Activity Detection Using Deep Convolutional Neural Networks

Daniel Sawyer
University of South Florida
danielsawyer@mail.usf.edu

Neilesh Sambhu
University of South Florida
nsambhu@mail.usf.edu

Abstract

Spatio-temporal localization of activities in long surveillance videos is a hard task, especially given the occurrence of simultaneous activities across different temporal and spatial scales. We tackle this problem using a cascaded region proposal and detection (CRPAD) framework for frame-level simultaneous action detection and tracking. We propose the use of a frame-level spatial detection model based on advances in object detection and a temporal linking algorithm that models the temporal dynamics of the detected activities. We show results on the VIRAT dataset through the Activities in Extended Video (ActEv) challenge in the TrecVID competition [1, 2].

1. Introduction

We aim to address the problem of spatio-temporal action detection: given long video sequence with multiple activities, our goal is to detect, classify and track every action and their subsequent actors at the frame-level. Compared with action detection and recognition, the task of action detection and tracking presents an even stiffer challenge due to arbitrary temporal duration of actions, simultaneous actions and large intra-class variation within individual action tubes. There have been prior works on the problem of action detection, leveraging advances in deep learning [3, 4]. The common approach has been to detect actions in individual frames or short frame snippets and then temporally link such spatio-temporal regions (called action tubes) to detect action segments in long videos. Such methods process motion and appearance based features separately and fuse them for the final detection.

We follow a similar approach and propose a cascaded region proposal and detection (CRPAD) framework for frame-level simultaneous action detection and tracking. The proposed approach has two major parts: (1) a spatial action proposal network based on YOLO [4] and (2) a probabilistic temporal linking model that takes the frame-wise spatial detections and outputs actions tubes such as those in [5].

Related Work Advances in object detection and recognition have inspired a variety of approaches to action detection. There have been several approaches to temporal detection of actions in untrimmed videos [6, 7, 8, 9] and spatial detection in trimmed videos [10, 11]. There have, however, been fewer approaches that tackle the problem of spatio-temporal localization of actions in videos [12, 5, 13]. Some approaches to temporal segmentation have the underlying assumption that there are no simultaneous actions occurring in the same temporal segment. Surveillance videos such as those in the VIRAT dataset[14] pose a different set of problems: (1) there can be multiple simultaneous activities, (2) a actor or object can have multiple action labels and (3) there are large intra-class variations and small inter-class variations. The latter poses a significantly different challenge to traditional approaches to object detection, where there are less intra-class variations and more inter-class variations.

2. Proposed Approach

As shown in Figure 1, a feature extraction network and a frame-wise action detection network are combined to generate frame-level action proposals. These frame-wise detection outputs are combined linked temporally to generate spatio-temporal action detections. In this section, we discuss the model architecture and training procedures in detail.

2.1. You Only Look Once Architecture

The You Only Look Once (YOLO) architecture classifies objects within images and creates the corresponding bounding boxes [4]. The original YOLO architecture is followed by further improvements in YOLOv2 [15] and YOLOv3 [16]. We use Redmon's proposed architecture but not their pre-trained model. More specifically, we train our own YOLO-based network using the VIRAT-V1 dataset: `https://data.kitware.com/#collection/56f56db28d777f753209ba9f`.

2.2. YOLOv1

YOLO distinguishes itself from other object-detection networks in that YOLO uses a single convolutional neural

