Using k-poselets for detecting people and localizing their keypoints

Georgia Gkioxari*, Bharath Hariharan*, Ross Girshick and Jitendra Malik
University of California, Berkeley - Berkeley, CA 94720
{gkioxari,bharath2,rbg,malik}@berkeley.edu

Abstract

A k-poselet is a deformable part model (DPM) with k parts, where each of the parts is a poselet, aligned to a specific configuration of keypoints based on ground-truth annotations. A separate template is used to learn the appearance of each part. The parts are allowed to move with respect to each other with a deformation cost that is learned at training time. This model is richer than both the traditional version of poselets and DPMs. It enables a unified approach to person detection and keypoint prediction which, barring contemporaneous approaches based on CNN features [14], achieves state-of-the-art keypoint prediction while maintaining competitive detection performance.

1. Introduction

Our goal in this paper is to develop an effective computational approach to object recognition that supports detection and keypoint localization in a common framework. We will show that this can be accomplished using a representation based on *k-poselets*. These are a proper generalization of poselets (Bourdev & Malik [5]) in the same way that in natural language processing bigrams and trigrams are a generalization of unigrams. This generalization is based on learned spatial relations between parts, as in the deformable part models of Felzenszwalb *et al*. [10].

We start with some background. In the sliding window paradigm for object detection, the deformable part model (DPM) is the dominant method today, thanks to its consistently impressive showing on the PASCAL VOC detection challenge. Instead of training a single, monolithic HOG detector [7], these models have a mixture of components, and each component contains "parts" that can translate relative to the detection window at some deformation cost. The flexibility introduced through these mechanisms enables the detector to cope with changes in appearance due to intra-category variation as well as pose. Both the part appearance templates (which are HOG filters) and the deformation costs are jointly and discriminatively trained in the latent SVM formalism. The parts in the DPM framework are discovered in an unsupervised manner, which is sufficient for bounding box prediction, but does not lead to consistent information for predicting keypoints. So for the stated objective of our paper—a common framework for object and keypoint detection—DPMs fall short. Indeed, given that DPM training does not make use of keypoint annotations, it would be surprising if it could do a good job of keypoint localization.

An alternative approach that does exploit keypoint annotations is based on poselets, where one uses similarity in keypoint configurations to mine for semantically meaningful discriminative parts. Random windows are sampled from objects, and the keypoint configuration in each sampled window is used to get similar windows. This list of similar windows defines a training set for a single HOG-based classifier, a *poselet*. A poselet's training set is associated with a consistent keypoint configuration (unlike in DPM training), so by design when there is a strong activation of a poselet in a test image, one can make considerably tighter predictions that are useful for analyzing attributes [4], segmenting objects [6], recognizing actions [19], and locating keypoints [17]. Results on various benchmark tasks suggest that poselets are comparable to DPMs for object detection (in spite of the extra supervision), though they perform better than DPMs for more fine-grained tasks such as keypoint prediction. This suggests that there must be room for improvement in the poselet-based recognition pipeline.

One obvious location of sub-optimality in the poselet pipeline is that each poselet is trained independently of the others. Figure 1 illustrates this point. The images show examples of two different poselets indicated by blue and red rectangles respectively. If we have a training set of positive examples corresponding to a visual pattern that is distinctive, and more or less rigid, like a front view of a face (blue rectangles), then this approach works reasonably well. However, suppose the training set of examples corresponds to a part of a human arm (red rectangles), then the associated poselet will perform poorly. The local visual pattern is just a pair of parallel lines, and we can find them all over

* Authors contributed equally

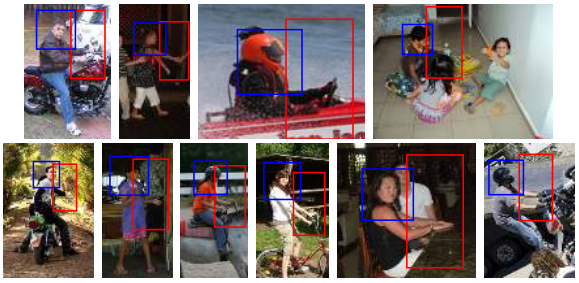

Figure 1: Example of two different poselets shown in blue and red, respectively. The blue (face) poselet would train well as a single HOG template. The red (arm) poselet would fail to train due to the non discriminative signal contained in it. Instead, using a 2-poselet will train both templates jointly and also learn the deformation needed to capture the pose.

the background in natural images. But note that while the arm poselet cannot be trained well as a single HOG template, if we searched for it in the context of the face poselet, the false positives would be considerably reduced. It is also worth observing that the arm poselet is not in a fixed configuration with respect to the face poselet. If the relationship was rigid we could train a monolithic HOG template that includes both the face and arm, but the relative shifts between the face and arm preclude that.

