Pose Estimation of Anime Characters

Final Report

Liran Li
The University of British Columbia
liliran@cs.ubc.ca

Haoyu Yang
The University of British Columbia
yhymason@gmail.com

1 ABSTRACT

While human pose estimation is a well established problem in computer vision, little attention has been paid to the domain of anime characters. We adopt the problem definition of Khungurn et al. [MANPU 2016], and evaluate a pre-trained stacked hourglass model on anime drawings for the pose estimation task. Our experiment shows promising results for this transfer of domain knowledge. We also demonstrate the use of a Maya script that has the prospect of helping casual users to export joint annotations suitable for pose estimation tasks.

2 INTRODUCTION

Pose estimation has a range of applications, from interactive installations that react to the body to augmented reality, animation, fitness uses, and more. A good pose estimation system must be robust to occlusion and severe deformation, successful on rare and novel poses, and invariant to changes in appearance due to factors including, but not limited to, clothing and lighting [1].

Although much research has been done in order to tackle the problem of human pose estimation, the amount of attention being paid to anime character pose estimation is yet very limited. With the rapid growth of digital entertainment industry, the need for robust anime character pose estimation systems is increasing. Animators who work with anime character models usually pay more efforts to construct accurate skeletons because these anime characters often have non-realistic body shapes that can cause confusion to automatic rigging tools. Moreover, the lack of pose estimation methods designed for anime characters makes it difficult to automatically generate customized anime characters that have a specific anime art style. Therefore, it will be valuable to create a pose estimation system to support applications of anime character poses.

The problem of anime character pose estimation is not as well defined as the problem of human pose estimation. Artists may create characters of unnatural body shapes to achieve desired artistic effects, realistic or not. The attempt by Khungurn et al. [2] was to prescribe the distribution of anime characters to be that of Danbooru [3], a repository of high-quality anime-style art. In particular, the anime drawings dataset in [2] was 2,000 hand-picked images from Danbooru, labelled by volunteers. While there are abundant anime drawings for image classification, an equivalent for pose estimation did not exist due to lack of joint annotation. To aid their training process, Khungurn et al. therefore generated a large set of training images with joint annotations using animated 3D models. Note that their training images came from a different distribution than the drawings; however, training on the 3D model images improved performance on the target distribution.

We closely follow Khungurn et al. [2] by adopting the same distribution of anime characters [1] and using the same set of images for evaluation. However, our approach differs from [2] in the following two respects:

- We directly apply the stacked hourglass network for human pose estimation to anime character pose estimation. Our experiments show that transferring knowledge from real human images to anime characters could be a promising approach.
- We leverage the accessibility of the Maya software to generate synthetic training images. We developed a python script[2] that has a prospect of helping many more Maya users to export joint annotations, thus alleviating the lack of synthetic training data.

The rest of the paper is structured as follows. Section 3 reviews two previous works on which we base our study. Section 4 describes the pre-processing of anime drawings dataset and generating our own synthetic dataset. Section 5 shows the results of evaluating models on the anime drawings dataset. We conclude our study in section 6

3 RELATED WORK

There has been a wealth of works on pose estimation. Some of the past state-of-the-art networks include DeepCut [4], DeepPose [5] and stacked hourglass [1]. Instead of the first two models, we prefer the last one, which gives a minimal design that achieved state-of-the-art performance back in 2016. The stacked hourglass network makes no assumption on the problem domain, so that transfer learning from a pre-trained model has minimal penalty resulting from the network design.

[1] From now on we use the word drawings interchangeably with anime characters in the context of training.
[2] https://github.com/Traeume/GraphicsProject

The Stack Hourglass (SH) architecture [1] can be visualized as hourglasses stacked on top of each other. Each hourglass is a sequence of convolution operations that first reduce feature dimensions (picture size) while increasing the number of channels, and then upsample to increase feature dimensions while reducing number of channels. They also combined residual signals between layers. In other words, SH is a straight forward convolutional neural network with residual connections.

