# DOD-CNN: DOUBLY-INJECTING OBJECT INFORMATION FOR EVENT RECOGNITION

*Hyungtae Lee*[⋆†] *Sungmin Eum*[⋆†] *Heesung Kwon*[⋆]

[⋆]US Army Research Laboratory [†]Booz Allen Hamilton Inc.

## ABSTRACT

Recognizing an event in an image can be enhanced by detecting relevant objects in two ways: 1) indirectly utilizing object detection information within the unified architecture or 2) directly making use of the object detection output results. We introduce a novel approach, referred to as Doubly-injected Object Detection CNN (DOD-CNN), exploiting the object information in both ways for the task of event recognition. The structure of this network is inspired by the Integrated Object Detection CNN (IOD-CNN) where object information is indirectly exploited by the event recognition module through the shared portion of the network. In the DOD-CNN architecture, the intermediate object detection outputs are directly injected into the event recognition network while keeping the indirect sharing structure inherited from the IOD-CNN, thus being 'doubly-injected'. We also introduce a batch pooling layer which constructs one representative feature map from multiple object hypotheses. We have demonstrated the effectiveness of injecting the object detection information in two different ways in the task of malicious event recognition.

***Index Terms***— IOD-CNN, object detection, event recognition, malicious crowd dataset, malicious event recognition

## 1. INTRODUCTION

When figuring out what is happening in a photo, we typically make use of its relevant objects depicted in the scene. For example, if a bride, a bridegroom, and flowers are shown, we can naturally infer that it is highly likely to be related to a wedding event. Similarly, automatically recognizing such an event in an image should be empowered by exploiting relevant object information.

There are two approaches to exploit the object detection information in assisting the task of event recognition. The first approach is to make use of a separately constructed object detection module and its output for boosting the event recognition. In this approach, the object detection results can either be directly fed into the event recognition module [1, 2, 3] or be integrated with the event recognition output via a late fusion [4, 5, 6, 7, 8, 9, 10, 11]. The second approach is to transfer the object information by sharing the network weights between the object detection and event recognition and co-learning them in a unified architecture.

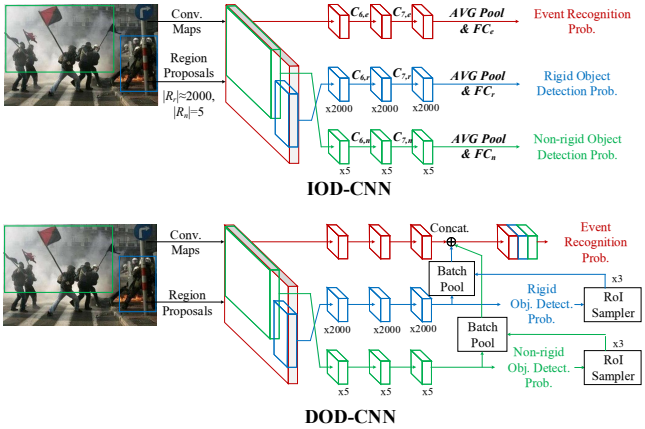


**Fig. 1**. **IOD-CNN and DOD-CNN Architectures.** Red, blue, and green arrows indicate the computational flow responsible for event recognition, rigid object detection, and non-rigid object detection, respectively.

Eum et al. [12] showed the effectiveness of the second approach by devising the IOD-CNN (Integrated Object Detection CNN) architecture which consists of three networks for event recognition, rigid object detection, and non-rigid object detection, where some initial layers are shared across all three tasks. Eum et al. claim that the enhancement in the event recognition performance was achieved by indirectly transferring the relevant object information by sharing certain portion of the weights in the network during the co-learning of multiple tasks. In [12], we also note that applying a late fusion over three different tasks bring an additional performance increase. This observation provides another evidence that both approaches (direct and indirect usage of object information) are complementary to each other when pulling up the event recognition performance.

In this paper, we adopt a function to directly inject the

object detection output into the event recognition module to complement the IOD-CNN architecture which only utilizes the object information indirectly. We refer to our novel architecture as DOD-CNN (Doubly-injected Object Detection CNN). In this architecture, rigid object detection and non-rigid object detection network output feature maps are extracted from the intermediate layers for multiple regions-of-interests (RoIs). It then selects a certain number of feature maps with highest classification probabilities ($RoI\ Sampler$) and aggregates them into one single map by using the *batch pooling*. Feature maps for rigid object detection and non-rigid object detection are then concatenated with the event recognition feature map at the same layer depth (after $C_7$ layers), where the concatenation is performed across the channel direction. The architectural comparison between the DOD-CNN and the IOD-CNN is illustrated in Figure 1.

