SketchParse : Towards Rich Descriptions for Poorly Drawn Sketches using Multi-Task Hierarchical Deep Networks

Ravi Kiran Sarvadevabhatla*
ravika@gmail.com

Isht Dwivedi
isht.dwivedi@gmail.com

Abhijat Biswas
abhijatbiswas@gmail.com

Sahil Manocha
sahil.manocha1995@gmail.com

Venkatesh Babu R.
venky@cds.iisc.ac.in

ABSTRACT

The ability to semantically interpret hand-drawn line sketches, although very challenging, can pave way for novel applications in multimedia. We propose SketchParse, the first deep-network architecture for fully automatic parsing of freehand object sketches. SketchParse is configured as a two-level fully convolutional network. The first level contains shared layers common to all object categories. The second level contains a number of expert sub-networks. Each expert specializes in parsing sketches from object categories which contain structurally similar parts. Effectively, the two-level configuration enables our architecture to scale up efficiently as additional categories are added. We introduce a router layer which (i) relays sketch features from shared layers to the correct expert (ii) eliminates the need to manually specify object category during inference. To bypass laborious part-level annotation, we sketchify photos from semantic object-part image datasets and use them for training. Our architecture also incorporates object pose prediction as a novel auxiliary task which boosts overall performance while providing supplementary information regarding the sketch. We demonstrate SketchParse's abilities (i) on two challenging large-scale sketch datasets (ii) in parsing unseen, semantically related object categories (iii) in improving fine-grained sketch-based image retrieval. As a novel application, we also outline how SketchParse's output can be used to generate caption-style descriptions for hand-drawn sketches.

CCS CONCEPTS

•**Computing methodologies** →**Image segmentation;** *Shape representations; Transfer learning;*

KEYWORDS

object segmentation,sketch,transfer learning, deep learning, multi-task learning

1 INTRODUCTION

Hand-drawn line sketches have long been employed to communicate ideas in a minimal yet understandable manner. In this paper, we explore the problem of parsing sketched objects, i.e. given a freehand line sketch of an object, determine its salient attributes (e.g. category, semantic parts, pose). The ability to understand sketches in terms of local (e.g. parts) and global attributes (e.g. pose) can drive novel applications such as sketch captioning, storyboard animation [16] and automatic drawing assessment apps for art teachers. The onset of deep network era has resulted in architectures which can impressively recognize object sketches at a coarse (category) level [34, 37, 48]. Paralleling the advances in parsing of photographic objects [13, 20, 43] and scenes [4, 8, 27], the time is ripe for understanding sketches too at a fine-grained level [19, 47].

A number of unique challenges need to be addressed for semantic sketch parsing. Unlike richly detailed color photos, line sketches are binary (black and white) and sparsely detailed. Sketches exhibit a large amount of appearance variations induced by the range of drawing skills among general public. The resulting distortions in object depiction pose a challenge to parsing approaches. In many instances, the sketch is not drawn with a 'closed' object boundary, complicating annotation, part-segmentation and pose estimation. Given all these challenges, it is no surprise that only a handful of works exist for sketch parsing [15, 36]. However, even these approaches have their own share of drawbacks (Section 2).

To address these issues, we propose a novel architecture called SketchParse for fully automatic sketch object parsing. In our approach, we make three major design decisions:

Design Decision #1 (Data): To bypass burdensome part-level sketch annotation, we leverage photo image datasets containing part-level annotations of objects [6]. Suppose I is an object image and C_I is the corresponding part-level annotation. We subject I to a sketchification procedure (Section 3.2) and obtain S_I. Thus, our training data consists of the sketchified image and corresponding part-level annotation pairs ($\{S_I, C_I\}$) for each category (Figure 1).

Design Decision #2 (Model): Many structurally similar object categories tend to have common parts. For instance, 'wings' and 'tail' are common to both birds and airplanes. To exploit such shared semantic parts, we design our model as a two-level network of disjoint experts (see Figure 2). The first level contains shared layers common to all object categories. The second level contains a number of experts (sub-networks). Each expert is configured for parsing sketches from a super-category set comprising of categories

*This author and co-authors are all affiliated with Video Analytics Lab, Indian Institute of Science, Bangalore, INDIA 560012.

MM '17, October 23–27, 2017, Mountain View, CA, USA.
 ISBN 978-1-4503-4906-2/17/10…$15.00
DOI: https://doi.org/10.1145/3123266.3123270

with structurally similar parts[1]. Instead of training from scratch, we instantiate our model using two disjoint groups of pre-trained layers from a scene parsing net (Section 4.1). We perform training using the sketchified data (Section 5.1) mentioned above. At test time, the input sketch is first processed by the shared layers to obtain intermediate features. In parallel, the sketch is also provided to a super-category sketch classifier. The label output of the classifier is used to automatically route the intermediate features to the appropriate super-category expert for final output i.e. part-level segmentation (Section 5.2).

Design Decision #3 (Auxiliary Tasks): A popular paradigm to improve performance of the main task is to have additional yet related auxiliary targets in a multi-task setting [1, 17, 24, 30]. Motivated by this observation, we configure each expert network for the novel auxiliary task of 2-D pose estimation.

At first glance, our approach seems infeasible. After all, sketchified training images resemble actual sketches only in terms of stroke density (see Figure 1). They seem to lack the fluidity and unstructured feel of hand-drawn sketches. Moreover, SketchParse's base model [5], originally designed for photo *scene* segmentation, seems an unlikely candidate for enabling transfer-learning based sketch *object* segmentation. Yet, as we shall see, our design choices result in an architecture which is able to successfully accomplish sketch parsing across multiple categories and sketch datasets.

Contributions:

- We propose SketchParse – the first deep hierarchical network for fully automatic parsing of hand-drawn object sketches (Section 4). Our architecture includes object pose estimation as a novel auxiliary task.
- We provide the largest dataset of part-annotated object sketches across multiple categories and multiple sketch datasets. We also provide 2-D pose annotations for these sketches.
- We demonstrate SketchParse's abilities on two challenging large-scale sketch object datasets (Section 5.2), on unseen semantically related categories, (Section 6.1) and for improving fine-grained sketch-based image retrieval (Section 6.2).
- We outline how SketchParse's output can form the basis for novel applications such as automatic sketch description (Section 6.2).