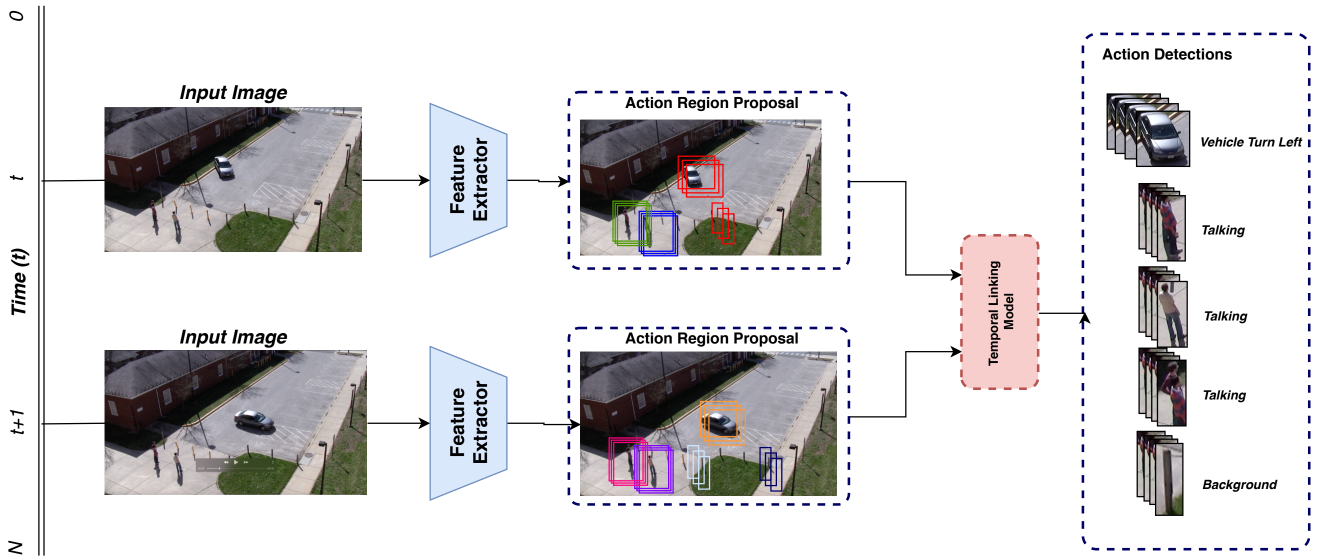

Figure 1. **Overall architecture**: The proposed approach is shown here. There are three basic components to the approach: a feature extraction network, a frame-wise action detection network and a temporal linking model.

network to predict bounding boxes for images and classify those images at the same time [4]. Older approaches perform object detection and classification with two separate classifiers. Moreover, instead of looking through an image only once, the systems use sliding windows to ensure that the classifier runs across the entire image. Consequently, by having these intricate systems, the pipelines are inherently slow and difficult to improve because each network is trained separately, whereas YOLO has only a single network to optimize.

The primary benefits of YOLO are speed and generalizability. However, regarding accuracy, YOLO is not as good as the state-of-the-art detection systems. YOLO is excellent at identifying objects quickly but has difficulty with pinpointing certain objects, particular those which are small relative to the entire image [4]. However, the difficulty with detecting small objects has mostly been resolved in YOLOv3.

YOLO utilizes features from all parts of the image when classifying each bounding box, enabling global reasoning for an image. Regarding the architecture, YOLO divides an image into a grid with $S * S$ cells, where each cell that contains the center of an object is accountable for detecting that object.

For each cell within the $S * S$ grid, there are B bounding boxes. Each bounding box has a corresponding confidence score defined as the probability of detecting an object multiplied by the intersection over union of the predicted bounding box and the ground truth bounding box. In other words, the confidence score indicates how likely a bounding box is to contain an object as well as how correct the predicted bounding box is to the ground truth. Mathematically, confidence can be expressed as follows: $confidence = P(Object) * IOU_{prediction\ and\ ground\ truth\ (PGT)}$. Along with the confidence score, each bounding box has four additional values: x, y, w, h. The (x, y) point represents the center of the bounding box. The w and h represent the width and height, respectively.

Moreover, each cell within the $S * S$ grid has a single set of conditional class probabilities C. Given there is an object within a cell, we want to determine the probability that object is of a specific class: $P(Class_i|Object)$. Because of the constraint that each cell contains only one set of probabilities, we can encounter issues when two objects of the same class are located in the same cell, as both objects would be encompassed within one bounding box.

When it comes time for testing, we must multiply the aforementioned conditional class probabilities by the confidence scores for each cell:

$$P(Class_i|Object) * confidence$$

$$= (\frac{P(Class_i) * P(Object)}{P(Object)}) * (P(Object) * IOU_{PGT})$$

$$= P(Class_i) * P(Object) * IOU_{PGT}$$

The output of this expression gives confidence scores for each class for each bounding box, where the value encodes both the probability of a bounding box containing an object belonging to that class and the suitability of the bounding box to the object.

As shown in Figure 2, YOLO has 24 convolutional layers and 2 subsequent fully connected layers [4]. YOLO alternates between $1 * 1$ reduction layers and then $3 * 3$ convolutional layers.

