Neural Networks 92 (2017) 39–46

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

2017 Special Issue

Application of structured support vector machine backpropagation to a convolutional neural network for human pose estimation

Peerajak Witoonchart[*], Prabhas Chongstitvatana[*]

Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, 17th floor, Engineering 4 Building (Charoenvidsavakham), Phayathai Road, Wang Mai, Pathumwan, Bangkok 10330, Thailand

ARTICLE INFO

Article history:
Available online 16 February 2017

Keywords:
Back propagation
Convolutional neural network
Deformable part model
Human pose estimation
Structured support vector machine

ABSTRACT

In this study, for the first time, we show how to formulate a structured support vector machine (SSVM) as two layers in a convolutional neural network, where the top layer is a loss augmented inference layer and the bottom layer is the normal convolutional layer. We show that a deformable part model can be learned with the proposed structured SVM neural network by backpropagating the error of the deformable part model to the convolutional neural network. The forward propagation calculates the loss augmented inference and the backpropagation calculates the gradient from the loss augmented inference layer to the convolutional layer. Thus, we obtain a new type of convolutional neural network called an Structured SVM convolutional neural network, which we applied to the human pose estimation problem. This new neural network can be used as the final layers in deep learning. Our method jointly learns the structural model parameters and the appearance model parameters. We implemented our method as a new layer in the existing Caffe library.

1. Introduction

Generic structural prediction for object parts is similar to the human pose estimation (HPE) problem, e.g., labeling and bounding car parts with boxes, labeling and bounding face parts with boxes, and labeling and bounding bus parts with boxes. The pose estimation problem based on a still two-dimensional (2D) image is defined as finding the human joints or parts in an image containing one human. This problem is difficult due to variations in the color of clothes and because some parts are partially or totally occluded. Previous state-of-the-art pose estimation solutions are based mainly on the success of the pictorial structure programming (Felzenszwalb & Huttenlocher, 2005), which was developed by Felzenszwalb (Felzenszwalb & Huttenlocher, 2005). This method rapidly became the standard approach for object localization. Ramanan (2007) adopted this method for HPE and it is now the standard method for HPE. Ramanan's method for solving the HPE involves clustering the subparts, before extracting the histogram of oriented gradients (HOG) features for each subpart and learning support vector machine (SVM) filters for each of the subparts. Pictorial structures are then created and the parts are populated, before the subparts are filtered with these SVM filters to relearn them structurally. This method has been improved in various ways, where one way involves finding a better structure, e.g., Tian, Zitnick, and Narasimhan (2012) proposed a pictorial structure tree model with added latent variables, Tian and Sclaroff (2010) carefully designed leaf node variations and latent nodes, which control the variations of the leaf nodes, while Pishchulin, Andriluka, Gehler, and Schiele (2013) added a loopy model for inference. Dantone, Gall, Leistner, and Van Gool (2014) focused on clustering parts into multimodal decomposable models. Cherian, Mairal, Alahari, and Schmid (2014) tried to improve the pictorial structure by obtaining a better prior model by parameterizing the geometric variables. However, if the model is improved, all of these methods must learn the structural model parameters. Latent SVM (Yang & Ramanan, 2011) is the standard method for learning these model parameters.

Currently, deep learning and feature learning are popular methods for finding features for classification, detection, and segmentation, including a deep convolutional network for facial point detection (Sun, Wang, & Tang, 2013), a deep network for pedestrian detection (Sermanet, Kavukcuoglu, Chintala, & Lecun, 2013), pose estimation with a deep network (Ouyang, Chu, & Wang, 2014), facial feature tracking with a restricted Boltzmann machine (Wu, Wang, & Ji, 2013), shape prior detection using deep learning for object segmentation (Chen, Yu, Hu, & Xunxun, 2013), and object detection with a deep network (Szegedy, Toshev, &

[*] Corresponding authors. Fax: +66 0 2218 6955.
E-mail address: peerajak@gmail.com (P. Witoonchart).

http://dx.doi.org/10.1016/j.neunet.2017.02.005
0893-6080/© 2017 Elsevier Ltd. All rights reserved.

Erhan, 2013). The most similar method to that proposed in our study is that described by Yang, Ouyang, Li, and Wang (2016), which resulted in a great improvement in the percentage of correct parts (PCP), where the PCP accuracy was high as 81%. They formulated the problem as the end-to-end backpropagation of SVM hinge loss, whereas we formulate the problem as a structured SVM (SSVM) (Tsochantaridis, Hofmann, Joachims, & Altun, 2004).

Our approach starts from Girshick's claim that a convolutional neural network (CNN) (LeCun, Bottou, Bengio, & Haffner, 1998) is a deformable part model (DPM) (Girshick, Iandola, Darrell, & Malik, 2015). However, they did not backpropagate the error from the DPM model to the CNN. If a DPM is a CNN, then the error must be backpropagated to the lower layer. Thus, we backpropagate the error of the DPM back to the CNN using the SSVM loss function.