This motivates the design of k-poselets. One can think of a k-poselet as a DPM with k parts, where each of the parts is aligned to a specific configuration of keypoints based on ground-truth annotations. A separate HOG template will be used to model the appearance model of each part (k HOG templates total). The parts can move with respect to each other with a deformation cost that is learned jointly with the HOG templates. So this model is richer (if $k > 1$) than traditional poselets because we now have more tolerance to deformation than if we were using a single HOG template. It is also richer than a single DPM component with multiple parts because the extra supervisory information will enable a k-poselet to predict keypoint locations. Our parts are aligned using keypoint annotations as done in the poselet philosophy. The example in Figure 1 is a 2-poselet.

In this paper we will develop a k-poselet based framework for person detection and keypoint prediction. As in the standard poselet pipeline [3] we will train an ensemble of k-poselet detectors. At test time, we cluster the activation of these detectors to generate people hypotheses. We tie the activation of these different detectors using the human torso as a common reference—each detector can produce its best guess of the torso of the associated person, and we can cluster all such activations whose torso predictions are "close enough". Given a person hypothesis (a cluster of k-poselet activations), we can make multiple inferences: an overall confidence score for the detection, a bounding box that should encompass the visible extent of the person, and a detailed keypoint pose estimate.

The experimental part of the paper seeks to compare three models: DPMs, traditional poselets, and k-poselets. This is a delicate task as both DPMs and poselets exist in a "tweaked" form, and these tweaks can easily contribute several percentage points in performance on benchmarks (*e.g.*, mean AP of DPM went up from 32.3 to 33.7 from release 4 to release 5). In the case of poselets there is a fair amount of complexity and the training code remains unavailable. These factors have hindered the widespread adoption of poselets as a tool for recognition.

We made two major changes to the poselet training procedure with respect to [3]. Instead of the Dalal-Triggs version of HOG features used there, we use the HOG implementation from the DPM software release [10], thus making the baseline template detector exactly the same. Secondly, [3] does context-specific rescoring of each poselet score (from q to Q in their terminology). This is an $O(n^2)$ step that adds considerably to the computational complexity at runtime, for a relatively small gain, and therefore we avoid doing it. There are also a few minor changes inspired by [9], which both clean up and simplify the training procedure. Code is available at `http://www.cs.berkeley.edu/~bharath2/kposelets.html`.

The paper is organized as follows. Section 2 describes related work on person detection and keypoint prediction. In Section 3 we describe how we collect and train k-poselets. Section 4 describes how we use k-poselet detections to create and score person hypotheses. In Section 5 we state our results.

2. Related work

Person detection and keypoint localization are two tasks that have progressed significantly the last few years. The most relevant techniques from object detection, namely poselets [3] and DPM [10], were discussed in Section 1. In a related twist on DPMs, Azizpour and Laptev [2] propose to train them with manually specified part annotations, but do not show results on person detection or pose estimation. Until recently, techniques based on bottom-up region proposals [23] have not been competitive on the person category. However, in contemporaneous work, Girshick *et al*. [14] show large gains in person detection performance by using CNNs to classify region proposals.

The task of keypoint localization has a long history of research. Fischler and Elschlager [13] introduced pictorial structure models (PSM) in 1973. Felzenszwalb and Huttenlocher [11] later rephrased PSM under a probabilistic framework and gave an efficient matching algorithm. In PSM, the body parts are represented as boxes and their spatial relations are captured by tree-structured graphs. Ramanan [21], Andriluka *et al*. [1] and Eichner *et al*. [8] exploited appearance cues to parse the human body. Johnson and Everingham [18] replaced a single PSM with a mixture

of PSMs in order to capture more variety in pose. Yang and Ramanan [25] improved on PSM mixtures by using a mixture of templates for each part. Sapp *et al*. [22] enhanced the appearance models, including contour and segmentation cues. Wang *et al*. [24] used a hierarchy of poselets for human parsing. Pishchulin *et al*. [20] used PSM with poselets as parts for the task of human body pose estimation. Gkioxari *et al*. [17] trained arm specific detectors and show state-of-the-art results on the PASCAL VOC dataset for arm pose prediction, but did not propose an approach for whole-body pose estimation.

