Human Arm Pose Modeling with Learned Features using Joint Convolutional Neural Network

Chongguo Li Nelson H.C. Yung Edmund Y. Lam
Department of Electrical and Electronic Engineering,
The University of Hong Kong, Pokfulam, Hong Kong
{cgli, nyung, elam}@eee.hku.hk

Abstract

This paper proposes a new approach to model arm pose configuration from color images based on the learned features and arm part structure constraints. It aims to model human arm pose without assuming of a particular clothing style, action category and background. It uses an energy model that describes the dependence relationships among arm joints and parts. A joint convolutional neural network (J-CNN) based on multi-scaled images is then developed for feature extraction of joints and parts, where the local rigidity of arm part is used to constrain the occurrence between the joints and arm parts in a dynamic programming inference. The experimental results show better performance than alternative approaches using hand-crafted features for arm pose modeling.

1 Introduction

The objective of this study is to automatically identify the positioning of human body parts or joints, using real life images or videos of people as input. Generally, the low-level features are important for human arm pose modeling, as they capture the invariant properties of the joints or arm parts. The high-level pose configuration, on the other hand, can restrict the solution space, as inferences that violate the human physical structure can be eliminated. To solve the estimation problem, one may apply probabilistic models and compute pose configuration based on the visual likelihood and pose prior [1]; alternatively, one may convert it to a classification problem to learn the relationship between the low-level features and high-level poses [2, 3]. In this paper, we focus on the learned features of arm joints and the local rigidity of arm parts using convolutional neural network, which is capable of learning features from images for various objectives.

Most studies on human pose modeling involve still images, and 3D arm pose models are often projected to 2D images for silhouette matching. For instance, Moeslund et al. [4] use a two parameters based screw-axis 3D arm model for exhaustive silhouette matching. They can reduce the solution space at the expense of depth errors as a result. For simple clothing styles and plain backgrounds, background estimation and skin color detection are used for arm detection [5], followed by a silhouette fitting with a projected 3D sticks-figure model. The pictorial structure proposed by Fishchler and Elschlager [6] is also widely used in arm pose modeling; among its extensions, Felzenszwalb and Huttenlocher [1] incorporate a Bayesian model to pictorial structure for deformable objects recognition, including arm parts detection. The mixture of parts method proposed by Yang and Ramanan [2] uses histogram of gradient (HOG) features of multiple points for each body part to increase the detection ability. Structured max-margin models [7] are also used for parameters learning, but the performance is limited by the hand-crafted feature HOG, as it is hard to capture the properties of body parts but their gradients. This structured max-margin model and HOG feature are also used in a multi-modal, decomposable model for articulated human pose estimation by Sapp and Taskar [3], combining global and local pose cues to improve the estimation performance.

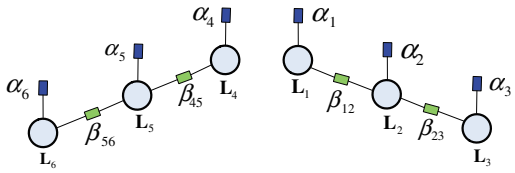


Figure 1. The graphical model of an arm pose.

2 Energy Model

2.1 Graphical model of both arms

Assume we have an arm model with J joints, and the location of the j^{th} joint is written as $\mathbf{L}_j$. Let $\Gamma = \{\mathbf{L}_1, \ldots, \mathbf{L}_J\}$ be the joint location variable for a particular arm pose configuration involving all J joints. As shown in Figure 1, each joint or arm part is attached a relevant potential, where α_i and β_{ij} evaluate the occurrences of the i^{th} joint and the arm part with the i^{th} and j^{th} joints, respectively. The location $\mathbf{L}_{ij}$ of the arm part contains its two joints' location $\mathbf{L}_i$ and $\mathbf{L}_j$. Note that a smaller potential means a higher probability of occurrence of the joint or arm part.

2.2 Energy function

Given an image I, we can derive its energy for each possible arm pose configuration Γ by summing all relevant potentials, i.e.,

$$E(I, \Gamma) = \sum_{i=1}^{J} \alpha_i(I, \mathbf{L}_i) + \sum_{ij \in \varepsilon} \beta_{ij}(I, \mathbf{L}_{ij}), \qquad (1)$$

where $J = 6$ is the size of all arm joints, and $\varepsilon = \{\{1,2\},\{2,3\},\{4,5\},\{5,6\}\}$ is the set of joint index pair for all arm parts. We aim to minimize E over all possible Γ, such that