We evaluated the proposed approach on the Malicious Crowd Dataset [11]. The experiments demonstrate that utilizing the object detection information in both direct (injecting the feature maps) and indirect (transferring the information via shared weights) ways are effective in enhancing malicious event recognition performance.

Our contributions can be summarized as:

1. Develop a novel architecture, DOD-CNN, which directly injects the object detection output into the event recognition network.

2. Introduce a batch pooling layer which aggregates multiple feature maps corresponding to multiple RoIs into a single representative feature map.

3. Demonstrate the effectiveness of DOD-CNN on the Malicious Crowd Dataset.

## 2. IOD-CNN

IOD-CNN [12] consists of shared layers, a RoI pooling layer, and three separate modules each responsible for event recognition, rigid object detection, and non-rigid object detection, respectively, as shown in Figure 1. Original architecture uses three fully connected (FC) layers (known as $FC_6$, $FC_7$, and $FC_8$) for each module. To adapt this architecture for DOD-CNN while considering the memory issues, the three FC layers in each module are replaced by two convolutional layers ($C_6$ and $C_7$), one average pooling layer, and one FC layer, where the output dimension of the FC layer is set to match the number of events or objects. Memory requirement significantly increases as the width of the last FC layer has to match the enlarged concatenated feature maps, and thus, the first and the second FC layers were pulled out. However, this modification cannot avoid a performance loss (from 93.6% to 90.7% in Table 1) because newly adopted convolutional layers is learned from scratch instead of finetuning from the FC layers of any predefined network trained over a large-scale dataset.

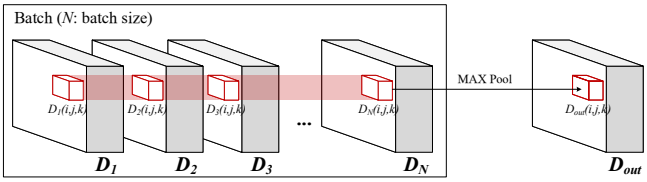


**Fig. 2**. **Batch Pooling Layer.** Batch pooling layer takes a batch with size $N$, and outputs a batch with size of one.

IOD-CNN receives an input image and passes it through the shared layers. The RoI pooling layer takes in the per-image feature map which is the output of the shared layers along with three sets of RoIs generated for three tasks. While an entire region of the input image is used as the RoI for event recognition, selective search [13] and multi-scale sliding windows [14, 15, 16, 17] are used to generate the RoIs for rigid and non-rigid object detection, respectively. For the rigid object detection, approximately 2000 RoIs are generated for each image while five RoIs are considered for the non-rigid object detection. For each RoI, per-RoI feature map is computed via RoI pooling and is fed into its corresponding task-specific module.

## 3. DOD-CNN: DOUBLY-INJECTING OBJECT INFORMATION

### 3.1. Architecture

DOD-CNN adopts a novel function to pull out the intermediate output of object detection networks and directly inject them into event recognition networks, when compared to the baseline IOD-CNN architecture, as shown in Figure 1. Output feature maps from the $C_7$ layers in all three modules are concatenated across the channel direction. This concatenated map is pooled via an average pooling and then fed into the subsequent $FC_e$ layer of the event recognition module. If AlexNet [18] is used as the backbone of our DOD-CNN, all maps share the same dimension as $6 \times 6 \times 256$ and the concatenated map becomes $6 \times 6 \times 768$. Accordingly, layers taking this concatenated map as an input should proportionally increase the filter depth, which then leads to a memory issue requiring an architectural modification when inheriting from IOD-CNN.

For each image, the network evaluates multiple RoIs for rigid and non-rigid object detections. Among multiple feature maps corresponding to these RoIs, we only use a selected number of RoIs with highest classification probabilities (three RoIs for our experiments). These selected feature maps are then transformed into a single map via the *batch pooling layer*. Note that the feature map selection process is performed to disregard the RoIs which are likely to contain no object information.

