Contextual Action Recognition with R*CNN

Georgia Gkioxari
UC Berkeley
gkioxari@berkeley.edu

Ross Girshick
Microsoft Research
rbg@microsoft.com

Jitendra Malik
UC Berkeley
malik@berkeley.edu

Abstract

*There are multiple cues in an image which reveal what action a person is performing. For example, a jogger has a pose that is characteristic for jogging, but the scene (*e.g. *road, trail) and the presence of other joggers can be an additional source of information. In this work, we exploit the simple observation that actions are accompanied by contextual cues to build a strong action recognition system. We adapt RCNN to use more than one region for classification while still maintaining the ability to localize the action. We call our system R*CNN. The action-specific models and the feature maps are trained jointly, allowing for action specific representations to emerge. R*CNN achieves 90.2% mean AP on the PASAL VOC Action dataset, outperforming all other approaches in the field by a significant margin. Last, we show that R*CNN is not limited to action recognition. In particular, R*CNN can also be used to tackle fine-grained tasks such as attribute classification. We validate this claim by reporting state-of-the-art performance on the Berkeley Attributes of People dataset.*[1]

1. Introduction

Consider Figure 1 (a). How do we know that the person highlighted with the red box is working on a computer? Could it be that the computer is visible in the image, is it that the person in question has a very specific pose or is it that he is sitting in an office environment? Likewise, how do we know that the person in Figure 1 (b) is running? Is it the running-specific pose of her arms and legs or do the scene and the other people nearby also convey the action?

For the task of action recognition from still images, the pose of the person in question, the identity of the objects surrounding them and the way they interact with those objects and the scene are vital cues. In this work, our objective is to use all available cues to perform activity recognition.

Formally, we adapt the Region-based Convolutional Network method (RCNN) [11] to use more than one region when making a prediction. We call our method *R*CNN*. In R*CNN, we have a primary region that contains the person in question and a secondary region that automatically discovers contextual cues.

Figure 1. Examples of people performing actions.

How do we select the secondary region? In other words, how to we decide which region contains information about the action being performed? Inspired by multiple-instance learning (MIL) [31, 21] and Latent SVM [9], if I is an image and r is a region in I containing the target person, we define the score of action α as

$$\text{score}(\alpha; I, r) = \mathbf{w}_{\text{P}}^{\alpha} \cdot \boldsymbol{\phi}(r; I) + \max_{s \in R(r;I)} \mathbf{w}_{\text{S}}^{\alpha} \cdot \boldsymbol{\phi}(s; I), \quad (1)$$

where $\boldsymbol{\phi}(r; I)$ is a vector of features extracted from region r in I, while $\mathbf{w}_{\text{P}}^{\alpha}$ and $\mathbf{w}_{\text{S}}^{\alpha}$ are the primary and secondary weights for action α respectively. $R(r; I)$ defines the set of candidates for the secondary region. For example, $R(r; I)$ could be the set of regions in the proximity of r, or even the whole set of regions in I. Given scores for each action, we use a softmax to compute the probability that the person in r is performing action α:

$$P(\alpha | I, r) = \frac{\exp(\text{score}(\alpha; I, r))}{\sum_{\alpha' \in A} \exp(\text{score}(\alpha'; I, r))}. \quad (2)$$

The feature representation $\boldsymbol{\phi}(\cdot)$ and the weight vectors $\mathbf{w}_{\text{P}}^{\alpha}$ and $\mathbf{w}_{\text{S}}^{\alpha}$ in Eq. 1 are learned *jointly* for all actions

[1] Source code and models are available at https://github.com/gkioxari/RstarCNN

1550-5499/15 $31.00 © 2015 IEEE
DOI 10.1109/ICCV.2015.129

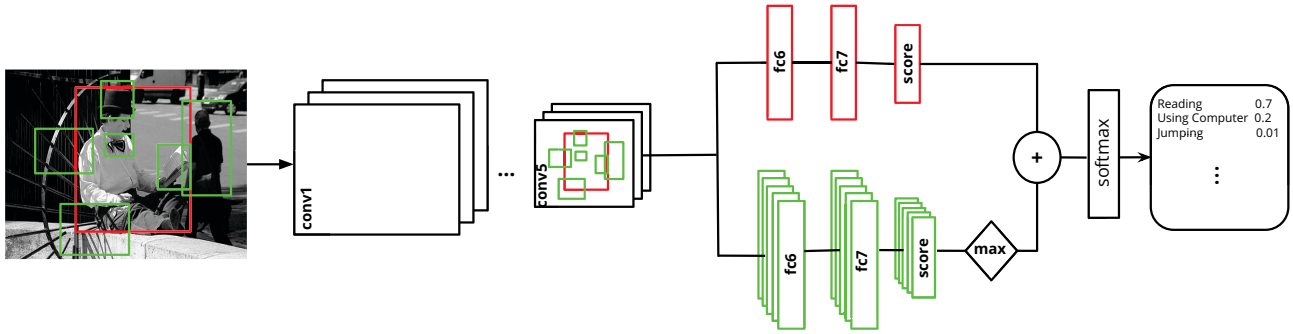

Figure 2. Schematic overview of our approach. Given image I, we select the primary region to be the bounding box containing the person (red box) while region proposals define the set of candidate secondary regions (green boxes). For each action α, the most informative secondary region is selected (*max* operation) and its score is added to the primary. The *softmax* operation transforms scores into probabilities and forms the final prediction.

