Transfer Learning for Pose Estimation of Illustrated Characters

Shuhong (Shu) Chen
University of Maryland - College Park
shuhong@cs.umd.edu

Matthias Zwicker
University of Maryland - College Park
zwicker@cs.umd.edu

Abstract

Human pose information is a critical component in many downstream image processing tasks, such as activity recognition and motion tracking. Likewise, a pose estimator for the illustrated character domain would provide a valuable prior for assistive content creation tasks, such as reference pose retrieval and automatic character animation. But while modern data-driven techniques have substantially improved pose estimation performance on natural images, little work has been done for illustrations. In our work, we bridge this domain gap by efficiently transfer-learning from both domain-specific and task-specific source models. Additionally, we upgrade and expand an existing illustrated pose estimation dataset, and introduce two new datasets for classification and segmentation subtasks. We then apply the resultant state-of-the-art character pose estimator to solve the novel task of pose-guided illustration retrieval. All data, models, and code will be made publicly available.

1. Introduction

Human pose estimation is a foundational computer vision task with many real-world applications, such as activity recognition [38], 3D reconstruction [23], motion tracking [42], virtual try-on [12], person re-identification [36], etc. The generic formulation is to find, in a given image containing people, the positions and orientations of body parts; typically, this means locating landmark and joint keypoints on 2D images, or regressing for bone transformations in 3D.

The usefulness of pose estimation is not limited to the natural image domain; in particular, we focus on the domain of illustrated characters. As pose-guided motion retargeting of realistic humans rapidly advances [16], there is growing potential for automatic pose-guided animation [19], a traditionally labor-intensive task for both 2D and 3D artists. Pose information may also serve as a valuable prior in illustration colorization [56], keyframe interpolation [44], 3D character reconstruction [5] and rigging [54], etc.

With deep computer vision, we have been able to leverage large-scale datasets [34, 1, 49] to train robust estimators of human pose [20, 8, 15]. However, little work has been done to solve pose estimation for illustrated characters. Previous pose estimation work on illustrations by Khungurn *et al*. [25] presented a 2D keypoint detector, but relied on a publicly-unavailable synthetic dataset and an ImageNet-trained backbone. In addition, the dataset they collected for supervision lacked variation, and was missing keypoints and bounding boxes required for evaluation under the more modern COCO standard [34].

Facing these challenges, we constructed a 2D keypoint detector with state-of-the-art performance on illustrated characters, built upon domain-specific components and efficient transfer learning architectures. We demonstrate the effectiveness of our methods by implementing a novel illustration retrieval system. We summarize our contributions:

- A state-of-the-art pose estimator for illustrated characters, transfer-learned from both domain-specific and task-specific source models. Despite the absence of synthetic supervision, we outperform previous work by 10-20% PDJ@20 [25].
- An implementation of our proposed pose estimator that solves the novel task of pose-guided character illustration retrieval.
- Datasets for our model and its components, including: an updated COCO-compliant version of Khungurn *et al*.'s [25] pose dataset with 2x the number of samples and more diverse poses; a novel 1062-class Danbooru [2] tagging rulebook; and a character segmentation dataset 20x larger than those currently available.

2. Related Work

2.1. The Illustration Domain

Though there has been work on caricatures and cartoons [7, 40], we focus on anime/manga-style drawings where characters tend to be less abstract. While there is work for more traditional problems like lineart cleaning [43] and sketch extraction [31], more recent studies include sketch colorization [56], illustration segmentation

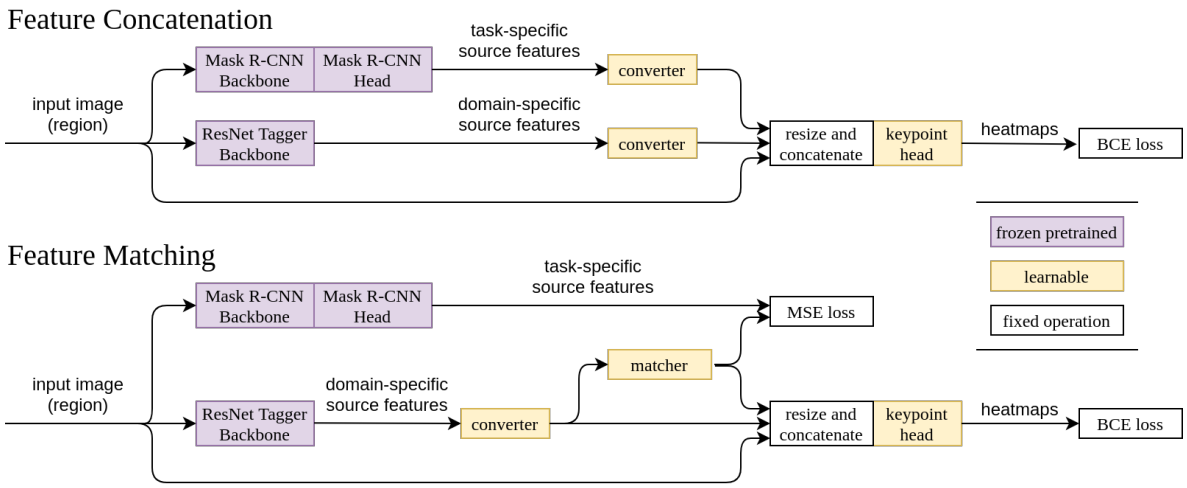

Figure 1. A schematic outlining our two transfer learning architectures: feature concatenation, and feature matching. Note that source feature specificity is with respect to the target; i.e. task-specific means "related to pose estimation" and domain-specific means "related to illustrations". While both designs require the pretrained Mask R-CNN components during training, feature matching discards them during inference, instead relying on the trained matcher network.

[55], painting relighting [57], image-to-image translation with photos [26], and keyframe interpolation [44].

Available models for illustrated tasks typically rely on small manually-collected datasets. For example, the AniSeg [33] character segmenter is trained on less than 1000 examples. While larger datasets are becoming available (e.g. Danbooru [2] now with 4.2m tagged illustrations), the labels are noisy and long-tailed, leading to poor model performance [3, 27]. Works requiring pose information may use synthetic renders of anime-style 3D models [25, 19], but the models are usually not publicly available. In this work, we present a cleaner tag classification task, a large character segmentation dataset, and an upgraded COCO keypoint dataset; these will all be made available upon publication, and may serve as a valuable prior for other tasks.