2 RELATED WORK

Semantic Parsing (Photos): Existing deep-learning approaches for semantic parsing of photos can be categorized into two groups. The first group consists of approaches for scene-level semantic parsing (i.e. output an object label for each pixel in the scene) [5, 8, 27]. The second group of approaches attempt semantic parsing of objects (i.e. output a part label for each object pixel) [13, 20, 43]. Compared to our choice (of a scene parsing net), this latter group of object-parsing approaches seemingly appear better candidates for the base architecture. However, they pose specific difficulties for adoption. For instance, the hypercolumns approach of Hariharan et al. [13] requires training separate part classifiers for each class. Also, the evaluation is confined to a small number of classes (animals and human beings). The approach of Liang et al. [20] consists of a complex multi-stage hybrid CNN-RNN architecture evaluated on only two animal categories (`horse` and `cow`). The approach of Xia et al. [43] fuses object part score evidence from scene, object and part level to obtain impressive results for two categories (human, large animals). However, it is not clear how their method can be adopted for our purpose.

[1]For example, categories `cat`, `dog`, `sheep` comprise the super-category *Small Animals*.

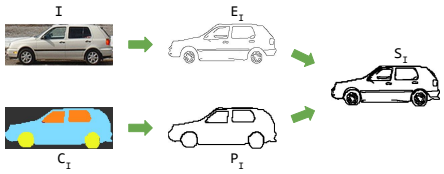

Figure 1: An illustration of our sketchification procedure (Section 3.2). The edge image E_I, corresponding to the input photo I, is merged with the part and object contours P_I derived from ground-truth labeling C_I, to obtain the final sketchified image S_I. We train SketchParse using S_I instances as inputs and C_I instances as the corresponding part-labelings.

Semantic Object Parsing (Sketches): Only a handful of works exist for sketch parsing [15, 36]. Existing approaches require a part-annotated dataset of sketch objects. Obtaining such part-level annotations is very laborious and cumbersome. Moreover, one-third of a dataset [15] evaluated by these approaches consists of sketches drawn by professional artists. Given the artists' relatively better drawing skills, incorporating such sketches artificially reduces the complexity of the problem. On the architectural front, the approaches involve tweaking of multiple parameters in a hand-crafted segmentation pipeline. Existing approaches label individual sketch strokes as parts. This requires strokes within part interiors to be necessarily labelled, which can result in peculiar segmentation errors [36]. Our method, in contrast, labels object regions as parts. In many instances, the object region boundary consists of non-stroke pixels. Therefore, it is not possible to directly compare with existing approaches. Unlike our category-scalable and fully automatic approach, these methods assume object category is known and train a separate model per category (E.g. `dog` and `cat` require separate models). Existing implementations of these approaches also have prohibitive inference time – parsing an object sketch takes anywhere from 2 minutes [36] to 40 minutes [15], rendering them unsuitable for interactive sketch-based applications. In contrast, our model's inference time is fraction of a *second*. Also, our scale of evaluation is significantly larger. For example, Schneider et al.'s method [36] is evaluated on 5 test sketches per category. Our model is evaluated on 100 sketches per category. Finally, none of the previous approaches exploit the hierarchical category-level groupings which arise naturally from structural similarities [51]. This renders them prone to drop in performance as additional categories (and their parts) are added.

Sketch Recognition: The initial performance of handcrafted feature-based approaches [35] for sketch recognition has been surpassed in recent times by deep CNN architectures [37, 48]. The sketch

router classifier in our architecture is a modified version of Yang et al.'s Sketch-a-Net [48]. While the works mentioned above use sketches, Zhang et al. [50] use sketchified photos for training the CNN classifier which outputs class labels. We too use sketchified photos for training. However, the task in our case is parsing and not classification.

Class-hierarchical CNNs: Our idea of having an initial coarse-category net which routes the input to finer-category experts can be found in some recent works as well [2, 45], albeit for object classification. In these works, coarse-category net is intimately tied to the main task (viz. classification). In our case, the coarse-category CNN classifier serves a secondary role, helping to route the output of a parallel, shared sub-network to the finer-category parsing experts. Also, unlike above works, the task of our secondary net (classification) is different from the task of experts (segmentation).

Domain Adaptation/Transfer Learning: Our approach can be viewed as belonging to the category of domain adaptation techniques [3, 28, 32]. These techniques have proven to be successful for various problems, including image parsing [14, 31, 39]. However, unlike most approaches wherein image modality does not change, our domain-adaptation scenario is characterized by extreme modality-level variation between source (image) and target (freehand sketch). This drastically reduces the quantity and quality of data available for transfer learning, making our task more challenging.

Multi-task networks: The effectiveness of addressing auxiliary tasks in tandem with the main task has been shown for several challenging problems in vision [1, 17, 24, 30]. In particular, object classification [26], detection [8], geometric context [40], saliency [18] and adversarial loss [23] have been utilized as auxiliary tasks in deep network-based approaches for semantic parsing. The auxiliary task we employ – object viewpoint estimation – has been used in a multi-task setting but for object classification [11, 52]. To the best of our knowledge, we are the first ones to design a custom pose estimation architecture to assist semantic parsing.

3 DATA PREPARATION

We first summarize salient details of two semantic object-part photo datasets.

3.1 Object photo datasets

PASCAL-Parts: This 10,103 image dataset [41] provides semantic part segmentation of objects from the 20 object categories from the PASCAL VOC2010 dataset. We select 11 categories (`aeroplane`, `bicycle`, `bus`, `car`, `cat`, `cow`, `dog`, `flying bird`, `horse`, `motorcycle`, `sheep`) for our experiments. To obtain the cropped object images, we used object bounding box annotations from PASCAL-parts.

CORE: The Cross-category Object REcognition (CORE) dataset [12] contains segmentation and attribute information for objects in 2800 images distributed across 28 categories of vehicles and animals. We select 8 categories from CORE dataset based on their semantic similarity with PASCAL-part categories (e.g. CORE category `crow` is selected since it is semantically similar to the PASCAL-part category `bird`).

To enable estimation of object pose as an auxiliary objective, we annotated all the images from PASCAL-Parts and CORE datasets with 2-D pose information based on the object's orientation with respect to the viewing plane. Specifically, each image is labeled with one of the cardinal ('North','East', 'West','South') and intercardinal directions ('NE','NW','SE','SW') [42]. We plan to release these pose annotations publicly for the benefit of multimedia community.

Next, we shall describe the procedure for obtaining sketchified versions of photos sourced from these datasets.