$\alpha \in A$ using a CNN trained with stochastic gradient descent (SGD). We build on the Fast RCNN implementation [10], which efficiently processes a large number of regions per image. Figure 2 shows the architecture of our network.

We quantify the performance of R*CNN for action recognition using two datasets: PASCAL VOC Actions [7] and the MPII Human Pose dataset [2]. On PASCAL VOC, R*CNN yields 90.2% mean AP, improving the previous state-of-the-art approach [28] by 6 percentage points, according to the leaderboard [1]. We visualize the selected secondary regions in Figure 3 and show that indeed the secondary models learn to pick auxiliary cues as desired. On the larger MPII dataset, R*CNN yields 26.7% mean AP, compared to 5.5% mean AP achieved by the best performing approach, as reported by [25], which uses holistic [32] and pose-specific features along with motion cues.

In addition to the task of action recognition, we show that R*CNN can successfully be used for fine-grained tasks. We experiment with the task of attribute recognition and achieve state-of-the-art performance on the Berkeley Attributes of People dataset [4]. Our visualizations in Figure 8 show that the secondary regions capture the parts specific to the attribute class being considered.

2. Related Work

Action recognition. There is a variety of work in the field of action recognition in static images. The majority of the approaches use holistic cues, by extracting features on the person bounding box and combining them with contextual cues from the whole image and object models.

Maji *et al*. [20] train action specific poselets and for each instance create a poselet activation vector that is classified using SVMs. They capture contextual cues in two ways: they explicitly detect objects using pre-trained models for the *bicycle, motorbike, horse* and *tvmonitor* categories and exploit knowledge of actions of other people in the image. Hoai *et al*. [16] use body-part detectors and align them with respect to the parts of a similar instance, thus aligning their feature descriptors. They combine the part based features with object detection scores and train non-linear SVMs. Khosla *et al*. [33] densely sample image regions at arbitrary locations and scales with reference to the ground-truth region. They train a random forest classifier to discriminate between different actions. Prest *et al*. [26] learn human-object interactions using only action labels. They localize the action object by finding recurring patterns on images of actions and then capture their relative spatial relations. The aforementioned approaches are based on hand-engineered features such as HOG [5] and SIFT [19].

CNNs achieve state-of-the-art performance on handwritten digit classification [18], and have recently been applied to various tasks in computer vision such as image classification [17, 28] and object detection [11] with impressive results. For the task of action recognition, Oquab *et al*. [23] use a CNN on ground-truth boxes for the task of action classification, but observe a small gain in performance compared to previous methods. Hoai [15] uses a geometrical distribution of regions placed in the image and in the ground-truth box and weights their scores to make a single prediction, using fc7 features from a network trained on the ImageNet-1k dataset [6]. Gkioxari *et al*. [12] train body part detectors (*head, torso, legs*) on pool5 features in a sliding-window manner and combine them with the ground-truth box to jointly train a CNN.

Our work is different than the above mentioned approaches in the following ways. We use bottom up region proposals [30] as candidates for secondary regions, instead of anchoring regions of specific aspect ratios and at specific locations in the image, and without relying on the reference

provided by the ground-truth bounding box. Region proposals have been shown to be effective object candidates allowing for detection of objects irrespective of occlusion and viewpoint. We jointly learn the feature maps and the weights of the scoring models, allowing for action specific representations to emerge. These representations might refer to human-object relations, human-scene relations and human-human relations. This approach is contrary to work that predefines the relations to be captured or that makes use of hand-engineered features, or features from networks trained for different tasks. We allow the classifier to pick the most informative secondary region for the task at hand. As we show in Section 4, the selected secondary region is instance specific and can be an object (*e.g.*, cell phone), a part of the scene (*e.g.*, nearby bicycles), the whole scene, or part of the human body.

Scene and Context. The scene and its role in vision and perception have been studied for a long time. Biederman *et al*. [3] identify five classes of relationships (presence, position, size, support and interposition) between an object and its setting and conduct experiments to measure how well humans identify objects when those relationships are violated. They found that the ability to recognize objects is much weaker and it becomes worse as violations become more severe. More recently, Oliva and Torralba [22] study the contextual associations of objects with their scene and link various forms of context cues with computer vision.

Multiple-Instance Learning. Multiple instance learning (MIL) provides a framework for training models when full supervision is not available at train time. Instead of accurate annotations, the data forms bags, with a positive or a negative label [21]. There is a lot of work on MIL for computer vision tasks. For object detection, Viola *et al*. [31] use MIL and boosting to obtain face detectors when ground truth object face locations are not accurately provided at train time. More recently, Song *et al*. [29] use MIL to localize objects with binary image-level labels (is the object present in the image or not). For the task of image classification, Oquab *et al*. [24] modify the CNN architecture [17], which divides the image into equal sized regions and combines their scores via a final max pooling layer to classify the whole image. Fang *et al*. [8] follow a similar technique to localize concepts useful for image caption generation.