2.2. Transfer Learning & Domain Adaptation

Transfer learning and domain adaptation have been defined somewhat inconsistently throughout the vision and natural language processing literature [50, 10], though generally the former is considered broader than the latter. In this paper, we use the terms interchangeably, referring to methods that leverage information from a number of related source domains and tasks, to a specific target domain and task. Typically, much more data is available for the source than the target, motivating us to transfer useful related source knowledge in the absence of sufficient target data [50]. For deep networks, the simplest practice is to pretrain a model on source data, and fine-tune its parameters on target data; however, various techniques have been studied that work with different levels of target data availability.

Much of the transfer learning work in vision focuses on extreme cases with significantly limited target domain data, with emphasis around the task of image classification. In the few-shot learning case, we may be given as few as ten (or even one) samples from the target, inviting methods that embed prototypical target data into a space learned through prior source knowledge [52]. In particular, it is common to align parameters of feature extractors across domains, by directly minimizing pairwise feature distances or by adversarial domain discrimination [35, 48]. If the source and target are similar enough, it is possible to perform domain adaptation in the complete absence of labeled target data. This can be achieved by matching statistical properties of extracted features [45], or by converting inputs between domains through cycle-consistent image translation [22].

2.3. Pose Estimation

With the availability of large-scale human pose datasets [34, 1], the vision community has recently been able to make great strides in pose estimation. A naive baseline was demonstrated by Mask R-CNN [20], which extended their detection and segmentation framework to predict single-pixel masks of joint locations. Other work such as RMPE take an approach tailored to pose estimation, deploying spatial transformer networks with pose-guided NMS and region proposal [15]. Around the same time, OpenPose proposed part affinity fields as a bottom-up alternative to the more common heatmap representation of joints [8]. Human pose estimation work continues to make headway, extending beyond keypoint localization to include dense body part labels [18] and 3D pose estimation [24, 29, 37].

2.4. Pose Estimation Transfer

Most transfer learning for pose estimation adapts from synthetically-rendered data to natural images. For example, by using mocaps and 3D human models, SURREAL [49] provides 6m frames of synthetic video, complete with a variety of datatypes (2D/3D pose, RGB, depth, optical flow, body parts, etc.). CNNs may be able to directly generalize pose from synthesized images [49], and can further close the domain gap using other priors like motion [11]. Outside of synthetic-to-real, Cao *et al*. [30] explore domain adaptation for quadruped animal pose estimation, achieving generalization from human pose through adversarial domain discrimination with pseudo-label training.

The closest prior work to our topic was done by Khungurn *et al*. [25], who collected a modest AnimeDrawings-Dataset (ADD) of 2k character illustrations with joint keypoints, and a larger synthetic dataset of 1m frames rendered from MikuMikuDance (MMD) 3D models and mocaps. Unfortunately, the MMD dataset is not publicly available, and ADD contains mostly standard forward-facing poses. In addition, ADD is missing bounding boxes and several face keypoints, which are necessary for evaluation under the modern COCO standard [34]. We remedy these issues by training a bounding box detector from our new character segmentation dataset, labeling missing annotations in ADD, and labeling 2k additional samples in more varied poses.

Khungurn *et al*. perform transfer from an ImageNet-pretrained GoogLeNet backbone [46] and synthetic MMD data. In the absence of MMD, we instead transfer from a stronger backbone trained on a new illustration-specific classification task, as well as from a task-specific model pretrained on COCO keypoints. We use our subtask models and data to implement a number of transfer techniques, from naive fine-tuning to adversarial domain discrimination. In doing so, we significantly outperform Khungurn *et al*. on their reported metrics by 10-20%.

3. Method & Architectures

We provide motivation and architecture details for two variants of our proposed pose estimator (feature concatenation and feature matching), as well as two submodules critical for their success (a class-balanced tagger backbone and a character segmentation model). Architectures for baseline comparison models are described in Sec. 5.1.

3.1. Pose Estimation Transfer Model

We present two versions of our final model: feature concatenation, and feature matching. In this section, we assume that region proposals are given by a separate segmentation model (Sec. 3.3), and that the domain-specific backbone is already available (Sec. 3.2); here, we focus on combining source features to predict keypoints (Fig. 1).

The goal is to perform transfer simultaneously from both a domain-specific classification backbone (Sec 3.2) and a task-specific keypoint model (Mask R-CNN [20]). Here, we chose Mask R-CNN as it showed significantly better out-of-the-box generalization to illustrations than OpenPose [8] (Tab. 1). Taking into account that the task-specific model already achieves mediocre performance on the target domain, the feature concatenation model simply stacks features from both sources (Fig. 1). In order to perform the concatenation, it learns shallow feature converters for each source to decrease the feature channel count and allow bilinear sampling to a common higher resolution. The combined features are fed to the head, consisting of a shallow converter and two ResNet blocks.

The final output is a stack of 25 heatmaps, 17 for COCO keypoints and 8 for auxiliary appendage midpoints (following Khungurn *et al*. [25]). We apply pixel-wise binary cross-entropy loss on each heatmap, targeting a normal distribution centered on the ground-truth keypoint location with standard deviation proportional to the keypoint's COCO OKS sigma [34]; the sigmas for auxiliary midpoints are averaged from endpoints of the body part. At inference, we gaussian-smooth the heatmaps and take the maximum pixel value index as the keypoint prediction.

Although feature concatenation produces the best results (Tab. 1), it is very inefficient. At inference, it must maintain the parameters of both source models, and run both forward models for each prediction; Mask R-CNN is particularly expensive in this regard. We thus also provide a feature matching model, inspired by the methods used in Luo *et al*. [35]. As shown in Fig. 1, we simultaneously train an additional matching network that predicts features from the expensive task-specific model using features from the domain-specific model, optimized with feature-wise mean-squared error. This way, the pretrained Mask R-CNN still helps training, but is not necessary at inference. Despite its simplicity, feature matching retains most performance benefits from both source models, while also being significantly lighter and faster than the concatenation architecture.

3.2. ResNet Tagger

The domain-specific backbone for our model (Fig. 1) is a pretrained ResNet50 [21] fine-tuned as an illustration tagger. The tagging task is equivalent to multi-label classification, in this case predicting the labels applied to an image by the Danbooru imageboard moderators [2]. The 392k unique Danbooru tags cover topics including colors, clothing, character interactions, image composition, metainfo, and even artists and copyrights.

