Context-aware CNNs for person head detection

Tuan-Hung Vu* Anton Osokin† Ivan Laptev*

INRIA/ENS

Abstract

Person detection is a key problem for many computer vision tasks. While face detection has reached maturity, detecting people under a full variation of camera view-points, human poses, lighting conditions and occlusions is still a difficult challenge. In this work we focus on detecting human heads in natural scenes. Starting from the recent local R-CNN object detector, we extend it with two types of contextual cues. First, we leverage person-scene relations and propose a Global CNN model trained to predict positions and scales of heads directly from the full image. Second, we explicitly model pairwise relations among objects and train a Pairwise CNN model using a structured-output surrogate loss. The Local, Global and Pairwise models are combined into a joint CNN framework. To train and test our full model, we introduce a large dataset composed of $369,846$ *human heads annotated in* $224,740$ *movie frames. We evaluate our method and demonstrate improvements of person head detection against several recent baselines in three datasets. We also show improvements of the detection speed provided by our model.*

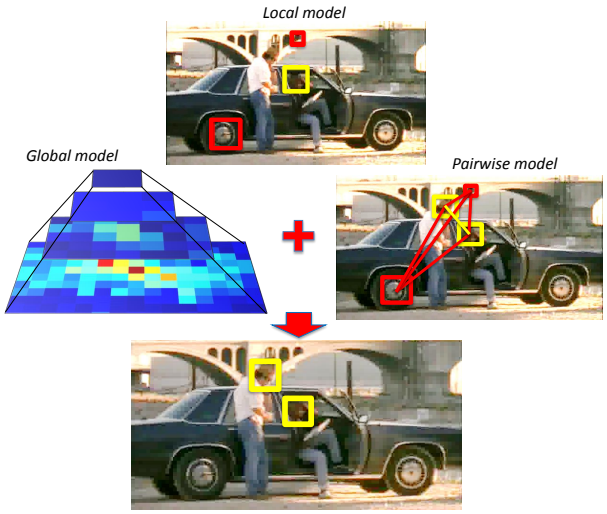

Figure 1. Results of head detection for a sample movie frame. The output of our method (bottom) is obtained from the combination of Local, Global and Pairwise CNN models. Bounding boxes illustrate detections: yellow – correct, red – false. Links between detections correspond to the pairwise potentials of our model: yellow – attractive, red – repulsive.

1. Introduction

Common images and videos primarily focus on people. Indeed, about 35% of pixels in movies and YouTube videos as well as about 25% of pixels in photographs belong to people [19]. This strong bias together with the growing amount of daily videos and photographs urge reliable methods for person analysis in visual data.

Person detection is a key component for many tasks including person identification, action recognition, age and gender recognition, autonomous driving, cloth recognition and many others. While face detection has reached maturity [22], the more general task of finding people in images and video still remains to be very challenging. For example, state-of-the-art object detectors [13] reach only 65% Average Precision for the person class on the Pascal VOC benchmark. Common difficulties arise from variations in human pose, background clutter, motion blur, low image resolution, occlusions and poor lighting conditions.

Recent advances in Convolutional Neural Networks (CNN) [20] have brought significant progress in image classification [18] and other vision tasks. In particular, CNN-based object detectors such as R-CNN [13] have shown large gains compared to previous models [12, 38]. Most of existing methods, however, treat objects independently and model appearance inside object bounding boxes only. Meanwhile, information available in the scene around objects [34] as well as relations among objects [8] are known to provide complementary contextual cues for recognition. Such cues are likely to be particularly helpful when object appearance lacks discriminative cues due to low image resolution, poor lighting and other factors.

In this work we build on the recent CNN model for object detection [13] and extend it to contextual reasoning. We particularly focus on person detection and aim to lo-

*WILLOW project-team, Département d'Informatique de l'Ecole Normale Supérieure, ENS/INRIA/CNRS UMR 8548, Paris, France

†SIERRA project-team, Département d'Informatique de l'Ecole Normale Supérieure, ENS/INRIA/CNRS UMR 8548, Paris, France

cate human heads on images coming from video data. The choice of heads is motivated by frequent occlusions of other body parts. When visible, however, other body parts and the rest of the scene constrain locations of heads in the image. Moreover, interactions between people put constraints on the relative positions and appearance of heads. We aim to leverage such constraints for detection by introducing the following two models.

First, we propose a *Global* CNN model which we train to predict coarse locations and scales of objects given the full low-resolution image on the input. In contrast to our base *Local* model limited to object appearance only, the Global model uses all pixels of the image for prediction. Interestingly, we find this simple model to provide quite accurate localization of heads across positions and scales of the image. Second, we introduce a *Pairwise* CNN model that explicitly models relations among pairs of objects. Motivated by Desai et al. [8], we build a joint score function for multiple object hypotheses in the image. This score function considers the relative positions, scales and appearance of heads. All parameters of the score function depend on the image data and are learned by optimizing a structured-output loss function. Our final joint model combines Local, Global and Pairwise CNN models (see Figure 1).

To train and test our model, we introduce a new large dataset with $369,846$ human heads annotated in $224,740$ video frames from 21 movies. We show the importance of our large dataset for training and evaluate our method on the new and two existing datasets. The results demonstrate improvements of the proposed contextual CNN model compared to other recent baselines including R-CNN [13] on all three datasets. We also demonstrate a speed-up of object detection provided by our Global model. Our new dataset and the code are publicly available from the project webpage [1].

The rest of the paper is organized as follows. We review related work in Section 2. Section 3 describes the parts of our contextual CNN model. Section 4 introduces datasets followed by the presentation of experimental results in Section 5. Section 6 concludes the paper.

2. Related works

The history of object detection with neural networks dates back to the 90s [37], but methods of these group have started to outperform others, e.g DPM [12], only after the seminal work of Krizhevsky et al. [18]. Szegedy et al. [31] and Sermanet et al. [29] applied CNN as a sliding window detector at multiple scales. The R-CNN model [13] is a combination of a CNN and a support vector machine (SVM) operating on object proposals generated by the selective search [36]. The pipeline of our Local model is similar to the one of R-CNN (see Section 3.1 for details).

The use of image context was proposed to support object detection in [34]. Contextual information can be modeled at a global scene level as well as at the level of object relations. For example, Murphy et al. [24] propose a CRF model for jointly solving the task of object detection and scene classification. Modolo et al. [23] uses context forest to predict object location and to speed-up object detection using global scene information. Erhan et al. [10] uses CNN to predict coordinates of object bounding boxes. Our Global CNN model predicts likely locations and scales of objects by producing a multi-scale heat map for the whole image.

Desai et al. [8] models spatial constellations of objects in the image and constructs an energy with unary and pairwise potentials. Unary potentials represent the confidence of object hypotheses based on the local image evidence, while pairwise potentials model spatial arrangement of objects in the image. Hoai and Zisserman [14] substitute the pairwise dependencies with a latent variable that represents the preferable configuration of object hypotheses. In both works [8, 14] binary potentials do not depend on the actual image data, moreover, unary potentials are trained independently of the joint model. Our Pairwise model exploits object context, i.e. builds a graphical model (an energy function) reasoning about multiple image locations jointly. Our approach is richer compared to [8] and [14] as it allows pairwise dependencies to be conditioned on the image data and we can train the base detector jointly with the graphical model on top of it.

Our Pairwise CNN model incorporates the structured-output loss. The idea of combining the structured-prediction objective with neural networks has been explored in [2, 20]. Recently Domke [9] and Chen et al. [4] use the dual message passing formulation of the inference task to construct a joint objective of the CNN parameters and the message-passing variables. This approach was applied to the small scale denoising and binary segmentation tasks in [9] and to the image tagging and word recognition tasks [4]. Jaderberg et al. [15] shows how to directly combine the structured SVM (SSVM) [32, 35] objective with the procedure of training a CNN for text recognition. CNNs with structured prediction have been recently explored for the task of human pose estimation. Chen and Yuille [5] propose a model with data-dependent pairwise potentials but the different parts of the model were trained separately. Tompson et al. [33] construct a specific NN that mimicked the behaviour of several rounds of a message-passing inference algorithm. Our Pairwise model is trained with an explicit structured-output surrogate loss with an external inference routine inside and enables to fine-tune all the parameters of the model jointly.