In this work, we treat the secondary region for each training example as an unknown latent variable. During training, each time an example is sampled, the forward pass of the CNN infers the current value of this latent variable through a max operation. This is analogous to latent parts locations and component models in DPM [9]. However, here we perform end-to-end optimization with an online algorithm (SGD), instead of optimizing a Latent SVM.

3. Implementation

Figure 2 shows the architecture of our network. Given an image I, we select the primary region to be the bounding box containing the person (knowledge of this box is given at test time in all action datasets). Bottom up region proposals form the set of candidate secondary regions. For each action α, the most informative region is selected through the *max* operation and its score is added to the primary (Eq. 1). The *softmax* operation transforms scores into estimated posterior probabilities (Eq. 2), which are used to predict action labels.

3.1. R*CNN

We build on Fast RCNN (FRCN) [10]. In FRCN, the input image is upsampled and passed through the convolutional layers. An adaptive max pooling layer takes as input the output of the last convolutional layer and a list of regions of interest (ROIs). It outputs a feature map of fixed size (*e.g.* 7×7 for the 16-layer CNN by [28]) specific to each ROI. The ROI-pooled features are subsequently passed through the fully connected layers to make the final prediction. This implementation is efficient, since the computationally intense convolutions are performed at an image-level and are subsequently being reused by the ROI-specific operations.

The test-time operation of FRCN is similar to SPPnet [14]. However, the training algorithm is different and enables fine-tuning all network layers, not just those above the final ROI pooling layer, as in [14]. This property is important for maximum classification accuracy with very deep networks.

In our implementation, we extend the FRCN pipeline. Each primary region r of an image I predicts a score for each action $\alpha \in A$ (top stream in Figure 2). At the same time, each region within the set of candidate secondary regions $R(r; I)$ independently makes a prediction. These scores are combined, for each primary region r, by a *max* operation over r's candidate regions (bottom stream in Figure 2).

We define the set of candidate secondary regions $R(r; I)$ as

$$R(r; I) = \{s \in S(I) : \text{overlap}(s, r) \in [l, u]\}, \quad (3)$$

where $S(I)$ is the set of region proposals for image I. In our experiments, we use Selective Search [30]. The lower and upper bounds for the overlap, which here is defined as the intersection over union between the boxes, defines the set of the regions that are considered as secondary for each primary region. For example, if $l = 0$ and $u = 1$ then $R(r; I) = S(I)$, for each r, meaning that all bottom up proposals are candidates for secondary regions.

3.2. Learning

We train R*CNN with stochastic gradient descent (SGD) using backpropagation. We adopt the 16-layer network architecture from [28], which has been shown to perform well for image classification and object detection.

During training, we minimize the log loss of the predictions. If $P(\alpha \mid I, r)$ is the softmax probability that action α is performed in region r in image I computed by Eq. 2, then the loss over a batch of training examples $B = \{I_i, r_i, l_i\}_{i=1}^{M}$ is given by

$$\text{loss}(B) = -\frac{1}{M}\sum_{i=1}^{M} \log P(\alpha = l_i \mid I_i, r_i), \qquad (4)$$

where l_i is the true label of example r_i in image I_i.

Rather than limiting training to the ground-truth person locations, we use all regions that overlap more than 0.5 with a ground-truth box. This condition serves as a form of data augmentation. For every primary region, we randomly select N regions from the set of candidate secondary regions. N is a function of the GPU memory limit (we use a Nvidia K40 GPU) and the batch size.

We fine-tune our network starting with a model trained on ImageNet-1K for the image classification task. We tie the weights of the fully connected primary and secondary layers (*fc6, fc7*), but not for the final scoring models. We set the learning rate to 0.0001, the batch size to 30 and consider 2 images per batch. We pick $N = 10$ and train for 10K iterations. Larger learning rates prevented fine-tuning from converging.

Due to the architecture of our network, most computation time is spent during the initial convolutions, which happen over the whole image. Computation does not scale much with the number of boxes, contrary to the original implementation of RCNN [11]. Training takes 1s per iteration, while testing takes 0.4s per image.

4. Results

We demonstrate the effectiveness of R*CNN on action recognition in static images. We use two datasets, PASCAL VOC Actions [7] and the MPII Human Pose dataset [2].

4.1. PASCAL VOC Action

The PASCAL VOC Action dataset consists of 10 different actions, *Jumping, Phoning, Playing Instrument, Reading, Riding Bike, Riding Horse, Running, Taking Photo, Using Computer, Walking* as well as examples of people not performing some of the above action, which are marked as *Other*. The ground-truth boxes containing the people are provided both at train and test time. During test time, for every example we estimate probabilities for all actions and compute AP.

4.1.1 Control Experiments

We experiment with variants of our system to show the effectiveness of R*CNN.