Khungurn *et al*. [25] use an ImageNet-trained GoogLeNet [46] backbone for their illustrated pose estimator, but we find that Danbooru fine-tuning significantly boosts transfer performance. There are publicly-available

Danbooru taggers [3, 27], but both their classification performance and feature learning capabilities are hindered by uninformative target tags and severe class imbalance. By alleviating these issues, we achieve significantly better transfer to pose estimation.

Most available Danbooru taggers [3, 27] take a coarse approach to defining classes, simply predicting the several thousand (6-7k) most frequent tags. However, many of these tags represent contextual information not present in the image; e.g. neon_genesis_evangelion (name of a franchise), or alternate_costume (fanmade/non-canon clothes). We instead only allow tags explicitly describing the image (clothing, body parts, etc.). Selecting tags by frequency also introduces tag redundancy and annotator disagreement. There are many high-frequency tags that share similar concepts, but are annotated inconsistently; e.g. hand_in_hair, adjusting_hair, and hair_tucking have vague wiki definitions for taggers, and many color tags are subjective (aqua_hair vs. blue_hair). To address these challenges, we survey Danbooru wikis to manually develop a rulebook of tag groups that defines more explicit and less redundant classes.

Danbooru tag frequencies form a long-tailed distribution, posing a severe class imbalance problem. In addition to filtering out under-tagged images (detailed in Sec. 4.2), we implement an inverse square-root frequency reweighing scheme to emphasize the learning of less-frequent classes. More formally, the loss on a sample is:

$$\mathcal{L}(y, \hat{y}) = \frac{1}{C} \sum_{i=0}^{C-1} w_i(y_i) BCE(y_i, \hat{y}_i) \tag{1}$$

$$w_i(z) = \frac{1}{2} \left(\frac{z}{r_i} + \frac{1-z}{1-r_i} \right) \tag{2}$$

$$r_i = \frac{\sqrt{N_i}}{\sqrt{N_i} + \sqrt{N - N_i}} \tag{3}$$

where C is the number of classes, $\hat{y} \in [0,1]^C$ is the prediction, $y \in \{0,1\}^C$ is the ground truth label, BCE is binary cross entropy loss, N is the total number of samples, and N_i is the number of positive samples in the i^{th} class. We found that plain inverse frequency weighing caused numerical instability in training, necessitating the square root.

3.3. Character Segmentation & Bounding Boxes

In order to produce bounding boxes around each subject in the image, we first train an illustrated character segmenter. As we assume one subject per image, we can derive a bounding box by enclosing the thresholded segmentation output. The single-subject assumption also removes the need for region proposal and NMS infrastructure present in available illustrated segmenters [33], so that our model may focus on producing clean segmentations only. Our segmentation model is based on DeepLabv3 [9], with three additional layers at the end of the head for finer segmentations at the input image resolution. We initialize with pretrained DeepLabv3 weights from PyTorch [39], and fine-tune the full model using pixel-wise binary cross-entropy loss.

4. Data Collection

Unless mentioned otherwise, we train with random image rotation, translation, scaling, flipping, and recoloring.

4.1. Pose Data

We extend the AnimeDrawingsDataset (ADD), first collected by Khungurn *et al*. [25]. The original dataset had 2000 illustrated full-body single-character images from Danbooru, each annotated with joint keypoints. However, ADD did not follow the now popularized COCO standard [34]; in particular, it was missing facial keypoints (eyes and ears) and bounding boxes. In order to evaluate and compare with modern pose estimators, we manually labeled the missing keypoints using an open-source COCO annotator [4] and automatically generated bounding boxes using the character segmenter described in Sec. 3.3. We also manually remove 57 images with multiple characters, or without the full body in view.

In addition, we improve the diversity of poses in ADD by collecting an additional 2043 samples. A major weakness of ADD is its lack of backwards-facing characters; only 5.45% of the entire 2k dataset had a back-related Danbooru tag (e.g. back, from_behind, looking_back, etc.). We specifically filtered for back-related images when annotating, resulting in a total of 850 in the updated dataset (21.25%). We also selected for other notably under-represented poses, like difficult leg tags (soles, bent_over, leg_up, crossed_legs, squatting, kneeling, etc.), arm tags (stretch, arms_up, hands_clasped, etc.), and lying tags (on_side, on_stomach).

Our final updated dataset contains 4000 illustrated character images with all 17 COCO keypoints and bounding boxes. We designate 3200 images for training (previously 1373), 313 for validation (previously 97), and 487 for testing (same as original ADD). For each input image, we first scale and crop such that the bounding box is centered and padded by at least 10% of the edge length on all sides. We then perform augmentations; flips require swapping left-right keypoints, and full 360-degree rotations are allowed.

4.2. ResNet Tagger Data

Our ResNet50 tagger is trained on a new subset of the 512px SFW Danbooru2019 dataset [2]. The original dataset contains 2.83m images with over 390k tags, but after filtering and retagging we arrive at 837k images with 1062 classes. The new classes are derived from manually-selected union rules over 2027 raw tags, as described in Sec. 3.2; the rulebook has 314 body-part, 545 clothing, and 203 miscellaneous (e.g. image composition) classes.

Model	OKS@50	OKS@75	PCKh@50	PDJ@20	PCPm@50	params	ms/img
Feature Concatenation (+new data)	**0.8982**	**0.7930**	**0.7866**	0.8403	0.8551	86.8m	217.7
Feature Concatenation	0.8827	0.7723	0.7762	0.8282	0.8435	86.8m	217.7
Feature Matching (+new data)	0.8953	0.7907	0.7851	**0.8423**	**0.8599**	9.9m	147.8
Feature Matching	0.8769	0.7680	0.7675	0.8251	0.8343	9.9m	147.8
Task Fine-tuning Only	0.8026	0.6481	0.7032	0.7666	0.7446	77.5m	174.5
Domain Features Only	**0.8607**	**0.7467**	0.7444	0.8076	**0.8215**	9.6m	143.7
Task Fine-tuning w/ Domain Features	0.8548	0.7209	**0.7544**	**0.8181**	0.8084	41.1m	147.8
Adversarial (DeepFashion2)	0.8321	0.6804	0.7108	0.7823	0.7778	9.9m	147.8
Adversarial (COCO)	0.8065	0.6362	0.6788	0.7607	0.7350	9.9m	147.8
Task-Pretrained (R-CNN)	**0.7584**	**0.6724**	**0.6960**	**0.7357**	**0.6679**	77.5m	174.5
Task-Pretrained (OpenPose)	0.4922	0.4222	0.4447	0.4796	0.4381	52.3m	128.2
Ours (equiv. to feat. concat.)	**0.8827**	**0.7723**	**0.7762**	**0.8282**	**0.8435**	86.8m	217.7
RF5 Backbone	0.8547	0.7358	0.7427	0.8015	0.8005	86.8m	217.7
ImageNet Backbone	0.8218	0.6919	0.7060	0.7649	0.7571	86.8m	217.7

Table 1. Performance of different architectures and ablations described in Sec. 5.1. Note that the parameter count and speed are measured in inference mode with batch size one.