3. Context-aware CNN model

This section presents main components of our contextual CNN model. In Section 3.1, we describe our Local model building on R-CNN [13]. In Section 3.2, we introduce the Global CNN model trained to score object proposals

using the context of the full image. Section 3.3 describes our extension of CNNs with a structured-output loss function aimed to model pairwise relations between objects.

3.1. Local model

Our Local model follows R-CNN [13] and uses selective search proposals [38] to restrict the set of object hypotheses. We extend the bounding box of each proposal with a small margin to capture local image context around objects. The image patch corresponding to each proposal is then resized to fit the input layer of the CNN. As we are interested in head detection, we select bounding boxes with square-like aspect ratios $\mathcal{R} \in [2/3, 3/2]$ and refer to them as candidates.

The R-CNN model is based on the AlexNet architecture [18] pre-trained on the ImageNet dataset [7]. We have considered several alternatives including VGG-S [3], VGG-verydeep-16 [30] and Oquab et al. [25]. In our experiments VGG-S slightly outperformed AlexNet but was significantly slower in both training and testing. VGG-verydeep-16 showed better performance but was much slower. The network of Oquab et al. [25] had better accuracy and similar speed compared to AlexNet (see Section 5.3 for details). For experiments in this paper we use the pre-trained network of Oquab et al. [25] extended by one fully-connected layer (with 2048 nodes) initialized randomly and followed by ReLu and DropOut.

To train the network, we optimize parameters by minimizing the sum of independent log-losses using stochastic gradient descent (SGD) with momentum. Differently from R-CNN which deploys the second pass of training using SVM, we use the outputs of CNN to score candidates. We found this training procedure to work better for our problem compared to the standard R-CNN training. More details on our training procedure can be found in Appendix A.1.

3.2. Global model

Our Global model uses image-level information to reason about locations of objects in the image. The Global model is a CNN that takes the whole image as input and outputs a score for each cell of a multi-scale heat map. The input image is isotropically rescaled and zero-padded to fit the standard CNN input of 224×224 pixels. The output of the network is defined as a multi-scale grid of scores, corresponding to object hypotheses with coarsely discretized locations and scales in the image (see Figure 3). Object hypotheses form a grid of $\mathrm{C} = 284$ square cells of four sizes (28x28, 56x56, 112x112 and 224x224 pixels) and the stride corresponding to the 50% of cell size. Except the output layer, the architecture of the Global CNN is identical to our Local model described in Section 3.1.

The Global CNN is trained with SGD, minimizing the sum of C log-loss functions, one per each grid cell $c \in \{1 \cdots \mathrm{C}\}$,

$$\ell(\boldsymbol{f}_c(\mathbf{x}), y_c) = \sum_{y \in \{0,1\}} \log(1 + \exp\left((-1)^{y_c + y + 1} f_{c,y}(\mathbf{x})\right)), \tag{1}$$

where $\boldsymbol{f}_c(\mathbf{x}) \in \mathbb{R}^2$ is the output of the network for grid cell c of input image $\mathbf{x}$; $y_c \in \{0, 1\}$ is the label indicating the class of the grid cell c: *background* or *head*. We set the label of a grid cell to *head* if the Intersection-over-Union (IoU) overlap-ratio between the cell and any ground-truth bounding box in the image $\mathbf{x}$ is larger than 0.3, otherwise the label is set to *background*.

Due to the coarse resolution of grid cells, our Global model does not provide accurate localization. We therefore use the Global model to rescore the candidates of Local and Pairwise models. For this purpose, we match each candidate with the corresponding grid cell and compute affine combination of their scores. Each candidate is matched to a grid cell with the maximum IoU overlap-ratio. The parameters of affine score combination are optimized by cross validation on the validation set.

3.3. Pairwise model

In this section we describe our Pairwise model that aims to jointly reason about multiple object candidates. Following Desai et al. [8] we formulate the model as a joint score function where variables correspond to object candidates. In the prior work [8, 12, 14] unary potentials of the score function are defined by the response of the local object detector at corresponding locations, whereas higher-order potentials model spatial relations between candidates. Our Pairwise model enriches the model of [8] by making all potentials of the score function (2) dependent on the image data and, in contrast to [5], allows to perform the joint training of all parameters. We describe details of our model in Section 3.3.1.

We train parameters of our model by minimizing the structured surrogate loss using stochastic gradient descent algorithm. The details of our training procedure are presented in Section 3.3.2.

3.3.1 Model formulation

Score function. Consider a set of $\mathcal{V}$ candidate bounding boxes (nodes) extracted from an image. Let each bounding box have a binary variable y_i, $i \in \mathcal{V}$ assigned to it. We associate label 1 with the object class and label 0 with the background class. We assume that the ground-truth labels $\hat{y}_i$ are available for all candidates in training images.

For each pair of nodes we choose an order based on the coordinates of corresponding bounding boxes: the left box is defined to be the first, the right one – the second. Let $\mathcal{E}$ denote the set of oriented pairs of candidates (set of edges). We cluster all edges based on relative locations and

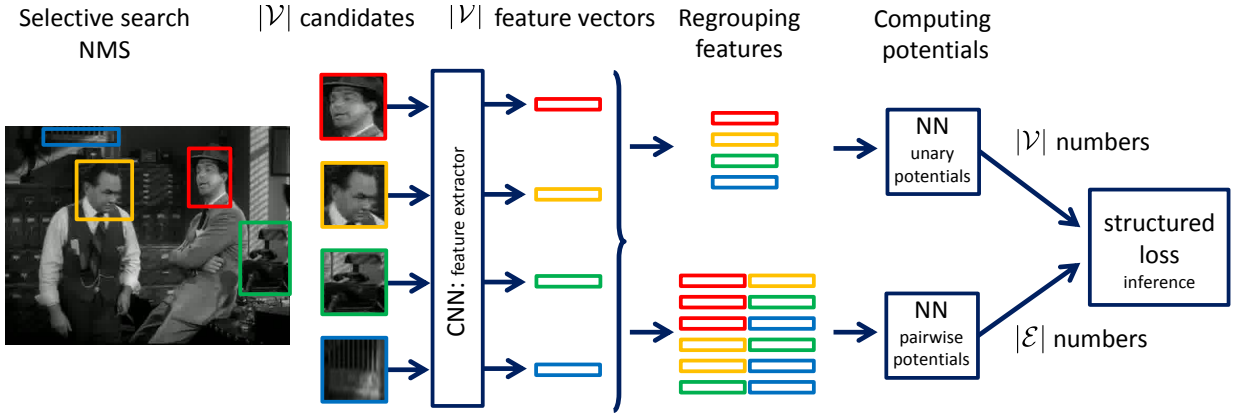

Figure 2. Pairwise model for the object detection task.

scales of bounding boxes[1] and denote the cluster index of edge $(i,j) \in \mathcal{E}$ by $k_{ij} \in \{1,\dots,K\}$.

Inspired by Desai et al. [8], we construct a joint score function $S(\boldsymbol{y};\boldsymbol{w})$ that ties together the labels of candidates in the same image:

$$S(\boldsymbol{y};\boldsymbol{w}) = \sum_{i\in\mathcal{V}} \theta_i^U(y_i;\boldsymbol{w}) + \sum_{(i,j)\in\mathcal{E}} \theta_{ij}^P(y_i,y_j,k_{ij};\boldsymbol{w}), \quad (2)$$

where $\boldsymbol{w}$ denotes trainable parameters, θ_i^U and θ_{ij}^P are unary and pairwise potentials depending on $\boldsymbol{w}$, and $\boldsymbol{y} = (y_i)_{i\in\mathcal{V}}$ is a vector of all binary variables.

