Learning Graphs to Model Visual Objects Across Different Depictive Styles

Qi Wu, Hongping Cai, and Peter Hall

Media Technology Research Centre, University of Bath, United Kingdom

Abstract. Visual object classification and detection are major problems in contemporary computer vision. State-of-art algorithms allow thousands of visual objects to be learned and recognized, under a wide range of variations including lighting changes, occlusion, point of view and different object instances. Only a small fraction of the literature addresses the problem of variation in depictive styles (photographs, drawings, paintings *etc.*). This is a challenging gap but the ability to process images of all depictive styles and not just photographs has potential value across many applications. In this paper we model visual classes using a graph with multiple labels on each node; weights on arcs and nodes indicate relative importance (salience) to the object description. Visual class models can be learned from examples from a database that contains photographs, drawings, paintings *etc.* Experiments show that our representation is able to improve upon Deformable Part Models for detection and Bag of Words models for classification.

Keywords: Object Recognition, Deformable Models, Multi-labeled Graph, Graph Matching

1 Introduction

Humans posses a remarkable capacity: they are able to recognize, locate and classify visual objects in a seemingly unlimited variety of depictions: in photographs, in line drawings, as cuddly toys, in clouds. Computer vision algorithms, on the other hand, tend to be restricted to recognizing objects in photographs alone, albeit subject to wide variations in points of view, lighting, occlusion, *etc.* There is very little research in computer vision on the problem of recognizing objects regardless of depictive style; this paper makes an effort to address that problem.

There are many reasons for wanting visual class objects that generalise across depictions. One reason is that computer vision should not discriminate between visual class objects on the basis of their depiction - a face is a face whether photographed or drawn. A second reason for being interested in extending the gamut of depictions available to computer vision is that not all visual objects exist in the real world. Mythological creatures, for example, have never existed but are recognizable nonetheless. If computer vision is to recognize such visual objects, it must emulate the human capacity to disregard depictive style with respect to recognition problems. The final reason will consider here is to note

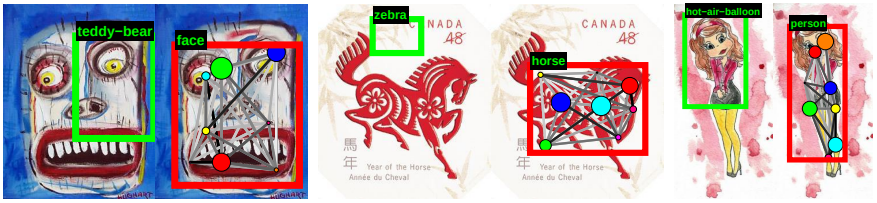

Fig. 1. Learning a model to recognize objects. Our proposed multi-labeled graph modelling method shows significant improvement for recognizing objects depicted in variety styles. The green boxes are estimated by using DPM [10], the red are predicted from our system. The text above the bounding box displays the predicted class category over a 50-classes dataset.

that drawings, paintings, *etc.* are models of objects: they are abstractions. This is obvious when one considers a child's drawing of a car in which all four wheels are shown – the child draws what they know of a car, not what is seen. In addition, a line drawing, for example, is much more compact in terms of information content than a photograph – drawings are abstractions in the sense that a lot of data is discarded, but information germane to the task of recognition is (typically) kept. This suggests that visual class models used in computer vision should exhibit a similarly high degree of abstraction.

The main contribution of this paper is to provide a modeling schema (a framework) for visual class objects that generalises across a broad collection of depictive styles. The main problem the paper addresses is this: ***how to capture the wide variation in visual appearance exhibited by visual objects across depictive styles***. This variation is typically much wider than for lighting and viewpoint variations usually considered for photographic images. Indeed, if we consider different ways to depict an object (or parts of an object) there is good reason to suppose that the distribution of corresponding features form distinct clusters. Its effect can be seen in Figure 1 where the currently accepted state-of-art method for object detection fails when presented with artwork. The same figure highlights our contribution by showing our proposal is able to locate (and classify) objects regardless of their depictive style.

The remainder of this paper first outlines the relevant background (Sec. 2), showing that our problem is hardly studied, but that relevant prior art exists for us to build upon. Sec. 3 describes our modeling schema, and in particular introduces the way in which we account for the wide variation in feature distributions, specifically - ***the use of multi-labels to represent visual words that exists in possibly discontinuous regions of a feature space.*** A visual class model (VCM) is now a graph with *multi-labeled nodes* and *learned weights*. Such novel visual class models can be learned from examples via an efficient algorithm we have designed (Sec. 4), and experimentally (Sec. 5) are shown to outperform state-of-art deformable part models at detection tasks, and state-of-art BoW methods for classification. The paper concludes, in Sec. 6, and points to future developments and applications.