To combat the class imbalance problem described in Sec. 3.2, we also rigorously filtered the dataset. We remove all images that are not single-person (solo, 1girl, or 1boy), are comics (comic, 4koma, doujinshi, etc.), or are smaller than 512px. Most critically, we remove all images with less than 12 positive tags; these images are very likely under-tagged, and would have introduced many false-negatives to the ground truth. The final subset of 837k images has significantly reduced class imbalance (median class frequency 0.38%, minimum 0.04%) compared to the datasets of available taggers (median 0.07%, min 0.01%) [3].

We split the dataset 80-10-10 train-val-test. As some tags are color-sensitive, we do not jitter the hue; similarly as some tags are orientation-sensitive, we allow up to 15-degree rotations and horizontal flips only.

4.3. Character Segmentation Data

To obtain character bounding boxes, we train a character segmentation model and enclose output regions at 0.5 threshold (Sec. 3.3). The inputs to our segmentation system are augmented composites of RGBA foregrounds (with transparent backgrounds) onto RGB backgrounds; the synthetic ground truth is the foreground alpha. The available AniSeg dataset [33] has only 945 images, with manually-labeled segmentations that are not pixel-perfectly aligned. We thus collect our own larger synthetic compositing dataset. Our background images are a mix of illustrated scenery (5.8k Danbooru images with scenery and no_humans tag) and stock textures (2.3k scraped [13] from the Pixiv Dataset [32]). We collect single-character foreground images from Danbooru with the transparent_background tag; 18.5k samples are used, after filtering images with text, non-transparency, or more than one connected component in the alpha channel. Counting each foreground as a single sample, this makes our new dataset roughly 20x larger than AniSeg. The foregrounds and backgrounds are randomly paired for compositing during training, with 5% chance of having no foreground. We hold out 2048 deterministic foreground-background pairs for validation and testing (1024 each).

5. Experiments

We used PyTorch [39] wrapped in Lightning [14]; some models use the R101-FPN keypoint detection R-CNN from Detectron2 [53]. All models can be trained with a single GTX1080ti (11GB VRAM). Unless otherwise mentioned, we trained models using the Adam [28] optimizer, with 0.001 learning rate and batch size 32, for 1000 epochs.

The ResNet backbone is trained on the Danbooru tag classification task using our new manual tagging rulebook (Sec. 4.2). The character segmenter used for bounding boxes is trained with our new character segmentation dataset (Sec. 4.3). Using the previous two submodules, we train the pose estimator using our upgraded version of the ADD dataset (Sec. 4.1). All data and code will be released upon publication.

5.1. Pose Estimation Transfer

Table 1 shows the performance of different architectures. We report COCO OKS [34], PCKh and PCPm [1], and PDJ (for comparison with Khungurn *et al*. [25]). From the top four rows, we see that our proposed feature concatenation and matching models perform the best out overall, and that the addition of our new data increases performance. We also observe that while concatenation performs marginally better than matching, matching is 8.8x more parameter efficient and one-third faster at inference.

keypoint	OKS@50	OKS@75	PCKh@50	PDJ@20	PDJ@20 [25]
nose	0.9466 (+0.4%)	0.8419 (+3.8%)	0.9918 (+0.2%)	0.9897 (+0.2%)	0.794 (+24.7%)
eyes	0.9795 (+1.1%)	0.9363 (+4.3%)	0.9928 (+0.0%)	0.9928 (+0.1%)	*0.890 (+11.6%)
ears	0.9589 (+1.3%)	0.8573 (+0.8%)	0.9836 (+0.1%)	0.9795 (-0.2%)	*0.890 (+10.1%)
shoulders	0.9825 (+2.8%)	0.9240 (+1.8%)	0.8973 (+2.6%)	0.9343 (+2.0%)	*0.786 (+18.9%)
elbows	0.8655 (+3.8%)	0.7320 (+6.4%)	0.7290 (+5.7%)	0.7916 (+4.2%)	0.641 (+23.5%)
wrists	0.7341 (+2.0%)	0.5657 (+2.4%)	0.6263 (+1.2%)	0.6961 (+1.5%)	0.503 (+38.4%)
hips	0.9630 (+0.0%)	0.8686 (+2.8%)	0.6704 (-1.1%)	0.7854 (+0.7%)	*0.786 (-0.1%)
knees	0.8686 (+2.8%)	0.7444 (+2.5%)	0.6643 (+2.9%)	0.7577 (+3.4%)	0.610 (+24.2%)
ankles	0.8090 (+1.3%)	0.6910 (-0.3%)	0.6263 (+1.0%)	0.7105 (+1.8%)	0.596 (+19.2%)

Table 2. Keypoint breakdown of our most performant "feature concatenation" model trained on our extended ADD dataset. In the center, we list the relative improvement of each metric when training on additional data. On the right, we display the PDJ@20 from Khungurn *et al*. [25], and report the relative difference from our best model. *Note that due to keypoint incompatibilities, we fill missing keypoint results from [25] using the most similar keypoints reported: "head" for eyes and ears, and "body" for shoulders and hips.

Model	F-1	pre.	rec.	IoU
Ours	**0.9472**	**0.9427**	**0.9576**	**0.9326**
YAAS SOLOv2	0.9061	0.9003	0.9379	0.9077
YAAS CondInst	0.8866	0.8824	0.8999	0.9158
AniSeg	0.5857	0.5877	0.5954	0.6651

Table 3. Comparison of our character segmentation and bounding box performance, described in Sec. 5.3.

