Video Pose Estimation with Convolutional Neural Networks and Recombination

Sam Toyer
The Australian National University
u5568237@anu.edu.au

Abstract

This paper presents a method for 2D human pose estimation across a sequence of video frames. Our approach is comprised of a Convolutional Neural Network (CNN) and graphical model for independent generation of pose candidate sets in each frame, followed by a recombination step which makes use of optical flow and limb deformation costs to produce a single, consistent sequence of poses. Evaluation on the Poses in the Wild data set validates the usefulness of inter-frame pose recombination over single-frame pose estimation alone, and shows that our method significantly improves upon past work in localisation of wrists and elbows.

1. Introduction

The aim of human pose estimation is to take a static image or sequence of video frames and output "skeletons" representing the locations of the joints[1] of any humans in those images, as depicted in Figure 1. Skeletons produced through pose estimation are useful for many higher-level computer vision tasks: for instance, knowing the locations of limbs can be helpful for identifying what kind of clothing a person is wearing [14, 15, 23], or for recognising actions [25]. Pose estimation can even be used to classify inanimate objects by observing how a person interacts with them [4]—for instance, a chair could be identified by a person in a sitting position above it. Improvement of existing pose estimation techniques can yield flow-on improvement in all of these applications.

Pose estimation is complicated by the wide range of plausible poses, the varied appearance of human bodies and clothing, occlusion of joints by objects or by other limbs, background clutter which resembles joints, and so on. Taking advantage of the motion information present in video sequences introduces another layer of complexity to this already challenging problem.

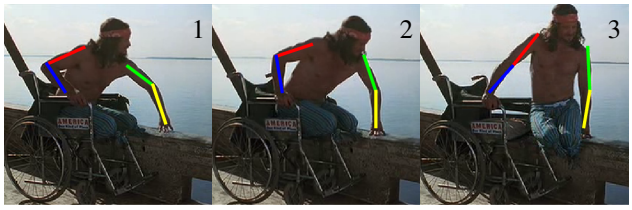

Figure 1. A video sequence annotated by our algorithm. Only shoulders, wrists and elbows are shown in this example.

We deal with this complexity by dividing our video pose estimation pipeline into two phases. Initially, a pose estimation procedure is applied independently to each frame in the video sequence. This procedure implicitly scores all possible poses in each video frame and returns a selection of the highest-scoring poses, which is pruned through non-maximum suppression to ensure that the returned set contains a diverse selection of poses. After a candidate pose set has been generated for each frame, the joints in each candidate pose are split up, and the complete set of joints is used to produce a single best position for each joint in each frame of the sequence.

Our candidate set generation procedure, described in Section 3, follows the approach of Chen and Yuille [2], which combines features from a deep CNN with a graphical model. The CNN yields features which are much more informative than the Histogram-of-Gradients (HoG) features used in previous work [3, 7, 17, 24], whilst the graphical model encourages joints to take on anatomically reasonable relative positions.

As explained in Section 4, incorporating temporal edges into graphical model-based pose estimation methods can result in an intractable inference problem. Independently generating a small set of candidate poses in each frame and using temporal edges to select the best candidate partially alleviates this problem. We employ a recombination method—as described by Cherian *et al*. [3]—to improve efficiency even further by allowing joints to be mixed-and-matched from different poses in each candidate set, thereby increasing the

[1] In the jargon of pose estimation, the term "joint" can be used to refer to any identifiable point on a person's body, including the elbows, wrists, head, and so on. Similarly, "limb" can be used to describe any connection made between joints for the purpose of constraining or visualising poses, and need not refer to an anatomical limb like an arm or leg.

effective size of the candidate sets under consideration whilst maintaining tractability.

In Section 5.1, we find that, by combining these two approaches, we obtain a significant in accuracy over Cherian *et al.*'s method, and a slight increase in accuracy over Chen and Yuille's method when estimating the positions of wrists and elbows. Wrists and elbows are the most difficult joints to detect due to their size and rapid motion, which makes our result significant, especially given that, until recently, the aforementioned approaches constituted the state-of-the-art in video and static image pose estimation, respectively.