3.2 Obtaining sketchified images

Suppose I is an object image. As the first step, we use a Canny edge detector tuned to produce only the most prominent object edges. Visually, we found the resulting image E_I to contain edge structures which perceptually resemble human sketch strokes compared to those produced by alternatives such as Sketch Tokens [21] and SCG [44]. We augment E_I with part contours and object contours from the part-annotation data of I and perform morphological dilation to thicken edge segments (using a square structured element of side 3) to obtain the sketchified image S_I (see Figure 1). To augment data available for training the segmentation model, we apply a series of rotations (±10, ±20, ±30 degrees) and mirroring about the vertical axis to S_I. Overall, this procedure results in 14 augmented images per each original, sketchified image. To ensure a good coverage of parts and eliminate inconsistent labelings, we manually curated 1532 object images from PASCAL-Parts and CORE datasets. Given the varying semantic granularity of part labels across these datasets, we manually curated the parts considered for each category [41]. Finally, we obtain the training dataset consisting of $1532 \times 14 = 21{,}448$ paired sketchified images and corresponding part-level annotations, distributed across 11 object categories.

We evaluate our model's performance on freehand line sketches from two large-scale datasets. We describe these datasets and associated data preparation procedures next.

3.3 Sketch datasets and augmentation

TU-Berlin: The TU-Berlin sketch database [10] contains 20,000 hand drawn sketches spanning 250 common object categories, with 80 sketches per object. For this dataset, only the category name was provided to the sketchers during the drawing phase.

Sketchy: The Sketchy database [33] contains 75,471 sketches spanning 125 object categories, with $500-700$ sketches per category. To collect this dataset, photo images of objects were initially shown to human subjects. After a gap of 2 seconds, the image was replaced by a gray screen and subjects were asked to sketch the object from memory. Compared to the draw-from-category-name-only approach employed for TU-Berlin dataset [10], this memory-based approach provides a larger variety in terms of multiple viewpoints and object detail in sketches. On an average, each object photo is typically associated with $5-8$ different sketches.

From both the datasets, we use sketches from only those object categories which overlap with the 11 categories from PASCAL-Parts mentioned in Section 3.1. For augmentation, we first apply morphological dilation (using a square structured element of side 3) to each sketch. This operation helps minimize the impact of loss in stroke continuity when the sketch is processed by deeper layers of

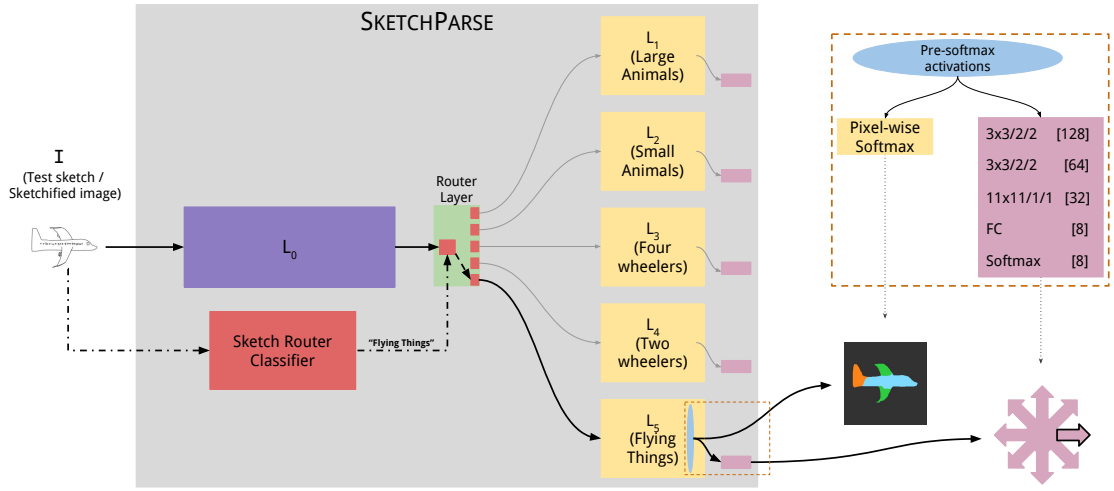

Figure 2: The first level L_0 of SketchParse (shaded purple) is instantiated with shallower layers of a scene parsing net. The second level consists of K expert super-category nets $L_1, L_2, \ldots L_K$ (shaded yellow) and is instantiated with deeper layers of the scene parsing net. Given the test sketch I, the Router Layer (shaded green) relays intermediate features produced by shared layers L_0 to the target expert (L_5). The pre-softmax activations (blue oval) generated in L_5 are used to obtain the part parsing. These activations are also used by our novel pose estimation auxiliary net (shaded light-pink). The architecture of the pose net can be seen within the brown dotted line box in top-right. Within the pose net, convolutional layers are specified in the format dimensions/stride/dilation [number of filters]. FC = Fully-Connected layer. The dash-dotted line connected to the router classifier is used to indicate that Router Layer is utilized only during inference.

the network. Subsequently, we apply a series of rotations (±10, ±20, ±30 degrees) and mirroring about the vertical axis on the dilated sketch. This produces 14 augmented variants per original sketch for use during training.

4 OUR MODEL (SKETCHPARSE)

4.1 Instantiating SketchParse levels

We design a two-level deep network architecture for SketchParse (see Figure 2). The first level is intended to capture category-agnostic low-level information and contains shared layers L_0 common to all N object categories. We instantiate first-level layers with the shallower, initial layers from a scene parsing net [5]. In this context, we wish to emphasize that our design is general and can accommodate any fully convolutional scene parsing network.

The categories are grouped into K smaller ($< N$), disjoint super-category subsets using meronym (i.e. part-of relation)-based similarities between objects [38]. For example, `dog,cat,sheep` are all grouped into the set *Small Animals*. The second level consists of K expert sub-networks $L_1, L_2, \ldots L_K$, each of which is specialized for parsing sketches associated with a super-category. We initialize these K experts using the deeper, latter layers from the scene parsing model. Suppose the total number of parts across all object categories within the i-th super-category is $n_i, 1 \leqslant i \leqslant K$. We modify the final layer for each expert network L_i such that it outputs n_i part-label predictions. In our current version of the architecture, $K = 5$ and $N = 11$. We performed ablative experiments to determine optimal location for splitting the layers of semantic scene parsing net into two groups. Based on these experiments, we use all the layers up to the *res5b* block as shared layers.