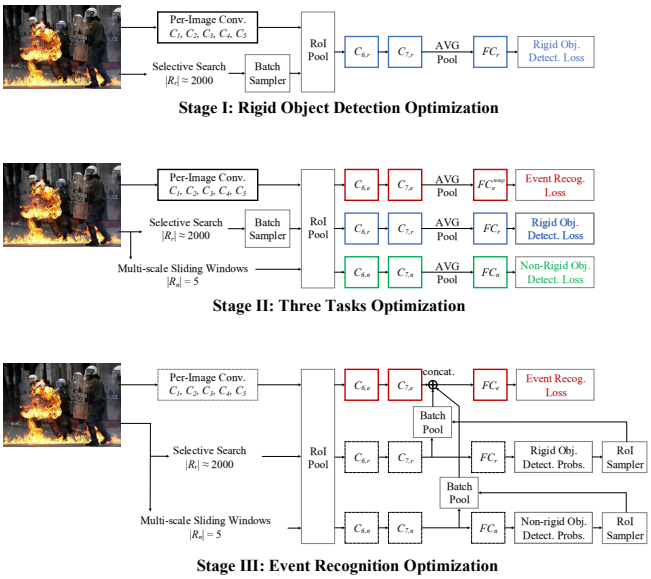


**Fig. 3**. **Three-stage Cascaded Optimization.** DOD-CNN is trained in three stages. **Black**, red, blue, and green boxes indicate the shared layers, event recognition module, rigid object detection module, and non-rigid object detection module, respectively. Black dashed boxes indicate the layers which are not updated in the corresponding stage.

**Batch Pooling Layer.** The function of this layer is to reduce the batch size as the output has to match the size of only one feature map. Batch pooling is computed as follows:

$$D_{out}(i,j,k) = \max\{D_l(i,j,k)\}_{l=1,2,\cdots,N}, \qquad (1)$$

where $D_l(i,j,k)$, $l = 1,2,\cdots,N$ is an activation at a spatial location $(i,j)$, $k^{th}$ channel in $l^{th}$ feature map in a batch, where a batch contains $N$ feature maps. $D_{out}$ is the output feature map of the batch pooling layer which contains the most significant activations among all the feature maps in the batch. The process of the batch pooling is illustrated in Figure 2.

When learning DOD-CNN, the back-propagation does not pass through the batch pooling layer to hold back the event recognition network from affecting the other object detection network optimization.

### 3.2. Training

DOD-CNN is trained using a mini-batch stochastic gradient descent (SGD) optimization approach. A batch contains two images comprising one malicious image and one benign image. For event recognition and non-rigid object detection modules, 1 and 5 RoIs are generated per image, thus 2 and 10 RoIs are used as one batch, respectively. For training rigid object detection, approximately 2000 RoIs are generated for each image while a batch only takes 64 RoIs randomly selected from the entire RoI set of each image. Accordingly, a large number of batches are necessary to cover the entire RoI set for training rigid object detection.

To deal with this issue, we adopt a three-stage cascaded optimization strategy to train the DOD-CNN. In the first stage, only the layers used in performing rigid object detection are trained to allow more training iterations. Then, as the second stage, all three tasks are co-optimized in an end-to-end fashion in the second stage. In this stage, the event recognition network does not use the intermediate output from rigid and non-rigid object detections. This stage is used to learn the shared layers and two object detection modules. Note that the first and second stages are equivalently used to train the IOD-CNN in [12]. In the third stage, event recognition network is trained by directly injecting the object detection feature maps into the event recognition module while the weights in the shared layers and object detection modules are fixed. While the weights in $C_{6,e}$ and $C_{7,e}$ layers are inherited from the second stage, $FC_e$ layer is newly initialized according to a Gaussian distribution with mean of 0 and standard deviation of 0.01. $FC_e$ layer cannot be inherited from the previous stage because this layer has a different depth when compared with the corresponding layer ($FC_e^{temp}$ in Figure 3) in the previous stage. Figure 3 illustrates the three stages used to train the DOD-CNN. For the first stage, we used the learning rate of 0.001, 50k iterations, and the step size of 30k. For the second and third stage, we trained with the learning rate of 0.0001, 20k iterations, and the step size of 12k.

To label the RoIs (for training purpose) in the rigid and non-rigid object detection, we have used 0.5 and 0.1 as the thresholds for the intersection over union (IoU) metric. We consider any RoI, whose IOU with respect to the ground truth bounding box is larger than the threshold, as a positive training example. For the rigid object detection, RoIs whose IoU is lower than 0.1 are used as negative training examples.