3. Collection and training of k-poselets

3.1. Collecting k-poselets

Our algorithm for collecting k-poselets follows [3]. As in [3], we first produce seed configurations by random sampling, and then pick other windows from the training set with similar keypoint configurations.

To sample a seed configuration for a k-poselet, we sample a training instance and sample k windows on the instance. For each window, we pick an aspect ratio from a predefined set, and place the window at a random position on the instance. We reject windows if their area is less than 50% within the instance bounding box, or if they contain fewer than four keypoints. To ensure each poselet in a k-poselet is different from the others, we discard windows that overlap highly with previously selected windows.

For each seed configuration, we pick similar instances from the training set. As in [3], for every instance we place the k windows on the instance so as to minimize the least squares error between the keypoints in the instance and the corresponding seed window. A high residual error indicates poor alignment and we reject instances for which *any* of the k windows yields an error higher than a threshold. This ensures that each of the k windows are well aligned in pose space, a modeling goal we share with poselets.

In addition, we reject instances for which the relative positions and scales of the k windows are very different from the relative locations and scales in the seed. This is critical since it ensures that the deformations of the k windows are unimodal, a modeling assumption we inherit from DPM.

The list of instances passing these tests forms the "positive list" for that k-poselet. If the cardinality of the positive list is small (less than 40), then that k-poselet is discarded. Figures 2 and 3 show the positive lists of two example 2-poselets and 1-poselets, respectively.

3.2. Training k-poselets

Detection models. Each k-poselet is described by a weight vector

$$w = \big(M_0, ..., M_{k-1}, d_1, ..., d_{k-1}, b\big), \qquad (1)$$

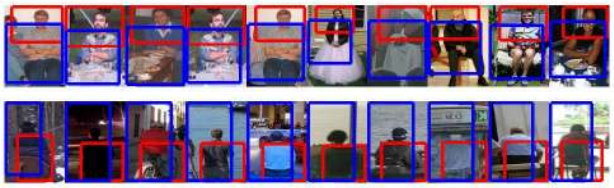

Figure 2: Each row shows examples collected for a k-poselet ($k = 2$), whose seed patch is shown in the first column. Red depicts examples of the first poselet, while blue examples of the second poselet.

Figure 3: Each row shows examples collected for a k-poselet ($k = 1$), whose seed patch is shown in the first column.

where M_i is the appearance template and d_i is the spatial deformation model for the i-th poselet of a k-poselet and b is a bias.

As in DPM, we fix the relative scales at which the k templates will be applied, but they can move in space relative to an anchor. We use u_i to denote the fixed scale at which the i-th poselet in a k-poselet occurs relative to the first poselet. Similarly we use v_i to denote the anchor location of the i-th poselet relative to the first poselet. A particular placement of the k-poselet can then be written as the tuple $l = (p_0, \ldots, p_{k-1}, \sigma_0)$, with the first poselet at scale σ_0, and poselet locations $p_i = (x_i, y_i)$ and scales $\sigma_i = \sigma_0 \cdot u_i$.

The score of a k-poselet placement can be decomposed into the scores of the individual templates at their particular locations, the deformation cost, and the bias. For a hypothesis $l = (p_0, ..., p_{k-1}, \sigma_0)$, the score is given by

$$\text{score}(l) = \sum_{i=0}^{k-1} M_i \cdot F(p_i) - \sum_{i=1}^{k-1} d_i \cdot f_d(\delta x_i, \delta y_i) + b, \quad (2)$$

where $F(p)$ are appearance features extracted at location p,

$$(\delta x_i, \delta y_i) = \big(\frac{x_i - x_0}{\sigma_0}, \frac{y_i - y_0}{\sigma_0}\big) - v_i \qquad (3)$$

is the scale-normalized spatial displacement from the anchor v_i, and

$$f_d(\delta x_i, \delta y_i) = (\delta x_i, \delta y_i, \delta x_i^2, \delta y_i^2) \qquad (4)$$

are deformation features. For appearance features, we use the HOG implementation from [10] to allow direct comparison with DPMs.

We compute $\{u_i\}$ and $\{v_i\}$ from the positive list collected for the k-poselet. We take u_i to be the mean scale and v_i to be the mean scale-normalized location (relative to the first poselet) at which the i-th poselet occurs.

At detection time, we apply a k-poselet exactly like a DPM, scoring each root poselet location by maximizing Eq. 2 over the locations of the non-root poselets.

It remains to specify the set of positive and negative examples to train a k-poselet. We explained earlier the construction of the aligned positive example lists. Note that since the k windows are specified for each instance in the positive list, we have no latent variables and training is a convex problem similar to a linear SVM, unlike standard weakly-supervised DPM training. Negatives are derived from image patches that don't contain people, and we follow standard hard negative mining [10]. In order to ensure that the deformation costs are convex functions, we use a positivity constraint on all quadratic deformation weights.