- **RCNN**. As a baseline approach we train Fast R-CNN for the task of action classification. This network exploits only the information provided from the primary region, which is defined as the ground-truth region.
- **Random-RCNN**. We use the ground-truth box as a primary region and a box randomly selected from the secondary regions. We train a network for this task similar to R*CNN with the *max* operation replaced by *rand*
- **Scene-RCNN**. We use the ground-truth box as the primary region and the whole image as the secondary. We jointly train a network for this task, similar to R*CNN, where the secondary model learns action specific weights solely from the scene (no *max* operation is performed in this case)
- **R*CNN** (l, u). We experiment with various combinations of values for the only free parameters of our pipeline, namely the bounds (l, u) of the overlaps used when defining the secondary regions $R(r; I)$, where r is the primary region
- **R*CNN** (l, u, n_S). In this setting, we use $n_S > 1$ secondary regions instead of one. The secondary regions are selected in a greedy manner. First we select the secondary region s_1 exactly as in R*CNN. The i-th secondary region s_i is selected via the *max* operation from the set $R(r; I) \cap R(s_1; I) \cap ... \cap R(s_{i-1}; I)$, where r is the primary region.

The Random- and Scene- settings show the value of selecting the most informative region, rather than forcing the secondary region to be the scene or a region selected at random.

Table 1 shows the performance of all the variants on the val set of the PASCAL VOC Actions. Our experiments show that R*CNN performs better across all categories. In particular, *Phoning*, *Reading*, *Taking Photo* perform significantly better than the baseline approach and Scene-RCNN. *Riding Bike*, *Riding Horse* and *Running* show the smallest improvement, probably due to scene bias of the images containing those actions. Another interesting observation is that our approach is not sensitive to the bounds of overlap (l, u). R*CNN is able to perform very well even for the unconstrained setting where all regions are allowed to be picked by the secondary model, $(l = 0, u = 1)$. In our basic R*CNN setting, we use one secondary region. However, one region might not be able to capture all the modes of contextual cues present in the image. Therefore,

we extend R*CNN to include n_S secondary regions. Our experiments show that for $n_S = 2$ the performance is the same as with R*CNN for the optimal set of parameters of $(l = 0.2, u = 0.75)$.

4.1.2 Comparison with published results

We compare R*CNN to other approaches on the PASCAL VOC Action test set. Table 2 shows the results. Oquab *et al*. [23] train an 8-layer network on ground-truth boxes. Gkioxari *et al*. [12] use part detectors for *head, torso, legs* and train a CNN on the part regions and the ground-truth box. Hoai [15] uses an 8-layer network to extract fc7 features from regions at multiple locations and scales inside the image and and the box and accumulates their scores to get the final prediction. Simonyan and Zisserman [28] combine a 16-layer and a 19-layer network and train SVMs on fc7 features from the image and the ground-truth box. R*CNN (with $(l = 0.2, u = 0.75)$) outperforms all other approaches by a substantial margin. R*CNN seems to be performing significantly better for actions which involve small objects and action-specific pose appearance, such as *Phoning*, *Reading*, *Taking Photo*, *Walking*.

4.1.3 Visualization of secondary regions

Figure 3 shows examples from the top predictions for each action on the test set. Each block corresponds to a different action. Red highlights the person to be classified while green the automatically selected secondary region. For actions *Jumping*, *Running* and *Walking* the secondary region is focused either on body parts (*e.g*. legs, arms) or on more instances surrounding the instance in question (*e.g*. joggers). For *Taking Photo*, *Phoning*, *Reading* and *Playing Instrument* the secondary region focuses almost exclusively on the object and its interaction with the arms. For *Riding Bike*, *Riding Horse* and *Using Computer* it focuses on the object, or the presence of similar instances and the scene.

Interestingly, the secondary region seems to be picking different cues depending on the instance in question. For example in the case of *Running*, the selected region might highlight the scene (*e.g*. road), parts of the human body (*e.g*. legs, arms) or a group of people performing the action, as shown in Figure 3.

Figure 4 shows erroneous predictions for each action on the val set (in descending score). Each block corresponds to a different action. The misclassified instance is shown in red and the corresponding secondary region with green. For *Riding Bike* and *Riding Horse*, which achieve a very high AP, the mistakes are of very low score. For *Jumping*, *Phoning* and *Using Computer* the mistakes occur due to confusions with instances of similar pose. In addition, for *Playing Instrument* most of the misclassifications are people performing in concert venues, such as singers. For *Taking Photo* and *Playing Instrument* the presence of the object seems to be causing most misclassifications. For *Running* and *Walking* they seem to often get confused with each other as well as with standing people (an action which is not present explicitly in the dataset).

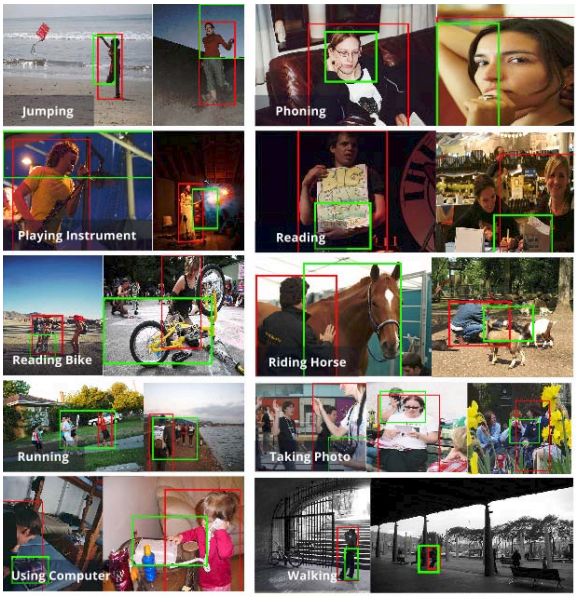

Figure 4. Top mistakes on the PASCAL VOC Action val set. The misclassified instance is shown in red, while the selected secondary region in green.