2 Related Work

Modeling visual object classes is an interesting open question of relevance to many important computer vision tasks such as object detection and classification. Of the many approaches to visual object classification, the bag-of-words (BoW) family [7, 19, 23, 22] is arguably the most popular and successful. It models visual object classes via histograms of "visual words", *i.e.* words being clusters in feature space. Although the BoW methods address many difficult issues, they tend to generalise poorly across depictive styles. The explanation for this is the formation of visual codewords in which clustering assumes low variation in feature appearance. To overcome this drawback, researchers use alternative low-level features that do not depend on photometric appearance, *e.g.*, edgelets [26, 12] and region shapes [15, 17]. However, even these methods do not generalise well. We argue that no single "monolithic" feature will cover all possible appearances of an object (or part), when depictive styles are considered. Rather, we expand the variation of a local feature appearance from different depiction sources by multi-labelling model graph nodes.

Deformable models of various types are widely used to model the object for detection tasks, including several kinds of deformable template models [4, 5] and a variety of part-based models [1, 6, 9–11, 13, 20]. In the constellation models from [11], parts are constrained to be in a sparse set of locations, and their geometric arrangement is captured by a Gaussian distribution. In contrast, pictorial structure models [9, 10, 13] define a matching problem where parts have an individual match cost in a dense set of locations, and their geometric arrangement is captured by a set of 'springs' connecting pairs of parts. In those methods, the Deformable Part-based Model (DPM) [10] is the most successful one. It describes an object detection system based on mixtures of multi-scale deformable part models plus a root model. By modeling objects from different views with distinct models, it is able to detect large variations in pose. However, when the variance comes from local parts, *e.g.* the same object depicted in different styles, it does not generalize well; this is exactly the problem we address.

Cross-depiction problems are comparably less well-explored. Edge-based HoG was explored in [16] to retrieve photographs with a hand sketch query. Li *et al* [21] present a method for the representation and matching of sketches by exploiting not only local features but also global structures, through a star graph. Matching visually similar images has been addressed using self-similarity descriptors [25], and learning the most discriminant regions with exemplar SVM is also capable of cross-depiction matching [27]. These methods worked well for matching visually similar images, but neither are capable of modeling object categories with high diversity. The work most similar to own in motivation and method is a graph based approach proposed in [32]. They use a hierarchical graph model to obtain a coarse-to-fine arrangement of parts, whereas we use a single layer. They use qualitative shape as node label; we use multiple labels, each a HOG features.

In summary, the problem of cross-depiction classification is little studied. We learn a graph with multi-labeled nodes and employ a learned weight vector to encode the importance of nodes and edges similarities. Such a model is unique as

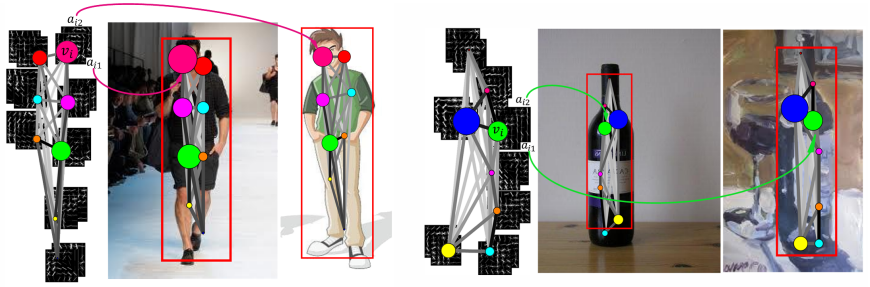

Fig. 2. Our multi-labeled graph model with learned discriminative weights, and detections for both photos and artworks. The model graph nodes are multi-labeled by attributes learned from different depiction styles (feature patches behind the nodes in the figure). The learned weight vector encodes the importance of the nodes and edges. In the figure, bigger circles represent stronger nodes, and darker lines denote stronger edges. And the same color of the nodes indicates the matched parts.

far as we know. We now describe the class model in greater detail: the formulation of the model, how to learn it, and its value to the problem of cross-depiction detection and classification.

3 Models

Our model of a visual object class is based around a graph of nodes and edges. Like Felzenszwalb *et al* [10], we label nodes with descriptions of object parts, but we differ in two ways. Unlike them, we label parts with multiple attributes, to allow for cross-depiction variation. Second, we differ in using a graph that defines the spatial relationship between node pairs using edge labels, rather than a star-like structure in which nodes are attached to a root. Furthermore, we place weights on the graph which are automatically learned using a method due to [3]. These weights can be interpreted as encoding relative salience. Thus a weighted, multi-labeled graph describes objects as seen from a single viewpoint. To account for variation in points of view we follow [10, 14, 8] who advocate using distinct models for each pose. They refer to each such model as a *component*, a term we borrow in this paper and which should not be confused with the *part* of an object.

We solve the problem of inter-depictive variation by using *multi-labeled* nodes to describe objects parts. These multiple attributes are learned from different depictive styles of images, which are more effective than attempting to characterize all attributes in a monolithic model, since the variation of local feature is much wider than the changes usually considered for photographic images, such as lighting changes *etc*. Moreover, it does not make sense that the parts of an object should be weighted equally during the matching for a part-based model. For example, for a person model, the head part should be weighted more than other parts like limbs and torso, because it is more discriminative than other

parts in the matching - a person's arms are easily confused with a quadruped's forelimbs, but the head part's features are distinctive. Beside the discrimination of node appearance, the relative location, edges, should be also weighted according to its rigidity. For instance, the edges between the head and shoulder should be more rigid than the edges between two deformable arms. Hence, in our model, a weight vector β is learned automatically to encode the importance of node and edge similarity. We refer to it as the *discriminative weight* formulation for a part based model. This advantage will be demonstrated with evidence in the experimental section.