Keypoint models. In the standard poselet framework, keypoint predictions are derived from the mean position of the keypoints relative to the poselet location and scale on the training data. The twist here is that we have multiple poselets within a single k-poselet and their abilities to localize a given keypoint might vary.

Denote by $\mu_J^{(i)}$ the mean offset of keypoint J from the position of the i-th poselet, in units of the scale of the i-th poselet. We can compute these from a k-poselet's positive list. The i-th poselet in a k-poselet votes for each keypoint with a learned weight $\alpha_J^{(i)}$. We then combine the predictions:

$$\mathbf{x}_J = \sum_{i=0}^{k-1} \alpha_J^{(i)} (\mu_J^{(i)} \cdot \sigma_i + p_i) \tag{5}$$

where $\{p_i, \sigma_i\}$ are the locations and scales of each constituent poselet in a k-poselet detection hypothesis.

The set of weights $\{\alpha_J^{(i)}\}$ are learnt by optimizing a linear regression problem. For a k-poselet and keypoint J, we minimize the sum of squared prediction errors across the examples in $\mathcal{S}$, the positive list for this k-poselet,

$$\min_{\alpha_J} \sum_{s \in \mathcal{S}} \|\mathbf{x}_J^{(s)*} - \mathbf{x}_J^{(s)}\|^2 \tag{6}$$

subject to $\sum_{i=0}^{k-1} \alpha_J^{(i)} = 1$ and $\alpha_J^{(i)} \geq 0$, where $\mathbf{x}_J^{(s)}$ is the prediction of keypoint J for instance s (Eq. 5) and $\mathbf{x}_J^{(s)*}$ is the ground-truth location of J in positive example s.

Calibration of k-poselet scores. During detection, a k-poselet scores a location l according to Eq. 2. Since different k-poselets are trained independent of each other, their scores are uncalibrated. We calibrate them by mapping the score s to the precision of that k-poselet at score s. Precision at score s is an estimate of the probability that an activation with score greater than s is true.

For each detector we compute and store a precision-recall curve on a validation set. An activation is marked as true if its predicted torso has a significant overlap with a ground-truth torso, and false otherwise. Then for a score s we can simply look up the precision at that score.

Model selection. Randomness in k-poselet sampling necessitates the collection of a large number of them in order to cover the training instances with narrowly-tuned detectors. Sampling a large number of k-poselets also means that the pool will contain redundancy. Therefore, we want to select a small number of k-poselets such that each instance is correctly detected by at least one of them at a high-precision score. We use Average Max Precision (AMP) [9] to measure whether a set $\mathcal{C}$ of k-poselets achieves high precision and high coverage:

$$\text{AMP}(\mathcal{C}) = \frac{1}{N} \sum_{n=1}^{N} \max_{c \in \mathcal{C}} \text{prec}_c(s_{n,c}), \tag{7}$$

where N is the total number of instances, $\text{prec}_c(s)$ is the precision of detector c at threshold s and $s_{n,c}$ is the maximum true positive score of detector c on instance n (or $-\infty$ if no true detection exists). The precision-recall curves computed for calibration are reused for AMP computation. We use greedy forward selection to approximately maximize Eq. 7, starting from an empty $\mathcal{C}$, until a fixed number of k-poselet detectors has been selected. In our experiments, we select 200 1- and 2-poselets (the number 200 is also the number of poselets used by [3]). We find that our procedure picks roughly thrice as many bigrams as unigrams (153 2-poselets and 47 1-poselets). In order to make comparisons with traditional poselets [3], we also do this selection on *only* the 1-poselets, again selecting 200.

4. Producing person hypotheses

4.1. Clustering

Given k-poselet activations in an image, we cluster the activations into person hypotheses. Similar to [3], we use agglomerative clustering. This requires us to define a similarity between activations, as well as a similarity between *clusters* of activations.

Because the k-poselets are discriminatively trained to be good predictors of the torso, we use intersection over union of the predicted torso bounds as a measure of the similarity between two activations.

For the similarity between clusters of activations, typically one uses the maximum similarity between two activations in the clusters ("single link") or the minimum similarity ("complete link"). However both of these would treat each activation in the cluster on an equal footing, discarding the activation scores. This is undesirable since higher

Figure 4: A visualization of the clusters on an image. For each cluster, we also show the predicted torso keypoints from all the activations in the cluster. The keypoints are color coded: magenta: right shoulder, yellow: left shoulder, green: right hip, dark blue: left hip. The torso predictions of all the activations of a cluster agree with each other. Even so, the clustering is strict enough to create two separate clusters for the woman and the child.

