IDENTIFYING VIOLENCE IN VIDEOS USING HIERARCHICAL ATTENTION NETWORKS

Shay Doner, Tamir Arbel, Nikita Shavit Dydo

Afeka College of Engineering, Tel Aviv, Israel

Abstract

Detecting violence in videos is a well-known problem that has various solutions. This work presents a novel approach to detecting violence in videos by using hierarchical attention networks (HAN) to analyze the context of each frame in relation to previous and future frames. The study evaluates the method using three existing violent behavior recognition datasets. In addition, ViTPose was chosen as the pose estimator due to its high accuracy and efficiency. The videos were transformed into a series of pose vectors, which were saved as the input for the HAN model. The HAN model was implemented using the Keras interface for TensorFlow framework, with modifications to receive pose estimation vectors instead of word embedding. The results showed improved accuracy in violent behavior recognition compared to previous methods, demonstrating the potential for this approach to contribute to the field of violence detection in videos.

1. Introduction

The concept of detecting violence in videos and identifying actions has existed for a long time [1], [2]. It has significant implications in the security and surveillance sectors, as surveillance cameras are widely used in various settings such as streets, schools, prisons, malls, etc. Given the increasing use of social media, it is crucial to develop automated methods to screen violent content. Thus, there has been a growing focus on finding solutions to this issue in recent years. There are various approaches to violence detection, with most utilizing computer vision and artificial intelligence to improve their algorithms [3]–[5]. Despite the advancements in computer vision technology and violence detection algorithms over the past decade, there are still challenges in dealing with violence detection in videos, such as detecting violence in a crowded scene [6] and the impact of context on the interpretation of actions [7]. Our team is taking a fresh approach to tackle the difficulties in violence detection by utilizing hierarchical attention networks (HAN) [8] to treat pose estimation data in analogy to text, by weighing temporal context, similar to how HAN was used for document classification.

By using skeleton key points, instead of images, as input we reduce the variability in the data. The reduced variability could help train model with less data since the variance in lighting, background etc. is removed. This study aims to increase the accuracy of violence detection by using hierarchical attention networks to analyze the context of each frame in relation to the context of previous and future frames, reducing issues like out of context false positives and overcrowded frames.

2. Methods

2.1 Data

We evaluated our method using 3 existing violent behavior recognition datasets [9]. The RWF-2000 (Real World Fighting) dataset comprises of 2000 surveillance video clips that mainly showcase one-on-one, multiple, and crowd violence. The hockey dataset comprises of 1000 videos, including both violent and non-violent actions in ice hockey games. The Expanded dataset contains 200 videos of violent and non-violent scenes sourced from movies and daily life. Detailed information and metadata of all datasets can be found in Table 1.

	RWF-2000	Hockey	Expanded dataset
Violent / Non-Violent videos	1000 / 1000	500 / 500	100 / 100
Train / Validation ratio	80% / 20%		
Resolution	360p / 480p / 720p	360p	720p
Size	1 - 10 [MB]	0.15 - 0.3 [MB]	0.3 - 3 [MB]
Length	5 [sec]	1 [sec]	2 [sec]
FPS	30	25	25 - 30
Frames	150	49	40 - 60

Table 1- Description of the datasets

2.2 Pre-Processing

The pose estimator serves as a crucial component in our machine's ability to analyze the video data and recognize various poses and movements within the frames. ViTPose was chosen as the pose estimator to tag the videos as it is known for its high accuracy and efficiency. By utilizing ViTPose, we were able to convert the video data into a series of pose vectors, which were then saved as the input for our machine. A 17-dimensional, 3-part pose estimation vectors were meticulously extracted from each frame of the videos to precisely capture the pose of each individual. The 17 key points represented specific body parts, while the 3 numbers - consisting of x and y coordinates and a confidence level - defined the location of each point within the frame space.