3.1 A Multi-labeled Graph Model

A *multi-labeled graph* is defined as $G^* = (V^*, E^*, A^*, B^*)$, where V^* represents a set of nodes, E^* a set of edges, A^* a set of multi-labeled attributes of the nodes and B^* a set of attributes of edges. Specifically, $V^* = \{v_1^*, v_2^*, ..., v_n^*\}$, n is the number of nodes. $E^* = \{e_{12}^*, ..., e_{ij}^*, ..., e_{n(n-1)^*}\}$ is the set of edges. $A^* = \{A_1^*, A_2^*, ..., A_n^*\}$ with each $A_i^* = \{a_{i1}^*, a_{i2}^*, ..., a_{ic_i}^*\}$ consists of c_i attributes. It is easy to see that a standard graph G is a special case of our defined *multi-labeled graph*, which restricts $c_i = 1$.

A visual object class model $M =< G^*, \beta >$ for an object with n parts is formally defined by a multi-labeled model graph G^* with n nodes and $n \times (n-1)$ directed edges. And the weight vector $\beta \in \mathcal{R}^{n^2 \times 1}$ encodes the importance of nodes and edges of the G^*. Both the model graph G^* and the weights vector β are learned from a set of labeled example graphs. Figure 2 shows two example models with their detections from different depictive style. The learning process depends on scoring and matching, so a description is deferred to Section 4.

We define a score function between a visual class model, G^*, and a putative object represented as a standard graph G, following [3]. The definition is such that the absence of the *VCM* in an image yields a very low score. Let Y be a binary assignment matrix $Y \in \{0,1\}^{n \times n'}$ which indicates the nodes correspondence between two graphs, where n and n' denote the number of nodes in G^* and G, respectively. If $v_i^* \in V^*$ matches $v_a \in V$, then $Y_{i,a} = 1$, and $Y_{i,a} = 0$ otherwise. The scoring function is defined as the sum of nodes similarities (which indicate the local appearance) and the edges similarities (which indicate the spatial structure of the objects) between the visual object class and the putative object.

$$S(G^*, G, Y) = \sum_{Y_{i,a}=1} S_V(A_i^*, a_a) + \sum_{\substack{Y_{i,a}=1 \\ Y_{j,b}=1}} S_E(b_{ij}^*, b_{ab}), \tag{1}$$

where, because we use multi-labels on nodes we define

$$S_V(A_i^*, a_a) = \max_{p \in \{1,2,...,c_i\}} S_A(a_{ip}^*, a_a), \tag{2}$$

with a_{ip}^*, the p_{th} attribute in $A_i^* = \{a_{i1}^*, a_{i2}^*, ..., a_{ip}^*, ...a_{ic_i}^*\}$, and S_A is the similarity measure between attributes.

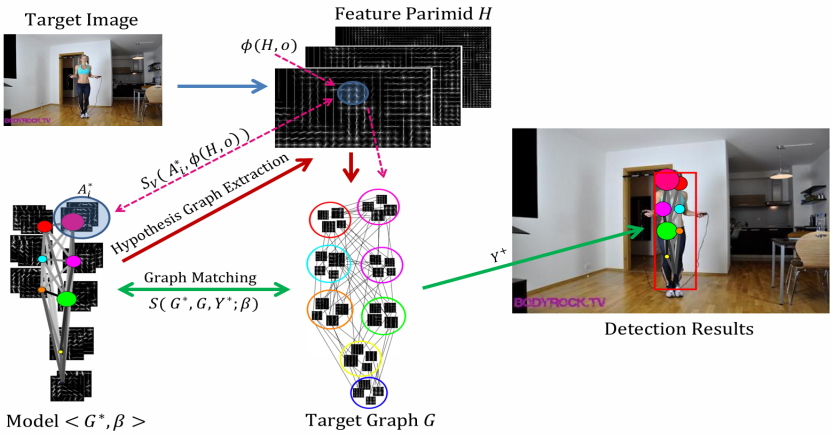

Fig. 3. Detection and matching process. A graph G will be firstly extracted from the target image based on input model $< G^*, \beta >$, then the matching process is formulated as a graph matching problem. The matched subgraph from G indicates the final detection results. $\phi(H, o)$ in the figure denotes the attributes obtained at position o.

To introduce the weight vector β into scoring, like [3], we parameterize Eq. 1 as follows. Let $\pi(i) = a$ denote an assignment of node v_i^* in G^* to node v_a in G, *i.e.* $Y_{i,a} = 1$. A joint feature map $\Phi(G^*, G, Y)$ is defined by aligning the relevant similarity values of Eq. 1 into a vectorial form as:

$$\Phi(G^*, G, Y) = [\cdots; S_V(A_i^*, a_{\pi(i)}); \cdots; S_E(b_{ij}^*, b_{\pi(i)\pi(j)}); \cdots]. \tag{3}$$

Then, by introducing weights on all elements of this feature map, we obtain a discriminative score function:

$$S(G^*, G, Y; \beta) = \beta \cdot \Phi(G^*, G, Y), \tag{4}$$

which is the score of a graph (extracted from the target image) with our proposed model $< G^*, \beta >$, under the assignment matrix Y.