The second group of Table 1 shows other architectures, roughly in order of method complexity. Here, as in Fig. 1, "task" source features refer to Mask R-CNN pose estimation features, and "domain" source features refer to illustration features extracted by our ResNet50 tag classifier.

"Task Fine-tuning Only" fine-tunes the pretrained Mask R-CNN head with its frozen default backbone; the last head layer is re-initialized to accommodate auxiliary appendage keypoints. This is vanilla transfer by fine-tuning a task-specific source network on a small task-specific target domain dataset.

"Domain Features Only" is our frozen ResNet50 backbone with a keypoint head. This is vanilla transfer by adding a new task head to a domain-specific source network.

"Task Fine-tuning w/ Domain Features" fine-tunes the pretrained Mask R-CNN head as above, but replaces the R-CNN backbone with our frozen ResNet50 backbone. This is a naive method of incorporating both sources, attempting to adapt the task source's pretrained prediction component to new domain features.

"Adversarial (DeepFashion2)" reuses the feature matching architecture, but performs adversarial domain discrimination instead of MSE matching. The discriminator is a shallow 2-layer convnet, trained to separate Mask R-CNN features of randomly sampled DeepFashion2 [17] images from ResNet features of Danbooru illustrations. As the feature maps to discriminate are spatial, we are careful to employ only 1x1 kernels in the discriminator; otherwise, the discriminator could pick up intrinsic anatomical differences. The matching network now fools the discriminator by adversarially aligning the feature distributions.

"Adversarial (COCO)" is the same adversarial architecture as above, but using COCO [34] images containing people instead of Deepfashion2.

While domain-features-only is the cheapest architecture overall, it is only slightly more efficient than feature matching, and loses all benefits of task-specific transfer. However, the performance drop from feature concatenation to domain-features-only and task-with-domain-features is not very large (2-3% OKS@50); meanwhile, there is a wide gap to task-fine-tuning-only. This shows that the domain-specific ResNet50 backbone trained on our new body-tag rulebook provides much more predictive power than the task-specific pretrained Mask R-CNN.

It is important to note that the adversarial models exhibited significant instability during training. After extensive hyperparameter tuning, the best DeepFashion2 model returns NaN loss at epoch 795, and the best COCO model fails at epoch 354; all other models safely exited at epoch 1000. DeepFashion2 likely outperforms COCO because the image composition is much more similar to that of Danbooru; images are typically single-person portraits with most of the body in view. Adversarial losses are notoriously difficult to optimize, and in our case destabilized training so as to perform worse than not having been used at all.

The fourth group of Table 4.3 shows out-of-the-box generalization to illustrations for Mask R-CNN [20] and OpenPose [8]. Despite OpenPose's more tailored design for pose estimation, the naive Mask R-CNN model is less-overfit to the natural image domain; we thus use the latter as our task-specific source.

Table 4.3 gives a keypoint breakdown and comparison with Khungurn *et al*. [25]. The results demonstrate that training on our additional more varied data improves the overall model performance; this is especially true for ap-

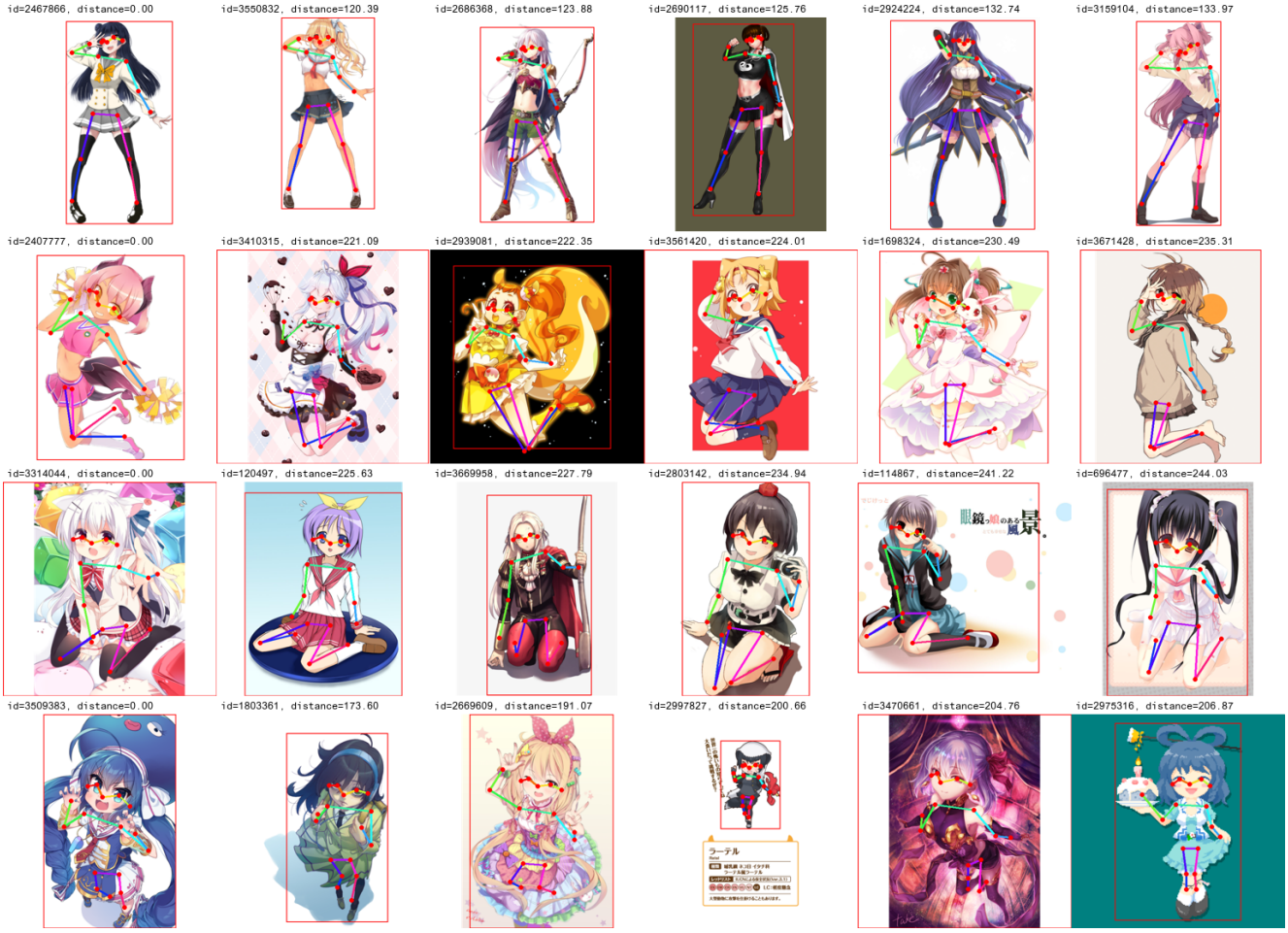

Figure 2. Application of our pose estimator to pose-based retrieval. From left to right, we show the query image (descriptor distance zero) followed by its five nearest neighbors (duplicate and NSFW images removed). Each illustration is annotated with its Danbooru ID, descriptor distance to the query, and the predicted bounding box with COCO keypoints.