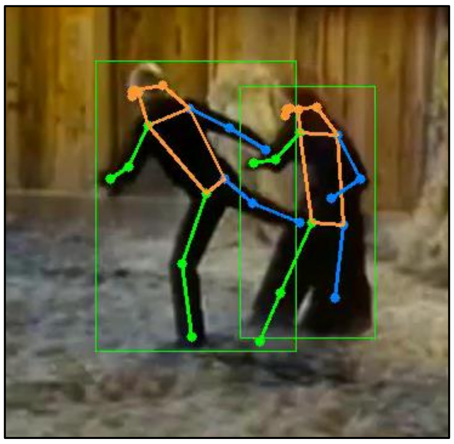

Figure 1- Pose estimation key points plotted of a frame from the expanded dataset.

This information was crucial for accurately representing the poses and movements in each video frame, leading to improved analysis and classification by the HAN model. This approach has several advantages over using raw video data, including reduced computational requirements and improved data standardization. By using the pose vectors as input, our machine can analyze and process the data more effectively and with greater precision. The pose estimations generated by ViTPose were saved in a Hierarchical Data File (HDF) format for efficient storage and retrieval of information from all video frames. The HDF was then fed into the HAN model for processing, enabling the model to classify and analyze the poses and movements in the video for improved accuracy in violent behavior recognition. The HDF file contained label categories, video files, frames, and pose embedding vectors. The label categories provided the ground truth for each video, indicating whether it contained violent or non-violent actions. The video files held the actual video data, while the frames represented individual frames. The pose embedding vectors were numerical representations of the poses within each frame, generated by ViTPose.

2.2.1 Pose Estimations Algorithms

Two algorithms, ViTPose and YOLOv5, were evaluated for the purpose of generating pose estimations.

	Backbone	AP^{Val}	AP^{Val}_{50}	AR
ViTPose	ViT-H	79.1 %	91.7 %	84.1 %
YOLOv5	Darknet-csp-d53-l	69.4 %	90.2 %	75.9 %

Table 2- Ablation study on Microsoft COCO validation set.

AR stands for Average Recall, a metric used to measure the average fraction of objects that are correctly detected out of all the objects present in the image. AP is a metric used to evaluate the accuracy of Pose Estimation algorithms. It calculates the average precision of an algorithm's predictions by comparing them to ground truth annotations. A positive is defined using object key point similarity (OKS): $OKS = \frac{\sum_i \exp\left(-\frac{d_i^2}{2s^2\kappa_i^2}\right)(v_i>0)}{\sum_i \delta(v_i>0)}$

Given the OKS, AP is calculated just as with IoU for object detection. $\boldsymbol{AP^{Val}}$ is calculated for $OKS = 0.5$ and $\boldsymbol{AP^{Val}}$ is averaged over $OKS = 0.5{:}\,0.05{:}\,.95$. [10]

Previous studies, as indicated in Table 2, revealed that the ViTPose algorithm provides more accurate and superior pose estimations compared to others [11], [12]. Hence, we decided to utilize it for our pose estimation process. The ViTPose algorithm, while favored, encountered difficulties in accurately detecting people and calculating poses in some videos. As a result, any videos lacking pose estimation were excluded from the dataset, as they couldn't be utilized as input for the HAN model. In order to completely isolate our model's errors, ground truth pose estimations should have been used. However, we found no dataset that includes violent and non-violent poses, and we weren't able to create one ourselves given the project's time constraints.

2.3 HAN model implementation

The HAN model was implemented using Keras interface for TensorFlow framework [13]. HAN was originally implemented for document classification [8], therefore, we have modified it to receive pose estimation vectors instead of word embedding.