4.2. MPII Human Pose Dataset

The MPII Human Pose dataset contains 400 actions and consists of approximately 40,000 instances and 24,000 images. The images are extracted from videos from YouTube. The training set consists of 15,200 images and 22,900 instances performing 393 actions. The number of positive training examples per category varies drastically [25]. The amount of training data ranges from 3 to 476 instances, with an average of 60 positives per action. The annotations do not include a ground-truth bounding box explicitly, but provide a point (anywhere in the human body) and a rough scale of the human. This information can be used to extract a rough location of the instance, which is used as input in our algorithm.

4.2.1 R*CNN vs. RCNN

We split the training set into train and val sets. We make sure that frames of the same video belong to the same split to avoid overfitting. This results in 12,500 instances in train and 10,300 instances in val. We train the baseline RCNN network and R*CNN. We pick $(l = 0.2, u = 0.5)$ due to the large number of region proposals generated by [30] (on average 8,000 regions per image).

AP (%)	Jumping	Phoning	Playing Instrument	Reading	Riding Bike	Riding Horse	Running	Taking Photo	Using Computer	Walking	mAP
RCNN	88.7	72.6	92.6	74.0	96.1	96.9	86.1	83.3	87.0	71.5	84.9
Random-RCNN	89.1	72.7	92.9	74.4	96.1	97.2	85.0	84.2	87.5	70.4	85.0
Scene-RCNN	88.9	72.5	93.4	75.0	95.6	98.1	88.6	83.2	90.4	71.5	85.7
R*CNN (0.0, 0.5)	89.1	80.0	**95.6**	81.0	**97.3**	98.7	85.5	**85.6**	93.4	71.5	87.8
R*CNN (0.2, 0.5)	88.1	75.4	94.2	80.1	95.9	97.9	85.6	84.5	92.3	**71.6**	86.6
R*CNN (0.0, 1.0)	**89.2**	77.2	94.9	**83.7**	96.7	**98.6**	87.0	84.8	93.6	70.1	87.6
R*CNN (0.2, 0.75)	88.9	79.9	95.1	82.2	96.1	97.8	**87.9**	85.3	94.0	71.5	**87.9**
R*CNN (0.2, 0.75, 2)	87.7	**80.1**	94.8	81.1	95.5	97.2	87.0	84.7	**94.6**	70.1	87.3

Table 1. AP on the PASCAL VOC Action 2012 val set. *RCNN* is the baseline approach, with the ground-truth region being the primary region. *Random-RCNN* is a network trained with primary the ground-truth region and secondary a random region. *Scene-RCNN* is a network trained with primary the ground-truth region and secondary the whole image. *R*CNN* (l, u) is our system where l, u define the lower and upper bounds of the allowed overlap of the secondary region with the ground truth. *R*CNN* (l, u, n_S) is a variant in which n_S secondary regions are used, instead of one.

AP (%)	CNN layers	Jumping	Phoning	Playing Instrument	Reading	Riding Bike	Riding Horse	Running	Taking Photo	Using Computer	Walking	mAP
Oquab *et al.* [23]	8	74.8	46.0	75.6	45.3	93.5	95.0	86.5	49.3	66.7	69.5	70.2
Hoai [15]	8	82.3	52.9	84.3	53.6	95.6	96.1	89.7	60.4	76.0	72.9	76.3
Gkioxari *et al.* [12]	16	84.7	67.8	91.0	66.6	96.6	97.2	90.2	76.0	83.4	71.6	82.6
Simonyan & Zisserman [28]	16 & 19	89.3	71.3	**94.7**	71.3	**97.1**	98.2	90.2	73.3	88.5	66.4	84.0
R*CNN	16	**91.5**	**84.4**	93.6	**83.2**	96.9	**98.4**	**93.8**	**85.9**	**92.6**	**81.8**	**90.2**

Table 2. AP on the PASCAL VOC Action 2012 test set. Oquab *et al.* [23] train an 8-layer network on ground-truth boxes. Gkioxari *et al.* [12] use part detectors for *head, torso, legs* and train a CNN. Hoai [15] uses an 8-layer network to extract fc7 features from regions at multiple locations and scales. Simonyan and Zisserman [28] combine a 16-layer and a 19-layer network and train SVMs on fc7 features from the image and the ground-truth box. R*CNN (with $(l = 0.2, u = 0.75)$) outperforms all other approaches by a significant margin.

On the val set, RCNN achieves 16.5% mean AP while R*CNN achieves 21.7% mean AP, across all actions. Figure 5 shows the performance on MPII val for RCNN and R*CNN. On the **left**, we show a scatter plot of the AP for all actions as a function of their training size. On the **right**, we show the mean AP across actions belonging to one out of three categories, depending on their training size.