4.2 Router Layer

From the above description (Section 4.1), SketchParse's design so far consists of a single shared sub-network and K expert nets. We require a mechanism for routing the intermediate features produced by shared layers to the target expert network. In addition, we require a mechanism for backpropagating error gradients from the K expert nets and update the weights of the shared layer sub-network during training. To meet such requirements, we design a Router Layer (shaded green in Figure 2). During the training phase, routing of features from the shared layer is dictated by ground-truth category and by extension, the super-category it is associated with. A branch index array is maintained for each training mini-batch. Since the ground-truth super-category for each training example is available, creation of branch index array requires only knowledge of the mini-batch label composition. The array entries are referenced during backward propagation to (a) recombine the gradient tensors in the same order as that of the mini-batch and (b) route error gradient from the appropriate branch to the shared layers during backpropagation.

To accomplish routing during test time, we use a K-way classifier (shaded red in Figure 2) whose output label corresponds to one of the K expert networks $L_1, L_2, \ldots L_K$. In this regard, we experimented with a variety of deep CNN architectures. Our initial attempts involved training custom CNNs solely on sketchified images or their deep feature variants. However, the classification performance was subpar. Therefore, we resorted to training the classifier using actual sketches. Our experiments show that fine-tuning a modified version of Sketch-a-Net [48], a CNN originally custom-designed

Architecture	aIOU %
Baseline (B)	66.99
B + Class-Balanced Loss Weighting (C)	69.56
B + C + Pose Auxiliary Task (P)	**70.26**

Table 1: Performance for architectural additions to a single super-category ('Large Animals') version of SketchParse.

Directions	Large Animals	Small Animals	4-Wheelers	2-Wheelers	Flying Things	AVG.
8	80.32	53.92	54.92	59.89	44.73	**58.45**
4	92.07	76.78	69.72	87.16	75.79	**80.44**

Table 2: Performance of our best SketchParse model (BCP-R5) on the pose auxiliary task.

for sketch classification, provides the most accurate classification (routing) performance.

4.3 Auxiliary (Pose Estimation) Task Network

The architecture for estimating the 2-D pose of the sketch (shaded pink and shown within the top-right brown dotted line box in Figure 2) is motivated by the following observations:

First, the part-level parsing of the object typically provides clues regarding object pose. For example, if we view the panel for *Two Wheelers* in Figure 5, it is evident that the relative location and spatial extent of 'handlebar' part for a `bicycle` is a good indicator of pose. Therefore, to enable pose estimation from part-level information, the input to the pose net is the tensor of pre-softmax pixelwise activations[2] generated within the expert part-parsing network. To capture the large variation in part appearances, locations and combinations thereof, the first two layers in the pose network contain dilated convolutional filters [46], each having rate $r = 2$ and stride $s = 2$ with kernel width $k = 3$. Each convolutional layer is followed by a ReLU non-linearity [25].

Second, 2-D pose is a global attribute of an object sketch. Therefore, to provide the network with sufficient global context, we configure the last convolutional layer with $r = s = 1$ and $k = 11$, effectively learning a large spatial template filter. The part combinations captured by initial layers also mitigate the need to learn many such templates – we use 32 in our implementation. The resulting template-based feature representations are combined via a fully-connected layer and fed to a 8-way softmax layer which outputs 2-D pose labels corresponding to cardinal and intercardinal directions. Note also that each super-category expert has its own pose estimation auxiliary net.

Having described the overall framework for SketchParsing, we next describe major implementation details of our training and inference procedure.

5 IMPLEMENTATION DETAILS

5.1 Training

SketchParse: Before commencing SketchParse training, the initial learning rate for all but the final convolutional layers is set to 5×10^{-4}. The rate for the final convolutional layer in each sub-networks is set to 5×10^{-3}. Batch-norm parameters are kept fixed during the entire training process. The architecture is trained end-to-end using a per-pixel cross-entropy loss as the objective function. For optimization, we employ stochastic gradient descent with a mini-batch size of 1 sketchified image, momentum of 0.9 and polynomial weight decay policy. We stop training after 20000 iterations. The training takes 3.5 hours on a NVIDIA Titan X GPU.

A large variation can exist between part sizes for a given super-category (e.g. number of 'tail' pixels is smaller than 'body' pixels in *Large Animals*). To accommodate this variation, we use a class-balancing scheme which weighs per-pixel loss differently based on relative presence of corresponding ground-truth part [9]. Suppose a pixel's ground-truth part label is c. Suppose c is present in N_c training images and suppose the total number of pixels with label c across these images is p_c. We weight the corresponding loss by $\alpha_c = M/f_c$ where $f_c = p_c/N_c$ and M is the median over the set of f_cs. In effect, losses for pixels of smaller parts get weighted to a larger extent and vice-versa.

Pose Auxiliary Net: The pose auxiliary network is trained end-to-end with the rest of SketchParse, using an 8-way cross-entropy loss. The learning rate for the pose module is set to 2.5×10^{-2}. All other settings remain the same as above.

Sketch classifier: For training the K-way sketch router classifier, we randomly sample 40% of sketches per category from TU-Berlin and Sketchy datasets[3]. We augment data with flip about vertical axis, series of rotations (±4, ±8, ±12 degrees) and sketch-scale augmentations (±3%, ±7% of image height). For training, the initial learning rate is set to 7×10^{-4}. The classifier is trained using stochastic gradient descent with a mini-batch size of 600 sketches, momentum of 0.9 and a polynomial weight decay policy. We stop training after 20000 iterations. The training takes 4 hours on a NVIDIA Titan X GPU. We intend to release our software framework (code and pre-trained models) to the community.

5.2 Inference

For evaluation, we used equal number of sketches (48) per category from TU-Berlin and Sketchy datasets except for `bus` category which is present only in TU-Berlin dataset. Thus, we have a total of $(48 \times 2 \times 10) + 48 = 1008$ sketches for testing. For sketch router classifier, we follow the conventional approach [37] of pooling score outputs corresponding to cropped (four corner crops and one center crop) and white-padded versions of the original sketch and its vertically mirrored version. Overall, the time to obtain part-level segmentation and pose for an input sketch is 0.25 seconds on average. Thus, SketchParse is an extremely suitable candidate for developing applications which require real-time sketch understanding.

6 EXPERIMENTS

To enable quantitative evaluation, we crowdsourced part-level annotations for all the sketches across 11 categories, in effect creating

[2]Shown as a blue oval in Figure 2.

