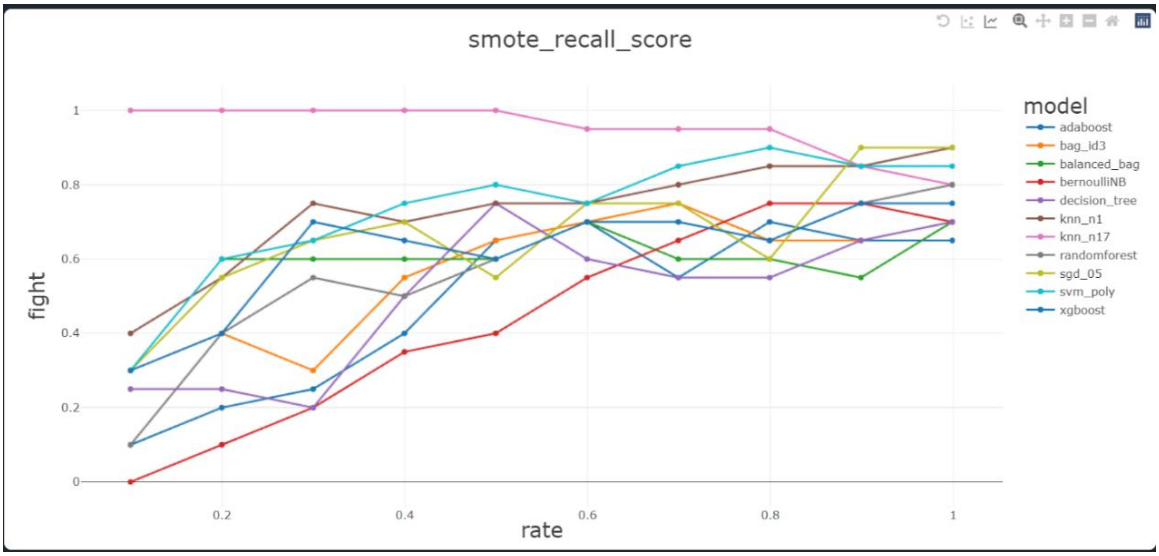
Thực tế, với loại dữ liệu này, số lượng các hành vi đánh nhau rất ít so với các hành vi không đánh nhau. Do đó, chúng tôi cố gắng mô phỏng lại trường hợp dữ liệu mất cân bằng và khảo sát một số cách giải quyết. Để xử lý mất cân bằng dữ liệu, chúng tôi sửa dụng thuật toán SMOTE. Kết quả thí nghiệm như sau:



Hình 8: Kết quả f1-score khi đã áp dụng SMOTE.



Hình 9: Kết quả precision khi đã áp dụng SMOTE.



Hình 10: Kết quả recall khi đã áp dụng SMOTE.

Chúng tôi lấy dữ liệu có tỷ lệ giữa video đánh nhau và video không đánh nhau lần lượt từ 1/10 tới 10/10. Sau đó, dùng SMOTE để nhận số lượng video đánh nhau bằng với số lượng video không đánh nhau. Chúng tôi tập trung vào sự mất cân bằng 1/10 để khảo sát độ hiệu quả của các model. Kết quả nhận thấy KNN model mang lại hiệu quả cao nhất với f1 score, precision score, và recall score lần lượt là 0.8, 0.65, 1. Ta có thể nhận thấy rằng KNN có hiệu suất được nâng lên đáng kinh

ngạc so với trước khi sử dụng SMOTE. Model có thể phát hiện 100% video chứa cảnh đánh nhau, so với việc nó không thể phát hiện ra video nào khi chưa sử dụng SMOTE.

4.3. Nhận xét

Sau khi áp dụng thuật toán SMOTE để xử lý mất cân bằng dữ liệu thì kết quả thí nghiệm tăng lên. Ngoài ra, chúng tôi còn áp dụng kỹ thuật bootstrap nhưng phương pháp này không hiệu quả bằng SMOTE. Vì thời gian có hạn nên phương pháp VAE chưa được áp dụng ở đây.

1. Model và dữ liệu được chúng tôi sử dụng trong DEMO là model SVM với dữ liệu cân bằng. Vì Type I error và Type II error thấp nhất trong các model, do đó, model giảm thiểu tối đa khả năng phán đoán sai (no_fight video thành fight video), và không phán đoán được video đánh nhau (fight video thành no_fight video).