2. Related work

Early attempts at human pose estimation often made use of pictorial structure models [5, 8], which exploit the fact that poses can be decomposed into joints which are likely to stay within certain distances of one another. Yang and Ramanan [24] extended this approach by introducing the notion of joint types to capture the different appearances which a joint can have depending on its orientation or other attributes. By treating the types of each joint as latent variables in a graphical model, and making deformation terms take into account the joint types on either end of a limb, Yang and Ramanan were able to obtain a significant increase in accuracy over earlier models. In this paper, we use a similar approach for generating pose candidate sets within frames.

Another common approach is to regress the (x, y) coordinates of joints directly from an image [20, 22], although this can lead to implausible poses being generated unless some constraints on the relative positions of limbs or on the overall pose are introduced. Using graphical models, as we have, can increase accuracy by eliminating pose candidates in which predicted joint positions are individually likely, but collectively implausible due to anatomical constraints.

The benefits of CNNs over traditional hand-engineered image features—as elucidated by LeCun and Bengio [13]—have recently led to major advances in image classification, object detection, and other fundamental tasks in computer vision [12, 18]. This has prompted increased investigation of CNNs for pose estimation.

Toshev and Szegedy [22] produced an early result in this area by proposing a cascade of CNN-based regressors, where the first regressor outputs an approximate location for each joint, and subsequent regressors are used to refine those approximations. Alternative approaches [9, 10, 16] instead map images to heatmaps for each joint, which has the advantage of being a less nonlinear mapping than that from images to joint coordinates. The heatmap approach also makes it straightforward to incorporate graphical models later in the pose estimation pipeline, since the heatmap values at different locations can be used as graphical model potentials [2, 21], as explained further in Section 3.1.

The motion information available in videos has previously been exploited using tracking [1, 11] or by extending graphical models for single-frame prediction with temporal links between joints in different frames [3, 7, 19]. Motion information has also been incorporated into CNN-based detectors by introducing motion features like optical flow at the input layers, which can sometimes increase accuracy over single-frame pose estimation [10]. As alluded to in Section 1, incorporating temporal links into graphical models can make inference intractable, and so past approaches in this area have used approximations. We avoid this problem by using recombination, as described in Section 4.1.

3. Single-frame candidate set generation

Our method for generating pose candidate sets largely follows that of Chen and Yuille [2], although we will use this section to describe the relevant parts in full.

Pose skeletons are represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of a set of joints $\mathcal{V}$ and a set of limbs $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. A complete pose $\mathbf{p} = (\mathbf{l}, \mathbf{t})$ is represented by a location $\mathbf{l_u}$ within an image $\mathbf{I}$ for each joint $u \in \mathcal{V}$, and discrete "types" $t_{uv} \in \{1, \ldots, T_{uv}\}$ and $t_{vu} \in \{1, \ldots, T_{vu}\}$ for each limb $(u, v) \in \mathcal{E}$.

Limb types are used to express the orientation and length of limbs. For example, one type might correspond to long forearms running from left to right. During inference, these types are introduced as latent variables, and type-dependent limb deformation costs are added to encourage joints to take anatomically reasonable relative positions. Further evidence for the type of a limb can be gleaned by inspecting small patches of an image around the endpoints of that limb. For instance, an image of a shoulder might give clues as to the direction in which the attached upper arm is pointing; this is discussed at greater length in Section 3.1.

Given an image and a complete pose $(\mathbf{l}, \mathbf{t})$ consisting of a set of joint locations $\mathbf{l}$ and a set of limb types $\mathbf{t}$, the full score $C(\mathbf{l}, \mathbf{t})$ of the pose can be decomposed into a sum of unary costs and pairwise costs, written as

$$C(\mathbf{l}, \mathbf{t}) = w_0 + \sum_{u \in \mathcal{V}} \phi_u(\mathbf{l}_u) + \sum_{(u,v) \in \mathcal{E}} \psi_{uv}(\mathbf{l}_u, \mathbf{l}_v, t_{uv}, t_{vu}), \tag{1}$$

where ϕ_u and ψ_{uv} are described in the next section, and w_0 is a bias term.

