A Machine Learning Approach to Detect Violent Behaviour from Video

David Nova[1], André Ferreira[2], and Paulo Cortez[1][0000−0002−7991−2090]*

[1] ALGORITMI Centre, Department of Information Systems, University of Minho, 4804-533 Guimarães, Portugal
dcvn99@gmail.com, pcortez@dsi.uminho.pt
[2] Department of Informatics, University of Minho, 4710-057 Braga, Portugal
alferreira@di.uminho.pt

Abstract. The automatic classification of violent actions performed by two or more persons is an important task for both societal and scientific purposes. In this paper, we propose a machine learning approach, based a Support Vector Machine (SVM), to detect if a human action, captured on a video, is or not violent. Using a pose estimation algorithm, we focus mostly on feature engineering, to generate the SVM inputs. In particular, we hand-engineered a set of input features based on keypoints (angles, velocity and contact detection) and used them, under distinct combinations, to study their effect on violent behavior recognition from video. Overall, an excellent classification was achieved by the best performing SVM model, which used keypoints, angles and contact features computed over a 60 frame image input range.

Keywords: Machine Learning · Support Vector Machine · Action Recognition · Pose Estimation · Video Analysis

1 Introduction

Nowadays, human behaviour is increasingly being recorded using digital cameras [2, 25, 14]. Following this increase of video data, there is a growing need for the development of intelligent video analysis systems, capable of providing value in several real-world domains, including video surveillance, human-robot interactions, entertainment and health applications, marketing and retail management [15, 3]. In this work, we focus on violent action detection, which is potentially useful in several real-world scenarios, such as to assist security personnel or to perform emergency calls.

Pose estimation or skeleton detection is a key tool for human action analysis from video [18, 11, 3]. In this paper, we use a pose estimation algorithm, running on the background of a video, to extract the key points of all human subjects. From the collected keypoint coordinates, we derive three new distinct features

* The work of P. Cortez was supported by Fundação para a Ciência e Tecnologia (FCT) within the Project Scope: UID/CEC/00319/2013.

(angles, velocity and contact detection), which are then merged with the keypoint coordinates, leading to an input feature vector. The generated dataset is used to train a Support Vector Machine (SVM), which predicts if there is a violent action. The set of features includes the velocity of all keypoints, six differently disposed angles and whether the persons are in contact or not. The extraction of features was done using a temporal frame step, which was set at 60 or 120 frames. A total of 32 different configuration experimental tests were performed. Overall, an excellent classification performance was achieved by the best performing model.

The paper is organized as follows. Section 2 presents the related work. Next, Section 3 describes the video data, machine learning approach and evaluation. Then, Section 4 details the experiments held and analyses the obtained results. Finally, Section 5 discusses the main conclusions.

2 Related Work

Due to the recent advances in deep learning algorithms, the use of convolutional neural networks (CNN) to predict actions on videos or images has seen a considerable growth. These methods are often paired with skeletons or other local features extracted from pose estimation algorithms, such as motion [5], to better model the action performed. Another deep learning approach was proposed in [27], which used a 3D CNN integrated with a Markov chain model to infer pose and motion from video frames. More recently, CNNs were used in [19] to extract visual information derived from temporal progression of joint coordinates.

For video object detection and classification, several other neural networks have been proposed. The ResNet architecture uses RGB images with encoded spatial-temporal features extracted from 3D skeleton keypoints [12, 21]. In 2018, the ResNet model was extended, under five distinct architectures, in [22]. Other new neural networks, handcrafted for the detection of actions, have also been proposed, such as the Attentional Recurrent Relational Network - Long Short Term Memory (ARNN-LSTM)[17].

Regarding the task of violent action detection, which is a subset of the general human action recognition, it has been mainly addressed by using hand-crafted input features, which often concentrate on visual cues found in images. For instance, in 1997, Vasconcelos and Lippman [24] used the the bodies in different frames to calculate the variation undergone throughout the video. In 2006, this method was improved by detecting blood regions on skin and analyzing them for motion intensity [6]. In 2002, motion trajectory and the orientation of different body limbs present on a frame was used to detect violent acts from video [8]. In 2014, the acceleration of body parts, as derived from the variation of the bodies on subsequent frames, was adopted in [9]. More recently, in 2016, LSTM deep learning networks were used to solve the problem of missing temporal information, on which visual features (extracted using CNN), optical flow images and acceleration flow maps were followed by an LSTM and were subject to a late fusion [10]. Expanding on this work, in 2017 the AlexNet architecture [16] was adopted [23]. The model used as inputs two subsequent frames at each step,

as a mean to encode a visual representation vector that was sent to a convolutional LSTM network. After processing the frames, several fully connected layers compute the final classification.