2. Cách dung model và cách xử lý số liệu có thể ảnh hưởng rất lớn đến hiệu quả của model. Thí dụ, khi chúng tôi không dùng SMOTE hoặc Bootstraps để cho dữ liệu về dạng cân bằng, KNN model là model tệ nhất trong tất cả vì nó không thể phát hiện ra bất cứ video nào có chứa hình ảnh đánh nhau. Tuy nhiên, sau khi dùng SMOTE hoặc Bootstraps, hiệu quả của model này được nâng lên đáng kinh ngạc.

3. Recall score cao chưa chắc đã là một model tốt. Chúng tôi đã dùng thử KNN với tiền xử lý dữ liệu SMOTE để chạy DEMO vì recall score là 1, nó có nghĩa rằng model có thể phát hiện ra 100% video có chứa cảnh đánh nhau. Tuy nhiên, precisions score thấp dẫn đến việc wrong predictions nhiều. Mặc dù video không hề có chứa cảnh bảo lực nhưng model vẫn cho ra kết quả có chứa cảnh bạo lực. Trung bình 4s sẽ alert một lần dẫn tới việc nhiễu thông tin.

Chương 5. ỨNG DỤNG DEMO

Ứng dụng được xây dựng với 3 lõi:

- Services trích xuất đặc trưng từ video.
- Services phân lớp.
- Chọn lọc mô hình phân lớp dựa trên quá trình kiểm tra.

Ứng dụng tái sử dụng code theo nguồn (https://github.com/antoine77340/video_feature_extractor) và tự chỉnh sửa để phù hợp với đồ án.

Code của nguồn trên được cập nhật theo bài báo sớm nhất 10/04/2020 (Hirokatsu Kataoka, Tenga Wakamiya, Kensho Hara, and Yutaka Satoh, "Would Mega-scale Datasets Further Enhance Spatiotemporal 3D CNNs" (link https://arxiv.org/abs/2004.04968)

Ứng dụng demo:

- Ngôn ngữ lập trình: Python 3.
- Môi trường chạy: anaconda-jupyter.
- Hỗ trợ real time với độ trễ 2s – 2.1s.
- Yêu cầu GPU Nvidia

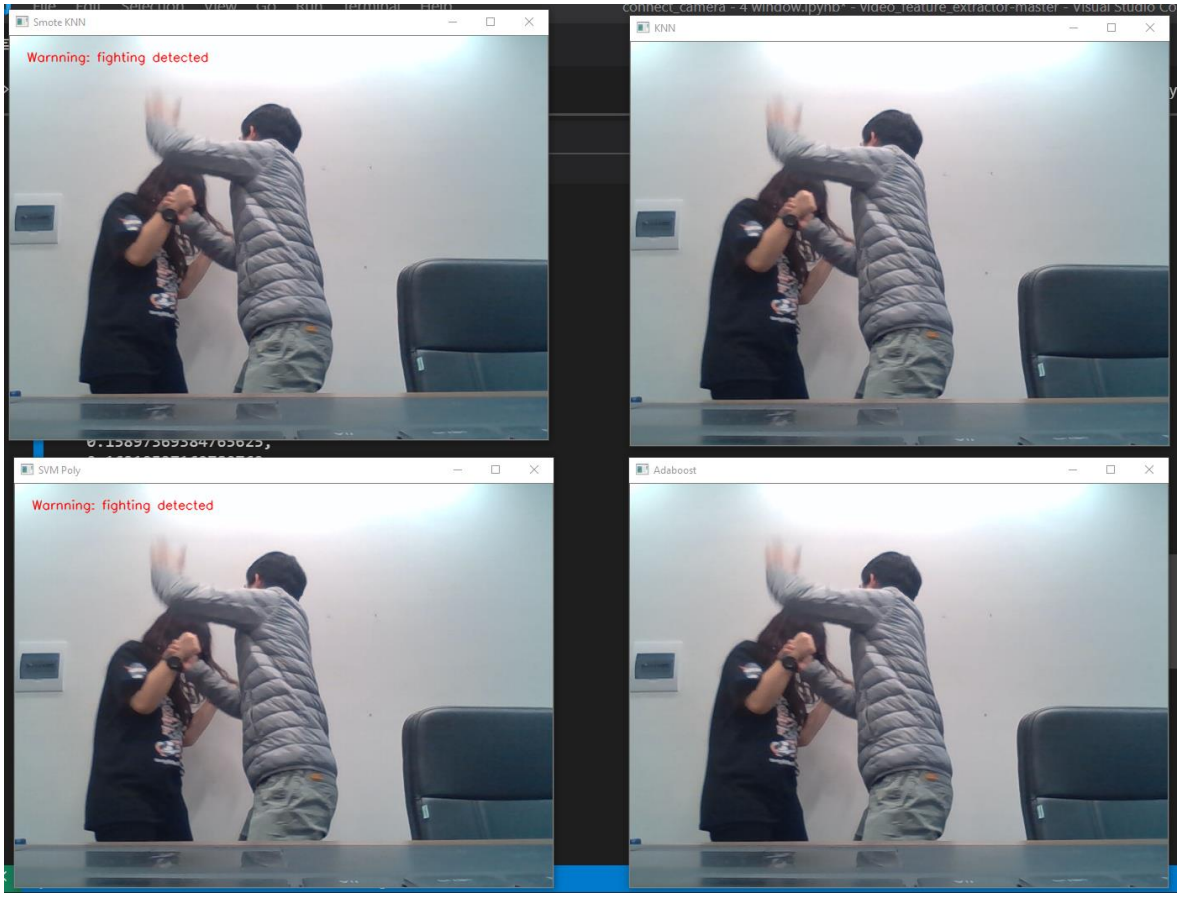
Hướng dẫn sử dụng: thư mục chưa project ứng dụng có 5 file gồm:

- connect_camera.ipynb chạy ứng dụng với mô hình phân lớp SVM cân bằng
- connect_camera 0.ipynb chạy ứng dụng với mô hình phân lớp KNN (k=17) với dữ liệu mất cân bằng qua sử lý Smote
- connect_camera 1.ipynb chạy ứng dụng với mô hình phân lớp KNN (k=17) với dữ liệu mất cân bằng chưa qua sử lý
- connect_camera 3.ipynb chạy ứng dụng với mô hình phân lớp SVM với dữ liệu mất cân bằng chưa qua xử lý.
- connect_camera – 4 window.ipynb chạy ứng dụng với mô hình phân lớp Adaboost với dữ liệu mất cân bằng chưa qua xử lý.

Cách chạy file như chạy file jupyter bình thường. Khi cửa sổ hiện lên, bấm "q" để thoát.

Trong thời gian cửa sổ hiện nên. Nếu thực hiện hành động trong camera, chương trình sẽ hiện lên cửa sổ cảnh báo nếu đó là hành vi bất thường.

Giao diện chương trình demo như sau:



Hình 11: Giao diện chương trình demo.

Chương 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. Kết luận

SMOTE rất phù hợp để xử lý các trường hợp mất cân bằng dữ liệu khi hiệu suất các mô hình đều tăng với tỷ lệ mất cân bằng từ 1:10 đến 10:10. Trong đó, với trường hợp dữ liệu cân bằng, SVM cho kết quả tốt nhất. Với trường hợp mất cân bằng 1:10 chưa áp dụng SMOTE, Balanced Bag cho kết quả tốt nhất. Với trường hợp mất cân bằng 1:10 qua áp dụng SMOTE, KNN với k = 17 cho kết quả tốt nhất.

6.2. Hướng phát triển

Bài toán có thể mở rộng ứng dụng trong giám sát an ninh tòa nhà, nơi công cộng. Cải thiện độ chính xác bằng các thuật toán hiện đại hơn như DL và thực hiện thời gian thực.

THAM KHẢO

[1] S. Aktı, G. A. Tataro¨glu, H. K. Ekenel, “Vision-based Fight Detection from Surveillance Cameras”, International Conference on Image Processing Theory, Tools and Applications, IPTA 2019.

[2] K. Simonyan, A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” Advances in Neural Information Processing Systems, 2014, pp. 568–576.

[3] S. Sudhakaran, O. Lanz, “Learning to Detect Violent Videos using Convolutional Long Short-Term Memory”, International Conference on Advanced Video and Signal based Surveillance (AVSS), 2017.

[4] S. Zhang, E. Staudt, T. Faltemier, and A. K. Roy-Chowdhury, “A camera network tracking (camnet) dataset and performance baseline,” in 2015 IEEE Winter Conference on Applications of Computer Vision. IEEE, 2015, pp. 365–372.

[5] W.-C. Wang, P.-C. Chung, C.-R. Huang, and W.-Y. Huang, “Event based surveillance video synopsis using trajectory kinematics descriptors,” in 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA). IEEE, 2017, pp. 250–253.

[6] C.-R. Huang, P.-C. J. Chung, D.-K. Yang, H.-C. Chen, and G.-J. Huang, “Maximuma posteriori probability estimation for online surveillance video synopsis,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 24, no. 8, pp. 1417–1429, 2014.

[7] H. Kataoka, T. Wakamiya, K. Hara, Y. Satoh, “Would Mega-scale Datasets Further Enhance Spatiotemporal 3D CNNs?”, 2020.