3.1. Image-dependent terms

The pairwise cost $\psi_{uv}(\mathbf{l}_u, \mathbf{l}_v, t_{uv}, t_{vu})$ can be decomposed into the sum of type-dependent deformation costs and Image Dependent Pairwise Relation (IDPR) terms, as expressed by

$$\begin{aligned}\psi_{uv}(\mathbf{l}_u, \mathbf{l}_v, t_{uv}, t_{vu}) &= \mathbf{w}^T_{uvt_{uv}} d(\mathbf{l}_v - \mathbf{l}_u - \mathbf{r}_{uvt_{uv}}) \\ &+ \mathbf{w}^T_{vut_{vu}} d(\mathbf{l}_u - \mathbf{l}_v - \mathbf{r}_{vut_{vu}}) \\ &+ w_{uv}\mathcal{I}_{uv}(\mathbf{l}_u, t_{uv}) + w_{vu}\mathcal{I}_{vu}(\mathbf{l}_v, t_{vu}),\end{aligned} \tag{2}$$

where $d(\mathbf{v}) = \begin{bmatrix} v_x^2 & v_y^2 & v_x & v_y \end{bmatrix}^T$ is a deformation feature, $\mathbf{r}_{uvt_{uv}}$ is the average displacement of a limb of type t_{uv}, and $\mathcal{I}$ represents an IDPR term, explained below.

Given a K-joint skeleton, we can define $p(j = u \mid \mathbf{I}(\mathbf{l}))$ to be the probability that the joint contained in the patch $\mathbf{I}(\mathbf{l})$ of the image $\mathbf{I}$ around $\mathbf{l}$ is the joint represented by $u \in \{1, \ldots, K\} \cup \{0\} = \mathcal{V} \cup \{0\}$, with the special value of $u = 0$ indicating that no joint is present. If we know that the patch $\mathbf{I}(\mathbf{l})$ contains a joint u, and $(u, v) \in \mathcal{E}$ is a limb, then we can define $p(t_{uv} = t \mid j = u, \mathbf{I}(\mathbf{l}))$ to be the probability that the limb between (u, v) has type $t_{uv} \in \{1, \ldots, T_{uv}\}$. Using this notation, we can define the IDPR term $\mathcal{I}$ as

$$\mathcal{I}_{uv}(\mathbf{l}_u, t) = \log p(t_{uv} = t \mid j = u, \mathbf{I}(\mathbf{l}_u)). \tag{3}$$

The inclusion of both $\mathcal{I}_{uv}$ and $\mathcal{I}_{vu}$ ensures that the visual cues given by the joints at either end of a limb can be used to infer the type of that limb.

The appearance term ϕ_u is defined similarly, and gives the log probability that a small, fixed-size patch of the image centered at $\mathbf{l}_u$ contains the joint u:

$$\phi_u(\mathbf{l}_u) = w_u \log p(j = u \mid \mathbf{I}(\mathbf{l}_u)). \tag{4}$$

3.2. Computing unaries and IDPR terms

To compute the unaries defined by (4) and the IDPR terms defined by (3), we train a CNN to output a distribution over joints and the types of neighbouring limbs for each patch of a given image $\mathbf{I}$, from which we may obtain the appearance and IDPR terms by marginalisation. If we let $t_u \in \prod_{(u,v)\in\mathcal{E}} \{1, \ldots, T_{uv}\}$ denote a combination of types for all limbs adjacent to a joint u, then we can write this distribution as

$$p(j = u, t_u = t \mid \mathbf{I}(\mathbf{l})) \tag{5}$$

for any location $\mathbf{l}$ in $\mathbf{I}$ and any joint $u \in \{1, \ldots, K\}$.

Our CNN architecture closely follows that of AlexNet [12], and is identical to that of [2]. To minimise wasted computation, we convert the final fully connected layers of the network to 1×4096 convolutions after the network has been trained. This allows us to evaluate (5) over a set of uniformly spaced patches of a full-resolution image in a single pass, rather than having to pass patches through the network one at a time [18].