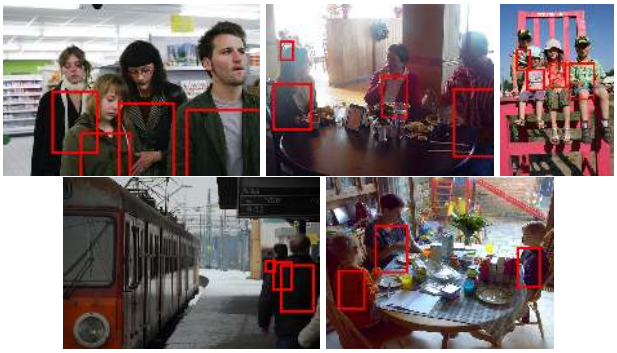

Figure 5: Visualizing recall after clustering. Each image shows the predicted torsos from the clusters that overlap highly with some ground truth instance. The clustering algorithm keeps highly overlapping people separate. The predicted torso is also a lot more invariant to pose variations, validating its use in clustering.

scoring activations are likely to make more reliable predictions. Therefore we follow a different strategy: we take the highest scoring activation in each cluster (the "leader") and measure the similarity between two clusters by the similarity between their leaders. Leaders are determined by comparing the calibrated, precision-mapped scores described in the previous section.

Figure 4 visualizes some of the clusters obtained in an image. For each cluster, we also show the predicted torso, and the predicted torso keypoints from all the activations in the cluster. As is clear from the figure, our clustering is able to resolve highly overlapping people, such as the woman and her child. Figure 5 shows that our clustering algorithm maintains recall, even when people are highly overlapping.

4.2. Scoring person hypotheses

For each cluster of activations we can also produce a score. We train a linear SVM to classify clusters as true or false detections using labels derived from ground-truth torso overlap. We use as features:

1. The precision-mapped score of the highest scoring activation of each k-poselet type.
2. The fraction of the highest scoring activation of each k-poselet type that is inside the image. For 2-poselets, we use the minimum of the two fractions. This allows us to disregard activations that are mostly outside the image.
3. The area of the highest scoring activation of each k-poselet type. We include this feature because very large or very small detections might be false positives.

5. Experiments

We train the k-poselets on the PASCAL VOC 2012 train dataset. We then evaluate the performance of k-poselets on two tasks: person detection and keypoint estimation. Below we describe each task, how we use k-poselets to solve them, and the results we achieve.

5.1. Detection task

We evaluate person detection using two metrics. The first metric requires us to predict a bounding box around the torso of each person in the image. Torso predictions that overlap by more than 50% with ground-truth torsos are considered true positives and others are false positives (duplicates are also false positives). This is exactly the PASCAL VOC detection metric, but applied to torsos. We do all tuning on the first half of VOC 2009 val and evaluate on the people images of the second half of VOC 2009 val. At test time, we use a simple torso predictor: the strongest activation in each cluster makes the prediction. For the score of the cluster, we can use the score of the highest scoring activation in the cluster. Alternatively, we can rescore the clusters using an SVM as described in section 4.2.

The second metric is the standard PASCAL VOC bounding box detection metric. This requires us to predict a box around the visible part of the person. We test on PASCAL 2007 and report the AP. Bounding box detection differs from torso prediction, since predicting the visible bounds requires us to reason about occlusions. We describe our bounding box predictor below.

Predicting the visible bounds. We find that the occlusion patterns for people typically cluster into a few modes, and so all we need to do is to predict these modes. We cluster the height of the bounding box (in units of torso height) into two clusters, and try to predict to which cluster a given person hypothesis belongs.

One could use the k-poselet activations to make this decision, but since the k-poselets haven't been trained to distinguish occluded keypoints from unoccluded ones, they

Figure 6: Examples of bounding box prediction. The red box is the torso, while the blue box is the predicted bounding box.

may not contain this information. Instead, we use the image directly. We compute HOG features on a large window in the image around the torso predicted by a person hypothesis. This window has an aspect ratio of 2:1 and is thrice as wide as the torso. To extract any information the k-poselets might have, we let each k-poselet activation vote for the height of the bounding box (in units of torso), and use the average of these votes weighted by the activation score as an additional feature. Using these features, we train a simple linear classifier (logistic regression). Figure 6 shows some examples where our bounding box predictor worked well.