$$\bar{\Gamma} = \arg\min_{\Gamma} E(I, \Gamma) \qquad (2)$$

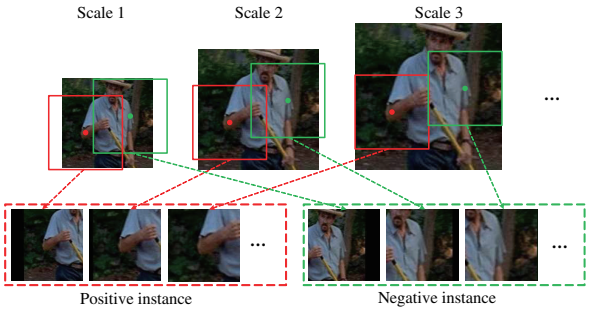


Figure 2. Multi-scale sampling for positive instance x_5 and negative instance $\neg x_5$ of the right elbow.

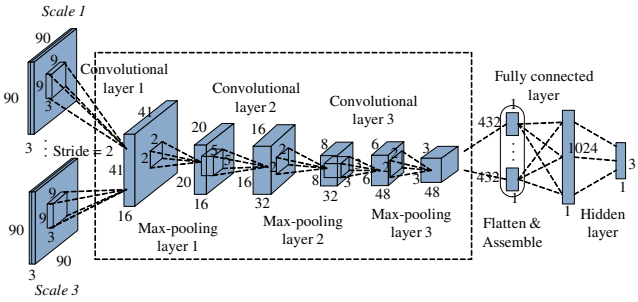


Figure 3. A basic structure of ConvNet for arm pose modeling with multi-scale input patches sharing parameters.

gives a configuration with the minimum energy. The potentials of the joints and arm parts can be defined as

$$\alpha_i(I, \mathbf{L}_i) = \|f_{\alpha_i}(x_i)\|_2 \quad (3)$$
$$\beta_{ij}(I, \mathbf{L}_{ij}) = \|f_{\beta_{ij}}(x_{ij})\|_2, \quad (4)$$

where the mapping functions f_{α_i} and $f_{\beta_{ij}}$ perform feature extraction and dimensionality reduction from the raw pixels of the image patches, respectively. x_i is the instance of the i^{th} joint at location $\mathbf{L}_i$, while x_{ij} is the instance of the arm part located between $\mathbf{L}_i$ and $\mathbf{L}_j$. $\alpha_i(x_i)$ and $\beta_{ij}(x_{ij})$ are the ℓ_2 norm of extracted features of instances for joints and arm parts in the feature space.

3 Features Learning

There are positive and negative instances in a training procedure. Below, we describe how we can map the output of the positive instances to an area close to the origin of the feature space, and the output of the negative instances to be far from the origin.

3.1 Positive and negative instances

For a unary potential α_i associated with the i^{th} joint, its positive instance x_i is an image patch centering at position $\mathbf{L}_i$, while its negative instance $\neg x_i$ is any patch not centering at $\mathbf{L}_i$. As for β_{ij}, which is associated with the i^{th} and j^{th} joints, the local rigidity of arm parts suggests that its positive instance x_{ij} is a set of image patches centering equidistantly from the two joints locations $\mathbf{L}_i$ and $\mathbf{L}_j$, and its negative instance $\neg x_{ij}$ is equidistant from two points that at least one of them is different from $\mathbf{L}_i$ or $\mathbf{L}_j$. Here, we use the middle point of each arm part as the center of the positive arm part instances. Furthermore, to capture more details of a specific joint and arm part, we make use of multiple scales of the images to sample the positive and negative instances. Figure 2 illustrates an example of sampling the positive and negative instances for the right elbow using a multi-scaled image.

3.2 Structure of J-CNN

As a powerful tool, convolutional neural network (CNN) [12] is widely used in object detection and recognition, location identification and feature representation in computer vision. In this work, CNN is also used for feature learning as the functions $f_{\alpha_i}(x)$ and $f_{\beta_{ij}}(x)$ in Eq.(3) and (4). Since they have the same network structure, we denote them as $f_{\alpha_i}(x) = \text{Conv}(x, \theta_{\alpha_i})$ and $f_{\beta_{ij}}(x) = \text{Conv}(x, \theta_{\beta_{ij}})$. This function Conv is implemented by a convolutional network (ConvNet) [8] structure. Since there is a set of CNNs to be trained on training dataset in a supervised manner, we propose a method to train them jointly. All the ConvNet units of all joints and arm parts are assembled together and trained jointly, so we call it joint CNN (J-CNN). Its overall cost summarizes all of individual error of each potential. J-CNN can take the advantage of computing capability of the current graphic processing unit (GPU) to train the CNNs in parallel, and can optimize the all potentials jointly at each iteration. Each individual CNN is independent, but just their parameters are trained jointly. As illustrated in Figure 3, an individual ConvNet unit contains three convolutional layers and three max-pooling layers alternatively, as well as two multilayer perceptrons (MLP) layers [8]. For an input instance with multi-scale patches with size 90×90, firstly the patches of a joint or arm part use a stride 2 to reduce input size and subsequently share the parameters of convolutional and max-pooling layers to derive their output. Then the outputs are flatten and assembled as the input of MLP layers to obtain the feature vector.