In this paper, we work with new visual hand-crafted features, such as angles, velocity and contact between two human subjects. These features are merged in order to encode temporal information and create a feature vector that is fed to a binary classification SVM model, aiming to predict violent behaviour.

3 Materials and Methods

3.1 Video data

In this work, we adopted the ISR-UoL 3D Social Activity Dataset [7], which contains a total of 93660 RGB images of multi-person actions, divided into 10 sessions. Each session contains 8 different acts and it is composed of a unique combination of two persons. A person can appear in another sessions, but not paired up in a similar combination. Every act represents a unique action repeated throughout the 10 different sessions. The specific personal nuances that each person exhibits when performing actions are herein captured and help increase the generalization of the dataset. An act can sometimes be divided into 4 mini-recordings, each with the same act. The dataset contains the skeleton data, the RGB data and the depth data for each image. There are a total of 8 distinct human actions:

1. handshake - 17460 video frames;
2. hug - 15084 frames;
3. help walk - 9592 frames;
4. help stand-up - 3123 frames;
5. **fight** - 16465 frames;
6. **push** - 18739 frames;
7. talk - 17895 frames; and
8. draw attention - 15920 frames.

The human actions were performed by a group of 6 people, from which 4 are male and 2 are female. For the binary target output class, we assumed the push and fight as violent or aggressive actions (around 31% of the frames), while the other classes are considered as non-violent. We note that the violent actions (fight and push) were clearly staged, in order to avoid physical injuries.

The images were derived from a set of small videos, with a duration of around 40 to 60 seconds, recorded at a frame rate of 30 frames per second. The videos capture the entire body of the subjects involved on an action, which allows for a total pose estimation inference. Figure 3.1 presents a few examples of the human actions present in the dataset.

Fig. 1. Examples of human actions from the dataset: draw attention (top left); hug (top right); handshake (bottom left); and fight (bottom right)

3.2 Machine Learning approach

The adopted pose estimation algorithm was OpenPose [4]. This algorithm was developed and it is currently supported by the CMU Perceptual Computing Lab. It uses Part Affinity Fields (PAFs) to learn how to connect the limbs of an individual and heatmaps to successfully identify multiple people in an image. It was developed using the programming language C++ and the machine learning library `Caffe`. Initially, the model was trained using the template used on the COCO keypoints dataset, with 18 different joints that when connected form a pose. More recently, a model with a total of 25 joints was released. This model considers all the previous points plus a higher concentration/detail on the foot. In this paper, we either retrieve the full 25 points or the first 9 points, which correspond to the human torso area.

Physical contact between two people, be it abrupt and quick or soft and slow, should be a strong indicator for identifying violent behaviour. In our work, it corresponds to the first extracted feature. After laying out the structure of the tree-shaped body and how the keypoints are placed on the model, we initially used box shape to envelope the body of a person. However, in preliminary experiments, we found that the box tend to include a large portion of the background of the image. Thus, wrap the human body with a more natural human-polygon based shape, aiming to detect a contact while reducing the background clutter to a minimum. We automatically detect a a contact when two human-polygons have an intersection area.

With the periodic information provided by OpenPose, it is possible to calculate the velocity of one or all the limbs of a person. The velocity feature can

be used to detect which limbs are moving quicker and more often than others. The feature is calculated comparing and then subtracting the positions of the current keypoints with the positions of the previous ones, then dividing by the time between two consecutive image frames. It should be noted that we use zero velocity values in case of the first frame of a video action scene. The velocity, in our experiments, is estimated for all the keypoints of a skeleton. For example, if 9 keypoints are used then we compute 9 velocity values.

Human anatomy restricts each joint to a specific range of angles, in which movement is allowed without any direct damage to the body integrity. As such, the angles on which certain joints are encountered on specific actions may be a clear indicator for the detection of such action. The angles were calculated using the following formula:

$$Angle = \frac{(\arctan(AB_y, AB_x) - \arctan(CB_x, CB_y)) \times 180}{\pi} \tag{1}$$

where AB_y and AB_x represent the distance, on the y and x axis, from point A to point B. Similarly, CB_x and CB_y represent the distances between points C and B. We extract 6 different angles when all 25 keypoints are used, which are localized on the right and left, elbows, shoulders and knees. When the keypoints are reduced to 9, only 4 angles are computed (by removing the right and left knees).