3.3. Producing the candidate set

Having evaluated appearance and IDPR terms for all joints and all locations in the image, we can now produce a set of high-scoring pose candidates for use in the recombination procedure (Section 4). Recall that poses are modelled as trees rooted at the head; the score of any subtree of the full pose tree rooted at joint u in location $\mathbf{l}_u$ is

$$S_u(\mathbf{l}_u) = \phi_u(\mathbf{l}_u) + \sum_{pa(v)=u} \max_{\mathbf{l}_v, t_{uv}, t_{vu}} \left[\psi_{uv}(\mathbf{l}_u, \mathbf{l}_v, t_{uv}, t_{vu}) + S(\mathbf{l}_v)\right], \tag{6}$$

where $\mathrm{pa}(v) = u$ iff u is the parent of v in the full pose tree.

At the leaves, this formula becomes $S_v(\mathbf{l}_v) = \phi_v(\mathbf{l}_v)$, which is trivial to compute for all locations in the image. Otherwise, given child scores $S_{v_1}(\mathbf{l}_{v_1}), \ldots, S_{v_C}(\mathbf{l}_{v_C})$ for the children $\{v : \mathrm{pa}(v) = u\}$ of some non-leaf joint u, it is possible to evaluate $S_u(\mathbf{l}_u)$ for all locations $\mathbf{l}_u$ in linear time using distance transforms [6]. If we have $T_{uv} = T$ for each $(u, v) \in \mathcal{E}$, then we must also perform maximisation over T^2 limb label combinations at each joint. Since this maximisation must be performed at a total of K joints, the overall time complexity of calculating the score $S_h(\mathbf{l}_h, \mathbf{I})$ of the root component for all N values of $\mathbf{l}_h$ is $O(T^2NK)$.

Given the maximum scores $S_h(\mathbf{l}_h)$ for a pose rooted at each possible head location $\mathbf{l}_h$, we can produce a set of M pose candidates by choosing the M highest-scoring head positions and backtracking to find the remainder of the pose. However, since recombination benefits from a diverse set of poses, we apply non-maximum suppression to ensure that our returned candidate pool contains only poses for which the pairwise intersection-over-union for detected wrists is no greater than some threshold.

3.4. Learning

Training for the single-frame candidate generation model begins with derivation of the mean limb displacement $\mathbf{r}_{uvt_{uv}}$ for each limb $(u, v) \in \mathcal{E}$ and each type $t_{uv} \in \{1, \ldots, T\}$ for that limb, where we have assumed for simplicity that each limb has the same number of types T. For a limb (u, v), this is achieved by calculating the displacement $\mathbf{l}_v - \mathbf{l}_u$ associated with each pose in the training set, then running K-means to find T centroids for the calculated displacements.

Having assigned a type to each limb in the training set, an image crop is made around each joint and labelled with the joint type and the types of all neighbouring limbs. We also include a set of patches not containing any people, which are labelled with a special negative label. The produced set of patches and labels is used to find parameters for the neural network described in Section 3.2 using stochastic gradient descent.

Finally, we can learn the bias w_0 and the weight sets $\{\mathbf{w}_{uvt_{uv}}\}$, $\{w_{uv}\}$, and $\{w_u\}$ for the pose cost (1). The cost associated with the limb locations, image, and limb types detected from a training sample can be represented as an inner product between a weight vector and a feature vector composed of all appearance, IDPR and deformation terms. If we classify all accurate predicted pose configurations as "positive" examples, and all other configurations—including

poses predicted for images in which no humans are present—as "negative" examples, then this can be viewed as the problem of finding weights for a structural SVM, which we do using the dual coordinate descent approach described in [24].

4. Pose estimation in videos

We have already seen in Section 3 how we can use a graphical model to do pose estimation within a single frame. Pose estimation in videos is similar to pose estimation in static images, except that we wish to enforce some sort of temporal consistency between poses. The obvious approach to this problem is to apply our existing graphical model to each frame, but to also add some temporal consistency term $\tau(p_t, p_{t+1}) = \sum_{u \in \mathcal{V}} \tau_u(\mathbf{l}_{u,t}, \mathbf{l}_{u,t+1})$, where p_t is the pose predicted at time t, and $\mathbf{l}_{u,t}$ denotes the predicted location of joint u at time t. If we had F frames in total, then the full cost would be

$$C(p_F) + \sum_{t=1}^{F-1} \left[C(p_t) + \tau(p_t, p_{t+1}) \right], \tag{7}$$

where we have abbreviated $C(\mathbf{l}_t, \mathbf{t}_t)$ as $C(p_t)$.