The score of any given k-poselet might not be optimal for scoring visible-bounds detections. So, we train an SVM to rescore bounding boxes, as described for torsos in section 4.2. However, here we use visible bounds overlap for defining training labels. For training the SVM, predicted boxes that overlap by more than 50% with ground truth are considered positive and the others negative. We train the bounding box predictor and the rescoring SVM on VOC 2009 val. Finally, we found that AMP selection on torsos tends to choose poselets that may be sub-optimal for the bounding box task. Therefore for bounding box detection we use the AMP criterion applied to bounding boxes.

Tables 1 and 2 shows our results on bounding box detection and torso detection respectively. In each case we observe that without doing any rescoring, adding 2-poselets to the pool provides a boost over just 1-poselets. The rescoring improves all numbers. Interestingly, after rescoring the AP of the 1-poselets and the $\{1,2\}$-poselets are comparable. This is likely because the rescoring SVM can use information present in the ensemble of poselets and thus reduce the performance gap. Moreover, the procedure for selecting poselets is greedy and does not take into account the final detection performance. A better selection procedure may make better use of the gains provided by 2-poselets. Nevertheless, as we show in Section 5.2, 2-poselets are more informative about pose and provide a significant boost for keypoint prediction.

Compared to previous approaches our final numbers on

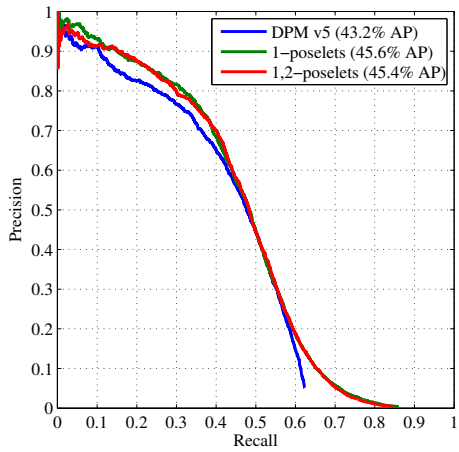

Figure 7: Precision-recall curves on PASCAL VOC 2007 test. Note that we not only achieve greater precision but also higher recall, compared to DPM [10].

method	AP
1-poselets (no rescoring)	34.6
$\{1,2\}$-poselets (no rescoring)	36.0
1-poselets (rescored, bbox AMP)	45.6
$\{1,2\}$-poselets(rescored, bbox AMP)	45.4
DPM	43.2
DPM grammar [15]	46.7
Poselets [3] (q scores)	45.7
Poselets [3] (Q scores)	**46.9**

Table 1: Bounding box detection average precision (%) on PASCAL VOC 2007 test.

method	DPM	1-poselets	$\{1,2\}$-poselets	1-poselets rescored	$\{1,2\}$-poselets rescored
AP	21.9	21.7	23.7	27.6	**27.8**

Table 2: Results on torso bounding box prediction on the second half of PASCAL VOC 2009 val (VAL2).

bounding box detection (after rescoring) are significantly better than DPMs, and match what [3] achieve without the (computationally expensive) context rescoring. Figure 7 shows the precision-recall curve. We also significantly outperform DPMs in the task of torso prediction (Table 2).

It's also notable that we trained each k-poselet detector using a small set of only 200 negative images in order to accelerate training, as suggested in [16].

5.2. Keypoint prediction task

For the keypoint prediction task, we use PASCAL VOC 2009 val where keypoint annotations were collected by [3]. We tune our parameters on the first half of that set ("VAL1") and evaluate on the second half ("VAL2").

Evaluation metrics. For the keypoint prediction task, we use the average precision of keypoints (APK) metric, proposed in [25], because it combines detection and pose estimation into a single task (unlike PCP). This task matches the stated goal of the paper, which is to provide a unified framework that performs well on both sub-tasks. APK measures the correctness of predicted keypoints by computing the precision-recall curve for each keypoint across all predictions. A keypoint prediction is considered correct if its distance to the corresponding ground-truth keypoint is less than $\alpha \cdot h$, where h is the height of the torso of the ground-truth instance. Average precision is reported for each keypoint individually.

Predicting keypoint locations. By design, k-poselets can be used to predict the bounds surrounding the person as well the location of its keypoints. However, different k-poselets should have different confidence levels for predicting a particular keypoint, *e.g.* a head & shoulder k-poselet should not make predictions of the ankles. We measure this confidence by mapping the k-poselet scores to keypoint detection precisions using the procedure described in Section 3. For each keypoint, only the most confident k-poselet (for that specific keypoint) predicts its location. The precision-recall curves are computed on VAL1.