OpenPose cannot distinguish between two different persons. In practice, this means that the keypoints outputted by the pose algorithm can differ in order and in an irregular and unreliable way. To solve this issue, we developed a tracking algorithm, described as follows. At the beginning of the algorithm, the tracking algorithm Kernelized Correlation Filters (KCF) [13] is initialized and the Region of Interest (ROI) is established overlapping the throat joint, saving into an array the order and the keypoints from every person detected on the first frame. Then, a conditional rule checks whether the actual frame is the second frame or not. If it is, the algorithm resets the tracker defining its ROI as the throat joint and then resets the counter. If it is not, it adds 1 to the counter and updates the tracker. After that, the algorithm draws the updated tracker boxes and changes the array created at the beginning, with the new keypoints, if the actual keypoints of the point 1 are inside the updated tracker box. Figure 2 exemplifies the distinct extracted features.

The extracted image features were stored in CSV files (one per each video scene), with each column corresponding to a different feature and each row to a different frame. Typically, each video action has a duration of four minutes. To achieve shorted classification results, we designed the SVM classifier to work with very short video sequences, with a length up to 2 (60 frames) or 4 seconds (120 frames). We achieve this, the CSV files were preprocessed in order to create a short video sequence input feature vector, which is fed the SVM. Thus, the feature vector contains the concatenation of all considered features (NF) for all short sequence images (SS) of the scene (SS =60 or SS =120 frames). Thus, the input feature vector length is $NF \times SI$.

Fig. 2. Examples of the features extracted: contact detection (top left); velocity (top right); angles (bottom left); and tracking (bottom right)

As for the learning model, we used a fast SVM variant, capable of working with high dimensional input features, and that used a linear kernel.

3.3 Evaluation

A confusion matrix allows to visualize the results obtained by a classification algorithm. Each row of the matrix corresponds to an instance of a predicted class while the columns represent the actual (desired) classes. For a given class c, the matrix exhibits the values of true positives (TP_c), false positives (FP_c), false negatives (FN_c) and true negatives (TN_c). Using this matrix, several performance metrics can be computed, such as Precision, Recall and F1-Score [26]:

$$\begin{aligned} Precision &= \frac{TP_c}{TP_c + FP_c} \\ Recall &= \frac{TP_c}{TP_c + FN_c} \\ F1\text{-}Score &= 2\frac{Precision_c * Recall_c}{Precision_c + Recall_c} \end{aligned} \tag{2}$$

To aggregate the violent and non-violent class metrics into a single measure, we adopted the weight-averaging method, which weights the measure according to the class prevalence in the data.

To validate the predictive models, we used the standard holdout train/test validation split [20]. The split was based on the dataset sessions: data from by data from sessions 1 to 7 were included in the training set (70%) and images from sessions 8 to 10 were used in the to test set (30%).

4 Results

4.1 Computational environment

The experiments were conducted using code written in the Python language. In particular, we adopted the Python API available at the `OpenPose` library [1] to interact with the pose algorithm. Since the experiments required a substantial computational effort, we conducted them using a dedicated machine with an Intel i7-7800X processor and a GeForce GTX 1080 Ti.

Several components of the designed computational experimentation were implemented using known Python modules, namely: `Shapely` – to detect the intersection of two distinct polygons; `csv` – to write the extract features into the CSV format; `Numpy` – to handle the feature vector, concatenating the features from different frames; and `scikit-learn` – to run the SVM algorithm and compute the classification performance metrics. We also adopted the `OpenCV` tracking API to load the images from the dataset and implement the KCF tracking algorithm. As for the SVM implementation, we used the L2 penalty, a soft margin parameter of $C = 1$, and a training that performs a maximum of 1000 iterations.

4.2 Violent action detection results

The number of executed experiments was $E = D \times F = 4 \times 8 = 32$ experiments, with D=*4* dataset combinations and F=*8* feature setups. The datasets include 25 or 9 keypoints, with a lenght of 60 or 120 frames: A – 25 keypoints and 120 frames; B – 25 keypoints and 60 frames; C – 9 keypoints and 60 frames; and D – 9 keypoints and 120 frames. As for the features, we explored the following setups: 1 – keypoints only; 2 – keypoints and angles; 3 – keypoints and velocities; 4 – keypoints and contact; 5 – keypoints, angles and velocities; 6 – keypoints, angles and contact; 7 – keypoints, contact and velocities; and 8 – all features. Table 1 presents the overall F1-score values for each tested configuration. In the table, each configuration (column **Model**) is represented by the respective combination letter and feature digit. Overall, the best result was obtained for the configuration that uses 60 frames and a set of input features composed of 25 keypoints, 6 angles and 1 contact (A6). For this selected model, detailed classification results are presented in Table 2 and Figure 3. The total number of features used by the model is NF = (25×2 (x and y axis) ×2 (two people))+(6×2 (people))+1=113. As shown in Figure 3, the proposed SVM presents high true positive (85%) and true negative (92%) rates. Globally, high quality Precision, Recall and F1-score classification measures were achieved (89%, Table 2).