Note, that different values of potentials in (2) can lead to exactly the same score function S. We rewrite Eq. (2) in the more compact form (the set of all representable functions of binary variables stays the same):

$$S(\boldsymbol{y};\boldsymbol{w}) = \sum_{i\in\mathcal{V}} y_i\, \theta_i^U(\boldsymbol{w}) + \sum_{(i,j)\in\mathcal{E}} y_i y_j\, \theta_{ij,k_{ij}}^P(\boldsymbol{w}) \quad (3)$$

where unary potentials θ_i^U and pairwise potentials $\theta_{ij,k_{ij}}^P$ are represented by real values.

Connecting the score function and the image. Now we connect the image with potentials of the score function (3) using several feed-forward neural networks. First, from the Local model described in Section 3.1 we create a feature extractor (FE), i.e. a function φ^E that constructs feature vector $\boldsymbol{f}_i$ for the image data $\boldsymbol{x}_i$ of candidate i: $\boldsymbol{f}_i = \varphi^E(\boldsymbol{x}_i,\boldsymbol{w}^E)$. Here $\boldsymbol{w}^E$ is a vector of trainable parameters of FE.

To connect features $\boldsymbol{f}_i$ with potentials in (3) we construct two additional feed-forward networks: the unary network (UN) and the pairwise network (PN). The unary network φ^U maps the feature vector $\boldsymbol{f}_i$ of a candidate i to the value of the corresponding unary potential, i.e. $\theta_i^U = \varphi^U(\boldsymbol{f}_i,\boldsymbol{w}^U)$. The pairwise network φ^P maps the concatenated feature vectors of its two candidates to a vector $\boldsymbol{\theta}_{ij}^P$ where the k-th component $\theta_{ij,k}^P$ corresponds to the one of K cluster indices, i.e. $\theta_{ij,k_{ij}}^P = \varphi_{k_{ij}}^P(\boldsymbol{f}_i,\boldsymbol{f}_j,\boldsymbol{w}^P)$. Vectors $\boldsymbol{w}^U$ and $\boldsymbol{w}^P$ are the trainable parameters of the UN and PN, correspondingly.

In our experiments we found the following architectures to work best. The FE was of the same structure as our Local model (based on the network of Oquab et al. [25]) leading to 2048 features. In both UN and PN we use just one fully-connected layer. Addition of more hidden layers did not improve results.

Precision-recall evaluation. Object detection methods are typically evaluated in terms of precision-recall (PR) and average precision (AP) values. To construct the precision-recall curve given the joint score (3), we follow the approach of Desai et al. [8]. For each candidate bounding box i, we compute an individual score $s_i(\boldsymbol{w})$ defined as the difference of the max-marginals of the joint score

$$s_i(\boldsymbol{w}) = \max_{\boldsymbol{y}:y_i=1} S(\boldsymbol{y};\boldsymbol{w}) - \max_{\boldsymbol{y}:y_i=0} S(\boldsymbol{y};\boldsymbol{w}). \quad (4)$$

The individual scores are used in the standard precision-recall evaluation pipeline [11].

When the number of candidates is small, i.e. $|\mathcal{V}| \le 20$, both maximization problems of (4) can be solved exactly using exhaustive search. When the number of candidates becomes larger, the exhaustive search becomes too slow. In this case one can use the cascade of QPBO [17] and TRW-S [16] methods to approximate s_i. Specifically, QPBO allows to quickly determine the optimal label for some candidates. On our dataset QPBO works surprisingly well, i.e., in many cases it is able to label all nodes. If some nodes are unlabeled by QPBO, one can apply the exhaustive search

[1] To cluster edges we apply k-means algorithm with $K = 20$ to a subset of oriented edges in training images. Edges in this subset connect object candidates with positive labels as well as any other candidates with high scores of the pre-trained Local model. For the clustering we use relative location features (horizontal and vertical displacements, ratio of sizes) converted to the log scale and normalized to have zero mean and unit standard deviation. Further details of the clustering are available in Appendix A.3.

when the number of unlabelled nodes is at most 20 and TRW-S otherwise.

We have tried using 16 and 32 candidates per image. The exact inference is tractable only in the first case. In this paper we use 16 candidates per image as the large number of candidates did not improve performance on our validation set.

3.3.2 Training the model

We train parameters of our model by minimizing a structured surrogate loss using the stochastic gradient descent algorithm[2]. The algorithm for parameter update consists of the following four steps:

1. Select the set of candidates by applying the non-maximum suppression [12] on top of the scores produced by the Local model.
2. Perform the forward pass through the model to compute potentials of the joint score function.
3. Perform the inference to compute the structured loss and its gradient (see below).
4. Back-propagate the gradient through the model.

We explain details of the algorithm below.

Structured surrogate loss. A structured loss is a function that maps the current values of parameters, image data $\boldsymbol{x} = (\boldsymbol{x}_i)_{i\in\mathcal{V}}$ and the ground-truth labeling $\hat{\boldsymbol{y}} = (\hat{y}_i)_{i\in\mathcal{V}}$ to a real number. A popular choice for the surrogate loss for structured-prediction tasks is the structured SVM (SSVM) objective [32, 35]:

$$\ell_{\text{SVM}}(\boldsymbol{w}, \hat{\boldsymbol{y}}, \boldsymbol{x}) = \max_{\boldsymbol{y}} \Big(S(\boldsymbol{y}; \boldsymbol{w}, \boldsymbol{x}) + h(\boldsymbol{y}, \hat{\boldsymbol{y}}) \Big) - S(\hat{\boldsymbol{y}}; \boldsymbol{w}, \boldsymbol{x}) \tag{5}$$

where $h(\boldsymbol{y}, \hat{\boldsymbol{y}}) \geq 0$ measures the agreement between the two labelings. Possible choices for h include the Hamming loss, the Hamming loss with penalties normalized by the frequency of classes, or higher-order losses making use of assumption that each ground-truth object is assigned to exactly one object candidate [26]. Notice, that in (5) the joint score S depends on parameters $\boldsymbol{w}$ and image data $\boldsymbol{x}$ implicitly through potentials θ^U and θ^P.

However, in our experiments we have observed that the SSVM loss is less suited for the detection task, i.e. optimizing the objective (5) does not lead to good results in terms of precision-recall measure. To tackle this problem, we propose a new surrogate loss which directly imposes penalties on the wrong values of individual scores (4) extracted from the joint score S. Specifically, this loss can be written as

$$\ell(\boldsymbol{w}, \hat{\boldsymbol{y}}, \boldsymbol{x}) = \sum_{i:\hat{y}_i=1} v(s_i(\boldsymbol{w}, \boldsymbol{x})) + \sum_{i:\hat{y}_i=0} v(-s_i(\boldsymbol{w}, \boldsymbol{x})) \tag{6}$$

[2] As common in the deep learning literature we ignore the non-differentiability issues and assume that in practice we can always compute the gradient.

where v can be any non-increasing function bounded from below. We use $v(t) = \log(1 + \exp(-t))$ which brings us closer to the training of conventional detector with a softmax loss.

Gradient of the structured loss. To optimize the structured loss w.r.t. the model parameters $\boldsymbol{w}$, we need to compute the gradient of the objective w.r.t. model parameters. We can always achieve this goal using the back-propagation method under two assumptions: 1) the gradient can be back-propagated through the modules of the model, i.e. all the partial derivatives of φ^E, φ^U, φ^P w.r.t. the input and the parameters can be computed; 2) the scores of the candidates (4) can be computed exactly.