3.2 Detection and Matching

To detect an instance of a visual class model (VCM) in an image we must find the standard graph in an image that best matches the given VCM. More exactly, we seek a subgraph of the graph G, constructed over a complete image, and is identified by the assignment matrix Y^+. We use an efficient approach to solve the problem of detection, which is stated as solving

$$Y^+ = \arg\max_{Y} S(G^*, G, Y; \beta), \tag{5a}$$

$$s.t. \sum_{i=1}^{n} Y_{i,a} \leq 1, \sum_{a=1}^{n'} Y_{i,a} \leq 1 \tag{5b}$$

where Eq.(5b) includes the matching constrains - only one node can match with at most one node in the other graph. To solve the NP-hard programming in Eq.5 efficiently, Torresani *et al.* [29] propose a decomposition approach for graph matching. The idea is to decompose the original problem into several simpler subproblems, for which a global maxima is efficiently computed. Combining the maxima from individual subproblems will then provide a maximum for the original problem. We make use of their general idea in an algorithm of our own design that efficiently locates graphs in images.

The graph G in Eq.(5a) is extracted from the target image as follows. First a dense multi-scale feature pyramid, H, is computed. Next a coarse-to-fine matching strategy is employed to locate each node of the *VCM* at k most possible locations in the image, based on the nodes similarity function S_V of Eq.(2). These possible locations are used to create a graph of the image. The 'image graph' is fully connected; corresponding features from H label the nodes; spatial attributes label the edges. This creates graph G.

Having found G the next step is to find the optimal subgraph by solving Eq. 5. During this step, we constrain the node v_i^* of the model graph G^* to be assigned (via Y) only to one of the k nodes it was associated with. In our experiments, to balance the matching accuracy and computational efficiency, we set $k = 10$. The optimal assignment matrix Y^+ between the model $< G^*, \beta >$ and the graph G, computed through Eq. (5), returns a detected subgraph of G that indicates the parts of the detected object. A detection and matching process is illustrated in Fig 3.

3.3 Mixture Models at Model Level

Our model also can be mixed using *components* as defined above and used in [10, 14, 8], so that different point of view (front/side) or poses (standing /sitting people) can be taken into account. A mixture model with m components is defined by a m-tuple, $\mathcal{M} = M_1, ..., M_m$, where $M_c =< G_c^*, \beta_c >$ is the multi-labeled *VCM* for the c-th component. To detect objects using a mixture model we use the matching algorithm described above to find the best matched subgraph that yields higher scoring hypothesis independently for each component.

4 Learning Models

Given images labeled with n interest points corresponding to n parts of the object, we consider learning a multi-labeled graph model G^* and weights β that together represent a visual class model. Because structure does not depend on fine-level details, we do not (nor should we) train an ssvm using depiction-specific features. The model learning framework is shown in Figure 4.

4.1 Learning the Model Graph G^*

For the convenience of description, consider a class-specific reference graph $G^{\triangle}$ (note that a reference graph is not created but is a mathematical convenience

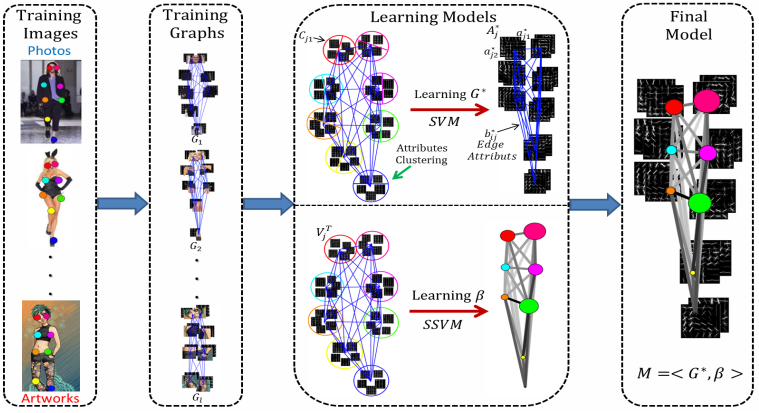

Fig. 4. Learning a class model, from left to right.(a): An input collection (different depictions) used for training. (b): Extract training graphs. (c): Learning models in two steps, one for G^*, one for β. (d): Combination as final class model

only, see [3] for details) and a labeled training graph set $T = (< G_1, y_1 >, ..., < G_l, y_l >)$ obtained from the labeled images. In each $< G_i, y_i > \in T$, we have n nodes, $n \times (n-1)$ edges and their corresponding attributes, defined as $G_i = (V_i, E_i, A_i, B_i)$, and y_i is an assignment matrix that denotes the matching between the training graph and the reference graph $G^{\triangle}$. Then, a sequence of nodes which match the same reference node $v_j^{\triangle} \in G^{\triangle}$ are collected over all the graphs in T. We define these nodes as $V_j^T = \{v_{j,1}^T, v_{j,2}^T, ..., v_{j,l}^T\}$ in which $v_{j,i}$ means the j-th node in training graph G_i. Then, the corresponding attributes set A_j^T can be extracted from the corresponding G_i to be used to learn the model graph G^* via the following process.