5 Conclusions

Currently, there is a vast amount of human daily activities that are recorded using digital cameras. Thus, automated systems capable of detecting interesting human behaviour from video are valuable in several real-world scenarios.

Table 1. Classification results for all experiments (best value in **bold**)

Model	F1-Score	Model	F1-Score	Model	F1-Score	Model	F1-Score
A1	0.87	A3	0.88	A5	0.88	A7	0.88
B1	0.87	B3	0.84	B5	0.85	B7	0.85
C1	0.87	C3	0.87	C5	0.81	C7	0.88
D1	0.84	D3	0.83	D5	0.83	D7	0.85
A2	0.88	A4	0.87	**A6**	**0.89**	A8	0.88
B2	0.84	B4	0.86	B6	0.85	B8	0.84
C2	0.84	C4	0.86	C6	0.82	C8	0.79
D2	0.84	D4	0.86	D6	0.83	D8	0.81

Table 2. Classification measures for the selected A6 model

Number of features (NF)	Precision	Recall	F1-Score
113	0.89	0.89	0.89

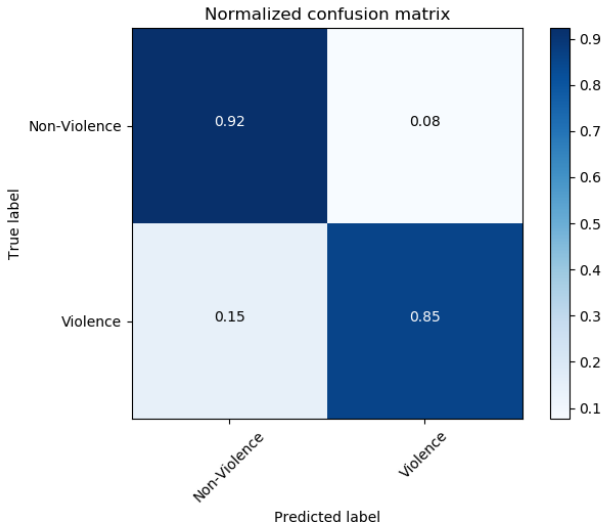

Fig. 3. Normalized confusion matrix for the selected A6 model

In this paper, we target violent action detection by using short video sequences (60 or 120 frames) and hand-engineered features: skeleton keypoints; angles and velocities computed over these keypoints; and contact detection based on human-polygon shapes. As the base learning classifier, we adopted the pop-

ular Support Vector Machine (SVM), with a linear kernel. A large set of 32 experiments was executed, by considering different short input video sequences (60 or 120 frames), number of keypoints (25 or 9), and velocity, angles and contact combinations. The system was tested on the publicly available ISR-UoL 3D Social Activity dataset, with a total of 93660 images reflecting 8 human actions. We have merged the 8 distinct actions into violent (fight and push) and non-violent examples (other actions). The best performing model used a short video sequence of 60 frames (1 second), 25 skeleton keypoints, 6 angels and human contact detection. Overall, an excellent classification performance was achieved, with a Precision, Recall and F1-score values of 89%.

In the future, we intend to extend this work by enriching the set of features (e.g., acceleration values, 3D skeleton data). We also plan to perform experiments that consider distinct video scenarios (e.g, with more than two people, inside a closer space such as a vehicle).