(7) corresponds to a complex, highly loopy graph, which makes it infeasible to find the $p_1, \ldots, p_F$ which minimises (7) for any nontrivial choice of τ. One way to reduce this computational burden is to restrict the set of poses which we consider to some limited set $\mathcal{P}_t$ in each frame; if we have exactly $|\mathcal{P}|$ candidate poses in each frame, then dynamic programming would allow us to minimise (7) in $O(|\mathcal{P}|^2 F)$ time. $|\mathcal{P}|^2$ can still be colossal when a large number of poses are considered in each frame, which restricts the applicability of this technique to situations in which $|\mathcal{P}|$ is small.

4.1. Recombination

Cherian *et al*. [3] avoid the penalty incurred by large pose candidate sets by taking a small, diverse set of $|\mathcal{P}|$ candidate poses and then considering all possible combinations of joints from each of those poses. Given $|\mathcal{P}|$ poses and K joints, this results in an effective candidate set of $|\mathcal{P}|^K$ poses in each frame, and ensures that the best joints in the pose candidate set, rather than just the best pose candidates themselves, are considered.

Now that we know we can efficiently perform inference on a large effective pose set, we can introduce temporal smoothing links between each joint $u \in \mathcal{V}$ in the pose at time t and its counterpart at time $t+1$ using a cost

$$\tau_u(\mathbf{l}_{u,t}, \mathbf{l}_{u,t+1}) = \lambda_\tau \| \mathbf{l}_{u,t+1} - \mathbf{l}_{u,t} - f_t(\mathbf{l}_{u,t}) \|^2, \tag{8}$$

where $f_t(\mathbf{l}_{u,t})$ is the optical flow at location $\mathbf{l}_{u,t}$ between frame t and frame $t+1$.

Additionally, to encourage the connected limbs chosen to make up each frame's final, recombined pose to be close to one another, we introduce a recombination cost ρ_v for each pair of limbs $(u, v), (v, w) \in \mathcal{E}$ (where $u \neq w$) which share a common joint v:

$$\rho_v(\mathbf{l}_v, \mathbf{l}'_v) = \lambda_\rho \| \mathbf{l}_v - \mathbf{l}'_v \|^2. \tag{9}$$

The complete cost which the recombination process must minimise is therefore given in (10); we have used $\mathcal{V}_S = \{v : \exists u \neq w : (u, v) \in \mathcal{E} \wedge (v, w) \in \mathcal{E}\}$ to denote the set of joints which are shared between two or more limbs, whilst $\mathbf{l}_v$ denotes the location of joint v in a recombined pose and $\mathbf{l}'_v$ denotes a location of joint a v which was discarded during recombination, but for which the location of some joint w *connected* to v was used.

$$\begin{aligned} \sum_{t=1}^{F-1} \left[C(p_t) + \sum_{v \in \mathcal{V}_S} \rho_v(\mathbf{l}_{v,t}, \mathbf{l}'_{v,t}) + \sum_{u \in \mathcal{V}} \tau_u(\mathbf{l}_{u,t}, \mathbf{l}_{u,t+1}) \right] \\ + C(p_F) + \sum_{v \in \mathcal{V}_S} \rho_v(\mathbf{l}_{v,F}, \mathbf{l}'_{v,F}) \end{aligned} \tag{10}$$

In order to make the minimisation of the full temporal cost (10) tractable, limbs are recombined starting at the head—which is typically the easiest joint to detect—then moving on to the neck, the shoulders, and so on until the full pose has been estimated in all frames.

Specifically, the algorithm starts by choosing a head position $\mathbf{l}_{h,t}$ at each time $t = 1, \ldots, F$ such that the chosen sequence of heads minimises the following cost, which corresponds to the parts of the full cost (7) that involve the head or any temporal links between heads in adjacent frames:

$$\phi_h(\mathbf{l}_{h,F}) + \sum_{t=1}^{F-1} \left[\phi_h(\mathbf{l}_{h,t}) + \tau_h(\mathbf{l}_{h,t}, \mathbf{l}_{h,t+1}) \right]. \tag{11}$$

If we have $|\mathcal{P}|$ candidate poses in each frame, each of which corresponds to a single head position candidate, then we can use dynamic programming to perform this minimisation in $O(|\mathcal{P}|^2 F)$ time.