To learn a node V_j^* in the model graph G^*, there are l positive training nodes V_j^T with their attributes A_j^T. All the attributes in A_j^T are labeled according to depictive styles. Instead of manually labelling the style for each image, we use K-means clustering based on chi-square distance to build c_j clusters automatically, C_{ji} denotes the i-th cluster for A_j^T, and attributes in the same cluster indicate the similar depictive styles. Accordingly, the attributes A_j^* for the node $V_j^* \in G^*$ actually include c_j elements, $A_j^* = \{a_{j1}^*, a_{j2}^*, ..., a_{jc_j}^*\}$. For each a_{ji}^*, it is learned by minimizing the following objective function:

$$E(a_{ji}^*) = \frac{\lambda}{2}\|a_{ji}^*\|^2 + \frac{1}{N}\sum_{s=1}^{N} \max\{0, 1 - f(a_s) < a_{ji}^*, a_s >\} \tag{6}$$

from N example pairs $(a_s, f(a_s))$, $s = 1, ..., N$, where

$$f(a_s) = \begin{cases} 1 & \text{if } a_s \in C_{ji} \\ -1 & \text{if } a_s \in \mathcal{N}_j \end{cases} \tag{7}$$

where $\mathcal{N}_j$ is the negative sample sets for the node V_j^* and a_s is a node attributes from the training set. In our experiments, we use all the attributes that are in T

but do not belong to A_j^T, and the background patch attributes to build the negative samples set. Hence, this learning process transfers to an SVM optimization problem, which is solved by using stochastic gradient descent [28]. Edges E^* and corresponding attributes B^* also can be learned in a similar way. We account for different depictive styles by constructing a distinct SVM for each one; so in effect the multi-labeled nodes in G^* are in fact multiple SVMs.

4.2 Learning the Weights β

The aim of this step is to learn a weight vector β to produce best matches of the reference graph $G^{\triangle}$ with the training examples $T = (< G_1, y_1 >, ..., < G_l, y_l >)$ of the class. Let $\hat{y}$ denote the optimal matching between the reference graph $G^{\triangle}$ and a training graph $G_i \in T$ given by

$$\hat{y}(G_i; G^{\triangle}, \beta) = \underset{y \in Y(G_i)}{\arg\max}\, S(G^{\triangle}, G_i, y; \beta), \tag{8}$$

where $Y(G_i) \in \{0,1\}^{n \times n'}$ defines the set of possible assignment matrix for the input training graph G_i. Inspired by the max-margin framework [30] and following [3], we learn the parameter β by minimizing the following objective function:

$$L_T(G^{\triangle}, \beta) = r(G^{\triangle}, \beta) + \frac{C}{l} \sum_{i=1}^{l} \Delta(y_i, \hat{y}(G_i; G^{\triangle}), \beta). \tag{9}$$

In this objective function r is a regularization function, $\Delta(y, \hat{y})$ a loss function, drives the learning process by measuring the quality of a predicted matching $\hat{y}_i$ against its ground truth y_i. The parameter C controls the relative importance of the loss term.

Cho *et al.* [3] propose an effective framework to transform the learning objective function in Eq. (9) into a standard formulation of the structured support vector machine (SSVM) by assuming the node and edge similarity functions are dot products of two attributes vectors. It is solved by using the efficient cutting plane method proposed by Joachims *et al.* [18], giving us the weight vector β to encode the importance of nodes and edges.

4.3 Features

Node Attributes. In our proposed model, we used a 31-d Histogram of Oriented Gradients (HOG) descriptor, following [10], which computes both directed and undirected gradients as well as a four dimensional texture-energy feature.

Edge Attributes. Considering an edge e_{ij} from node v_i to node v_j with polar coordinates (ρ_{ij}, θ_{ij}). We convert these distances and orientations to histogram features so that it can be used within dot products as in [3]. Two histograms (one for the length L_{ij}, and one for the angle P_{ij}) are built and concatenated to quantize the edge vectors, $b_{i,j} = [L_{ij}; P_{ij}]$. For length, we use uniform bins of size n_L in the log space with respect to the position of v_i, making the histogram

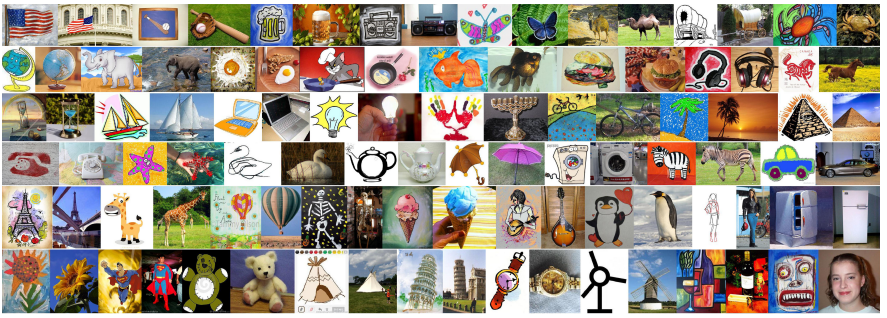

Fig. 5. Our photo-art dataset, containing 50 object categories. Each category is displayed with one art image and one photo image.