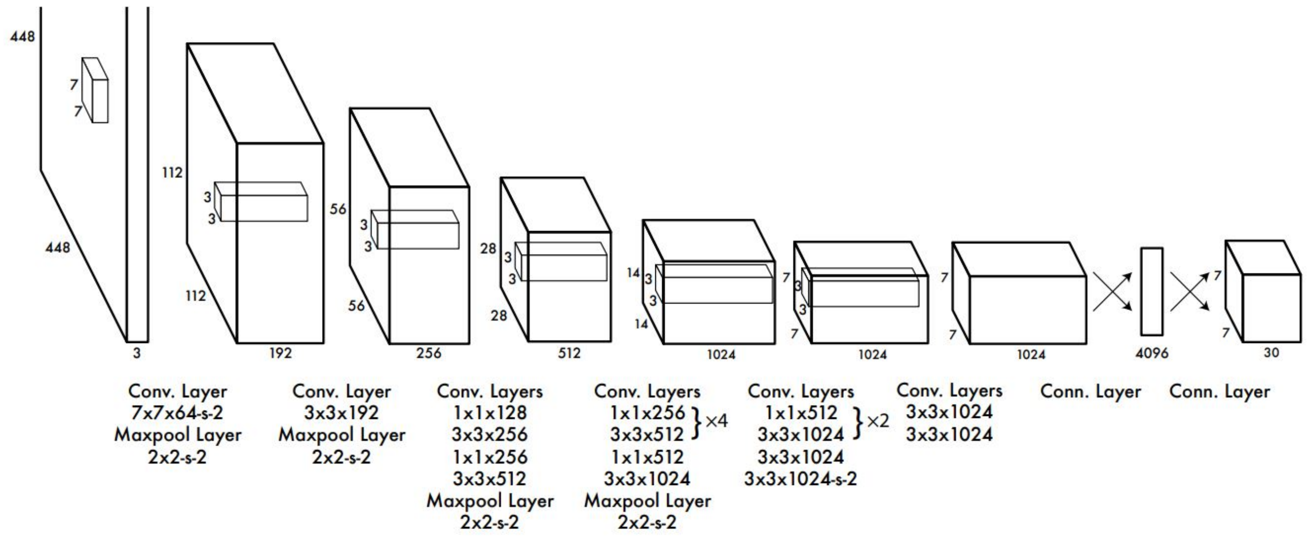

Figure 2. **YOLO network architecture**: The YOLO architecture has 24 convolutional layers and 2 subsequent fully connected layers. YOLO alternates between $1 * 1$ reduction layers and then $3 * 3$ convolutional layers [4].

2.3. YOLO9000

YOLOv2, called YOLO9000[15], which has made some significant improvements over its previous iteration. YOLO9000 received improvements in overall performance and speed by adding batch normalization and a high resolution classifier, which give a 6% increase in accuracy. YOLO9000 removes the fully connected layers and uses anchor boxes to predict the bounding boxes; this also speeds it up because you only need to predict the offsets instead of coordinates and also makes it easier for the network to learn.

YOLO9000 also shows better performance on fine-grained features. This allows it to pickup smaller objects or smaller features in larger objects. YOLO9000 also has multi-scaling training, which allows images/videos of varying resolution on the fly. This gives it more generalization with different size images and videos.

2.4. YOLOv3

YOLOv3[16] did not add too many new features or performance upgrades, but it does have a few. It uses Darknet-53 which has 53 convolutional layers, which is significantly larger than YOLO9000's Darknet-19. YOLOv3 is slightly slower than YOLO9000 but is much more accurate while still running in real-time. YOLOv3 also follows the trend of YOLO9000 in that it does even better on smaller objects and fine-grained features. However, it tends to do worse on medium to large objects.

2.5. Frame-wise Action Region Proposal

We use a spatial action detection network, similar to that proposed by Redmon *et al* [4]. We extract spatial features using a deep convolutional neural network, *Darknet-53*. The action proposal network is built on top of the final convolutional layer of the Darknet-53 network. An "*actionness*" map is constructed by convolving across the final activation map of the CNN feature extraction model and constructing a $S \times S$ grid for the image and class probabilities for each of the grids. We fix the grid size to be 13×13 and predict the probability of each class occurring in each grid. We use this grid to then regress the bounding box similar to what is proposed in [4].

Training Details: We first pretrain the feature extraction network, *Draknet-53* on the ImageNet dataset[17]. We then train the final detection layer on the target dataset, *VIRAT*. The network was trained end-to-end at the frame-level, with the minibatch size set to be 128. We also ensured that the minibatch is not dominated by any one sequence from a sigle video by careful selection of frame images.