[3]This translates to 26 sketches from TU-Berlin and 170 sketches from Sketchy for training, 6 and 30 sketches for validation.

Model	#parameters(M = millions)	Large Animals		Small Animals			4-Wheelers		2-Wheelers		Flying Things		Avg.
		cow	horse	cat	dog	sheep	bus	car	bicycle	motorbike	airplane	bird	
B-R5	88.4M	66.52	69.39	62.62	65.92	67.65	54.42	63.46	59.27	50.10	50.44	44.73	60.19
BC-R5	88.4M	67.69	70.02	64.57	68.54	70.02	66.79	66.96	63.74	53.15	55.50	47.72	62.90
BCP-R-1b	65.4M	60.47	22.09	24.38	22.76	23.87	19.96	20.55	18.21	19.50	21.94	20.13	25.78
BCP-R-11b	138.5M	64.87	65.69	63.78	65.24	65.69	64.91	62.50	57.44	48.71	51.47	43.84	59.17
BCP-R5	89M	68.10	69.45	65.47	68.37	70.74	67.17	66.48	62.66	54.49	55.47	48.77	**63.17**

Table 3: Comparing the full 5 super-category version of SketchParse (denoted BCP-R5) with baseline architectures.

the largest part-annotated dataset for sketch object parsing. We plan to publicly release the annotated sketch dataset and annotation software tool.

Evaluation procedure: For quantitative evaluation, we adopt the average IOU measure widely reported in photo-based scene and object parsing literature [22]. Consider a fixed test sketch. Let the number of unique part labels in the sketch be n_p. Let n_{ij} be the number of pixels of part-label i predicted as part-label j. Let $t_i = \sum_j n_{ij}$ be the total number of pixels whose ground-truth label is i. We first define part-wise Intersection Over Union for part-label i as $pwIOU_i = \frac{n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$. Then, we define sketch-average IOU (sIOU) = $\sum_i \frac{pwIOU_i}{n_p}$. For a given category, we compute sIOU for each of its sketches individually and average the resulting values to obtain the category's average IOU (aIOU) score.

Significance of class-balanced loss and auxiliary task: To determine whether to incorporate class-balanced loss weighting and 2-D pose as auxiliary task, we conducted ablative experiments on a baseline version of SketchParse configured for a single super-category ('Large Animals'). As the results in Table 1 indicate, class-balanced loss weighting and inclusion of 2-D pose as auxiliary task contribute to improved performance over the baseline model.

Determining split point in base model: A number of candidate split points exist which divide layers of the base scene parsing net [5] into two disjoint groups. We experimented with different split points within the scene parsing net. For each split point, we trained a full 5 super-category version of SketchParse model. Based on the results, we used the split point (*res5b*) which generated best performance for the final version viz. the SketchParse model with class-balanced loss weighting and pose estimation included for all super-categories. Note that we do not utilize the sketch router for determining the split point. From our experiments, we found the optimal split point results in shallow expert networks. This imparts SketchParse with better scalability. In other words, additional new categories and super-categories can be included without a large accompanying increase in number of parameters.

Full version net and baselines: We compare the final 5 super-category version (BCP-R5) of SketchParse (containing class-balanced loss weighting and pose estimation auxiliary task) with certain baseline variations – (i) B-R5: No additional components included (ii) BC-R5: Class-balanced loss weighting included in B-R5 (iii) BCP-R-1b: All categories are grouped into a single branch (iv) BCP-R-11b: A variant of the final version with a dedicated expert network for each category (i.e. one branch per category).

From the results (Table 3), we make the following observations: (1) Despite the challenges posed by hand-drawn sketches, our model performs reasonably well across a variety of categories (last row in Table 3). (2) Sketches from *Large Animals* are parsed the best while those from *Flying Things* do not perform as well. On closer scrutiny, we found that `bird` category elicited inconsistent sketch annotations given the relatively higher degree of abstraction in the corresponding sketches. (4) In addition to confirming the utility of class-balanced loss weighting and pose estimation, the baseline performances demonstrate that part (and parameter) sharing at category level is a crucial design choice, leading to better overall performance. In particular, note that having 1 category per branch (BCP-R-11b) almost doubles the number of parameters, indicating poor category scalability.

The performance of pose classifier can be viewed in Table 2. Note that simplifying the canonical pose directions (merging non-canonical directional labels with canonical directions) lends a dramatic improvement in accuracy.

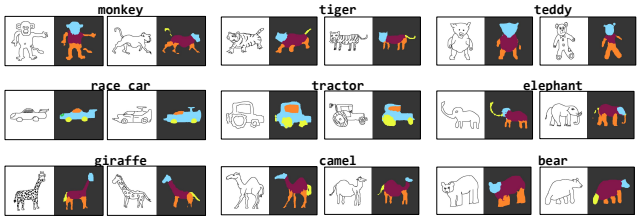

Figure 3: Part-parsing results for sketches from categories which are semantically similar to categories on which SketchParse is originally trained.

Qualitative evaluation: Rather than cherry-pick results, we use a principled approach to obtain a qualitative perspective. We first sort the test sketches in each super-category by their aIOU (average IOU) values in decreasing order. We then select 4 sketches located at 100-th, 75-th, 50-th, and 25-th percentile in the sorted order. These sketches can be viewed in Figure 5. The part-level parsing results reinforce the observations made previously in the context of quantitative evaluation.

6.1 Parsing semantically related categories

We also examine the performance of our model for sketches belonging to categories our model is not trained on but happen to be semantically similar to at least one of the existing categories. Since segmentation information is unavailable for these sketches, we show two representative parsing outputs per class. We include the classes `monkey`, `tiger`, `teddy-bear`, `camel`, `bear`, `giraffe`,