more sensitive to the position of nearby points. The log-distance histogram L_{ij} is constructed on the bins by a discrete Gaussian histogram centred on the bins for ρ_{ij}. For angle, we use uniform bins of size $2\pi/n_P$. The polar histogram P_{ij} is constructed on it in a similar way, except that a circular Gaussian histogram centered on the bin for $\theta_{i,j}$ is used. In this work, we used $n_L = 9, n_P = 18$.

5 Experimental Evaluation

Our class model has the potential to be used in many applications, here we demonstrate it in the task of cross-depiction detection and classification. Although there are several challenging object detection and classification datasets such as PASCAL VOC, ETHZ-shape classes and Caltech-256, most of the classes in these datasets do not contain objects that are depicted in different styles, such as painting, drawing and cartoons. Therefore, we augment photo images of 50 object categories, which frequently appear in commonly used datasets, to cover the large variety of art works. Each class contains around 100 images with different instances and approximately half of the images in each class are artworks and cover a wide gamut of style. Examples of each class are shown in figure 5.

5.1 Detection

In the detection task, we split the image set for each object class into two random partitions, 30 images for training (15 photos and 15 art) and the rest are used for testing. The dataset contains the groundtruth for each image in the form of bounding boxes around the objects. During the test, the goal is to predict the bounding boxes for a given object class in a target image (if any). The red bounding boxes in Fig. 1 are predicted in such way. In practice the detector outputs a set of bounding boxes with corresponding scores, and a precision-recall curve across all the test sets is obtained. We score a detector by the average precision (AP), which is defined as the area under the precision-recall curve across a test set, mAP(mean of the AP) is the average AP over all objects.

Fig. 6. Examples of high-scoring detections on our cross-depictive style dataset, selected from the top 20 highest scoring detections in each class. The framed images (last one in each row) illustrate false positives for each category. In each detected window, the object is matched with the learned model graph. In the matched graph, each node indicates a part of the object, and larger circles represent greater importance of a node, and darker lines denote stronger relationships.

Since our learning process (in Sec. 4) needs pre-labeled training graphs, n distinctive key-points have to be identified in the target images. In our experiment, we set $n = 8$. In order to ease the labelling process, rather than using

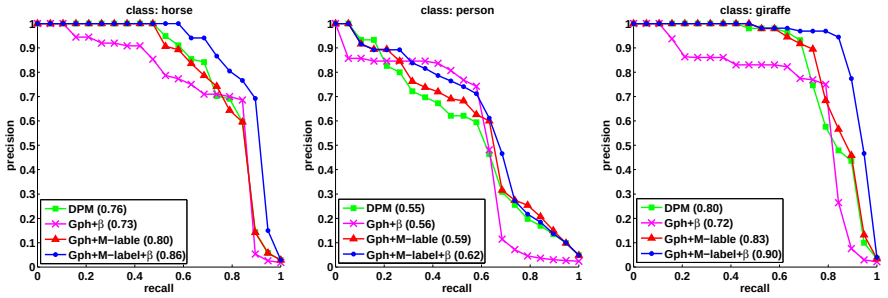

Fig. 7. Precision/Recall curves for models trained on the horse, person and giraffe categories of our cross-domain dataset. We show results for DPM, a single labeled graph model with learned β, our proposed multi-labeled model graph with and without learned β. In parenthesis we show the average precision score for each model.

the manually labeling process, we instead use a pre-trained DPM model to locate the object parts across the training set, as only an approximate location of the labeled parts is enough to build our initial model. This idea is borrowed from [34], which uses a pictorial structure [24] to estimate 15 key-points for the further learning of a 2.5D human action graph for matching. Also notice that DPM is only used to ease the training data labelling process, it is not used in our proposed learning and matching process. During the test process, we match each learned object class model with the hypothesis graph extracted from an input test image, as detailed illustrated in Sec 3.2. The detection score is computed via Eq. (5) and the predicted bounding box is obtained by covering all the matched nodes.

We trained a two component model, where the '*component*' is decided by the ground truth bounding box ratio as in DPM [10]. Each node in the model is multi-labeled by two labels (split automatically by *K-means* as illustrated in Sec. 4.1), that correspond to the attributes of the photo and art domains. Figure 6 shows some detections we obtain using our learned models. These results show that the proposed model can detect objects correctly across different depictive styles, including photos, oil paintings, children's drawings, stick-figures and cartoons. Moreover, the detected object parts are labeled by the graph nodes, and larger circles represent more important nodes, which are weighted more during the matching process, via β.