3.3 Supervised parameters training

3.3.1 Loss functions

We maintain a margin in the feature space to ensure that positive samples are close to the feature space origin, and the distances between negative samples and the origin are larger than a certain threshold. In order to satisfy the above criteria, the loss function based on the ℓ_2 norm is adopted for the J-CNN training, which was originally proposed by Hadsell et al. [9] for dimensionality reduction. The loss functions of potential α_i in Eq.(3) and potential β_{ij} in Eq.(4) are

$$L_{\alpha_i}(I, \mathbf{L}_i) = \frac{1}{2}\{\alpha_i(x_i)^2 + \max(0, \tau_i - \alpha_i(\neg x_i))^2\},$$
$$L_{\beta_{ij}}(I, \mathbf{L}_{ij}) = \frac{1}{2}\{\beta_{ij}(x_{ij})^2 + \max(0, \tau_{ij} - \beta_{ij}(\neg x_{ij}))^2\}$$

where α_i and β_{ij} are the potentials of joints and arm parts defined in Eq.(3) and (4), and τ_i and τ_{ij} are their corresponding margins. So, for an image I and its possible arm pose configuration Γ, the overall loss function of the J-CNN is a summation of the individual loss functions of the all joints and arm parts.

3.3.2 Mini-batching

In order to learn the parameters of J-CNN efficiently, we use the mini-batch stochastic gradient descent (SGD) method [10]. Mini-batch SGD incrementally updates the parameters performed on an average of gradients with respect to a batch of training instance rather than a single instance at each time. It can speed up the parameter optimization, since the computation of each gradient in a mini-batch is parallel and suitable for vectorization. The computed gradient at each iteration uses more training examples, so it also makes a smoother convergence.

However, in this paper, the mini-batch approach is used differently from its original version, where there are two parts in the overall loss function originally as described in Hadsell et al. [9]. At each iteration, firstly M images and the related joint annotations are randomly sampled as $(\mathbf{I}, \mathbf{\Gamma})$ from training dataset. Then mini-batch instances are generated for all joints and arm parts. They contain the positive and negative instances of the i^{th} arm joint, which can be written as $\mathbf{x}_i = \{x_i^{(1)} \dots x_i^{(M)}\}$ and $\neg\mathbf{x}_i = \{\neg x_i^{(1)} \dots \neg x_i^{(M)}\}$, respectively. Then the mini-batch instance of the joint for the current iteration is $\{\mathbf{x}_i, \neg\mathbf{x}_i\}$. Similarly, the mini-batch instance of arm part is $\{\mathbf{x}_{ij}, \neg\mathbf{x}_{ij}\}$.

The overall loss function $\mathcal{L}(\mathbf{I}, \mathbf{\Gamma})$ of current mini-batch contains two parts. One part, related to the positive instances, is defined as

$$\mathcal{L}^{+}(\mathbf{I}, \mathbf{\Gamma}) = \frac{1}{M} \sum_{m=1}^{M} \Big\{ \sum_{i=1}^{J} \alpha_i(x_i^{(m)}) + \sum_{ij \in \varepsilon} \beta_{ij}(x_i^{(m)}) \Big\}.$$

Another part related to the negative instances is

$$\mathcal{L}^{-}(\mathbf{I}, \mathbf{\Gamma}) = \sum_{m=1}^{M} \Big\{ \sum_{i=1}^{J} \frac{1}{N_i} \max(0, \tau_i - \alpha_i(\neg x_i^{(m)}))^2 + \sum_{ij \in \varepsilon} \frac{1}{N_{ij}} \max(0, \tau_{ij} - \alpha_{ij}(\neg x_{ij}^{(m)}))^2 \Big\},$$

where N_i is the size of the negative instance in the current $\neg\mathbf{x}_i$ satisfying $\tau_i - \alpha_i(\neg x_i^{(m)}) > 0$, and $N_i \leq M$ generally. N_{ij} of arm part also satisfy the corresponding restriction. N_i and N_{ij} make that training focus on the negative instances which are mapped within the margin of the feature space.