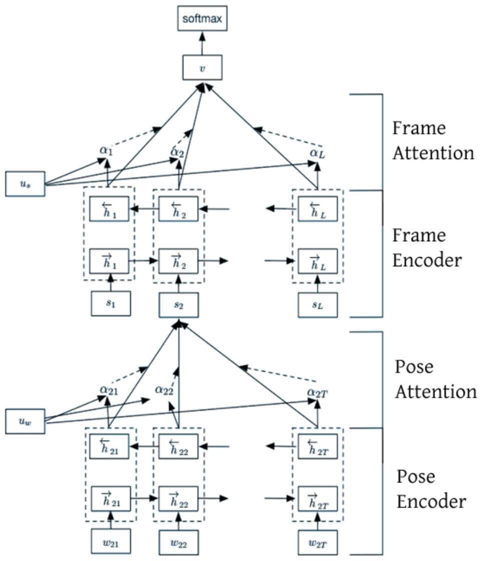

Figure 2 - HAN Architecture [8] modified for violence detection.

We used the 'keras.layers' library's pre-implemented Dense and GRU layers, and a custom implemented attention layer, as defined in the 2016 HAN paper [8]. which takes h_t as inputs and outputs s by applying the following operations:

$$u_t = \tanh(W_w h_t + b_w)\ , \qquad \alpha_i = \frac{\exp\,(u_t^T u_w)}{\sum_t \exp(u_t^T u_w)}$$

$$s = \sum_t \alpha_t h_t$$

The input for the model $n \times m$ matrix, where n is the maximal number of poses in frame, and m the length of pose estimation vector. The parameters W_w, b_w, u_W are trainable.

The inputs go into a fully connected layer with 64 dimensions and ReLu activation, and then into a bidirectional GRU layer with 100 units. The GRU's outputs (the poses annotations) are inserted to the custom attention layer, in order to create the frame representation. Keras Time Distributed wrapper is used to apply those layers to all the poses in a frame to create the frames' representations. Similar process is applied to the frames' representations to produce the video representation. The video representation is fed into a fully connected layers of 2 dimensions (that represent violent and non-violent classes) with SoftMax activation.

2.4 Training

The model is trained by Keras' fit function, using binary cross entropy as loss, and Keras' pre-implemented optimizers. For each dataset, a distinct model was constructed to enable comparison with the results reported in previous relevant studies. The training data for each model was stratify splinted into 90% for training and 10% for testing. The type of optimizer and the learning rate were selected as part of the hyperparameter tuning process.

2.5 Hyperparameters tuning

The selection of hyperparameters was carried out through a combination of grid search and k-fold cross-validation. In the initial stage, grid search with 5-fold cross-validation was applied to identify the best optimizer and learning rate. Next, a grid search with 10-fold cross-validation was done to determine the most suitable batch size and number of epochs. Finally, a 5-fold cross-validation was carried out to find the optimal number of GRU units and Danse layers dimensions respectively. The entire process was performed using the Hockey dataset, with 10% of the data reserved for future testing and to prevent data leakage. Due to hardware and time constraints, we were unable to conduct a full grid search of all hyperparameters, which is the ideal method for tuning parameters. Instead, we carried out a limited search of up to two hyperparameters at a time. Once we determined the hyperparameters from this search, we moved on to the next set.

2.6 Evaluation metrics

Accuracy and cross entropy loss were used to evaluate the results of the hyperparameters grid-search and cross-validation as used in previous relevant papers. Accuracy is defined as the proportion of correct predictions made by the model over the total number of predictions. Higher accuracy indicates that the model is making fewer prediction errors. Cross entropy loss measures the difference between the predicted probability distribution and the true distribution. A smaller value of cross entropy loss indicates that the model's predictions are closer to the true class labels.

3. Results

During the process of adjusting the hyperparameters, it was determined that the optimal combination was using the Adam optimizer with a learning rate of 0.0005. This combination was found to be the most effective among all the cross-validation tests conducted.

Optimizer	Learning Rate	Accuracy [%]
Adam	0.0005	89.7
Adam	0.01	89.2
Adam	0.001	88.1
Adam	0.0001	88.3
RMSprop	0.01	88.3
RMSprop	0.001	82.8
RMSprop	0.0005	86.3

Table 3- Grid-search result on optimizer and learning rate.