We evaluated different aspects of our system and compared them with a state-of-art method, DPM [10], which is a star-structured part-based model defined by a 'root' filter plus a set of parts filters. A two component DPM model is trained for each class following the setting of [10]. To evaluate the contribution of the mixture model and the importance of the weight β, we also implemented other two methods, multi-labeled graph without weight (Graph+M-label) and single-labeled graph with weight (Graph+β). The weight β can not be used on the DPM model, because it encodes no direct relation between nodes under the root.

	us-flag	bat	beer-mug	boom-box	butterfly	camel	wagon	crab	globe	eiffel-tower	elephant	fried-egg	frying-pan	giraffe	goldfish	hamburger	head-phones
DPM[10]	.871	.338	**.956**	.859	.718	.797	.897	.640	.961	.952	.853	.618	.757	.803	.901	.809	.808
G+β	.743	.343	.729	.683	.375	.420	.662	.633	.846	.798	.538	.627	.733	.716	.556	.650	.669
G+ml	.913	.423	.947	.913	.757	.816	.914	.723	.932	.971	.875	.778	.812	.831	.913	.883	.853
G+ml+β	**.917**	**.456**	.954	**.929**	**.839**	**.881**	**.942**	**.741**	**.993**	**.981**	**.887**	**.851**	**.838**	**.907**	**.962**	**.908**	**.879**

	horse	balloon	hourglass	skeleton	ice-cream	ketch	laptop	lightbulb	mandolin	menorah	bike	palm-tree	penguin	person	pyramid	refrigerator	rotary-phone
DPM[10]	.764	**.955**	.925	**.985**	.816	.905	.926	.751	.385	.882	.975	.899	.735	.554	**.840**	.731	.900
G+β	.733	.755	.881	.926	**.930**	.718	.780	.705	.487	.879	.933	.794	.599	.555	.719	.561	.713
G+ml	.799	.899	.953	.956	.894	.929	.954	.725	.415	.901	.996	.921	.742	.587	.760	.788	.916
G+ml+β	**.860**	.930	**.956**	.968	.911	**.976**	**.964**	**.807**	**.491**	**.933**	**.997**	**.936**	**.825**	**.616**	.808	**.820**	**.928**

	starfish	sunflower	superman	swan	teapot	teddy	teepee	tower-pisa	umbrella	wash-machine	watch	windmill	bottle	zebra	car	face	**mAP**
DPM[10]	.922	.881	.720	**.923**	.968	.962	.807	.964	**.924**	.957	.736	.889	.892	.914	.948	**.906**	.835
G+β	.713	.766	.584	.723	.632	.814	.872	.849	.724	.974	.744	.739	.639	.751	.851	.790	.711
G+ml	.942	.903	.758	.859	.992	.985	.922	.981	.794	.982	.774	.899	.911	.961	.965	.886	.858
G+ml+β	**.965**	**.923**	**.791**	.915	**.993**	**.991**	**.944**	**.991**	.898	**.985**	**.794**	**.965**	**.956**	**.973**	**.974**	.899	**.891**

Table 1. Detection results on our cross-depictive style dataset (50 classes in total): average precision scores for each class of different methods, DPM, a single labeled graph model with learned β, our proposed multi-labeled model graph with and without learned β. The mAP (mean of average precision) is shown in the last column.

Table 1 compares the detection results of using different models on our dataset. Our system achieves the best AP scores in 42 out of the 50 categories. DPM wins 7 times. Furthermore, our final mAP (.891) outperforms DPM (.835) by more than 5%. Figure 7 summarizes the results of different models applied on the person, horse and giraffe categories, chosen because these object classes appear commonly in many well-known detection datasets. The PR-curve of other classes can be found in the supplementary material. We see that the use of our multi-labeled graph model can significantly improve the detect accuracy. Further improvements are obtained by using discriminative weights β.

Our system is implemented by matlab, running on a Core i7 CPU@2.67GHz$\times$8 machine. The average training time for a single class is 4 to 5 minutes (parts labelling process is not included). The average testing time of a single image is 4.5 to 5 minutes, since the graph matching takes long time.

5.2 Classification

Our proposed model can also be adapted for classification. Training requires of learning a class model, exactly the same procedure as in the previous section. The testing process determines the class by choosing the class which has the best matching score with the query image.

Using our dataset we conduct experiments designed to test how well our proposed class model generalised across depictive styles. Like the detection experiments, we randomly split the image set for each object class into two partitions, 30 images for training (15 photos and 15 artworks) and the rest are used for test-

Methods	Art	Photos
BoW[31]	69.47 ± 1.1	80.38 ± 1.1
DPM[10]	80.29 ± 0.9	85.22 ± 0.6
Our	$\mathbf{89.06 \pm 1.2}$	$\mathbf{90.29 \pm 1.3}$

Table 2. Comparison of classification results for different test cases and methods.

ing. Unlike from the detection task, we test on photos and artworks separately to compare the performance on these two domains. The classification accuracy is determined as the average over 5 random splits.

For comparison with alternative visual class models we compare with two other methods: BoW and DPM. BoW classifier is chosen because it performs well and will help us assess the performance of such a popular approach to the problem of cross-depiction classification. We follow Vedaldi *et al* [31] using dense-sift features [2] and K-means ($K = 1000$) for visual word dictionary construction. Finally, it uses a SVM for classification. The second is the DPM [10], adapted to classification.