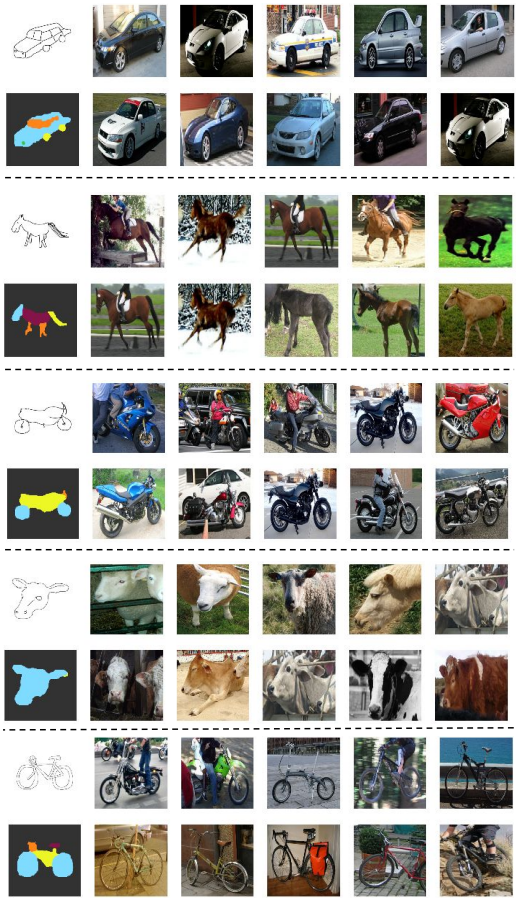

Figure 4: Five sketch-based image retrieval panels are shown. In each panel, the top-left figure is the query sketch. Its part parsing is located immediately below. Each panel has two sets of retrieval results for the presented sketch. The first row corresponds to Sangkloy et al.'s [33] retrieval results and the second contains our re-ranked retrievals based on part-graphs (Section 6.2).

elephant, race car and tractor which are semantically similar to categories already considered in our formulation. As the results demonstrate (Figure 3), SketchParse accurately recognizes parts it has seen before ('head', 'body', 'leg' and 'tail'). It also exhibits a best-guess behaviour to explain parts it is unaware of. For instance, it marks elephant 'trunk' as either 'legs' or 'tails' which is a semantically reasonable error given the spatial location of the part. These experiments demonstrate the scability of our model in terms of category coverage. In other words, our architecture can integrate new, hitherto unseen categories without too much effort.

6.2 Fine-grained retrieval

In another experiment, we determine whether part-level parsing of sketches can improve performance for existing sketch-based image retrieval approaches [33]. We use the PASCAL parts dataset [6] as the retrieval database. Given a TU-Berlin dataset [10] query sketch, the sketch and PASCAL images are projected onto a shared latent space using the Sketchy Siamese Network model [33] to obtain the sequence of retrieved images $D_1, D_2, \ldots D_T$. Suppose the test sketch is S and suppose the part-parsed version of S is P_S. We use a customized Attribute-Graph approach [29] and construct a graph G_S from P_S. The attribute graph is designed to capture spatial and semantic aspects of the part-level information at local and global scales. We use annotations from PASCAL-parts dataset to obtain part-segmented versions of $D_1, D_2, \ldots D_T$, which in turn are used to construct corresponding attribute graphs $G_{D_1}, G_{D_2}, \ldots G_{D_T}$. For re-ranking the retrieved images, we use Reweighted Random Walks Graph Matching [7] to compute similarity scores between G_S and $G_{D_i}, 1 \leqslant i \leqslant T$. During the graph matching process we enforce two constraints. *First*, a global node can only be matched to a global node of the other graph. *Second*, local nodes can only be matched if they correspond to the same type of part (*eg.* local nodes corresponding to legs can only be matched to other legs and cannot be matched to other body parts). For our experiments, we explore our re-ranking formulation for top-50 (out of 9620 images) of Sketchy model's retrieval results. In Figure 4, each panel corresponds to top-5 retrieval results for a particular sketch. The sketch and its parsing are displayed alongside the 5 nearest neighbors in latent space of Sketchy model (top row) and the top 5 re-ranked retrievals using our part-graphs (bottom row). The results show that our formulation exploits the category, part-level parsing and pose information to obtain an improved ranking.

6.3 Describing sketches in detail

Armed with the information provided by our model, we can go beyond describing a hand-drawn sketch by a single category label. For a given sketch, our model automatically provides its category, associated super-category, part-labels and their counts and 2-D pose information. From this information, we use a template-filling approach to generate descriptions – examples can be seen alongside our qualitative results in Figure 6. A fascinating application, inspired by the work of Zhang et al. [49], would be to use such descriptions to generate freehand sketches using a Generative Adversarial Network approach. We intend to explore this direction in our future work.

7 CONCLUSION

In this paper, we have presented SketchParse, the first deep-network architecture for fully automatic parsing of freehand object sketches. The originality of our approach lies in successfully repurposing a photo scene-segmentation net into a category-hierarchical sketch object-parsing architecture. The general nature of our transfer-learning approach also allows us to leverage advances in fully convolutional network-based scene parsing approaches, thus continuously improving performance. Another novelty lies in obtaining labelled training data *for free* by sketchifying photos from object-part datasets, thus bypassing burdensome annotation step. Our work stands out from existing approaches in the complexity of sketches, number of categories considered and semantic variety in categories. While existing works focus on one or two super-categories and build separate models for each, our scalable architecture can handle a larger number of super-categories, all with a single, unified