pendage keypoints, which are more variable than the head and torso. We also see significant improvement from results reported in Khungurn *et al*. The exception is the hips, for which we compare to their "body" keypoint at the navel. While this is not a direct comparison, our PDJ on hips is nevertheless low relative to other keypoints. This is because PDJ does not account for the intrinsic ambiguity of the hips; looking at the OKS, which accounts for annotator disagreement, we see that hip performance is actually quite high.

An important caveat is that the metrics are generally not comparable with those reported in human pose estimation. COCO OKS, for example, was designed using annotator disagreement on natural images [34]; however, illustrated character proportions deviate widely from the standard human form (i.e. bigger head and eyes). Characters also tend to take up more screen space proportional to body size (i.e. big hair and clothing), leading to looser thresholds normalized by bounding box size.

5.2. ResNet Tagger Backbone

We train our ResNet50 tagger backbone to produce illustration-specific source features 1. Taking into account the class imbalance, we accumulate gradients for an effective batch size of 512. Considering the minimum (0.04%) and median (0.38%) class frequencies, we may expect the smallest class to appear 0.2 times per batch, and the median class to appear 1.9 times per batch.

To demonstrate the effectiveness of our tag rulebook and class reweighing strategy, we report performance on pose estimation using two other ResNet50 backbones: the RF5 tagger [3], and the default ImageNet-pretrained ResNet50 from PyTorch [39]. While there are several Danbooru taggers available [3, 27], we chose to compare our backbone to the RF5 tagger [3] because it is the most architecturally similar to our ResNet50, and relatively better-documented. The backbones all share the same architecture and parame-

ter count, and are all placed into our feature concatenation transfer model for the ablation.

The backbone ablation results are shown in the last three rows of Table 1. As expected, a classifier trained with our novel body-part-specific tagging rulebook and class-balancing techniques significantly improves transfer to pose estimation. Note that our tagger also outperforms RF5 at classification (on shared target classes); please refer to the supplementary materials for more details.

5.3. Character Segmentation & Bounding Boxes

We compare the segmentation and bounding box performance of our system with that of publicly-available models. AniSeg [33] is a Faster-RCNN [41], and YAAS [58] provides SOLOv2 [51] and CondInst [47] models. These detectors may detect more than one character, and their bounding boxes are not necessarily tight around segmentations; for simplicity, we union all predicted segmentations of an image, and redraw a tight bounding box around the union. We evaluate all models on the same test set described in Sec. 4.3. Table 3 shows that training with our new 20x larger dataset outperforms available models in both mean F-1 (segmentation) and IoU (bounding boxes); we thus use it in our pipeline for bounding box prediction.

6. Application: Pose-guided Retrieval

An immediate application of our illustrated pose estimator is a pose-guided character retrieval system. We construct a proof-of-concept retriever that takes a query character (or user-specified keypoints and bounding box) and searches for illustrated characters in a similar pose. This system can serve as a useful search tool for artists, who often use reference drawings while illustrating.

Our pose retriever performs a simple nearest-neighbor search. The support images consist of single-character Danbooru illustrations with the full_body tag. Using our best-performing model, we extract bounding boxes and keypoint locations for each character, normalize the keypoints by the longest bounding box dimension, and finally store the pairwise euclidean distances between the normalized keypoints. This process ensures the pairwise-distance descriptor is invariant to translation, rotation, and image scale. At inference, we extract the descriptor from the query, and find the euclidean k-nearest neighbors from the support set.

In practice, we compute descriptors using all 25 predicted keypoints (17 COCO and 8 additional appendage midpoints). This makes the descriptor 300-dimensional (25 choose 2), which is generally too large for tree-based nearest neighbors [6]. However, since our support set consists of 136k points, we are still able to brute force search in reasonable time. Empirically, each query takes about 0.1341s for keypoint extraction (GPU) and 0.0638s for search (CPU).

To demonstrate the effectiveness of our pose estimator, we present several query results in Fig. 2. Our system works well on not only standard poses as shown in the first row, but also on more difficult poses for which illustrators would want references. Note that while our system has no awareness of perspective, it is able to effectively leverage keypoint cues to retrieve similarly foreshortened views in the last row. For more examples, please refer to our supplementary materials.

7. Conclusion & Future Work

While we may continue to improve the transfer performance through methods like pseudo-labeling [30] or cycle-consistent image translation [22], we can also begin extending our work to multi-character detection and pose estimation. While it is possible to construct a naive instance-based segmentation and keypoint estimation dataset by compositing background-removed ADD samples, we cannot expect a system trained on such data to perform well in-the-wild. Character interactions in illustrations are often much more complex than human interactions in real life, with much more frequent physical contact. For example, Danbooru has 43.6k images tagged with holding_hands and 59.1k with hugging, already accounting for 2.8% of the entire dataset. Simply compositing independent characters together would not be able to model the intricacies of the illustration domain; we would again need to expand our datasets with annotated instances of character interactions.

As a fundamental vision task, pose estimation also provides a valuable prior for numerous other novel applications in the illustrated domain. Our pose estimator opens the door to pose-guided retargeting for automatic character animation, better keyframe interpolation, pose-aware illustration colorization, 3D character reconstruction, etc.

In conclusion, we demonstrate state-of-the-art pose estimation on the illustrated character domain, by leveraging both domain-specific and task-specific source models. Our model significantly outperforms prior art [25] despite the absence of synthetic supervision, thanks to successful transfer from our new illustration tagging subtask focused on classifying body-related tags. In addition, we provide a single-region proposer trained on a novel character segmentation dataset 20x larger than those currently available, as well as an updated illustration pose estimation dataset with twice the number of samples in more diverse poses. Our model performance allows for successful application to the novel task of pose-guided character illustration retrieval, and paves the way for future applications in the illustrated domain.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[2] Anonymous, Danbooru community, and Gwern Branwen. Danbooru2020: A large-scale crowdsourced and tagged anime illustration dataset. `https://www.gwern.net/Danbooru2020`, January 2021.