Classification accuracy of different methods in various testing cases, are shown in table 2. It shows that our method outperforms the BoW and DPM method in all cases, especially when the test set are artworks only. Our multi-labeled modelling method effectively train nodes of the graph in separately depictive styles and then combine them in a mixture model to global optimization. Experimental results clearly indicate that our mixture model outperforms state of the art methods which attempt to characterize all depiction styles in a monolithic model. We also made tests on some of the cross-domain literature we cited such as [25, 32] and a method that is not depend on photometric appearance, using the edgelets [12]. A mixture-of-parts method [33] is also tested. But none of them work well on such a high-variety depiction dataset. We report DPM and BoW only because they consistently out-perform those methods.

6 Conclusion

There is a deep appeal in not discriminating between depictive styles, but instead considering images in any style, not just because it echoes an impressive human ability but also because it opens new applications. Our paper provides evidence that multi-label nodes are useful representations in coping with features that exhibit very wide, possibly discontinuous distributions. There is no reason to believe that such distributions are confined to the problem of local feature representation in art and photographs; it could be an issue in many cross-domain cases. For the future work, we want to more fully investigate the way in which the distribution of the description of a single object part is represented.

Acknowledgements

We thank the EPSRC for supporting this work through grant EP/K015966/1.

References

1. Amit, Y., Trouv, A.: Pop: Patchwork of parts models for object recognition. IJCV (2004)
2. Bosch, A., Zisserman, A., Muoz, X.: Image classification using random forests and ferns. In: ICCV (2007)
3. Cho, M., Alahari, K., Ponce, J.: Learning graphs to match. In: ICCV (2013)
4. Cootes, T.F., Edwards, G.J., Taylor, C.J., et al.: Active appearance models. TPAMI (2001)
5. Coughlan, J., Yuille, A., English, C., Snow, D.: Efficient deformable template detection and localization without user initialization. CVIU (2000)
6. Crandall, D., Felzenszwalb, P., Huttenlocher, D.: Spatial priors for part-based recognition using statistical models. In: CVPR (2005)
7. Csurka, G., Dance, C.R., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: ECCV (2004)
8. Dong, J., Xia, W., Chen, Q., Feng, J., Huang, Z., Yan, S.: Subcategory-aware object classification. In: CVPR (2013)
9. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV (2005)
10. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. TPAMI (2010)
11. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: CVPR (2003)
12. Ferrari, V., Jurie, F., Schmid, C.: From images to shape models for object detection. IJCV (2010)
13. Fischler, M.A., Elschlager, R.: The representation and matching of pictorial structures. Computers, IEEE Transactions on (1973)
14. Gu, C., Arbelaez, P., Lin, Y., Yu, K., Malik, J.: Multi-component models for object detection. In: ECCV (2012)
15. Gu, C., Lim, J.J., Arbeláez, P., Malik, J.: Recognition using regions. In: CVRP (2009)
16. Hu, R., Collomosse, J.: A performance evaluation of gradient field hog descriptor for sketch based image retrieval. CVIU (2013)
17. Jia, W., McKenna, S.: Classifying textile designs using bags of shapes. In: ICPR (2010)
18. Joachims, T., Finley, T., Yu, C.N.J.: Cutting-plane training of structural svms. Machine Learning (2009)
19. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR (2006)
20. Leibe, B., Leonardis, A., Schiele, B.: Robust object detection with interleaved categorization and segmentation. IJCV (2008)
21. Li, Y., Song, Y.Z., Gong, S.: Sketch recognition by ensemble matching of structured features. In: BMVC (2013)
22. Perronnin, F., Snchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: ECCV (2010)
23. Russakovsky, O., Lin, Y., Yu, K., Fei-Fei, L.: Object-centric spatial pooling for image classification. In: ECCV (2012)
24. Sapp, B., Toshev, A., Taskar, B.: Cascaded models for articulated pose estimation. In: ECCV (2010)

25. Shechtman, E., Irani, M.: Matching local self-similarities across images and videos. In: CVPR (2007)
26. Shotton, J., Blake, A., Cipolla, R.: Multiscale categorical object recognition using contour fragments. TPAMI (2008)
27. Shrivastava, A., Malisiewicz, T., Gupta, A., Efros, A.A.: Data-driven visual similarity for cross-domain image matching. ACM Transaction of Graphics (TOG) (2011)
28. Singer, Y., Srebro, N.: Pegasos: Primal estimated sub-gradient solver for svm. In: ICML (2007)
29. Torresani, L., Kolmogorov, V., Rother, C.: Feature correspondence via graph matching: Models and global optimization. In: ECCV (2008)
30. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. JMLR (2005)
31. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms (2008)
32. Wu, Q., Hall, P.: Modelling visual objects invariant to depictive style. In: BMVC (2013)
33. Yang, Y., Ramanan, D.: Articulated pose estimation with flexible mixtures-of-parts. In: CVPR (2011)
34. Yao, B., Fei-Fei, L.: Action recognition with exemplar based 2.5d graph matching. In: ECCV (2012)