The performance reported in Figure 5 is instance-specific. Namely, each instance is evaluated. One could evaluate the performance at the frame-level (as done in [25]), *i.e.* classify the frame and not the instance. We can generate frame-level predictions by assigning for each action the maximum score across instances in the frame. That yields 18.2% mean AP for RCNN and 23% mean AP for R*CNN.

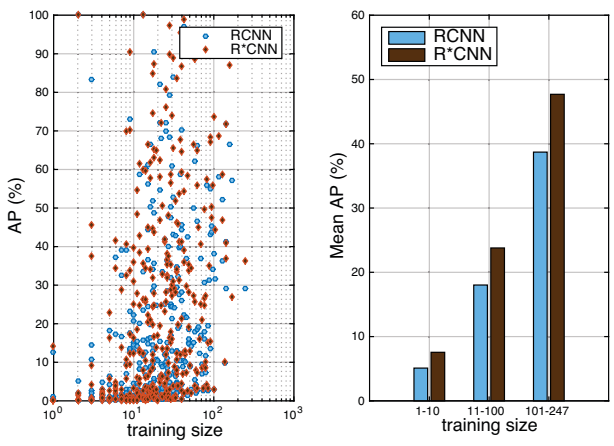

Figure 5. Performance on MPII val for RCNN (blue) and R*CNN (brown). **Left:** AP (%) for all actions as a function of their training size (x-axis). **Right:** Mean AP (%) for three discrete ranges of training size (x-axis).

4.2.2 Comparison with published results

In [25], various approaches for action recognition are reported on the test set. All the approaches mentioned use motion features, by using frames in the temporal neighborhood of the frame in question. The authors test variants of Dense Trajectories (DT) [32] which they combine with pose specific features. The best performance on the test set is 5.5% mean AP (frame-level) achieved by the DT combined with a pose specific approach.

We evaluate R*CNN on the test set[2] and achieve 26.7% mAP for frame-level recognition. Our approach does not use motion, which is a strong cue for action recognition in video, and yet manages to outperform DT by a significant margin. Evaluation on the test set is performed only at the frame-level.

Figure 6 shows the mean AP across actions in a descending order of training size. This figure allows for a direct comparison with the published results, as shown in Figure

[2]We sent our results to the authors of [25] for evaluation since test annotations are not publicly available.

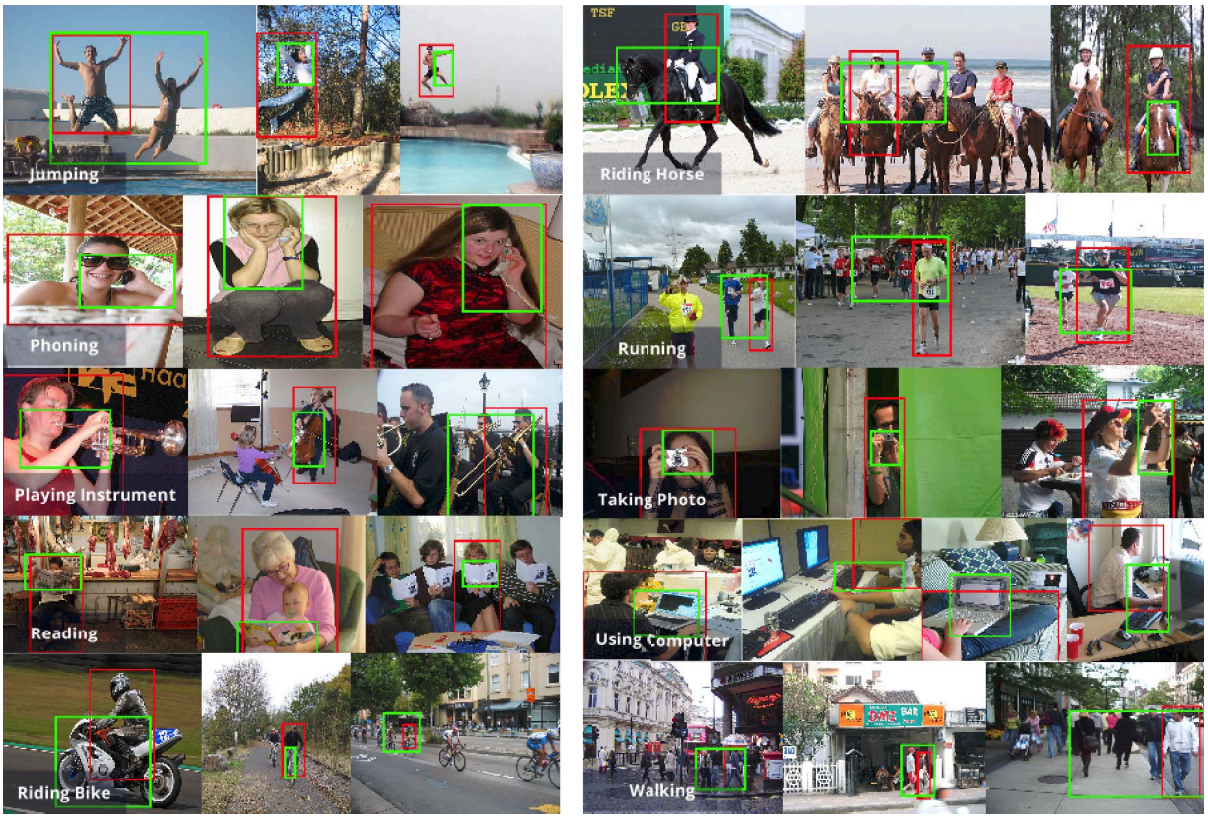

Figure 3. Top predictions on the PASCAL VOC Action test set. The instance in question is shown with a red box, while the selected secondary region with a green box. The nature of the secondary regions depends on the action and the image itself. Even within the same action category, the most informative cue can vary.