The predictions and ground truths are in the format of a heapmap, where each image pixel is assigned a value indicating the likelihood of being a certain joint. For a prediction, the pixel with the highest value for a joint is predicted as that joint. The joint predictors are independent even though they share previous layers. The loss used for SH is the squared error between predicted and ground truth heatmaps.

The input images to SH are scaled and cropped to 256-by-256 pixels with the person in the center, and the outputs are 64-by-64 pixel heatmaps. This model always predicts a pixel location for a joint, even if the joint is invisible when it falls outside of the image (cropped limbs) or is hidden behind another object. In cases where a joint is not visible, the model is not penalized for any prediction, and the authors argue that this can be remedied by a classifier that tells from the predicted heatmap whether not that joint is visible. This model was trained on Mpii dataset and achieved an average PCKh accuracy of 90.9%. The training images are randomly rotated and scaled as a scheme of augmentation.

The problem of human pose estimation is well standardized by state-of-the-art benchmark datasets including MPII Human Pose dataset and Leeds Sport Pose (LSP) Dataset. On the other hand, anime character pose estimation is not as well defined, and no public datasets existed.

Khungurn et al. [2] approached this problem by prescribing the distribution of anime characters to be that of the drawings on Danbooru, a repository of high-quality anime-style art. A set of 2,000 drawings were collected from Danbooru, annotated by volunteers, and used as the benchmark.

To aid their training process, Khungurn et al. generated another dataset consisting of over 1,000,000 images with joint annotations[3]. These synthesized images are from animated sequences of 3D characters, combined with random background images. In particular, they sampled from a pool of 2,100 character models and 6,500 motion sequences, both in MikuMikuDance[4] format. Note that the distribution of synthesized images is different from the target distribution of anime characters.

Khungurn et al. modified a pretrained GoogLeNet and trained the last layers on their dataset of synthesized images. They demonstrated that their models thus trained had a high PCKh accuracy on 3D anime characters while achieving competitive performance on LSP test set. They fine tuned their model on the 2,000 drawings and showed that the test accuracy was reasonably high. In particular, they achieved PDJ scores of 89% for heads, 59.6% for ankles and 50.3% for knees. However, their model failed on characters with flashy costumes or unnatural body proportions, which are common in anime characters.

With the experiment on LSP test set, [2] suggests similarity between human poses and anime character poses. We validate this similarity by directly applying a pre-trained stacked hourglass network to anime characters. It is also shown in [2] that training on synthesized images was effective. These synthesized images can be easily obtained from software such as Maya. To eventually engage the large community of Maya users, we developed a prototype script that extracts joint annotation from animation and camera settings. The lack of training images could be greatly alleviated with a similar plug-in made available to the community.

4 DATASETS AND METHOD

In order to validate the transfer of knowledge from human poses to anime character poses, we apply a pre-trained stacked hourglass model to the anime drawings dataset used by Khungurn et al. [2]. In addition, we develop a python script for the Maya software to help interested users to export joint annotations. We demonstrate the effectiveness of the script by generating our own synthetic dataset.

4.1 Annotation Conversion

The pre-trained hourglass model was obtained from [6]. The validation accuracy on the MPII human pose dataset averaged over all joints is 81.6%[5]. The MPII dataset contains 25,000 images with 16 annotated body joints.

In Khungurn et al. [2], a set of 2,000 drawings[6] were collected from Danbooru, annotated by volunteers, and used as the benchmark. Each character has 21 joints, with finer features including the nose tip, thumbs and tiptoes. The annotation of this drawings dataset is in a different format than MPII.

In order to evaluate the stacked hourglass model on the drawings dataset, we converted the annotation of drawings dataset into the MPII format. Joints contained in both annotations are left intact (i.e. shoulders, elbows, wrists, hips, knees and ankles).

Annotation in MPII has fields (”objpos” and ”scale”) indicating the rough pixel location and the bounding box of a person, while the drawings dataset [7] does not. These fields

[3]This synthetic dataset was not distributed by the authors.