Position sequences for any subsequent joint u can be found in much the same way, except that we must also include pairwise costs from the single-frame cost C, as well as recombination costs relative to the (previously localised) parent joint, yielding a full cost of

$$\begin{aligned} \sum_{t=1}^{F-1} \left[C_{uv}(l_{u,t}, \mathbf{l}_{v,t}, \mathbf{t}) + \rho_u(\mathbf{l}_{u,t}, \mathbf{l}'_{u,t}) + \tau_u(\mathbf{l}_{u,t}, \mathbf{l}_{u,t+1}) \right] \\ + C_{uv}(l_{u,F}, \mathbf{l}_{v,F}, \mathbf{t}) + \rho_u(\mathbf{l}_{u,F}, \mathbf{l}'_{u,F}), \end{aligned} \tag{12}$$

where $v = \mathrm{pa}(u)$ is the parent of u and $C_{uv}(\mathbf{l}_u, \mathbf{l}_v, \mathbf{t})$ are the terms of the single-frame cost (1) which either involve only u or involve both u and v.

As with the head, finding the appropriate sequence of joint locations for each remaining joint can be done in $O(|\mathcal{P}|^2 F)$ time with dynamic programming, meaning that the total runtime of the minimisation procedure is $O(K|\mathcal{P}|^2 F)$ for a K-joint skeleton.

4.2. Approximations and heuristics

In practice, the unary terms in the head sequence cost (11) and the cost (12) for subsequent joints can be approximated by the full, single-frame inference score $C(\mathbf{l}, \mathbf{t})$ for the specific candidate pose being considered. This not only makes implementation easier, but improves performance due to the fact that the single-frame inference scores are already computed during candidate set generation.

In addition to the costs listed above, we have used the "practical extensions" of [3], which include additional keypoints along limbs to constrain motion further, an additional term for wrists which encourages them to occupy regions of high motion, and a regularisation term which acts to constrain the absolute difference between joint positions across frames.

5. Experiments

To evaluate the performance of our model, we tested it on the Poses in the Wild [3] data set, which consists of a series of 16–30 frame sequences extracted from movies. Many sequences in this data set include a large degree of camera motion, rapidly moving subjects, cluttered backgrounds or occlusion of joints.

The single-frame pose estimation model was trained on the Frames Labelled in Cinema (FLIC) data set [17], with negatives drawn from the INRIA person data set.[2] The data set was augmented by rotating images through a 70° range in 5° increments, as done in [2].

For recombination, we chose a set of hand-tuned parameters which differed for each pair of joints, depending on the extent of motion of the joints.[3] At test time, 100 candidate poses were used in each frame and NMS was performed at a threshold of 95% of the intersection-over-union on each wrist.

We found it advantageous to perform candidate set generation independently at several scales. The highest-scoring poses over all scales were passed to the recombination stage to produce a final pose sequence.

5.1. Results

The accuracy of our algorithm on Poses in the Wild is depicted in Figure 3. For comparison, we have included the results of our algorithm in its previously described evaluation configuration and the results of our algorithm when only one candidate pose is generated for each frame, in which case it is equivalent to Chen and Yuille's method [2]. We have also included the results of Cherian *et al*. [3] on the same sequence. Timings for the different stages of the pose estimation pipeline during testing on a single sequence of Poses in the Wild are given in Table 1.

Stage	Flow	CNN	Candidate set gen.	Recomb.
Time	337s	1713s	159s	11s

Table 1. Run time of different stages of the pipeline when applied to all 30 frames of sequence 15 of Poses in the Wild. The experiment was performed using two Intel Xeon E5-2620 processors and an NVIDIA K80 GPU.

Note that results for Yang and Ramanan's [24] widely used pose estimator have been provided in Figure 3 solely to enable comparison with other work which has been evaluated against their system; whilst their system once represented the state of the art, and was widely benchmarked against as a result, it has since been surpassed in accuracy.