The overall loss of the current mini-batch instances is as the objective function for parameters training:

$$\mathcal{L}(\mathbf{I}, \mathbf{\Gamma}) = \frac{1}{2}\big\{\mathcal{L}^{+}(\mathbf{I}, \mathbf{\Gamma}) + \mathcal{L}^{-}(\mathbf{I}, \mathbf{\Gamma})\big\}. \tag{5}$$

3.3.3 Learning algorithm

Algorithm 1 summarizes the procedure of training for this J-CNN.

Algorithm 1 training algorithm for parameters of J-CNN.

Input: $\{(\mathbf{I}, \mathbf{\Gamma})\}$: training dataset; θ: randomly initialized parameters of all CNN; $\eta_\theta(t)$: learning rates of each CNN; τ: their corresponding margins; M: mini-batch size;

1: initial iteration counter $t = 0$;
2: **repeat**
3: $t := t + 1$;
4: randomly sample the images $\mathbf{I}$ for mini-batch and the negative joints location $\neg\mathbf{\Gamma}$ based on the annotated joints location $\mathbf{\Gamma}$ of $\mathbf{I}$;
5: prepare the positive mini-batch instances $\mathbf{x}$ and its negative mini-batch instances $\neg\mathbf{x}$ as of all joints and arm parts;
6: derive their features $f(\mathbf{x}) = \text{Conv}(\mathbf{x}, \theta)$ and $f(\neg\mathbf{x}) = \text{Conv}(\neg\mathbf{x}, \theta)$;
7: compute the overall loss function $\mathcal{L}(\mathbf{I}, \mathbf{\Gamma})$;
8: calculate gradients of each potential:
9: $\nabla\theta = \frac{\partial\mathcal{L}(\mathbf{I}, \mathbf{\Gamma})}{\partial\theta}$
10: update $\theta := \theta - \eta_\theta(t) \cdot \nabla\theta$;
11: **until** converge

Output: the learned parameters θ of all ConvNet

4 Inference

A test image contains the upper human body part and two arms generally. After constructing test instances in a pixel-by-pixel manner on its multi-scale images, its energy maps of the joints and arm parts are generated by the learned J-CNN. Then, there are two methods to infer arm joints $\bar{\Gamma}$ in Eq.(2):

1. minimizes the overall energy solely based on the generated energy maps of arm joints;

2. minimizes the overall energy by dynamic programming with local rigidity constraints based on all generated energy maps of joints and arm parts.

5 Experiment

The multi-scale image patches sizes used are 90×90, 126×126, and 162×162. The individual structure and setting of the J-CNN is shown in Figure 3. Theano and Pylearn2 are used on a CUDA Tesla K40 GPU for training. This GPU with 12GB memory makes it possible to train the J-CNN of all potentials jointly. Subsequently, we tested the proposed method on the FLIC [3] dataset which contains people with arbitrary clothing and action. The evaluation criterion for testing is Percent of Detected Joints (PDJ) as proposed by [3]. PDJ curve illustrates the estimation performance within a certain range of ratios. The sizes of training and testing samples are 17130 and 1016, respectively.

Figure 5 is an example of combining an input image and the generated energy map of each joint and arm part together. The area of inner white circle is recognized as correct [3] and the cyan point is the estimate. Figure 5(a) depicts the result of independent minimization of each arm joint and middle arm part without arm structure constraints. Although the main blue part (with low energy) is located inside the white circle, the estimated points without arm structure constraints may not. Figure 5(b) depicts the result of us-