During our search for the best batch size, we considered sizes that were powers of 2. We discovered that batch sizes smaller than 32 resulted in low accuracy and high loss. Our hardware memory constraints prevented us from testing with batch sizes larger than 128, so we focused on evaluating batch sizes of 32, 64, and 128. Although the results from these sizes were similar, batch size 64 proved to be the most consistent across all 10 folds, leading us to choose it as the batch size.

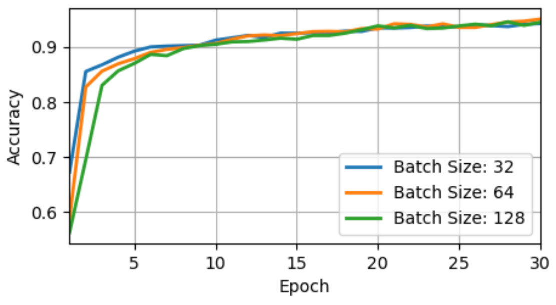

Figure 3 – Mean accuracy over 30 epochs with different batch sizes.

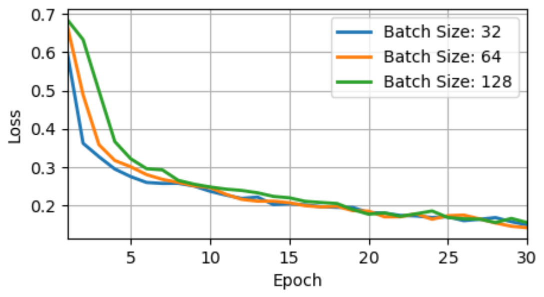

Figure 4 – Mean loss over 30 epochs with different batch sizes.

In our search for the optimal GRU unit size, we considered powers of 2 from 16 to 256, with the exception of 100 units as it was specified in the Han paper and had been used in all previous cross-validations and grid searches. After evaluating the various options, a GRU with 64 units was selected due to its ability to achieve the lowest loss and highest accuracy while producing stable results across folds.

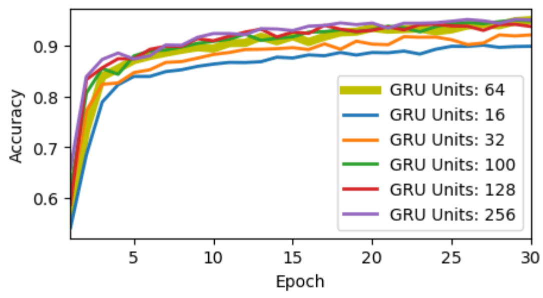

Figure 5 – Mean accuracy over 30 epochs with different GRU unit sizes.

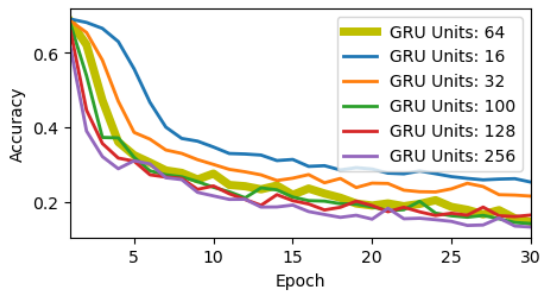

Figure 6 – Mean loss over 30 epochs with different GRU unit sizes.

The model displayed comparable behavior when searching for the ideal Dense layer dimension, utilizing dimensions that were a power of 2 ranging from 8 to 256. The size that produced the lowest loss and highest accuracy was determined to be 64.

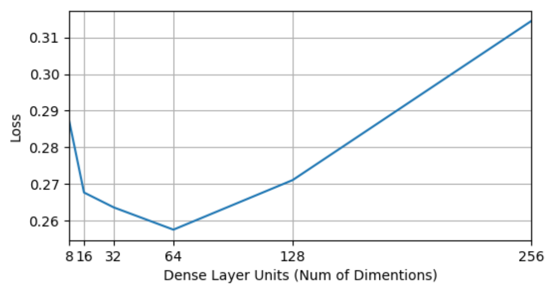

Figure 7 – Mean loss over 30 epochs with different GRU unit sizes.