Owing to the lack of released evaluation code, the recent results of Pfister *et al*. [16] are not listed in Figure 3. However, Pfister *et al*. also reported improvements on the state of the art in video pose estimation, so comparison with the results given in their paper—especially their evaluation on Poses in the Wild—may be of benefit to some readers.

6. Discussion and future work

Figure 3 shows that our algorithm significantly improves on the results of Cherian *et al*. [3] for elbows and wrists, which are typically the most difficult joints to detect. Further, it demonstrates that recombination yields an appreciable performance increase over independent pose estimation in each frame, especially for wrists. The results also show a slight improvement over Chen and Yuille's method [2] in some cases, and compare favourably with the reported results of Pfister *et al*. [16].

Chen and Yuille's pose estimation method performs impressively well on its own given that it does not make use of temporal information. This can be ascribed to its use of powerful CNN-based image features rather than the HoG features of previous pose estimation systems. This serves to explain our performance gains relative to Cherian *et al*.'s method, since the candidate set generation stage of our pipeline is an extension of Chen and Yuille's model.

Counterintuitively, the approach of Cherian *et al*. outperforms ours on shoulders at low thresholds. This could be because of the size of joint–type distribution (5) learnt by the CNN; in the graphical model used to generate pose candidate sets, the shoulder is attached to the upper arm, upper torso and base of the neck, so if 13 types were learnt for each limb, then there would be $13^3 = 2197$ different combinations of

[2] http://pascal.inrialpes.fr/data/human/

[3] All parameters are available online, along with the rest of the code for these experiments: https://github.com/qxcv/comp2560

(a) (b) (c) (d) (e) (f)

Figure 2. Common types of errors encountered during testing on the Poses in the Wild data set.

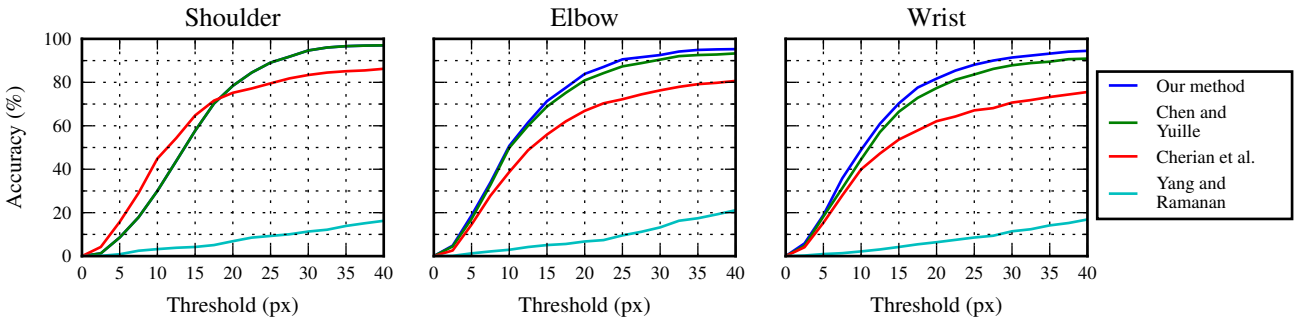

Figure 3. Accuracy on the Poses in the Wild data set for different joints. A joint is considered to be correctly detected in a single frame if the distance between its predicted position and its true position is less than some threshold. We have averaged accuracy for each joint over all frames, and tested at a wide range of thresholds to indicate precisely how close predictions are to the ground truth. Note that the curve for our algorithm in the shoulder plot is obscured by that for Chen and Yuille's algorithm.

adjacent limb types for each shoulder. Many of these combinations may not be well-represented in the training set, which would hurt performance at test time.

It may be possible to address this by adopting a joint-based type system [24] in place of a limb-based one. This would give greater control over the effective number of joint types, which could allow for the size of the joint–type distribution to be decreased significantly. This could also yield an increase in performance, since the CNN evaluation time for the final layer in the network would decrease, as would the time taken to copy and marginalise over the hundreds of thousands joint distribution values produced by applying the fully convolutional network to large images.