To start the back-propagation procedure, we compute the gradient of structured loss w.r.t. potentials θ^U_i, $\theta^P_{ij,k}$ of the joint score function S. Jaderberg et al. [15] have in details explained how to do this for the SSVM loss (5). Here we explain how to differentiate the loss (6). First, the gradient of the loss (6) w.r.t. the scores can be expressed as

$$\frac{d\ell}{ds_i} = (-1)^{\hat{y}_i+1} v'(s_i(-1)^{\hat{y}_i+1}),\ v'(t) = \frac{-\exp(-t)}{1+\exp(-t)}.$$

The gradient of the score (when existent) w.r.t. potentials can be computed exactly if we can compute all max-marginals exactly:

$$\frac{ds_i}{d\theta^U_p} = y^{i,1}_p - y^{i,0}_p,\ \ \frac{ds_i}{d\theta^P_{pq,k}} = (y^{i,1}_p y^{i,1}_q - y^{i,0}_p y^{i,0}_q)[k_{ij} = k]$$

where $y^{i,t}_q$ is the q-th component of $\boldsymbol{y}^{i,t} = \operatorname*{argmax}_{\boldsymbol{y}:y_i=t} S(\boldsymbol{y}; \boldsymbol{w})$ for $t \in \{0, 1\}$. Here, $[\cdot]$ is the Iverson bracket notation. Combining the two derivatives via the chain rule we get

$$\frac{d\ell}{d\theta^U_p} = \sum_{i\in\mathcal{V}} \frac{d\ell}{ds_i}\frac{ds_i}{d\theta^U_p},\ \ \frac{d\ell}{d\theta^P_{pq,k}} = \sum_{i\in\mathcal{V}} \frac{d\ell}{ds_i}\frac{ds_i}{d\theta^P_{pq,k}}.$$

Back-propagation of the gradient. The next step of the back-propagation procedure is to compute the derivatives of the loss w.r.t. parameters of the UN and PN

$$\frac{d\ell}{d\boldsymbol{w}^U} = \sum_{i\in\mathcal{V}} \frac{d\ell}{d\theta^U_i}\frac{d\theta^U_i}{d\boldsymbol{w}^U},\ \ \frac{d\ell}{d\boldsymbol{w}^P} = \sum_{(i,j)\in\mathcal{E}} \sum_{k=1}^{K} \frac{d\ell}{d\theta^P_{ij,k}}\frac{d\theta^P_{ij,k}}{d\boldsymbol{w}^P} \tag{7}$$

and w.r.t. the output of the feature extractor

$$\frac{d\ell}{d\boldsymbol{f}_i} = \frac{d\ell}{d\theta^U_i}\frac{d\theta^U_i}{d\boldsymbol{f}_i} + \sum_{j:(i,j)\in\mathcal{E}} \frac{d\ell}{d\theta^P_{ij,k_{ij}}}\frac{d\theta^P_{ij,k_{ij}}}{d\boldsymbol{f}_i} + \sum_{j:(j,i)\in\mathcal{E}} \frac{d\ell}{d\theta^P_{ji,k_{ji}}}\frac{d\theta^P_{ji,k_{ji}}}{d\boldsymbol{f}_i}. \tag{8}$$

Notice that all the derivatives of potentials w.r.t. parameters and features can be computed by propagating the gradient through networks φ^U and φ^P. Finally, propagation of the gradient (8) through φ^E gives us the direction of the update for parameters $\boldsymbol{w}^E$ of the FE.

4. Datasets

In this section we present our new head detection dataset, HollywoodHeads (HH), and discuss two other datasets we use for evaluation: TVHI [27, 14] and Casablanca [28].

4.1. HollywoodHeads dataset

HollywoodHeads dataset contains $369,846$ human heads annotated in $224,740$ video frames from 21 Hollywood movies[3]. The movies vary in genres and represent different time epochs. To create annotation, we have manually annotated tracks of human heads in action-rich movie clips. For each head track, head bounding boxes, i.e., the smallest axis-parallel rectangles including all visible pixels of the head, were manually annotated on several key frames. The bounding boxes on remaining frames were linearly interpolated and manually verified to be correct. In total, we have collected $2,380$ clips with $3,872$ human tracks, spanning over 3.5 hours of video. The dataset is divided into the training, validation and test subsets which have no overlap in terms of movies[3]. Given the redundancy of consequent video frames, we have temporally subsampled videos in the validation and test subsets. In summary, the training set of HollywoodHeads contains $216,719$ frames from 15 movies, the validation set contains $6,719$ frames from 3 movies and the test set contains $1,302$ frames from another set of 3 movies. Human heads with poor visibility (e.g., strong occlusions, low lighting conditions) were marked by the "difficult" flag and were excluded from the evaluation. The HollywoodHeads dataset is available from [1].

4.2. TVHI dataset

The extended TV Human Interaction (TVHI) dataset [27, 14] consists of $1,313$ frames of TV show episodes annotated with bounding boxes of human upper bodies. Frames are split into the two sets: 599 for training and 714 for testing. To evaluate head detection using upper-body annotation, we have applied bounding-box regression to the output of head detectors [22]. The parameters of regression were tuned on the TVHI training subset for each tested method.

4.3. Casablanca dataset

The Casablanca dataset [28] contains $1,466$ frames from the movie "Casablanca". The frames are annotated with head bounding boxes, however, the annotation of frontal heads is typically reduced to face bounding boxes and, therefore differs in the scale and aspect ratio from the HollywoodHeads annotation. Given some mistakes in the original annotation of [28], we have added missing bounding boxes for heads of all people in the foreground. We have also applied bounding-box regression [22] to compensate for differences in annotation policies.

[3]List of movies used in HollywoodHeads dataset. Training set: *American beauty*, *As Good As It Gets*, *Big Fish*, *Big Lebowski*, *Bringing out the dead*, *Capote*, *Clerks*, *Crash*, *Dead Poets Society*, *Double Indemnity*, *Erin Brockovich*, *Fantastic 4*, *Fargo*, *Fear And Loathing In Las Vegas*, *Fight Club*. Validation set: *Five Easy Pieces*, *Forrest Gump*, *Gang Related*. Test set: *Gandhi*, *Charade*, *I Am Sam*.

Test set	Local	Local Global	Local Pairwise	Local Pairwise Global
Casablanca	71.8	72.1	72.5	**72.7**
HH	71.8	72.5	71.9	**72.7**
TVHI	87.8	89.5	89.2	**89.8**

Table 1. Performance (% AP) of different context-aware models on three datasets: Casablanca, HollywoodHeads (HH) and TVHI.

5. Experiments

This section presents our experimental results. First, we demonstrate the effect of different combinations of proposed models (Section 5.1) and provide the comparison with the state-of-the-art (Section 5.2). Section 5.3 compares different architectures of the Local model. We then justify the need of our new large dataset for training (Section 5.4) and show improvements in computational complexity that can be achieved with the Global model (Section 5.5).

To evaluate the detection performance, we use the standard Average Precision (AP) measure based on the Precision-Recall (PR) curve [11]. Detections having high overlap ratio with the ground truth (IoU > 0.5) are considered as true positives. Multiple detections assigned to the same ground truth are penalized and declared as false positives. Matches to "difficult" head annotations are ignored in the evaluation, i.e. such detections are considered neither as true positives nor as false positives.

5.1. Results of context-aware models

We compare performance of the following four models: the Local model (Sec .3.1), the combination of the Local and Global models (Section 3.2), the combination of the Local and Pairwise models (Section 3.3) and the combination of all the three proposed models. The performance of head detection is evaluated on HollywoodHeads, Casablanca and TVHI datasets. Qualitative results of the Global and Pairwise models are illustrated in Figures 3 and 4 respectively. Table 1 presents quantitative results for all models. We observe that the Global and Pairwise models consistently improve the performance of the baseline Local model. The combination of all three models demonstrates the best performance on all three datasets.