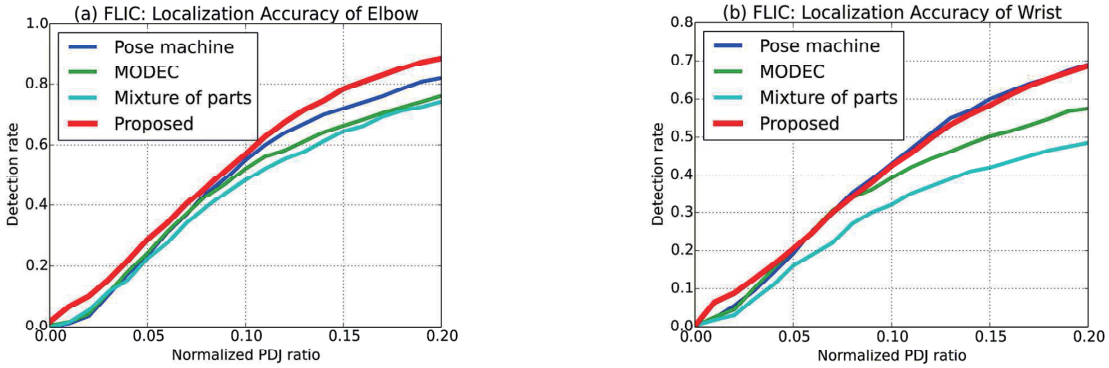
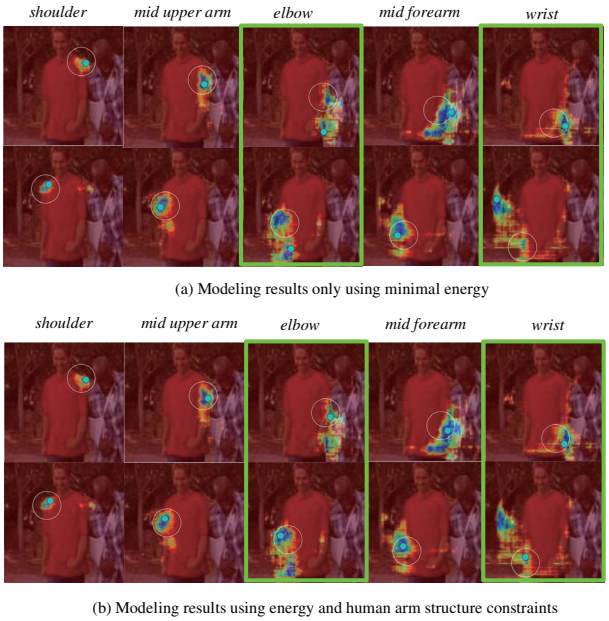


Figure 4. The PDJ curves of the proposed methods and other methods for elbow and wrist.



(a) Modeling results only using minimal energy

(b) Modeling results using energy and human arm structure constraints

Figure 5. A modeling example based on minimal energy and human arm structure constraints.

ing the local rigidity constraints for the overall energy minimization, i.e., the point of one arm part is located in the middle of its two end joints. The estimation is visually more accurate. Figure 4 shows the PDJ curves of our proposed method with rigidity constraints and other methods e.g. pose machine [11], MODEC [3] and mixture of parts [2]. It has a noticeable improvement for elbow estimation and has better performance on wrist when the PDJ ratio is small.

6 Conclusions

This work proposes a method to learn the features of both joints and arm parts by the J-CNN. It further incorporates the local rigidity property of arm part for arm pose inference. Experiment on FLIC shows a better performance than the hand-crafted feature based methods. In the future, the energy model can be enhanced by the pairwise potentials between joints of each arm part to improve its modeling ability.

Acknowledgement

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research.

References

[1] P.F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *International Journal of Computer Vision*, vol.61, no.1, pp.55–79, 2005.

[2] Y. Yang and D. Ramanan, "Articulated pose estimation with flexible mixtures-of-parts," in *Computer Vision and Pattern Recognition*, pp.1385–1392, 2011.

[3] B. Sapp and B. Taskar, "MODEC: Multimodal decomposable models for human pose estimation," in *IEEE Computer Vision and Pattern Recognition*, pp. 3674–3681, 2013.

[4] T. B. Moeslund and E. Granum, "Modelling and estimating the pose of a human arm," *Machine Vision and Applications*, vol. 14, pp. 237–247, 2003.

[5] S. Salti, O. Schreer, and L. D. Stefano, "Real-time 3D arm pose estimation from monocular video for enhanced HCI," in *ACM Vision networks for behavior analysis*, pp.1–8, 2008.

[6] M. A. Fischler and R. A. Elschlager, "The representation and matching of pictorial structures," *IEEE Transactions on Computers*, vol. 100, pp. 67–92, 1973.

[7] M. P. Kumar, A. Zisserman, and P. H. Torr, "Efficient discriminative learning of parts-based models," in *IEEE International Conference on Computer Vision*, pp.552–559, 2009.

[8] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *International Joint Conference on Neural Networks*, pp.2809–2813, 2011.

[9] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *IEEE Computer Vision and Pattern Recognition*, pp.1735–1742, 2006.

[10] O. Delalleau and Y. Bengio, "Parallel stochastic gradient descent," CIAR Summer School, 2007.

[11] V. Ramakrishna, D. Munoz, M. Hebert, J. A. Bagnell, and Y. Sheikh, "Pose Machines Articulated Pose Estimation via Inference Machines," In *European Conference on Computer Vision*, pp.33–47, 2014.

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