1(b) in [25].

Figure 7 shows some results on the test set. We highlight the instance in question with red, and the secondary box with green. The boxes for the instances were derived from the point annotations (some point on the person) and the rough scale provided at train and test time. The predicted action label is overlaid in each image.

Even though R*CNN outperforms DT, there is still need of movement to boost performance for many categories. For example, even though the MPII dataset has a many examples for actions such as *Yoga*, *Cooking or food preparation* and *Video exercise workout*, R*CNN performs badly on those categories (1.1% mean AP). We believe that a hybrid approach which combines image and motion features, similar to [27, 13], would perform even better.

4.3. Attribute Classification

Finally, we show that R*CNN can also be used for the task of attribute classification. On the Berkeley Attributes of People dataset [4], which consists of images of people and their attributes, *e.g. wears hat*, *is male* etc, we train R*CNN as described above. The only difference is that our loss is no longer a log loss over softmax probabilities, but the cross entropy over independent logistics because attribute prediction is a multi-label task. Table 3 reports the performance in AP of our approach, as well as other competing methods.

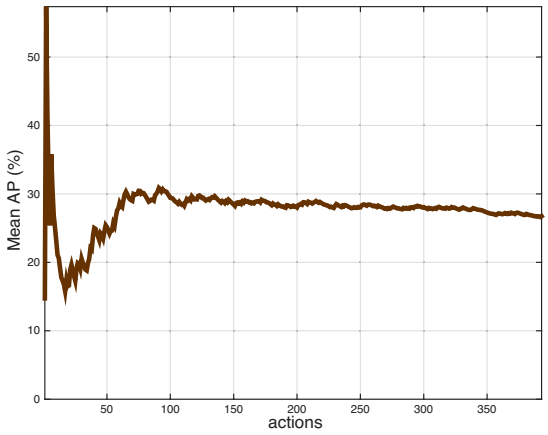

Figure 6. Mean AP (%) on MPII test for R*CNN across actions in descending order of their training size. A direct comparison with published results, as shown in Figure 1(b) in [25], can be drawn.

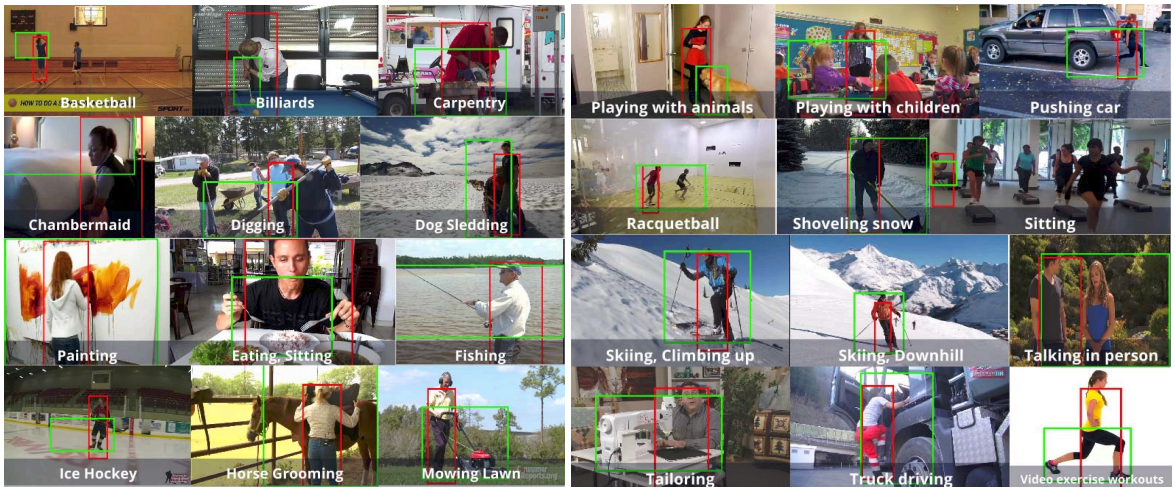

Figure 7. Predictions on the MPII test set. We highlight the person in question with a red box, and the secondary region with a green box. The predicted action label is overlaid.