References

1. Openpose: Real-time multi-person keypoint detection library for body, face, and hands estimation. https://github.com/CMU-Perceptual-Computing-Lab/openpose, accessed: 2018-09-14
2. Afsar, P., Cortez, P., Santos, H.: Automatic visual detection of human behavior: A review from 2000 to 2014. Expert Syst. Appl. **42**(20), 6935–6956 (2015). https://doi.org/10.1016/j.eswa.2015.05.023, https://doi.org/10.1016/j.eswa.2015.05.023
3. Afsar, P., Cortez, P., Santos, H.M.D.: Automatic human trajectory destination prediction from video. Expert Syst. Appl. **110**, 41–51 (2018). https://doi.org/10.1016/j.eswa.2018.03.035, https://doi.org/10.1016/j.eswa.2018.03.035
4. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR (2017)
5. Chéron, G., Laptev, I., Schmid, C.: P-CNN: pose-based CNN features for action recognition. CoRR **abs/1506.03607** (2015), http://arxiv.org/abs/1506.03607
6. Clarin, C.T., Dionisio, J.A.M., Echavez, M.T., Naval, P.C.: Dove : Detection of movie violence using motion intensity analysis on skin and blood (2006)
7. Coppola, C., Faria, D., Nunes, U., Bellotto, N.: Social activity recognition based on probabilistic merging of skeleton features with proximity priors from rgb-d data. In: Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS). pp. 5055–5061 (2016)
8. Datta, A., Shah, M., Lobo, N.D.V.: Person-on-person violence detection in video data. In: Object recognition supported by user interaction for service robots. vol. 1, pp. 433–438 vol.1 (Aug 2002). https://doi.org/10.1109/ICPR.2002.1044748
9. Deniz, O., Serrano, I., Bueno, G., Kim, T.: Fast violence detection in video. In: 2014 International Conference on Computer Vision Theory and Applications (VISAPP). vol. 2, pp. 478–485 (Jan 2014)
10. Dong, Z., Qin, J., Wang, Y.: Multi-stream deep networks for person to person violence detection in videos. In: Tan, T., Li, X., Chen, X., Zhou, J., Yang, J., Cheng, H. (eds.) Pattern Recognition. pp. 517–531. Springer Singapore, Singapore (2016)

11. Du, W., Wang, Y., Qiao, Y.: Rpan: An end-to-end recurrent pose-attention network for action recognition in videos. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 3745–3754 (Oct 2017). https://doi.org/10.1109/ICCV.2017.402
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR **abs/1512.03385** (2015), http://arxiv.org/abs/1512.03385
13. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. CoRR **abs/1404.7584** (2014), http://arxiv.org/abs/1404.7584
14. Herath, S., Harandi, M.T., Porikli, F.: Going deeper into action recognition: A survey. CoRR **abs/1605.04988** (2016), http://arxiv.org/abs/1605.04988
15. Kong, Y., Fu, Y.: Human Action Recognition and Prediction: A Survey. ArXiv e-prints (Jun 2018)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012), http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
17. Li, L., Zheng, W., Zhang, Z., Huang, Y., Wang, L.: Skeleton-based relational modeling for action recognition. CoRR **abs/1805.02556** (2018), http://arxiv.org/abs/1805.02556
18. Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal LSTM with trust gates for 3d human action recognition. CoRR **abs/1607.07043** (2016), http://arxiv.org/abs/1607.07043
19. Luvizon, D.C., Picard, D., Tabia, H.: 2d/3d pose estimation and action recognition using multitask deep learning. CoRR **abs/1802.09232** (2018), http://arxiv.org/abs/1802.09232
20. Ng, A.: Machine Learning Yearning. deeplearning.ai (2018)
21. Pham, H., Khoudour, L., Crouzil, A., Zegers, P., Velastin, S.A.: Exploiting deep residual networks for human action recognition from skeletal data. CoRR **abs/1803.07781** (2018), http://arxiv.org/abs/1803.07781
22. Pham, H., Khoudour, L., Crouzil, A., Zegers, P., Velastin, S.A.: Learning and recognizing human action from skeleton movement with deep residual neural networks. CoRR **abs/1803.07780** (2018), http://arxiv.org/abs/1803.07780
23. Sudhakaran, S., Lanz, O.: Learning to detect violent videos using convolutional long short-term memory. CoRR **abs/1709.06531** (2017), http://arxiv.org/abs/1709.06531
24. Vasconcelos, N., Lippman, A.: Towards semantically meaningful feature spaces for the characterization of video content. In: Proceedings of International Conference on Image Processing. vol. 1, pp. 25–28 vol.1 (Oct 1997). https://doi.org/10.1109/ICIP.1997.647375
25. Wang, Q.: A survey of visual analysis of human motion and its applications. CoRR **abs/1608.00700** (2016), http://arxiv.org/abs/1608.00700
26. Witten, I., Frank, E., Hall, M., Pal, C.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Franscico, USA, San Francisco, CA, 4th edn. (2017)
27. Zolfaghari, M., Oliveira, G.L., Sedaghat, N., Brox, T.: Chained multi-stream networks exploiting pose, motion, and appearance for action classification and detection. CoRR **abs/1704.00616** (2017), http://arxiv.org/abs/1704.00616