5.2. Comparison with the state-of-the-art methods

We compare our approach against several baselines: the CNN-based object detector [13] (R-CNN), DPM-based face

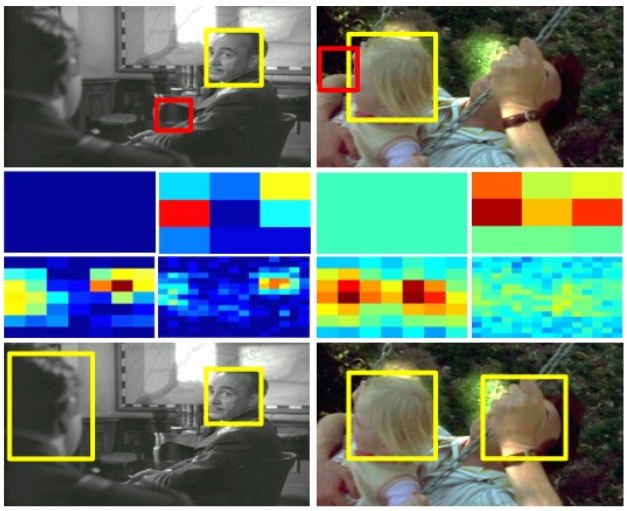

Figure 3. Qualitative results for the Global model. The top row shows detections produced by the Local model. The middle row illustrates the multi-scale score map produced by our Global model. Red color correspond to high score values for the "head" class, blue color – to low score values. The bottom row demonstrates detections by the combination of the Local and Global models.

detector [22] (DPM Face) as well as other methods reporting results on TVHI [14] (UBC+S) and Casablanca [28] (VJ-CRF). We have trained R-CNN[4] object detector on human heads using the training subset of HollywoodHeads dataset. The CNN model was first fine-tuned on all region proposals used to train our Local model. Given memory limitations, the SVM phase of R-CNN training was done on a subset of training images. For the DPM-based face detector we have used the vanilla DPM model provided by [22]. Results of other methods were taken from original papers.

Results of all compared methods are presented in Figure 5. Our joint model outperforms other methods on all three datasets. Consistently with other recent evaluations, we observe the advantage of CNN-based methods compared to other baselines. As expected, methods trained to detect faces achieve lower recall on the head detection task given the large variation of view points in natural images. Our method significantly outperforms R-CNN on two out of three datasets and performs slightly better than R-CNN on the TVHI dataset.

Note that our evaluation on the Casablanca dataset differs from [28] due to the improved annotation and the use of VOC evaluation procedure. Our results using the original evaluation setup by Ren [28] are reported in Appendix B. Additional results of our method are available from the project web-page [1] and in Appendix C.

5.3. Architectures of the Local model.

In this section we compare performance and speed of different architectures of the Local model. We consider

[4] https://github.com/rbgirshick/rcnn

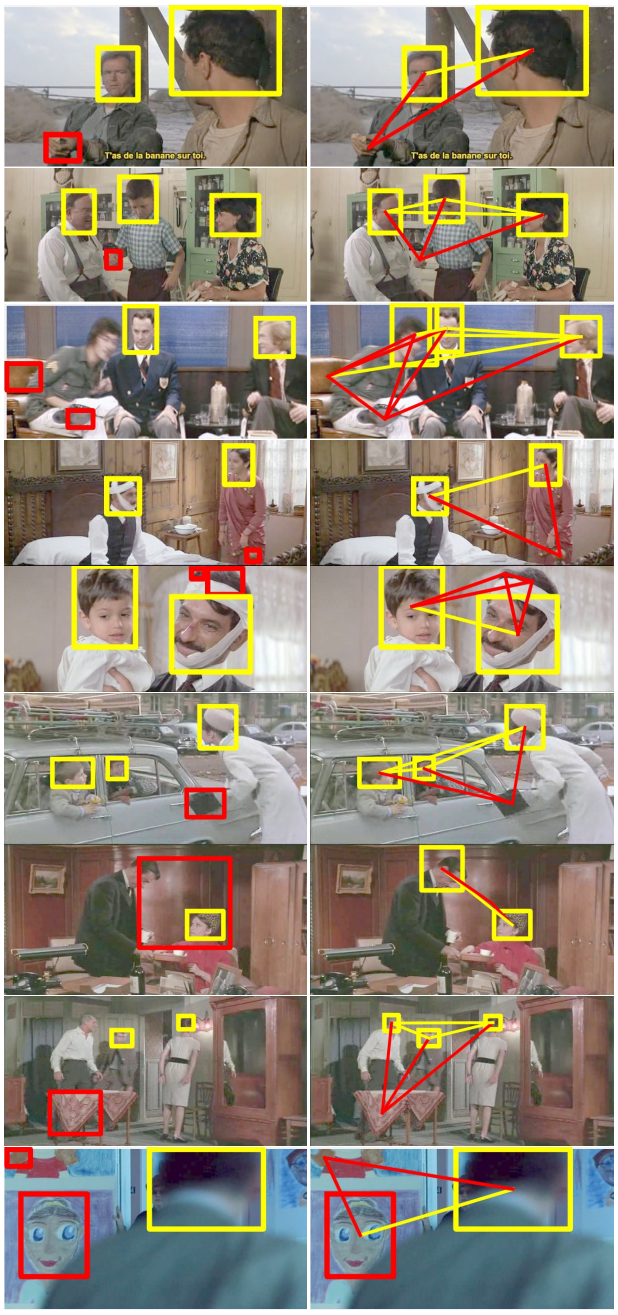

Figure 4. Qualitative results for the Pairwise model. For each video frame we show results of the Local model (left) and the Pairwise model (right). For both methods we choose the score thresholds such that the precision equals the recall on the validation set. The plotted bounding boxes show the detections with the scores above the selected thresholds. Yellow boxes correspond to correct detections, red – to false positives. For the Pairwise model we show the strength of links between the candidates detected by at least one method. Links above a strength threshold (attractive) are plotted yellow and others – red (repulsive).

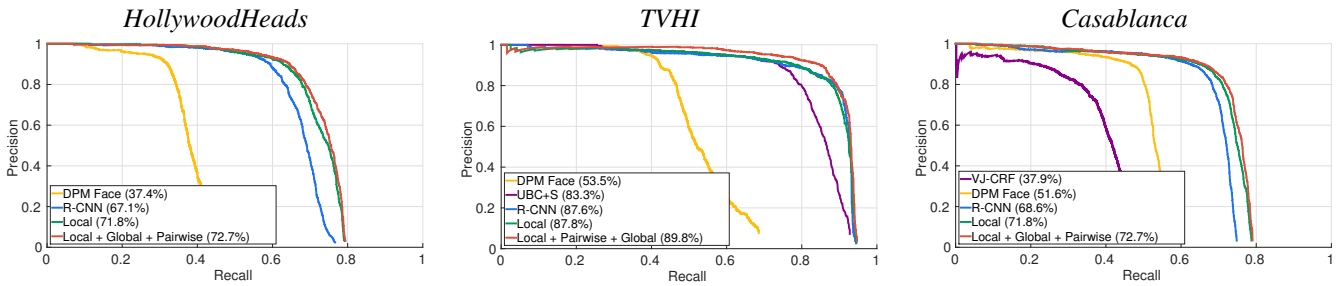

Figure 5. Results of our method compared to the state-of-the-art on the three datasets.

	AlexNet	Oquab	VGG-S	verydeep-16
AP	76.3	76.7	77.2	**78.5**
Train speed	**445**	284	147	30
Test speed	**1490**	980	510	74

Table 2. Performance (% AP) of Local models of different architectures on the HollywoodHeads validation set. Bottom lines report training and testing speed, measured by the number of image patches processed per second.

Test set	4 movies	8 movies	15 movies
Casablanca	51.2	62.5	**72.7**
HollywoodHeads	63.3	67.7	**72.7**
TVHI	88.6	88.8	**89.8**

Table 3. Performance of models trained on the training sets of different sizes. We report % AP for each test set.