[4]MikuMikuDance is an open-source animation tool developed in Japan that has gained popularity among anime and VOCALOID fans.

[5]This is the validation accuracy of a 1-stack hourglass model. The accuracy increases with the number of stacks [1]

[6]This dataset is available at [7].

are important for the stacked hourglass model as the model requires the human figure to be roughly in the center of its input. By quick inspection, we found that the vast majority of the characters are indeed drawn in the center of the image, with the character filling the image either in height or in width. These field were therefore always set to the center of the image, and the width or height of the image, respectively.

Other differences in the two annotations are immaterial. Joints present in the drawings dataset but not in MPII were simply left out. Joints present in MPII but not in the drawings dataset were replaced either with a similar joint, or calculated with an average or offset of other joints.

We use the MPII annotation format in all of our experiments.

4.2 Synthetic Dataset

Khungurn et al. [2] showed that training on synthesized images improves model performance on the target distribution. Given the large community of animators, it should be reasonably easy to collect synthetic images into a dataset. We develop a script that has the prospect of helping casual Maya users to export suitable joint annotations for pose estimation tasks.

In this study, we follow [2] and focus on animated 3D models from MikuMikuDance (MMD). MMD is an open-source animation tool developed in Japan. It provides a simple interface and has gained popularity among anime and VOCALOID fans.

The approach taken by Khungurn et al. [2] was to sample from a large pool of MMD files, parse the kinematics, and render while exporting joint annotations. However, we note that developer support for MMD is limited, and documentations for MMD file formats are scarce and incomplete. Even though members of the community have independently developed tools to animate physics and solve inverse kinematics[7], they differ in the extent to which they approximate the actual MMD animation. Therefore, rendered images and joint annotations could deviate from the motion intended by the animator. Besides, this approach does not engage casual users of MMD.

Instead, we use the well maintained commercial software Maya, and our script exports joint annotations exactly matching the sequence crafted by the animator. Our script encodes knowledge of MMD models and finds most joints correctly, therefore reducing user intervention. For joints missing from MMD models we approximate them using other joints (e.g. averaging left and right hips to find pelvis). Our script requires no dependencies.

We test our script by generating synthetic images similar to [2]. In order to work with MMD, we use MMD4Maya, a plug-in that imports MMD format character models and motion sequences[8]. Our workflow is typical of a casual animator. Within Maya, we import MMD animation, add in background images and lighting, adjust the camera, and playback the animation to ensure that the rendered sequence is aesthetically pleasing. For a fair comparison with the drawings dataset, we change camera and render settings until there are no cropped limbs. After rendering a whole sequence, we run our script within Maya to record the joint annotations. Throughout the process, we use the Arnold renderer.

Our proof-of-concept synthetic dataset consists of 5 different character models, 4 different motion sequences and 5 background images that were taken in real world. This dataset contains 1,400 images, all of which are square shaped with dimensions 512*512 or 1024*1024. These images are then randomly shuffled and divided into a training set of 1,300 images and a validation set of 100 images. The overall time spent generating our dataset is about 4 hours (on Microsoft Surface Pro 5 with Intel Core i7 with minimal GPU). For simplicity, we do not incorporate toon shading. Figure 1 shows two example from our synthetic dataset.

Using our script with Maya allows users to generate data suitable for pose estimation tasks. By engaging more users, a large variety of anime character models and body poses could be covered.

4.3 Transfer of Domain Knowledge

Our goal is to evaluate the performance of a pre-trained SH model on the drawings dataset, thus validating the transfer of knowledge from human poses to the domain of anime characters. We choose the PyTorch implementation of SH found at [6] and use their 1-stack[9] hourglass model pre-trained on MPII. For completeness, we further fine-tune the pre-trained model on both the drawings dataset and our synthetic dataset. For fine-tuing, we freeze all weights in SH except at the last fully connected and convolution output layers, resulting in 70,000 trainable weights, compared to a total of 3.5M weights in the model. Note that [6] works with the MPII format of annotations, and complies with the SH procedures [1] by first cropping input images into square boxes containing the human figure in the center; during training, it applies random rotation as data augmentation.