### 3.3. Event Recognition

While IOD-CNN only uses event recognition network for testing, DOD-CNN needs to utilize all the networks. This is because DOD-CNN directly utilizes the intermediate output and classification probabilities from the object detection networks. This may increase a memory size. Please note that replacing *FC* by *C* significantly reduced the required memory size, which will compensate the increased memory size caused by including the object detection networks for testing.

## 4. EXPERIMENTS

### 4.1. Dataset

We have selected the Malicious Crowd Dataset [11, 19] as it provides the appropriate components in evaluating the effect of using object information for event recognition. The dataset contains 1133 crowd images and is equally split into malicious and benign classes. For evaluation, half of each

| Method | | AP (%) |
|---|---|---|
| IOD-CNN | Original [12] | 93.6 |
| | Modified | 90.7 |
| DOD-CNN | | **94.6** |

**Table 1**. **Event recognition** average precision (AP).

| Task | Single-task | IOD-CNN | DOD-CNN |
|---|---|---|---|
| E | 89.9 | 90.7 | **94.6** |
| R | **8.1** | 7.8 | 7.8 |
| N | 30.4 | **37.2** | **37.2** |

**Table 2**. **Single task versus multitask performance.** Task: **E**: Event Recognition, **R**: Rigid Object Detection, **N**: Non-rigid Object Detection.

class is used for training and the remaining is used for testing. Along with the event class labels, bounding box annotations for three rigid objects (*police*, *helmet*, and *car*) and two non-rigid objects (*fire* and *smoke*) are provided. Details on how these objects are selected are given in [11].

### 4.2. Performance Evaluation

To demonstrate the effectiveness of exploiting object detection output for event recognition, we compare the DOD-CNN with two baselines: IOD-CNN [12] and its modified version which replaces three *FC* layers with two convolutional layers and one *FC* for each module. Their event recognition accuracies are shown in Table 1. DOD-CNN outperforms the two baselines by at least 1.0% AP which shows that it is a better practice to include (in addition to the indirect sharing) the direct injection of the intermediate object detection network outputs to the event recognition module. We also note that modified IOD-CNN architecture under-performs than the original architecture by 2.9%. This performance drop may have been caused by using the layers which are not inherited from the network trained on a large-scale Places dataset [20].

We have also carried out an experiment to analyze how the performance of each task changes when applying IOD-CNN (indirect sharing) or DOD-CNN (both indirect and direct injection of object information). Note that the optimization of DOD-CNN and IOD-CNN share the first and second stages in the training process while DOD-CNN takes an additional third stage where only the event recognition module is trained with directly injected object detection output features. Therefore, task-wise performances for rigid and non-rigid object detection modules in the IOD-CNN and DOD-CNN are naturally equivalent. Table 2 shows that, when the tasks are co-learned, non-rigid object detection performance is boosted while rigid object detection performance is sacrificed.

| | IOD-CNN | DOD-CNN: Direct Injection Location | | |
|---|---|---|---|---|
| | | $C_6$ | $C_7$ | $C_6$ & $C_7$ |
| AP (%) | 90.7 | 91.4 | **94.6** | 93.2 |

**Table 3**. **Performance comparison w.r.t. the injection location of object detection output.** Note that IOD-CNN does not adopt the direct injection of object detection output.

| Method | w/o DBF | w/ DBF | gain |
|---|---|---|---|
| IOD-CNN [12] | 93.6 | 94.2 | +0.6 |
| DOD-CNN | 94.6 | 94.7 | +0.1 |

**Table 4**. **Late fusion performance (DBF [21]).**

### 4.3. Ablation Studies

**Injection Location of Object Detection Output.** To find the most effective way to directly inject the object detection output into the event recognition module via feature map concatenation, we have conducted an experiment to explore three different concatenation locations in the network. Table 3 shows that the best location to adopt the feature map concatenation is after $C_7$, which is closer to the end of the network.

**Effect of Late Fusion.** We have also applied a late fusion approach over the final scores of all three tasks. We have selected the dynamic belief fusion (DBF) [21] as the fusion method. Table 4 presents the event recognition performance of IOD-CNN and DOD-CNN after applying the late fusion. The performance of the IOD-CNN was enhanced via a late fusion by 0.6% AP, while only 0.1% AP has been gained by the combination of DOD-CNN and DBF. This observation suggests that the DOD-CNN contains the functionality of the late fusion embedded into a network itself, as it directly integrates the intermediate outputs of the three tasks. Thus, practically, DOD-CNN requires no additional late fusion approaches at the end of the network for performance boost.