Another way of addressing this problem would be to train the network with more data, or perform more aggressive augmentations on an existing data set. This would be perfectly computationally feasible given that training time is linear in the number of training samples used. However, introducing more data would be a less elegant fix than modifying the existing system to be more efficient.

Figure 2 illustrates a number of common failure modes. Whilst our algorithm is robust to minor occlusions—where a hand lies slightly outside a frame, for instance—of the kind shown in frames (a) and (d)–(f), self-occlusion of subjects and partial occlusion of several joints have proven more challenging, as in frames (b) and (f). Frames (a) and (c) also show situations in which limb-like objects have confused the detector. Finally, rapid limb motion was responsible for a large number of failures, including examples (d)–(f).

7. Conclusion

We have presented a two-stage pose estimation algorithm: firstly, a candidate pose set is generated for each frame using a graphical model incorporating CNN-derived features. Secondly, a temporally consistent sequence of poses is produced by recombining the poses in each frame's candidate set. We find that a CNN-based pose estimator for individual frames can yield superior accuracy to a temporally-aware pose estimator without CNN-based features; we ascribe this result to the informativeness of CNN-produced image features. By making use of temporal smoothing as well as CNN-based features, our complete pipeline increases accuracy even further for fast-moving joints like wrists and elbows, which are typically the hardest joints to localise.

Acknowledgements We would like to thank the authors of [2] and [3] for making their code publicly available.

References

[1] M. Andriluka, S. Roth, and B. Schiele. Monocular 3D pose estimation and tracking by detection. In *CVPR*, 2010. 2

[2] X. Chen and A. Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *NIPS*, 2014. 1, 2, 3, 5, 6

[3] A. Cherian, J. Mairal, K. Alahari, and C. Schmid. Mixing body-part sequences for human pose estimation. In *CVPR*, 2014. 1, 2, 4, 5, 6

[4] V. Delaitre, D. F. Fouhey, I. Laptev, J. Sivic, A. Gupta, and A. A. Efros. Scene semantics from long-term observation of people. In *ECCV*. 2012. 1

[5] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005. 2

[6] P. F. Felzenszwalb and D. P. Huttenlocher. Distance transforms of sampled functions. *Theory of Computing*, 8(1):415–428, 2012. 3

[7] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *CVPR*, 2008. 1, 2

[8] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, (1):67–92, 1973. 2

[9] A. Jain, J. Tompson, M. Andriluka, G. W. Taylor, and C. Bregler. Learning human pose estimation features with convolutional networks. *arXiv:1312.7302*, 2013. 2

[10] A. Jain, J. Tompson, Y. LeCun, and C. Bregler. MoDeep: A deep learning framework using motion features for human pose estimation. In *ACCV*. 2014. 2

[11] Q. Ji. 3D face pose estimation and tracking from a monocular camera. *Image and Vision Computing*, 20(7):499–511, 2002. 2

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

[13] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10), 1995. 2

[14] S. Liu, J. Feng, Z. Song, T. Zhang, H. Lu, C. Xu, and S. Yan. "Hi, magic closet, tell me what to wear!". In *ACMMM*, 2012. 1

[15] S. Liu, Z. Song, G. Liu, C. Xu, H. Lu, and S. Yan. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *CVPR*, 2012. 1

[16] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In *ICCV*, 2015. 2, 5

[17] B. Sapp and B. Taskar. MODEC: Multimodal decomposable models for human pose estimation. In *CVPR*, 2013. 1, 5

[18] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv:1312.6229*, 2013. 2, 3

[19] L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard. Tracking loose-limbed people. In *CVPR*, 2004. 2

[20] M. Sun, P. Kohli, and J. Shotton. Conditional regression forests for human pose estimation. In *CVPR*, 2012. 2

[21] J. 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

[22] A. Toshev and C. Szegedy. DeepPose: Human pose estimation via deep neural networks. In *CVPR*, 2014. 2

[23] K. Yamaguchi, M. H. Kiapour, L. E. Ortiz, and T. L. Berg. Parsing clothing in fashion photographs. In *CVPR*, 2012. 1

[24] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011. 1, 2, 4, 5, 6

[25] A. Yao, J. Gall, G. Fanelli, and L. J. Van Gool. Does human action recognition benefit from pose estimation? In *BMVC*, 2011. 1