[3] Matthew Baas. Danbooru2018 pretrained resnet models for pytorch. `https://rf5.github.io`, July 2019.

[4] Justin Brooks. COCO Annotator. `https://github.com/jsbroks/coco-annotator/`, 2019.

[5] Philip Buchanan, Ramakrishnan Mukundan, and Michael Doggett. Automatic single-view character model reconstruction. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, pages 5–14, 2013.

[6] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[7] Kaidi Cao, Jing Liao, and Lu Yuan. Carigans: Unpaired photo-to-caricature translation. *arXiv preprint arXiv:1811.00222*, 2018.

[8] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *IEEE transactions on pattern analysis and machine intelligence*, 43(1):172–186, 2019.

[9] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[10] Hal Daume. Domain adaptation vs. transfer learning, Nov 2007.

[11] Carl Doersch and Andrew Zisserman. Sim2real transfer learning for 3d human pose estimation: motion to the rescue. *arXiv preprint arXiv:1907.02499*, 2019.

[12] Haoye Dong, Xiaodan Liang, Xiaohui Shen, Bochao Wang, Hanjiang Lai, Jia Zhu, Zhiting Hu, and Jian Yin. Towards multi-pose guided virtual try-on network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9026–9035, 2019.

[13] Nandaka et al. Pixivutil2. `https://github.com/Nandaka/PixivUtil2`, 2021.

[14] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019.

[15] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2334–2343, 2017.

[16] Oran Gafni, Oron Ashual, and Lior Wolf. Single-shot freestyle dance reenactment. *arXiv preprint arXiv:2012.01158*, 2020.

[17] Yuying Ge, Ruimao Zhang, Xiaogang Wang, Xiaoou Tang, and Ping Luo. Deepfashion2: A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5337–5345, 2019.

[18] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7297–7306, 2018.

[19] Koichi Hamada, Kentaro Tachibana, Tianqi Li, Hiroto Honda, and Yusuke Uchida. Full-body high-resolution anime generation with progressive structure-conditional generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. PMLR, 2018.

[23] Zeng Huang, Yuanlu Xu, Christoph Lassner, Hao Li, and Tony Tung. Arch: Animatable reconstruction of clothed humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3093–3102, 2020.

[24] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014.

[25] Pramook Khungurn and Derek Chou. Pose estimation of anime/manga characters: a case for synthetic data. *Proceedings of the 1st International Workshop on coMics ANalysis, Processing and Understanding*, 2016.

[26] Junho Kim, Minjae Kim, Hyeonwoo Kang, and Kwanghee Lee. U-gat-it: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. *arXiv preprint arXiv:1907.10830*, 2019.

[27] Kichang Kim, Rachmadani Haryono, and Arthur Guo. Deepdanbooru. `https://github.com/KichangKim/DeepDanbooru`, 2019.

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[29] Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J Black, and Peter V Gehler. Unite the peo-

ple: Closing the loop between 3d and 2d human representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6050–6059, 2017.

[30] Chen Li and Gim Hee Lee. From synthetic to real: Unsupervised domain adaptation for animal pose estimation. *arXiv preprint arXiv:2103.14843*, 2021.

[31] Chengze Li, Xueting Liu, and Tien-Tsin Wong. Deep extraction of manga structural lines. *ACM Transactions on Graphics (TOG)*, 36(4):1–12, 2017.

[32] Jerry Li. Pixiv dataset. `https://github.com/jerryli27/pixiv_dataset`, 2019.

[33] Jerry Li and Tazik Shahjahan. Aniseg. `https://github.com/jerryli27/AniSeg`, 2020.

[34] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[35] Zelun Luo, Yuliang Zou, Judy Hoffman, and Li Fei-Fei. Label efficient learning of transferable representations across domains and tasks. *arXiv preprint arXiv:1712.00123*, 2017.

[36] Jiaxu Miao, Yu Wu, Ping Liu, Yuhang Ding, and Yi Yang. Pose-guided feature alignment for occluded person re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 542–551, 2019.

[37] Gyeongsik Moon and Kyoung Mu Lee. I2l-meshnet: Image-to-lixel prediction network for accurate 3d human pose and mesh estimation from a single rgb image. *arXiv preprint arXiv:2008.03713*, 2020.

[38] Tewodros Legesse Munea, Yalew Zelalem Jembre, Halefom Tekle Weldegebriel, Longbiao Chen, Chenxi Huang, and Chenhui Yang. The progress of human pose estimation: a survey and taxonomy of models applied in 2d human pose estimation. *IEEE Access*, 8:133330–133348, 2020.

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[40] Omid Poursaeed, Vladimir Kim, Eli Shechtman, Jun Saito, and Serge Belongie. Neural puppet: Generative layered cartoon characters. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3346–3356, 2020.

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[42] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, pages 1297–1304. Ieee, 2011.

[43] Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. Mastering sketching: adversarial augmentation for structured prediction. *ACM Transactions on Graphics (TOG)*, 37(1):1–13, 2018.

[44] Li Siyao, Shiyu Zhao, Weijiang Yu, Wenxiu Sun, Dimitris N Metaxas, Chen Change Loy, and Ziwei Liu. Deep animation video interpolation in the wild. *arXiv preprint arXiv:2104.02495*, 2021.

[45] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pages 443–450. Springer, 2016.

[46] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[47] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. *arXiv preprint arXiv:2003.05664*, 2020.

[48] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.

[49] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 109–117, 2017.

[50] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[51] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. *Advances in Neural Information Processing Systems*, 2020.

[52] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020.

[53] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[54] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. Rignet: Neural rigging for articulated characters. *arXiv preprint arXiv:2005.00559*, 2020.

[55] Lvmin Zhang, Yi Ji, and Chunping Liu. Danbooregion: An illustration region dataset. In *ECCV (13)*, pages 137–154, 2020.

[56] Lvmin Zhang, Chengze Li, Tien-Tsin Wong, Yi Ji, and Chunping Liu. Two-stage sketch colorization. *ACM Transactions on Graphics (TOG)*, 37(6):1–14, 2018.