AlexNet architecture [18], VGG-S [3], VGG-verydeep-16 [30] provided with the MatConvNet framework [39][5] and the model of Oquab et al. [25]. All models were pre-trained on the ImageNet dataset [7] and fine-tuned on the training set of the HollywoodHeads dataset as the Local model. In Table 2 we report values of AP produced by different models together with the train/test speed. We measure the speed as the number of image patches processed per second. For each model we choose the size of the training batch such that the training speed is maximal. In all cases it happens to be the maximum batch size that fits into the GPU memory. Experiments of this section were run on NVIDIA TITAN X with 12G RAM.

5.4. Size of the training set

In this experiment we analyze the amount of training data required to train our models. Our full training set is constructed from 15 movies. We also examine the use of smaller training sets corresponding to the first 8 movies and the first 4 movies of the full training set respectively. We use each training set to train parameters of our full model and evaluate it on three datasets. Corresponding results are reported in Table 3. We observe that the amount of the training data and, maybe more importantly, its diversity helps to improve the performance.

[5] http://www.vlfeat.org/matconvnet/pretrained/

% left	100	30	20	10	6	4	2
R-CNN	67.1	65.0	63.9	59.0	53.7	48.9	41.3
Local	71.8	68.3	66.8	60.2	53.4	48.8	41.9

Table 4. Performance of the R-CNN method and of our Local model (% AP) on the test set of HollywoodHeads with different percentage of candidates left after filtering using the Global model.

5.5. Complexity reduction with the Global model

Here we show that the Global model can suppress false candidates and reduce the computational complexity of R-CNN and our Local model at test time. We achieve this by transferring scores of the Global model detection proposals. We then filter out low-score candidates and thus reduce the number of candidates that have to pass through Local CNN. We evaluate the performance of detectors with different percentage of candidates left after the filtering. Table 4 presents results of this experiment. We observe that detection performance remains high despite aggressive filtering by the scores of the Global model.

6. Conclusion

In this work we have addressed the task of detecting people in still images. We proposed two context-aware CNN-based models. To train and evaluate our method, we have collected the new large-scale HollywoodHeads dataset consisting of movie frames and human head annotations. The combination of our context-aware models and the CNN-based local detector achieves state-of-the-art results on our dataset and the two existing human detection datasets, TVHI and Casablanca.

We believe that our context-aware models can be extended to tackle general object classes. In particular, the Microsoft COCO dataset [21] contains many small object classes with implied spatial constraints. Another possible direction for future work is to take into account motion information to extend our methods to perform long-term tracking.

Acknowledgements This work was partly supported by the ERC grant Activia (no. 307574) and the MSR-INRIA Joint Center.

A. Implementation details

A.1. Local model

To train the Local model, we assign each candidate region to the positive (head) or negative (background) class. For a given image, we make this assignment based on the intersection-over-union (IoU) overlap ratio o of the candidate bounding box with the best matching ground-truth bounding box. Specifically, candidates with $o > 0.6$ are labeled as positives and candidates with $o < 0.5$ are labeled as negatives. The remaining candidates are considered ambiguous and are not used at the training. Following [13] we exploit the context padding. Each candidate is resized to 188×188 square patch which is extended with 18 pixels on each side filled from the original image. The input images of our CNN are of size 224×224. For each image, we form a training batch by sampling 64 proposals such that the balance between classes is roughly maintained.

We initialize parameters of the network using the ImageNet pre-trained network of Oquab et al. [25]. We optimize the parameters of the network by minimizing the sum of independent log-losses with a stochastic gradient descent (SGD) algorithm with momentum 0.9 and weight decay 0.0005. We initialize the learning rate at 0.01, and decrease it several times by a factor of 10 after the validation error reaches saturation.

A.2. Global model

The Global model takes the whole image (isotropically rescaled and zero-padded to size 224×224) as the input and provides a vector of 284 numbers as the output. Each element of the output vector is associated with a cell of our multi-scale grid. For each cell we construct a target objective: 1 is the corresponding image patch has at least 0.3 IoU ratio with at least one ground-truth object bounding box. To train the Global model we optimize the sum of independent log-losses with an SGD algorithm. We initialize the model with the ImageNet pre-trained network [25]. The learning rate of SGD is set to 0.0001, momentum – to 0.9, weight decay – to 0.0005.

A.3. Pairwise model

The number of candidates from one image that our Pairwise model can process is quite limited due to the complexity of the inference procedure. To select the "good" candidates out of the thousands produced by the selective search [36] we use the non-maximum suppression (with threshold 0.3) on top of the scores provided by the Local model. We find that 16 candidates per image produced this way provide good balance between quality and speed.

To construct clusters of candidate pair (edges) incorporating the layout information we use the three features representing the vertical and horizontal displacements and the ratio of the candidate sizes. To be precise, if the position of each candidate is defined by a tuple (x_i, y_i, w_i, h_i) we define the size of the candidate as $s_i = (w_i + h_i)/2$, its horizontal position as $x_i^c = x_i + w_i/2$ and its vertical position as $y_i^c = y_i + h_i/2$. For the two candidates sorted such that $x_i \leq x_j$ we compute the features as follows: $f_{ij}^1 = \log(s_i/s2)$, $f_{ij}^2 = \varphi((x_j^c - x_i^c)/s_i)$, $f_{ij}^3 = \varphi((y_j^c - y_i^c)/s_i)$, where $\varphi(x) = \text{sign}(x)\log(|x| + 1)$. All the three features are normalized to have zero mean and unit standard deviation on the training set. We find that increasing number of clusters beyond 20 does not improve the performance.

To train the Pairwise model we assign each selected candidate a target binary label based on the maximum IoU ratio with the ground-truth bounding boxes (threshold 0.5). We form a training batch from 64 candidates coming from 4 different images. The FE part of the model is initialized from the Local model. The weights of the UN and PN were initialized randomly using zero-mean Gaussians with standard deviation 0.01. The structured surrogate objective is optimized with and SGD with momentum 0.9, weight decay 0.000005, and learning rate 0.00001. We decreased the learning rate by a factor of 10 after 4 passes over the training data.

A.4. Combining models

Local and Pairwise models. We now describe the process of computing the scores of the joint model. First, we compute the scores of the Local model for all candidates and perform the non-maximum suppression [12] using NMS threshold 0.3. The 16 top-scoring detections produced by NMS are then used as input for the Pairwise model. This number of candidates is sufficient on scenes with a few people, but can cause the drop of recall for crowded scenes (especially for some scenes of Casablanca dataset). To compensate for this drop, we combine scores produced by the Local and Pairwise models s_l,s_p respectively. For candidates with both scores existing, we use the affine combination $s_{lp} = \alpha s_l + (1 - \alpha)s_p + \beta$. For candidates with the score of the Pairwise model non-existent, we use the score of the Local model $s_{lp} = s_l$. Parameters $\alpha \in [0, 1]$ and $\beta \in [-10, 10]$ are selected by maximizing AP on the validation set using grid search.

Local, Pairwise and Global models. To combine scores s_{lp} with the Global model, we associate each candidate with the output cell of the Global model having maximum IoU overlap-ratio. The score of the joint model s^* is computed as an affine combination of the detection score s_{lp} and the grid cell score s_g, i.e. $s^* = \gamma s_{lp} + (1-\gamma)s_g$ where $\gamma \in [0, 1]$ is obtained by maximizing AP on the validation set.

A.5. Implementation details

All our experiments were run on NVIDIA GPUs using MATLAB-based MatConvNet [39] framework with the cuDNN backend [6]. To avoid speed bottlenecks, we found

it important to crop and resize image patches corresponding to object proposals using GPU which can be easily implemented using e.g. NVIDIA Performance Primitives (NPP) library provided in the CUDA package[6].