## 5. CONCLUSION

We have developed a CNN-based event recognition approach which is referred to as DOD-CNN. This architecture is inspired by the architecture of IOD-CNN, where object detection information is indirectly exploited for enhancing the event recognition performance by utilizing the shared layers between those two. In this architectural integration, our network also adopts the function of using object detection output for event recognition into the architecture. The experimental results show that taking advantage of both indirect and direct way of injecting the object information for the event recognition within DOD-CNN is highly effective in boosting the performance.

## 6. REFERENCES


[1] Tim Althoff, Hyun Oh Song, and Trevor Darrell, "Detection bank: An object detection based video representation for multimedia event recognition," in *ACM MM*, 2012.

[2] Mihir Jain, Jan C. van Gemert, and Cees G. M. Snoek, "What do 15,000 object categories tell us about classifying and localizing actions?," in *CVPR*, 2015.

[3] Yu-Wei Chao, Zhan Wang, Yugeng He, Jiaxuan Wang, and Jia Deng, "HICO: A benchmark for recognizing human-object interactions in image," in *ICCV*, 2015.

[4] Limin Wang, Zhe Wang, Wenbin Du, and Yu Qiao, "Object-scene convolutional neural networks for event recognition in images," in *CVPR Workshops*, 2015.

[5] Limin Wang, Zhe Wang, Sheng Guo, and Yu Qiao, "Better exploiting OSCNNs for better event recognition in images," in *ICCV Workshops*, 2015.

[6] Ryan M. Robinson, Hyungtae Lee, Michael J. McCourt, Amar Marathe, Heesung Kwon, Chau Ton, and William Nothwang, "Human-autonomy sensor fusion for rapid object detection," in *IROS*, 2015.

[7] Hyungtae Lee, Heesung Kwon, Ryan M. Robinson, Daniel Donavanik, William D. Nothwang, and Amar R. Marathe, "Task-conversions for integrating human and machine perception in a unified task," in *IROS*, 2016.

[8] Hyungtae Lee, Heesung Kwon, Ryan M. Robinson, William D. Nothwang, and Amar R. Marathe, "An efficient fusion approach for combining human and machine decisions," in *SPIE Defense+Commercial Sensing (DCS)*, 2016.

[9] Yilun Cao*, Hyungtae Lee*, and Heesung Kwon, "Enhanced object detection via fusion with prior beliefs from image classification," in *ICIP*, 2017, (* indicates an equal contribution.).

[10] Yu-Wei Chao, Yunfan Liu, Xieyang Liu, Huayi Zeng, and Jia Deng, "Learning to detect human-object interactions," in *WACV*, 2018.

[11] Hyungtae Lee*, Sungmin Eum*, Joel Levis*, Heesung Kwon, James Michaelis, and Michael Kolodny, "Exploitation of semantic keywords for malicious event classification," in *ICASSP*, 2018, (* indicates an equal contribution.).

[12] Sungmin Eum*, Hyungtae Lee*, Heesung Kwon, and David Doermann, "IOD-CNN: Integrating object detection networks for event recognition," in *ICIP*, 2017, (* indicates an equal contribution.).

[13] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, and A.W.M. Smeulders, "Selective search for object recognition," *International Journal on Computer Vision (IJCV)*, vol. 104, no. 2, pp. 154–171, February 2013.

[14] Paul Viola and Michael Jones, "Rapid object detection using a boosted cascade of simple features," in *CVPR*, 2001.

[15] Navneet Dalal and Bill Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.

[16] Pedro Felzenszwalb, Ross Girshick, David McAllester, and Deva Ramanan, "Object detection with discriminatively trained part based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 32, no. 9, pp. 1627–1645, September 2010.

[17] Hyungtae Lee, Vlad Morariu, and Larry Davis, "Qualitative pose estimation by discriminative deformable part models," in *ACCV*, 2012.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "ImageNet classification with deep convolutional neural networks," in *NIPS*, 2012.

[19] Sungmin Eum, Hyungtae Lee, and Heesung Kwon, "Going deeper with CNN in malicious crowd event classification," in *SPIE Defense+Commercial Sensing (DCS)*, 2018.

[20] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva, "Learning deep features for scene recognition using Places database," in *NIPS*, 2014.

[21] Hyungtae Lee, Heesung Kwon, Ryan M. Robinson, William D. Nothwang, and Amar R. Marathe, "Dynamic belief fusion for object detection," in *WACV*, 2016.