Once all the hyperparameters were established, the model was constructed and trained on three separate occasions, each time using different a data set.

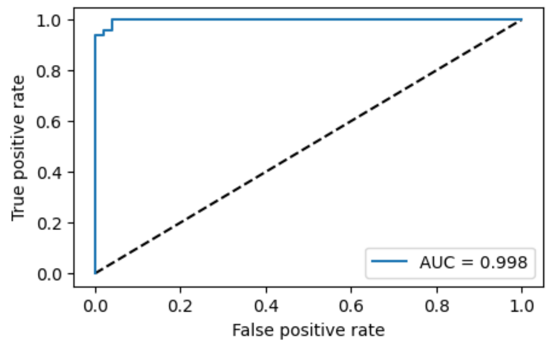

Figure 8 – ROC of the model test results on Hokey dataset.

As shown in Figure 8, the model an AUC score of 0.998. In addition, the confusion matrix presented in Figure 9 provides a representation of the model's performance, including True Positives, False Positives, True Negatives, and False Negatives.

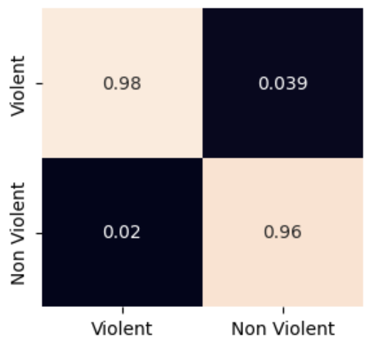

Figure 9– Confusion matrix of the model test results on Hokey dataset.

In the following a comparison of our results to optimal accuracy among previous papers. 3D-CNN[14], LRCN[15] and ECA TSM[9] use the images as inputs, while SPIL[16] uses skeleton key points, in a similar approach to ours.

Algorithm	Hokey	RWF-2000	Expended dataset
3D-CNN	94.4	82.75	91.7
LRCN	97.1	77	92.3
ECA TSM	98.995	89.23	95.0
SPIL	96.8	89.3	98.5
Ours	**98.0**	**78.1**	**95.0**

Table 4- Comparison of our model's accuracy with previous papers results

4. Conclusions

In this paper we used an original approach to segmenting violent videos is innovative, incorporating the use of pose estimation to decrease variability and a hierarchical attention neural network to prioritize the contextual significance of movement and actions. We tested and confirmed our method on three datasets that are commonly used in violence detection. Despite having limited resources for hyperparameter tuning, our results were comparable to state-of-the-art algorithms and indicate the potential for even higher performance with more optimal tuning. Our training and testing error also includes error from the pose estimation process, which would ideally be eliminated by using ground truth pose key points. The near-1 AUC score and high accuracy rate on the extended dataset demonstrate the effectiveness of our approach, outpacing most competitive algorithms.

5. Future goals

The proposed solution shows promise as a system for detecting violence in videos, through its use of a unique approach to violence classification. With further optimization and advancements in real-time pose estimation algorithms, this technology could be integrated into a real-time system. By utilizing a key points approach, the model can be extended to a 3D skeleton key points model for 3D violence detection and context analysis, potentially leading to improved results.

References

[1] X. Peng and C. Schmid, “Multi-region Two-Stream R-CNN for Action Detection,” in *Computer Vision – ECCV 2016*, Cham, 2016, pp. 744–759. doi: 10.1007/978-3-319-46493-0_45.

[2] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei, “End-To-End Learning of Action Detection From Frame Glimpses in Videos,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2678–2687. Accessed: Jan. 19, 2023. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2016/html/Yeung_End-To-End_Learning_of_CVPR_2016_paper.html

[3] E. Bermejo Nievas, O. Deniz Suarez, G. Bueno García, and R. Sukthankar, “Violence Detection in Video Using Computer Vision Techniques,” in *Computer Analysis of Images and Patterns*, vol. 6855, P. Real, D. Diaz-Pernil, H. Molina-Abril, A. Berciano, and W. Kropatsch, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 332–339. doi: 10.1007/978-3-642-23678-5_39.