[57] Lvmin Zhang, Edgar Simo-Serra, Yi Ji, and Chunping Liu. Generating digital painting lighting effects via rgb-space geometry. *ACM Transactions on Graphics (TOG)*, 39(2):1–13, 2020.

[58] zymk9 and huaji0353. Yet-another-anime-segmenter. `https://github.com/zymk9/Yet-Another-Anime-Segmenter`, 2020.

8. Supplementary Materials

8.1. Tagger Classification Comparison

In the main paper, we show that our tagger (trained on our new tag rulebook with class-balanced weighing) significantly improves transfer to pose estimation. Here, we show classification results in comparison to the RF5 Danbooru tagger [3], a publicly-available model with the same ResNet50 architecture. RF5 predicts the presence of the top 6000 most common tags in the dataset; 1207 of these are present in our new rulebook, and can be used to predict 1032 of the 1062 total new classes. As we can see from Table 4 below, our model performs much better at classifying the same tags.

Model	Ours	RF5
F-2	0.4744	0.2297
precision	0.3022	0.1238
recall	0.5786	0.3360
accuracy	0.9760	0.9496
F-1	0.4249	0.1910
precision	0.4236	0.1898
recall	0.4458	0.2235
accuracy	0.9851	0.9727

Table 4. Comparison of our Danbooru tagger to RF5 [3]. Metrics are calculated using per-class optimal thresholds for either F-1 or F-2, and averaged across all classes shared between models. Note that this means F-1 and F-2 cannot be directly calculated from their respective precision and recall statistics in the table.

8.2. Pose Retrieval Additional Results

We display several more pose-based illustration retrieval results in Fig. 3; the images are taken from the Danbooru dataset [2]. The first two rows show challenging sitting positions, on which our model still performs well qualitatively. Despite the differences in orientation, our rotation-invariant descriptor is still able to identify the poses as similar. Rows 3-5 show some more standard poses. Notice that in row 4, the first and second neighbors are variations of the same character in the same pose; it is very common to find a set of such variations uploaded to Danbooru together, and our model may help identify them. In the last two rows, we show failure cases of our model, where incorrect predictions on the query result in neighbors with different poses.

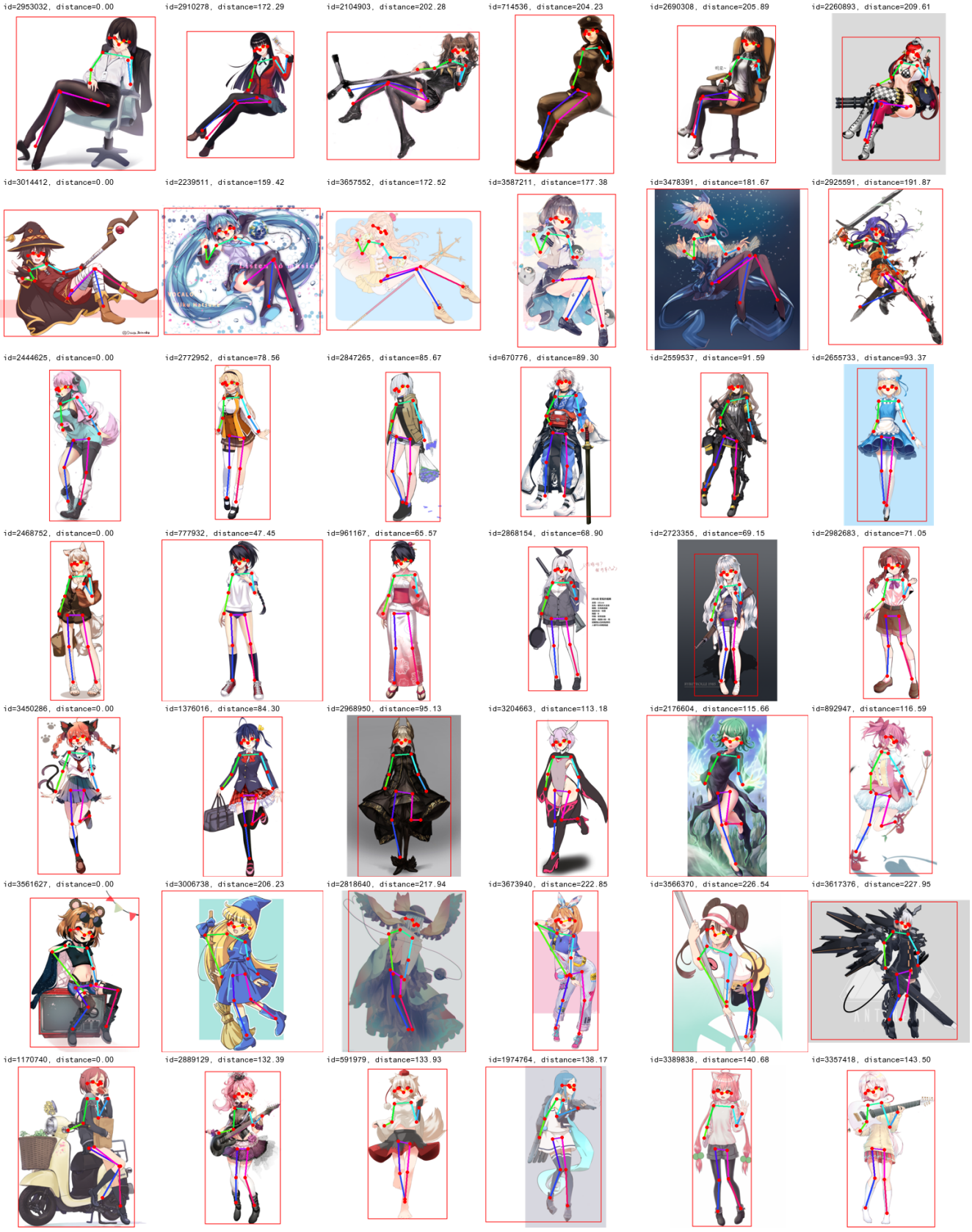

Figure 3. Additional pose-based retrieval results. From left to right, we show the query image (descriptor distance zero) followed by its five nearest neighbors (duplicate and NSFW images removed). Each illustration is annotated with its Danbooru ID, descriptor distance to the query, and the predicted bounding box with COCO keypoints.