Running times. We report the running times of different parts of our model measured on NVIDIA TITAN X. The forward and backward passes of the Local model on a batch of 64 proposals take 0.08s and 0.18s, respectively. The forward and backward passes of the Global model on a batch of 32 images, take 0.06s and 0.12s, respectively. The Pairwise model consists of several parts: feature extractor, unary network, pairwise network, structured loss. For a batch of 64 candidates taken from 4 images (16 candidates from each) the forward pass through a feature extractor network takes 0.07s, the unary network – 0.003s, the pairwise network – 0.003s. The backward pass through these networks takes 0.2s, 0.004s and 0.004s, correspondingly. The computation of the structured loss and its derivatives takes 0.01s per image. Overall, the forward and backward passes through a joint Pairwise model take 0.36s for a batch coming from 4 images.

B. Evaluation on the original Casablanca dataset

To compare our results with the exact results reported in [28], we evaluate head detection on the Casablanca dataset using the original set of annotations and the evaluation procedure used in [28]. Figure 6 demonstrates corresponding precision-recall curves. Our method significantly outperforms VJ-CRF [28] as well as other baselines.

As mentioned in Section 4.3, the original Casablanca dataset [28] contains many cases of missing and imprecisely localized head annotations. Figure 7 (left) depicts some examples with missing annotations. To provide more conclusive results in Section 5, we have improved original annotation by adding missing and correcting existing annotations on all test frames defined in [28], see Figure 7 (right). Despite our effort, some crowded scenes may still contain missing annotations of very small heads.

C. Qualitative results

C.1. Global model

In this section we illustrate multi-scale grids of scores produced by the Global model (see Section 3.2). Each output consists of 1×1, 3×3, 7×7 and 15×15 score grids corresponding to grids of cells with 28×28, 56×56, 112×112 and 224×224 pixels. Figures 8 illustrates the output of the Global model for a few test examples. Note high responses at positions and scales corresponding to human heads in the image.

[6] https://github.com/aosokin/cropRectanglesMex

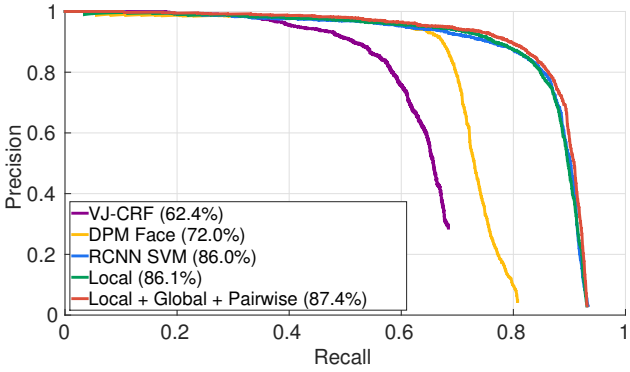

Figure 6. Results of head detection in the original Casablanca dataset.

Figure 7. (Left): Examples of original annotations in Casablanca dataset [28]. (Right): Our corrected annotation with added head bounding boxes marked with cyan.

C.2. Pairwise model

In Figure 9 we provide a few qualitative results of our Pairwise model. The bounding boxes and the links in this figure have the same meaning as the ones in Figure 4 of the main paper. We use the same thresholds for the links and the candidates as in Figure 4 of the main paper.

References

[1] "http://www.di.ens.fr/willow/research/headdetection." 2, 6, 7

[2] L. Bottou, Y. LeCun, and Y. Bengio, "Global training of document processing systems using graph transformer networks," in *CVPR*, 1997. 2

[3] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *BMVC*, 2014. 3, 8

[4] L. Chen, A. G. Schwing, A. L. Yuille, and R. Urtasun, "Learning deep structured models," in *ICML*, 2015. 2

[5] X. Chen and A. L. Yuille, "Articulated pose estimation with image-dependent preference on pairwise relations," in *NIPS*, 2014. 2, 3

[6] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, "cuDNN: Efficient primitives for deep learning," arXiv, Tech. Rep. 1410.0759, 2014. 9

[7] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009. 3, 8

[8] C. Desai, D. Ramanan, and C. C. Fowlkes, "Discriminative models for multi-class object layout," *IJCV*, vol. 95, no. 11, pp. 1–12, 2011. 1, 2, 3, 4

[9] J. Domke, "Structured learning via logistic regression," in *NIPS*, 2013. 2

[10] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *CVPR*, 2014. 2

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010. 4, 6

[12] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE TPAMI*, vol. 32, no. 9, pp. 1627–1645, 2010. 1, 2, 3, 5, 9

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014. 1, 2, 3, 6, 9

[14] M. Hoai and A. Zisserman, "Talking heads: Detecting humans and recognizing their interactions," in *CVPR*, 2014. 2, 3, 6, 7

[15] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep structured output learning for unconstrained text recognition," in *ICLR*, 2015. 2, 5

[16] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE TPAMI*, vol. 28, no. 10, pp. 1568–1583, 2006. 4

[17] V. Kolmogorov and C. Rother, "Minimizing non-submodular functions with graph cuts – a review," *IEEE TPAMI*, vol. 29, no. 7, pp. 1274–1279, 2007. 4

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *NIPS*, 2012. 1, 2, 3, 8

[19] I. Laptev, "Modeling and visual recognition of human actions and interactions," Habilitation à diriger des recherches en informatique, École normale superieure, Paris, France, July 2013. 1

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient based learning applied to document recognition," *Proceedings of IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. 1, 2

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *ECCV*, 2014, pp. 740–755. 8

[22] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles," in *ECCV*, 2014, pp. 720–735. 1, 6, 7

[23] D. Modolo, A. Vezhnevets, and V. Ferrari, "Context forest for efficient object detection with large mixture models," arXiv, Tech. Rep. 1503.00787, 2015. 2

[24] K. Murphy, A. Torralba, and W. Freeman, "Using the forest to see the trees: a graphical model relating features, objects and scenes," in *NIPS*, vol. 16, 2003, pp. 1499–1506. 2

[25] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014. 3, 4, 8, 9

[26] A. Osokin and P. Kohli, "Perceptually inspired layout-aware losses for image segmentation," in *ECCV*, 2014. 5

[27] A. Patron-Perez, M. Marszalek, I. Reid, and A. Zisserman, "Structured learning of human interactions in tv shows," *IEEE TPAMI*, vol. 34, no. 12, pp. 2441–2453, 2012. 6

[28] X. Ren, "Finding people in archive films through tracking," in *CVPR*, 2008, pp. 1–8. 6, 7, 10

[29] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *ICLR*, 2014. 2

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015. 3, 8

[31] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *NIPS*, 2013. 2

[32] B. Taskar, C. Guestrin, and D. Koller, "Max-margin markov networks," in *NIPS*, 2003. 2, 5

[33] J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," in *NIPS*, 2014. 2

[34] A. Torralba, "Contextual priming for object detection," *IJCV*, vol. 53, no. 2, pp. 169–191, 2003. 1, 2

[35] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research (JMLR)*, vol. 6, pp. 1453–1484, 2005. 2, 5

[36] 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, 9

[37] R. Vaillant, C. Monrocq, and Y. LeCun, "An original approach for the localization of objects in images," in *International Conference on Artificial Neural Networks (ICANN)*, 1993. 2

[38] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders, "Segmentation as selective search for object recognition," in *ICCV*, 2011. 1, 3

[39] A. Vedaldi and K. Lenc, "MatConvNet – convolutional neural networks for MATLAB," in *Proceeding of the ACM Int. Conf. on Multimedia*. 8, 9