[4] F. D. M. de Souza, G. C. Cha, E. A. do Valle, and A. de A Araujo, “Violence Detection in Video Using Spatio-Temporal Features,” in *2010 23rd SIBGRAPI Conference on Graphics, Patterns and Images*, Gramado, Aug. 2010, pp. 224–230. doi: 10.1109/SIBGRAPI.2010.38.

[5] W.-F. Pang, Q. He, Y. Hu, and Y.-X. Li, *Violence Detection in Videos Based on Fusing Visual and Audio Information.* 2021, p. 2264. doi: 10.1109/ICASSP39728.2021.9413686.

[6] T. Senst, V. Eiselein, A. Kuhn, and T. Sikora, “Crowd Violence Detection Using Global Motion-Compensated Lagrangian Features and Scale-Sensitive Video-Level Representation,” *IEEE Trans. Inf. Forensics Secur.*, vol. PP, pp. 1–1, Jul. 2017, doi: 10.1109/TIFS.2017.2725820.

[7] D. Freire-Obregón, P. Barra, M. Castrillón-Santana, and M. D. Marsico, “Inflated 3D ConvNet context analysis for violence detection,” *Mach. Vis. Appl.*, vol. 33, no. 1, p. 15, Dec. 2021, doi: 10.1007/s00138-021-01264-9.

[8] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, “Hierarchical Attention Networks for Document Classification,” in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, California, Jun. 2016, pp. 1480–1489. doi: 10.18653/v1/N16-1174.

[9] Q. Liang, Y. Li, B. Chen, and K. Yang, “Violence behavior recognition of two-cascade temporal shift module with attention mechanism,” *J. Electron. Imaging*, vol. 30, no. 04, Jul. 2021, doi: 10.1117/1.JEI.30.4.043009.

[10] M. R. Ronchi and P. Perona, “Benchmarking and Error Diagnosis in Multi-instance Pose Estimation,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Oct. 2017, pp. 369–378. doi: 10.1109/ICCV.2017.48.

[11] Y. Xu, J. Zhang, Q. Zhang, and D. Tao, “ViTPose: Simple Vision Transformer Baselines for Human Pose Estimation.” arXiv, Oct. 12, 2022. Accessed: Jan. 13, 2023. [Online]. Available: http://arxiv.org/abs/2204.12484

[12] D. Maji, S. Nagori, M. Mathew, and D. Poddar, “YOLO-Pose: Enhancing YOLO for Multi Person Pose Estimation Using Object Keypoint Similarity Loss,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, New Orleans, LA, USA, Jun. 2022, pp. 2636–2645. doi: 10.1109/CVPRW56347.2022.00297.

[13] M. Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems”.

[14] C. Ding, S. Fan, M. Zhu, W. Feng, and B. Jia, “Violence Detection in Video by Using 3D Convolutional Neural Networks,” in *Advances in Visual Computing*, vol. 8888, G. Bebis, R. Boyle, B. Parvin, D. Koracin, R. McMahan, J. Jerald, H. Zhang, S. M. Drucker, C. Kambhamettu, M. El Choubassi, Z. Deng, and M. Carlson, Eds. Cham: Springer International Publishing, 2014, pp. 551–558. doi: 10.1007/978-3-319-14364-4_53.

[15] J. Donahue *et al.*, “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description”.

[16] Y. Su, G. Lin, J. Zhu, and Q. Wu, “Human Interaction Learning on 3D Skeleton Point Clouds for Video Violence Recognition,” in *Computer Vision – ECCV 2020*, vol. 12349, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 74–90. doi: 10.1007/978-3-030-58548-8_5.