All of our training or fine-tuning was performed on a GeForce GTX 980 desktop graphics card with each epoch taking a few minutes.

5 RESULTS AND DISUSSION

A visualization of predictions from all models can be found in Figure 5.

[7] See e.g. [8] and [9]

[8] MMD character model files have extension .pmd or .pmx, and MMD motion sequence files have extension .vmd.

[9] This was a convenient choice for our limited GPU resouces

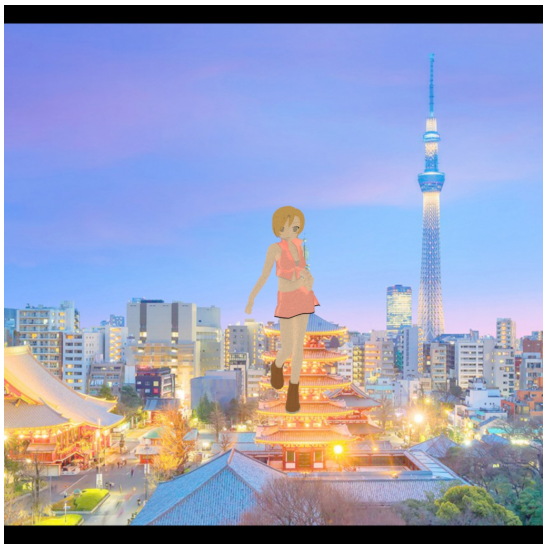

Figure 1: Synthetic images with random background images, generated using Maya Arnold renderer. MMD models were imported into Maya with the MMD4Maya plug-in.

5.1 Evalutaion Criterion

We evaluate all models on the drawings validation set [7]. The accuracy criterion we use is the Percentage of Correct Keypoints (PCK@0.5) score, while Khungurn et al. [2] use Percentage of Detected Joints (PDJ@0.2). The two scores both find the fraction of joints predicted within a threshold distance from the ground truth. We note that these two scores, albeit different in the threshold, can still be compared meaningfully in our experiments. In the PCK code we use, the half-head distance was actually set to 1/20 of the cropped image size. This distance is usually shorter than the half-head length of the characters, and is close to 0.2 of the torso length (length from e.g. left shoulder to right hip). However, for inappropriately scale images, for instance due to our annotation conversion, this assumption fails.

The drawings dataset provides a division of the 2,000 drawings into training (1,500), validation (100) and test (400) sets. Because we did not attempt to optimize over hyper-parameters, we use the validation set for evaluation[9].

5.2 Pre-Train Model

A pre-trained 1-stack hourglass model gives a PCKh@0.5 score of 0.5130 on the validation set; that is, out of every 100 joints, the pre-trained model predicts about 51 joints within a half-head distance from the ground truth. In Figure 2, we can see the challenge posed by drawings. The bottom image is close to being photo-realistic, and the pre-trained model predicts most joints correctly. Note that, while the right hand of the character is not found by the model, the left and right ankles are predicted correctly. It seems that, for drawn figures with realistic appearance, the pre-trained model works reasonably well. Even though we validate the pre-trained model on the drawings dataset, we use the mean and standard deviation from MPII dataset during data normalization. This ensures that the full knowledge of human pose estimation is being assessed. The validation accuracies can be found in Table 1.

Some of the PDJ results from Khungurn et al. [2] are shown in Figure 4. While Khungurn et al. evaluated their model on the test set and we evaluated on the validation set, we will assume that the results are still comparable. Khungurn et al. trained a modified GoogLeNet on their synthetic dataset consisting of over 1,000,000 examples, which they refer to as SYNTH. Assuming that PDJ@0.2 and PCK@0.5 are roughly comparable, we find that the pre-trained SH performs almost as well as SYNTH. Human pose estimation methods may prove promising on anime characters.