To apply latent SVM (Yang & Ramanan, 2011) with deep learning, the normal method involves extracting features with a deep neural network and performing latent SVM learning as two distinct stages using the same pipeline, i.e., feature extraction, caching the result from the first stage, and then submitting it to the latent SVM learning algorithm in the second stage. For example Girshick (Girshick et al., 2015) extracted a pyramid of features from a CNN in the feature extraction stage, before caching the extracted features, and then learning with a latent SVM during the second stage. In the second stage, the latent SVM then learned all of the model parameters by switching between SVM optimization and inference combinatorial optimization. However, this type of method has inherent problems because it cannot learn the parameters extracted by the deep learning feature from the inference optimization error because they are conducted in two distinct stages. The learnable feature extraction parameters cannot be updated based on the error of the latent SVM. Thus, we propose SSVM CNN to address this problem.

In future research, we plan to use deep CNN as a feature extractor because it is known to perform well (Yang et al., 2016). However, in the present study, we determined the feasibility of this method by showing that minimizing the loss with the SSVM CNN can be employed for part-based detection. There may be cases where the losses are minimized well but part-based detection is not achieved. If this method is feasible, we aim to employ deep CNN as the front end feature extractor instead of HOG and the SSVM neural network as the back end for HPE. In this study, we demonstrated that it is feasible to use our SSVM CNN method as the back end in the planned system.

2. Our method

HPE is the problem of estimating human joint positions. The joints are considered to have been estimated correctly if the predicted error in the difference among pixels is less than a certain threshold. The HPE problem involves the following challenges: (1) the size of a human can be small or large depending on how close the camera is to the human when obtaining an image; (2) body parts can differ in their appearance, e.g., the appearance of a hand can be a fist or palm, while a limb can be vertical or horizontal; (3) human pictures are taken in three-dimensional (3D) contexts, but there may be many 2D pose appearances for the same 3D pose, and there are many possible 3D poses. To address these challenges, the following methods have been implemented in previous studies. (1) Multiscale human detection, which is sometime called an image pyramid or feature pyramid. An image of a feature $\mathbf{x}$ is scaled to all the layer values $l \in \mathrm{L}$, where $\mathrm{L} = \{1, \ldots, l_n\}$. (2) Mixture of part types. To address the different possible appearances of each human part, each human part model is designed so it comprises multiple different part types. The body parts obtained from training images are clustered into part types based on their relative joint positions in the image coordinates with respect to neighboring joints. The underlying assumption of this clustering method is that the same group of relative joint positions will appear similar. (3) Co-occurrence model. This model consider how two neighboring parts co-occur according to a system of biases. Each type of mixture of neighboring nodes has an associated bias. These measures are incorporated into a well-known pictorial structure model where their edges are quantified under the assumption that the energy required for placing parts only varies quadratically based on the relative distance, such as the energy required for stretching or compressing springs from their anchor positions with respect to their parent nodes. These models were developed directly from that proposed by Yang and Ramanan (2011), where they combined these three models into a single large model, i.e., the coocurrence model, deformable model, and appearance model. The first two models are called structural models in our study.

We propose a new method for learning the models mentioned above. We propose to jointly learn all the models, all their parameters, in the way the neural network learns their parameters by learning all of them using stochastic subgradient descent. In contrast to Yang and Ramanan (2011), who proposed learning the appearance model prior to structural training, our method can learn all the parameters by random initialization. We note that the DPM unary filters method is exactly equal to a convolutional operation during DPM inference. During DPM inference, each unary filter determines the "sliding dot product" between feature matrices and weight matrices. This operation is exactly equal to the weights of the convolutional layer in the CNN acting on their input matrices. Thus, we design the DPM unary filters, which vary according to the human parts and human part types, as the convolution layer in the CNN. This DPM filters defines the appearance model's weights because they give the feature similarity scores. We design the cooccurrence model's weights as parameters on the loss augmented inference layer. We design the DPM pairwise weights, deformable weights, or simply the spring term's weights as other types of parameters on the loss augmented inference layer. These designs are shown in Fig. 1. In this section, we show how an SSVM based on DPM can be implemented as a two-layer neural network, where the first layer is the convolutional layer (Convssvm layer in Fig. 1) and the other is the loss augmented inference layer (loss augmented inference layer in Fig. 1). By transforming the structured form of the model into a neural network in the model, our proposed method can jointly learn the structural model and appearance model, and then backpropagate the error to learn the underlying learnable parameters that can be extracted from the appearance model features (CNN layer in Fig. 1). Thus, our proposed method translates the SSVM model into a neural network model, and thus it inherits the neural network's innate ability to backpropagate the error to a lower layer, as well as calculating the exact SSVM loss and learning the original SSVM with a subgradient-based method.

2.1. DPM problem formulation

Our approach starts by formulating the DPM as an instance of SSVM learning. As shown by Ratliff, Bagnell, and Zinkevich (2007), the SSVM can be learned by subgradient descent. Thus, we start by formulating the detection problem.

Let $\mathbf{x_i}$ represent a matrix of RGB values for the ith training data and $\mathbf{y_i}$ represents the ith training bounding boxes label, where (top left column, top left row, bottom right column, bottom right row) are denoted by $[x_1, y_1, x_2, y_2]$. Each row of $\mathbf{y_i}$ represents each bounding box for each part in a similar manner to the method proposed by Yang and Ramanan (2011). Therefore, $\mathbf{y_i}$ is a matrix of $numparts \times 4$. The bold characters indicate that they are either vectors or matrices. First, we scale $\mathbf{x_i}$ to multiple scales. To define this scaling, let $\mathrm{L} = \{1, \ldots, l_n\}$ denote the set of all scaling levels.