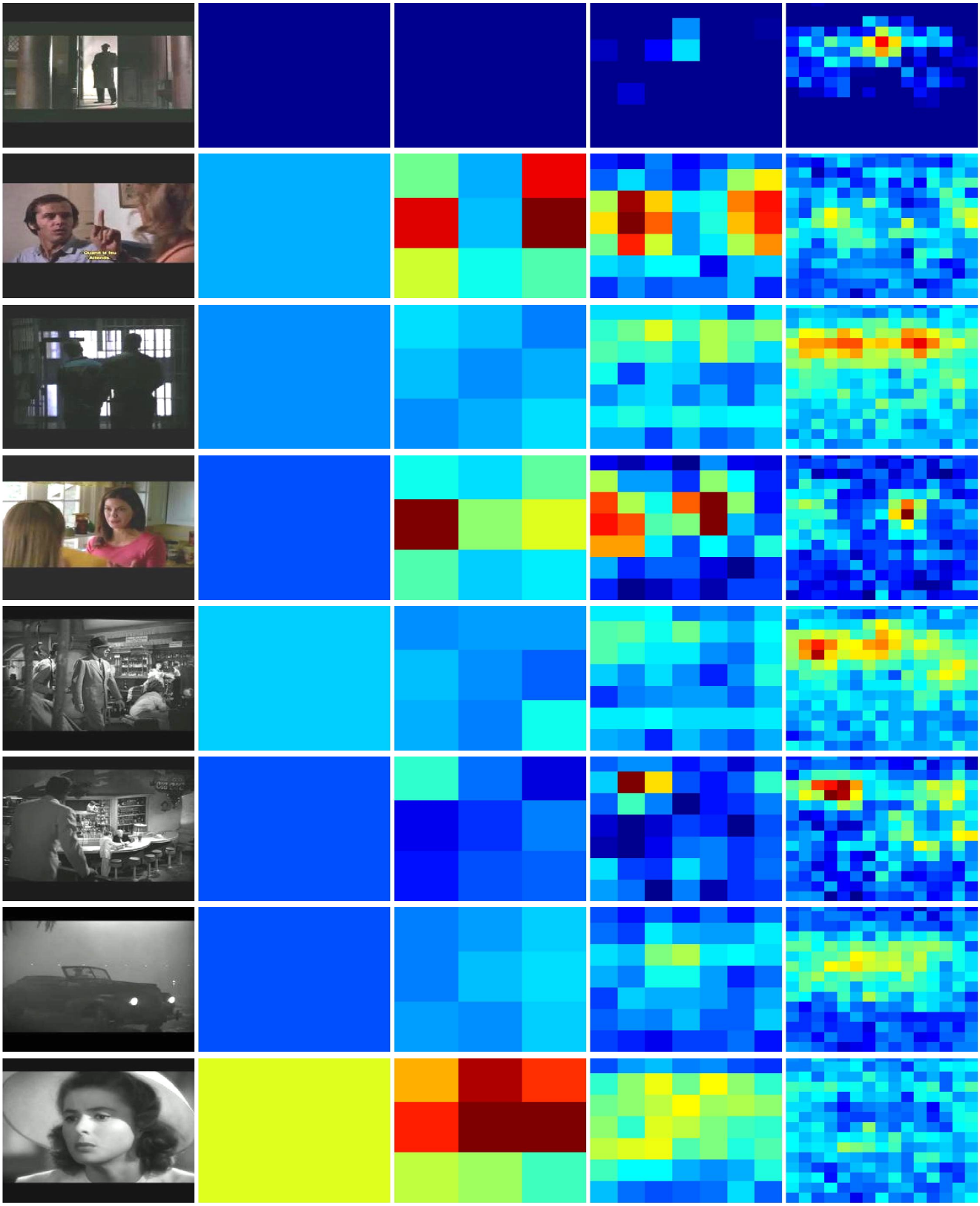

Figure 8. Qualitative results for our Global model. Columns from the left to the right correspond to original images and the output of the Global model at different resolutions. Red color corresponds to cells with high scores for the “head” class, blue color indicates cells with low scores for the “head” class.

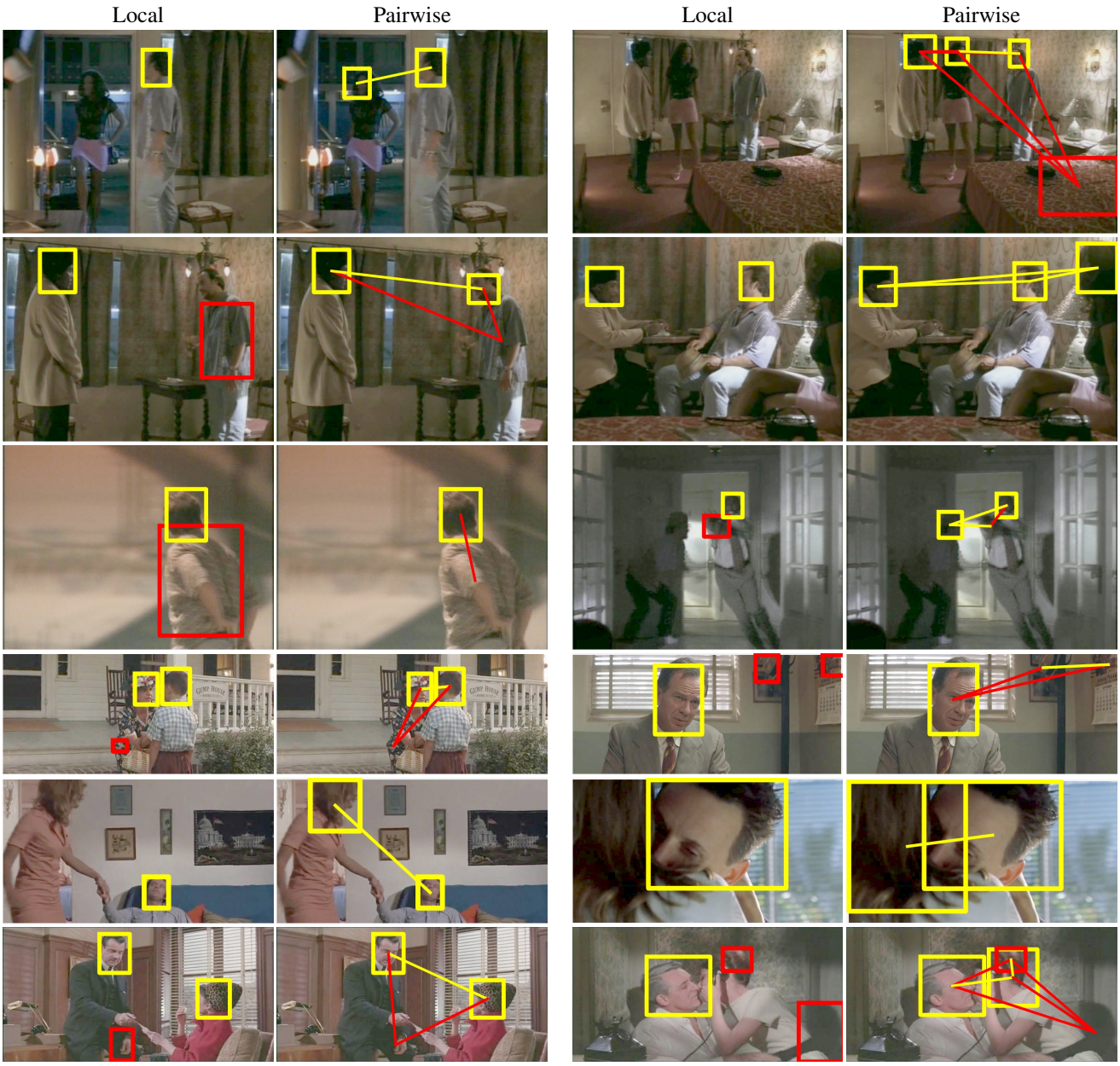

Figure 9. Qualitative results of our Pairwise model. For each frame we show the result of the local detector (left) and the result of the Pairwise model (right). For both methods we choose the threshold in such a way that precision equals recall on the validation set. We show only boxes with scores above the fixed threshold. The yellow bounding boxes correspond to correct detections, red – to false positive detections. For the Pairwise model we show links between candidates detected by the Local, Pairwise or both models. Links above a fixed strength threshold (attractive) are plotted with yellow, other (repulsive) links are marked by red.