Results. We report numbers on the second half of PASCAL VOC 2009 val (VAL2). We compare our performance with two leading techniques in person detection and keypoint prediction, DPM [10] and Y&R [25]. For DPM, we trained a person model on our training set as well as a linear regressor from the deformed location of the part filters to the location of the keypoints. For Y&R, we used their released model, which was trained on the PARSE dataset. We did not train a Y&R model on our training set, since it requires fully visible training instances and only 6% of our training set includes such instances. We used the detections of the Y&R model on our test set with NMS threshold of 0.5 to produce candidate person hypotheses and keypoint predictions. We also tried using our $\{1, 2\}$-poselets clusters to filter their detections. To be more precise, from all the Y&R detections that overlapped more than 0.4 with the torso prediction of each cluster, we picked the detection that scored the highest to make keypoint predictions for that cluster. The Y&R final detection inherits the score of the cluster.

Table 3 shows average precision using the APK criterion for the different approaches. Note that Y&R do not have a prediction for Nose directly. We compute the mean AP for all keypoints (column mAP) and also for the arm and leg keypoints for a comparison with Y&R (column mAP*)

The task of keypoint prediction on VAL2 is challenging, especially compared to other pose estimation datasets, such as Buffy Stickmen [12] or the PARSE dataset [21], since it contains people under a wide variety of occlusion, clutter and appearance patterns. A part or a keypoint of an instance is evaluated as long as it has a valid keypoint annotation. This protocol is different than the one used in [17], where an instance was evaluated only if all its arm keypoints had valid annotations. Our protocol makes the task much harder, since heavily occluded instances are considered in the evaluation. In addition, [17] did not report APK numbers, since armlets were not trained to detect people and assumed knowledge of ground truth at test time. Figure 8 shows examples of keypoint predictions on some test instances in VAL2 (we cropped the image around the instance in question for display only). We depict the correct keypoint predictions with green and the false predictions with black dots. Predictions that correspond to keypoints that don't have ground-truth annotations (due to no consensus among annotators) are marked with a black cross. For better visualization we connect the keypoints with red colored lines. We also show the strictness of the metric by placing a blue circle of radius equal to the tolerance of the metric centered on the ground truth Right Shoulder (blue square).

6. Conclusion

We have shown that an ensemble of k-poselets provides a unified formalism for person detection and keypoint prediction. In this work we also have simplified the poselet training pipeline and provided source code, so that it can be widely used. Applications such as attribute detection, action recognition, semantic segmentation could be built in a fairly straightforward way. Looking ahead, this machinery can use features other than HOG, such as CNN-based features which have been shown to perform well for the task of object detection [14].

Acknowledgments. This work was supported by the Intel Visual Computing Center, ONR SMARTS MURI N000140911051, ONR MURI N000141010933, a Google Research Grant and a Microsoft Research fellowship.

References

[1] M. Andriluka, S. Roth, and S. Bernt. Pictorial structures revisited: People detection and articulated pose estimation. *CVPR*, 2009.

[2] H. Azizpour and I. Laptev. Object detection using strongly-supervised deformable part models. In *ECCV*, 2012.

[3] L. Bourdev, S. Maji, T. Brox, and J. Malik. Detecting people using mutually consistent poselet activations. In *ECCV*, 2010.

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

[5] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009.

[6] T. Brox, L. Bourdev, S. Maji, and J. Malik. Object segmentation by alignment of poselet activations to image contours. In *CVPR*, 2011.

APK [$\alpha = 0.2$]	Nose	R_Shoulder	R_Elbow	R_Wrist	L_Shoulder	L_Elbow	L_Wrist	R_Hip	R_Knee	R_Ankle	L_Hip	L_Knee	L_Ankle	*mAP*	*mAP**
DPM [10]	39.1	25.2	7.7	1.8	25.1	7.8	1.3	8.5	1.3	1.0	9.7	1.5	1.3	10.1	7.7
Y&R [25] + NMS	-	13.7	7.7	**4.3**	15.5	8.9	**5.2**	3.0	3.3	**3.9**	4.0	3.1	**4.1**	-	6.4
Y&R [25] + k-poselets	-	19.3	7.7	3.6	19.0	8.7	5.1	3.8	3.0	3.7	4.2	2.7	3.1	-	7.0
1-poselets	**44.9**	24.9	9.7	3.4	26.4	10.9	3.0	9.0	2.5	3.0	8.4	2.8	2.2	11.6	8.8
$\{1,2\}$-**poselets**	42.9	**27.1**	**12.2**	3.4	**27.3**	**11.8**	2.8	**10.6**	**4.4**	3.8	**11.4**	**4.9**	3.2	**12.7**	**10.2**

Table 3: APK average precision (%) on the second half of PASCAL VOC 2009 val (VAL2). We report keypoint prediction AP for DPM, Y&R with NMS, Y&R filtered with $\{1,2\}$-poselets, 1-poselets and $\{1,2\}$-poselets. $\{1,2\}$-poselets are better on almost all keypoints.