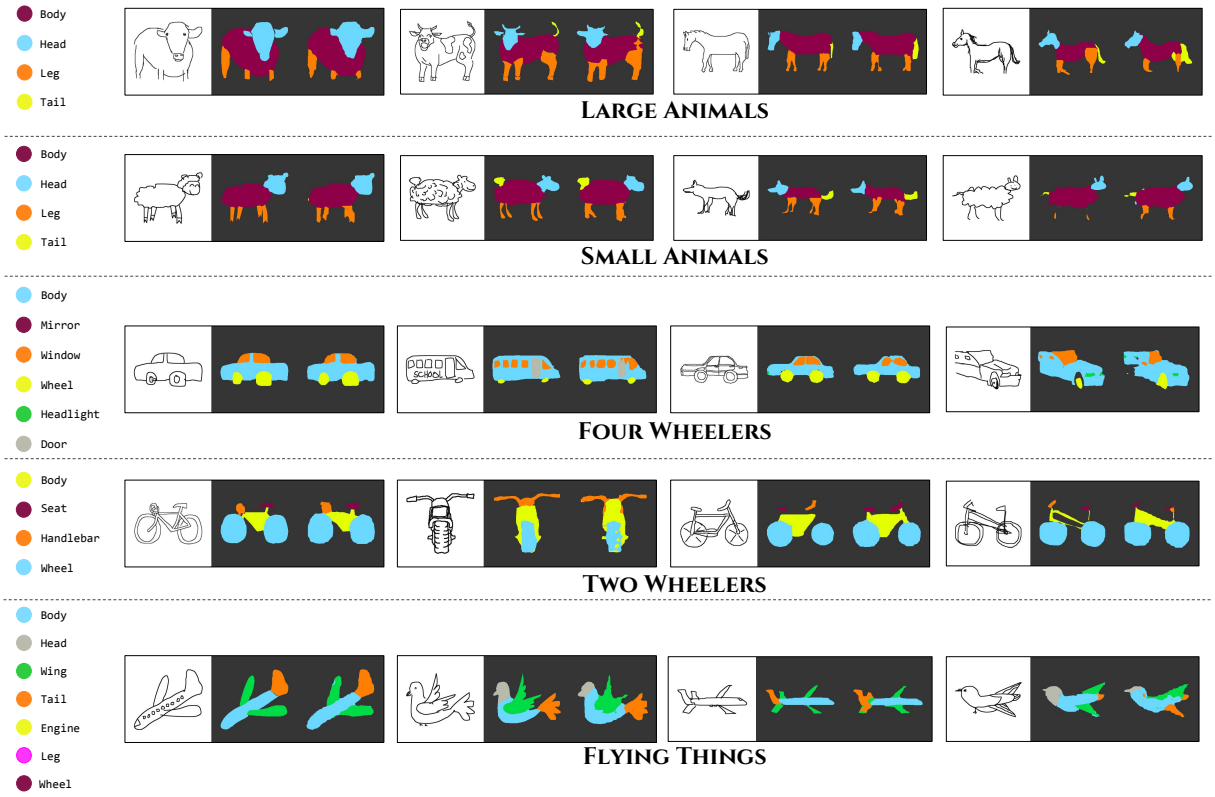

Figure 5: Qualitative part-parsing results across super-categories. Each row contains four panels of three images. In each panel, the test sketch is left-most, corresponding part-level ground-truth in the center and SketchParse's output on the right. The four panels are chosen from the 100th, 75th, 50th, 25th percentile of each super-category's test sketches sorted by their average IOU. The background is color-coded black in all the cases.

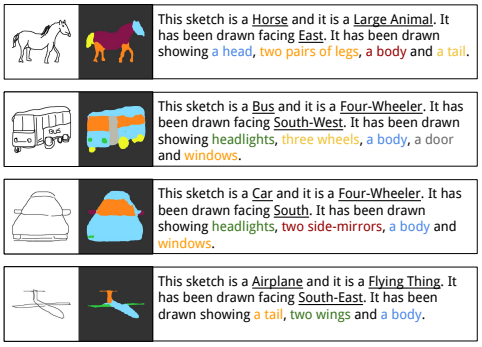

Figure 6: Some examples of fine-grained sketch descriptions. Each panel above shows a test sketch (left), corresponding part-parsing (center) and the description (last column). Note that in addition to parsing output, we also use the outputs of auxiliary pose network and router classifier to generate the description. The color-coding of part-name related information in the description aligns with the part color-coding in the parsing output. See Section 6.3 for additional details.

model. Finally, the utility of SketchParse's novel multi-task architecture is underscored by its ability to enable applications such as fine-grained sketch description and improving sketch-based image retrieval.

Please visit http://val.cds.iisc.ac.in/sketchparse for code, additional details and resources related to the work presented in this paper.

Acknowledgements: The authors would like to thank BS Vivek for his efforts in designing the pose annotation tool, Qualcomm Research India for their support to Ravi Kiran Sarvadevabhatla via the Qualcomm Innovation Fellowship and NVIDIA for providing a K40 GPU grant.

REFERENCES

[1] Abrar H Abdulnabi, Gang Wang, Jiwen Lu, and Kui Jia. 2015. Multi-task CNN model for attribute prediction. *IEEE Transactions on Multimedia* 17, 11 (2015), 1949–1959. 2, 3

[2] Karim Ahmed, Mohammad Haris Baig, and Lorenzo Torresani. 2016. Network of Experts for Large-Scale Image Categorization. In *14th European Conference on Computer Vision (Part-VII)*. Springer International Publishing, 516–532. 3

[3] Alessandro Bergamo and Lorenzo Torresani. 2010. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In *NIPS*. 181–189. 3

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. 2015. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In *ICLR*. 1

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. 2016. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *arXiv preprint arXiv:1606.00915* (2016). 2, 4, 6

[6] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. 2014. Detect What You Can: Detecting and Representing Objects using Holistic Models and Body Parts. In *CVPR*. 1, 7

[7] Minsu Cho, Jungmin Lee, and Kyoung Mu Lee. 2010. Reweighted Random Walks for Graph Matching. In *ECCV*. Springer-Verlag, 492–505. 7

[8] Jifeng Dai, Kaiming He, and Jian Sun. 2016. Instance-Aware Semantic Segmentation via Multi-Task Network Cascades. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 2, 3

[9] David Eigen and Rob Fergus. 2015. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of IEEE ICCV*. 2650–2658. 5

[10] Mathias Eitz, James Hays, and Marc Alexa. 2012. How do humans sketch objects? *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 44. 3, 7

[11] Mohamed Elhoseiny, Tarek El-Gaaly, Amr Bakry, and Ahmed Elgammal. 2016. A Comparative Analysis and Study of Multiview CNN Models for Joint Object Categorization and Pose Estimation. In *Proceedings of ICML*, Vol. 48. JMLR.org, 888–897. 3

[12] Ali Farhadi, Ian Endres, and Derek Hoiem. 2010. Attribute-centric recognition for cross-category generalization. In *IEEE CVPR*. IEEE, 2352–2359. 3

[13] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. 2015. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE CVPR*. 447–456. 1, 2

[14] Seunghoon Hong, Junhyuk Oh, Honglak Lee, and Bohyung Han. 2016. Learning Transferrable Knowledge for Semantic Segmentation with Deep Convolutional Neural Network. In *Proceedings of the IEEE CVPR*. 3

[15] Zhe Huang, Hongbo Fu, and Rynson W. H. Lau. 2014. Data-driven Segmentation and Labeling of Freehand Sketches. *Proceedings of SIGGRAPH Asia* (2014). 1, 2

[16] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, and George Fitzmaurice. 2014. Draco: bringing life to illustrations with kinetic textures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 351–360. 1

[17] Maksim Lapin, Bernt Schiele, and Matthias Hein. 2014. Scalable multitask representation learning for scene classification. In *Proceedings of the IEEE CVPR*. 1434–1441. 2, 3