AP (%)	CNN layers	Is Male	Has Long Hair	Has Glasses	Has Hat	Has T-Shirt	Has Long Sleeves	Has Shorts	Has Jeans	Has Long Pants	mAP
PANDA [34]	5	91.7	82.7	70.0	74.2	49.8	86.0	79.1	81.0	96.4	79.0
Gkioxari *et al*. [12]	16	**92.9**	**90.1**	77.7	**93.6**	72.6	**93.2**	**93.9**	**92.1**	**98.8**	**89.5**
RCNN	16	91.8	88.9	81.0	90.4	73.1	90.4	88.6	88.9	97.6	87.8
R*CNN	16	92.8	88.9	**82.4**	92.2	**74.8**	91.2	92.9	89.4	97.9	89.2

Table 3. AP on the Berkeley Attributes of People test set. PANDA [34] uses CNNs trained for each poselet type. Gkioxari *et al*. [12] detect parts and train a CNN jointly on the whole and the parts. RCNN is our baseline approach based on FRCN. Both RCNN and R*CNN do not use any additional part annotations at training time. [12] and R*CNN perform equally well, with the upside that R*CNN does not need use keypoint annotations during training.

Figure 8. Results on the Berkeley Attributes of People test set. We highlight the person in question with a red box, and the secondary region with a green box. The predicted attribute is overlaid.

Figure 8 shows results on the test set. From the visualizations, the secondary regions learn to focus on the parts that are specific to the attribute being considered. For example, for the *Has Long Sleeves* class, the secondary regions focus on the arms and torso of the instance in question, while for *Has Hat* focus is on the face of the person.

Conclusion

We introduce a simple yet effective approach for action recognition. We adapt RCNN to use more than one region in order to make a prediction, based on the simple observation that contextual cues are significant when deciding what action a person is performing. We call our system *R*CNN*. In our setting, both features and models are learnt jointly, allowing for action-specific representations to emerge. R*CNN outperforms all published approaches on two datasets. More interestingly, the auxiliary information selected by R*CNN for prediction captures different contextual modes depending on the instance in question.

R*CNN is not limited to action recognition. We show that R*CNN can be used successfully for tasks such as attribute classification. Our visualizations show that the secondary regions capture the region relevant to the attribute considered.

Acknowledgments

This work was supported by the Intel Visual Computing Center and the ONR SMARTS MURI N000140911051. The GPUs used in this research were generously donated by the NVIDIA Corporation.

References

[1] http://pascallin.ecs.soton.ac.uk/challenges/voc/voc2012/, 2012. 2

[2] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014. 2, 4

[3] I. Biederman, R. J. Mezzanotte, and J. C. Rabinowitz. Scene perception detecting and judging objects undergoing relational violations. *Cognitive Psychology*, 1982. 3

[4] L. Bourdev, S. Maji, and J. Malik. Describing people: Poselet-based attribute classification. In *ICCV*, 2011. 2, 7

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2

[6] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Competition 2012 (ILSVRC2012). http://www.image-net.org/challenges/LSVRC/2012/. 2

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 2010. 2, 4

[8] H. Fang, S. Gupta, F. N. Iandola, R. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt, C. L. Zitnick, and G. Zweig. From captions to visual concepts and back. In *Computer Vision and Pattern Recognition (CVPR)*, 2015. 3

[9] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *TPAMI*, 2010. 1, 3

[10] R. Girshick. Fast R-CNN. In *ICCV*, 2015. 2, 3

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1, 2, 4

[12] G. Gkioxari, R. Girshick, and J. Malik. Actions and attributes from wholes and parts. In *ICCV*, 2015. 2, 5, 6, 8

[13] G. Gkioxari and J. Malik. Finding action tubes. In *CVPR*, 2015. 7

[14] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014. 3

[15] M. Hoai. Regularized max pooling for image categorization. In *BMVC*, 2014. 2, 5, 6

[16] M. Hoai, L. Ladicky, and A. Zisserman. Action recognition from weak alignment of body parts. In *BMVC*, 2014. 2

[17] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 3

[18] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1989. 2

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 2

[20] S. Maji, L. Bourdev, and J. Malik. Action recognition from a distributed representation of pose and appearance. In *CVPR*, 2011. 2

[21] O. Maron and T. Lozano-Pérez. A framework for multiple instance learning. In *NIPS*, 1998. 1, 3

[22] A. Oliva and A. Torralba. The role of context in object recognition. *Trends in cognitive sciences*, 2007. 3

[23] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014. 2, 5, 6

[24] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Weakly supervised object recognition with convolutional neural networks. 2014. 3

[25] L. Pishchulin, M. Andriluka, and B. Schiele. Fine-grained activity recognition with holistic and pose based features. In *GCPR*, 2014. 2, 5, 6, 7

[26] A. Prest, C. Schmid, and V. Ferrari. Weakly supervised learning of interactions between humans and objects. *PAMI*, 2012. 2

[27] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 7

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 2, 3, 4, 5, 6

[29] H. O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, and T. Darrell. On learning to localize objects with minimal supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014. 3

[30] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *IJCV*, 2013. 2, 3, 5

[31] P. Viola, J. Platt, and C. Zhang. Multiple instance boosting for object detection. In *NIPS*, 2005. 1, 3

[32] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013. 2, 6

[33] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR*, 2011. 2

[34] N. Zhang, M. Paluri, M. Ranzato, T. Darrell, and L. Bourdev. PANDA: Pose aligned networks for deep attribute modeling. In *CVPR*, 2014. 8