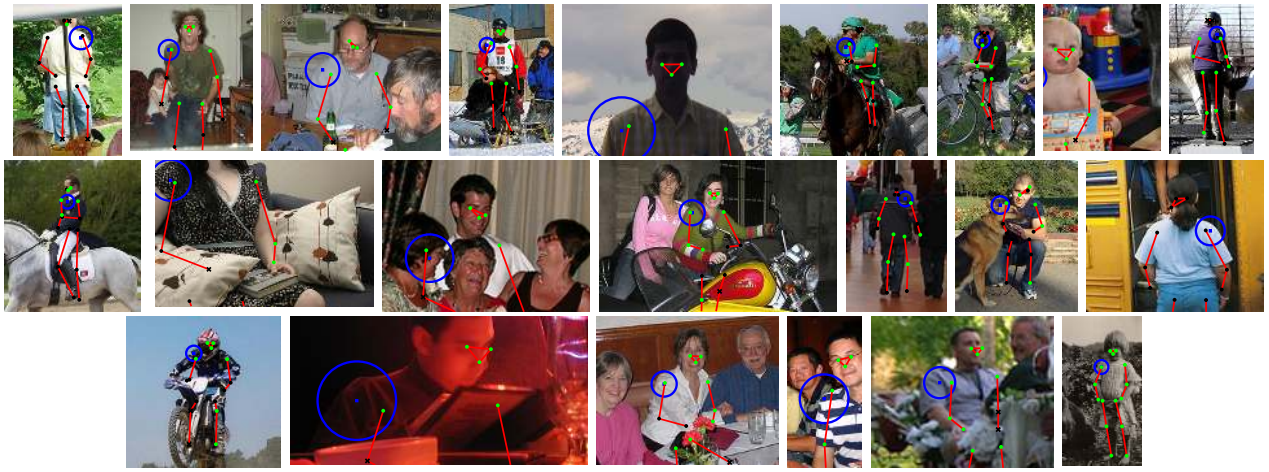

Figure 8: Examples of keypoint predictions on the second half of PASCAL VOC 2009 val (VAL2) using the APK metric with $\alpha = 0.2$. Green dots depict correct keypoint prediction while black dots false predictions. Black crosses depict predictions that correspond to keypoints that are not annotated. Red sticks are drawn between keypoints for better visualization. We show the strictness of the metric by drawing a blue circle of radius equal to the tolerance centered on the ground truth Right Shoulder keypoints, depicted with a blue square.

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

[8] M. Eichner and V. Ferrari. Better appearance models for pictorial structures. *BMVC*, 2009.

[9] I. Endres, K. J. Shih, J. Jiaa, and D. Hoiem. Learning collections of part models for object recognition. CVPR, 2013.

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

[11] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 2005.

[12] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. CVPR, 2008.

[13] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. on Computer*, 22(1), 1973.

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

[15] R. Girshick, P. Felzenszwalb, and D. McAllester. Object detection with grammar models. In *NIPS*, 2011.

[16] R. Girshick and J. Malik. Training deformable part models with decorrelated features. In *ICCV*, 2013.

[17] G. Gkioxari, P. Arbelaez, L. Bourdev, and J. Malik. Articulated pose estimation using discriminative armlet classifiers. CVPR, 2013.

[18] S. Johnson and M. Everingham. Clustered pose and nonlinear appearance models for human pose estimation. *BMVC*, 2010.

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

[20] L. Pishchulin, M. Andriluka, P. Gehler, and B. Schiele. Poselet conditioned pictorial structures. In *CVPR*, 2013.

[21] D. Ramanan. Learning to parse images of articulated bodies. NIPS, 2006.

[22] B. Sapp, A. Toshev, and B. Taskar. Cascaded models for articulated pose estimation. *ECCV*, 2010.

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

[24] Y. Wang, D. Tran, and Z. Liao. Learning hierarchical poselets for human parsing. In *CVPR*. IEEE, 2011.

[25] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures-of-parts. *TPAMI*, 2012.