2.6. Temporal Linking Model

We employ a probabilistic temporal linking, based on the Viterbi algorithm [18]. The bounding box proposals ($d_i \in D$) from the action proposal network defined in Section 2.5 are used as input to the temporal linking model. A temporal affinity score is computed for each pair of region proposals at time t (d_t) and $t+1$ (d_{t+1}). The region proposals with maximum temporal affinity are combined to form action tube proposals. The temporal affinity score between two regions, d^i_t and d^j_{t+1}, is given by

$$S_c(d^i_t, d^j_{t+1}) = (1-\beta) * (E_c(d^i_t) + E_c(d^j_{t+1})) + \psi_{d^i_t, d^j_{t+1}} \tag{1}$$

where $\psi_{d_t^i, d_{t+1}^j}$ is the distance between the center of the bounding boxes d_t^i, d_{t+1}^j and $E_c(\cdot)$ is the class confidence score of the given region proposal. The linking score is high for region proposals that share class confidence scores and overlap highly in consecutive frames.

3. Empirical Evaluation

We evaluate the efficacy of the proposed approach at each stage of the pipeline. For evaluating the spatial localization capacity of the region proposal network, we set a threshold of 50% Intersection over Union (IoU) and compute the accuracy of detected bounding boxes regardless of the predicted class. We evaluate overall framework by the probability of missed detection as defined in the Actev challenge TrecVid Competition[1, 2].

From our experiments, we see region proposal network has an accuracy of 57.23% across all activities on the validation set. This corresponds to the detection of 38.64% activities on the validation set, set a threshold of 10% temporal overlap between the groundtruth and frame-wise bounding boxes predictions. Overall, the proposed approach achieves a p-miss of 85% and 93.4% at the rate of 0.15 frames per second on the validation and test sets respectively and 68.12% p-miss at the rate of 1.0 false alarms per minute on the validation set.

4. Discussion and Future Work

As can be seen from the experimental evaluation, the proposed approach provides a platform for spatial localization of actions at the frame level without explicit temporal modeling. We believe that the use of temporal modeling at both the semantic and feature levels would help in improving the performance of the proposed approach. Additionally, we find that our approach is able to identify the differences in fine-grained action classes such as *vehicle turn left* and *vehicle turn right* with a high degree of accuracy, even without explicit temporal modeling.

References

[1] A. F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and trecvid," in *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, (New York, NY, USA), pp. 321–330, ACM Press, 2006. 1, 4

[2] G. Awad, A. Butt, K. Curtis, Y. Lee, J. Fiscus, A. Godil, D. Joy, A. Delgado, A. F. Smeaton, Y. Graham, W. Kraaij, G. Qunot, J. Magalhaes, D. Semedo, and S. Blasi, "Trecvid 2018: Benchmarking video activity detection, video captioning and matching, video storytelling linking and video search," in *Proceedings of TRECVID 2018*, NIST, USA, 2018. 1, 4

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 1

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016. 1, 2, 3

[5] G. Gkioxari and J. Malik, "Finding action tubes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 759–768, 2015. 1

[6] S. Stein and S. J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 729–738, ACM, 2013. 1

[7] H. Kuehne, J. Gall, and T. Serre, "An end-to-end generative framework for video segmentation and recognition," in *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pp. 1–8, IEEE, 2016. 1

[8] B. L. Bhatnagar, S. Singh, C. Arora, C. Jawahar, and K. CVIT, "Unsupervised learning of deep feature representation for clustering egocentric actions," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 1447–1453, AAAI Press, 2017. 1

[9] J.-B. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, and S. Lacoste-Julien, "Unsupervised learning from narrated instruction videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4575–4583, 2016. 1

[10] K. Soomro, H. Idrees, and M. Shah, "Action localization in videos through context walk," in *Proceedings of the IEEE international conference on computer vision*, pp. 3280–3288, 2015. 1

[11] L. Wang, Y. Qiao, X. Tang, and L. Van Gool, "Actionness estimation using hybrid fully convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2708–2717, 2016. 1

[12] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *Proceedings of the IEEE international conference on computer vision*, pp. 3164–3172, 2015. 1

[13] Y. Tian, R. Sukthankar, and M. Shah, "Spatiotemporal deformable part models for action detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2642–2649, 2013. 1

[14] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. Aggarwal, H. Lee, L. Davis, *et al.*, "A large-scale benchmark dataset for event recognition in surveillance video," in *Computer vision and pattern recognition (CVPR), 2011 IEEE conference on*, pp. 3153–3160, IEEE, 2011. 1

[15] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," *arXiv preprint*, 2017. 1, 3

[16] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018. 1, 3

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, Ieee, 2009. 3

[18] G. D. Forney, “The viterbi algorithm,” *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973. 3