[18] Xi Li, Liming Zhao, Lina Wei, Ming-Hsuan Yang, Fei Wu, Yueting Zhuang, Haibin Ling, and Jingdong Wang. 2016. DeepSaliency: Multi-task deep neural network model for salient object detection. *IEEE Transactions on Image Processing* 25, 8 (2016), 3919–3930. 3

[19] Yi Li, Timothy M. Hospedales, Yi-Zhe Song, and Shaogang Gong. 2014. Fine-Grained Sketch-Based Image Retrieval by Matching Deformable Part Models. In *BMVC*. 1

[20] Xiaodan Liang, Xiaohui Shen, Donglai Xiang, Jiashi Feng, Liang Lin, and Shuicheng Yan. 2016. Semantic Object Parsing With Local-Global Long Short-Term Memory. In *The IEEE CVPR*. 1, 2

[21] Joseph J Lim, C Lawrence Zitnick, and Piotr Dollár. 2013. Sketch tokens: A learned mid-level representation for contour and object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3158–3165. 3

[22] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE CVPR*. 3431–3440. 6

[23] Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. 2016. Semantic Segmentation using Adversarial Networks. In *NIPS Workshop on Adversarial Training*. 3

[24] Behrooz Mahasseni and Sinisa Todorovic. 2013. Latent multitask learning for view-invariant action recognition. In *Proceedings of the IEEE ICCV*. 3128–3135. 2, 3

[25] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*. 807–814. 5

[26] Vladimir Nekrasov, Janghoon Ju, and Jaesik Choi. 2016. Global Deconvolutional Networks for Semantic Segmentation. *CoRR* abs/1602.03930 (2016). 3

[27] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. 2015. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE ICCV*. 1520–1528. 1, 2

[28] Vishal M Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. 2015. Visual domain adaptation: A survey of recent advances. *IEEE signal processing magazine* 32, 3 (2015), 53–69. 3

[29] Nikita Prabhu and R Venkatesh Babu. 2015. Attribute-Graph: A Graph based approach to Image Ranking. In *Proceedings of the IEEE ICCV*. 1071–1079. 7

[30] Rajeev Ranjan, Vishal M Patel, and Rama Chellappa. 2016. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *arXiv preprint arXiv:1603.01249* (2016). 2, 3

[31] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. 2016. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE CVPR*. 3234–3243. 3

[32] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. 2010. Adapting visual category models to new domains. In *ECCV*. Springer, 213–226. 3

[33] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. 2016. The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies. *ACM Trans. Graph.* 35, 4, Article 119 (July 2016), 12 pages. 3, 7

[34] Ravi Kiran Sarvadevabhatla, Jogendra Kundu, and R. Venkatesh Babu. 2016. Enabling My Robot To Play Pictionary: Recurrent Neural Networks For Sketch Recognition. In *Proceedings of the ACMMM*. 247–251. 1

[35] Rosália G. Schneider and Tinne Tuytelaars. 2014. Sketch Classification and Classification-driven Analysis Using Fisher Vectors. *ACM Trans. Graph.* 33, 6, Article 174 (Nov. 2014), 174:1–174:9 pages. 2

[36] Rosália G. Schneider and Tinne Tuytelaars. 2016. Example-Based Sketch Segmentation and Labeling Using CRFs. *ACM Trans. Graph.* 35, 5, Article 151 (July 2016), 9 pages. DOI: https://doi.org/10.1145/2898351 1, 2

[37] Omar Seddati, Stephane Dupont, and Said Mahmoudi. 2015. Deepsketch: deep convolutional neural networks for sketch recognition and similarity search. In *13th International Workshop on Content-Based Multimedia Indexing (CBMI)*. IEEE, 1–6. 1, 2, 5

[38] Anja Theobald. 2003. An Ontology for Domain-oriented Semantic Similarity Search on XML Data. In *BTW 2003, Datenbanksysteme für Business, Technologie und Web, Tagungsband der 10. BTW-Konferenz, 26.-28. Februar 2003, Leipzig*. 217–226. 4

[39] Annegreet van Opbroek, M Arfan Ikram, Meike W Vernooij, and Marleen De Bruijne. 2015. Transfer learning improves supervised image segmentation across imaging protocols. *IEEE transactions on medical imaging* 34, 5 (2015), 1018–1030. 3

[40] Alexander Vezhnevets and Joachim M Buhmann. 2010. Towards weakly supervised semantic segmentation by means of multiple instance and multitask learning. In *IEEE CVPR*. IEEE, 3249–3256. 3

[41] Peng Wang, Xiaohui Shen, Zhe Lin, Scott Cohen, Brian Price, and Alan L Yuille. 2015. Joint object and part segmentation using deep learned potentials. In *Proceedings of the IEEE ICCV*. 1573–1581. 3

[42] Wikipedia. 2017. Cardinal direction — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Cardinal_direction. (2017). 3

[43] Fangting Xia, Peng Wang, Liang-Chieh Chen, and Alan L. Yuille. 2016. Zoom Better to See Clearer: Human and Object Parsing with Hierarchical Auto-Zoom Net. In *Proceedings of 14th European Conference in Computer Vision: Part V*. 648–663. 1, 2

[44] Ren Xiaofeng and Liefeng Bo. 2012. Discriminatively trained sparse code gradients for contour detection. In *Advances in neural information processing systems*. 584–592. 3

[45] Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. 2015. HD-CNN: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE ICCV*. 2740–2748. 3

[46] Fisher Yu and Vladlen Koltun. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122* (2015). 5

[47] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M. Hospedales, and Chen-Change Loy. 2016. Sketch Me That Shoe. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1

[48] Qian Yu, Yongxin Yang, Yi-Zhe Song, Tao Xiang, and Timothy Hospedales. 2015. Sketch-a-Net that Beats Humans. *BMVC* (2015). 1, 2, 3, 4

[49] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. 2016. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. *arXiv preprint arXiv:1612.03242* (2016). 7

[50] Yuqi Zhang, Yuting Zhang, and Xueming Qian. 2016. Deep Neural Networks for Free-Hand Sketch Recognition. In *17th Pacific-Rim Conference on Multimedia, Xi'an, China, September 15-16, 2016*. 3

[51] Bin Zhao, Fei Li, and Eric P Xing. 2011. Large-scale category structure aware image categorization. In *NIPS*. 1251–1259. 2

[52] Jiaping Zhao and Laurent Itti. 2016. Improved Deep Learning of Object Category using Pose Information. *CoRR* abs/1607.05836 (2016). http://arxiv.org/abs/1607.05836 3