The pre-trained model performs poorly on the other two images in Figure 2. This shows the large variance in art style that confuses the pre-trained network. We therefore further fine-tune this network on the drawings dataset and our synthetic dataset.

5.3 Fine-Tuning

We perform fine-tuning by initializing the model with pre-trained weights and freezing all weights except the prediction layer. We use the mean and standard deviation from the drawings or synthetic datasets as appropriate.

The validation accuracies of the 1-stack hourglass network fine-tuned on the drawings dataset are shown in Table 1. This fine-tuning took 98 epochs. The training loss went down

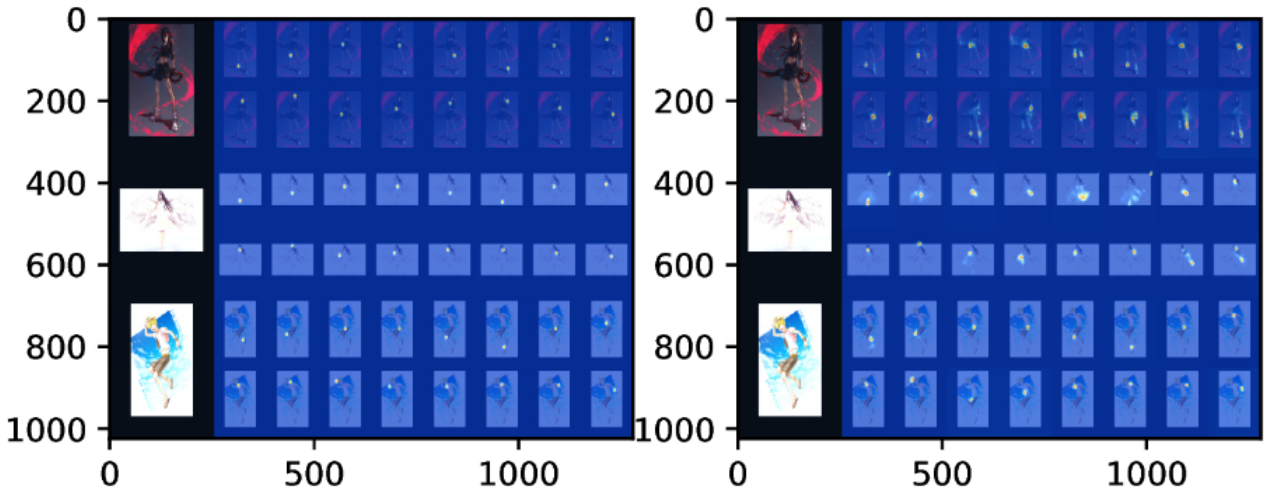

Figure 2: Three representative images from drawings validation set. On the left is the ground truth, and on the right is the prediction by pre-trained model with 1 stack. Note that the prediction of the bottom image closely matches the ground truth, while the other two don't.

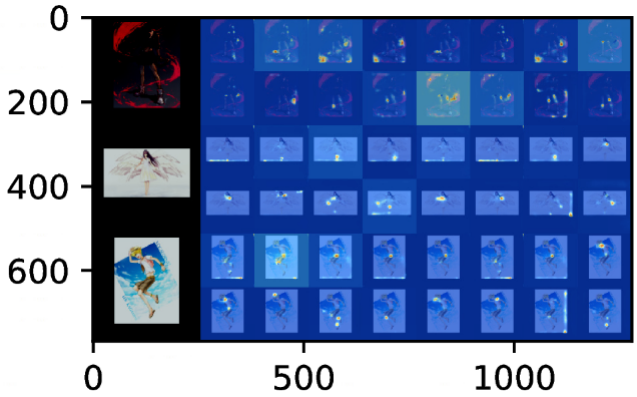

Figure 3: Model initalized with pre-trained weights, and trained on the drawings dataset. In the top drawing, we see that the fine-tuned model tends to guess all similar joints without discriminating left and right.

from 6.99×10^{-4} at epoch 1, to 6.62×10^{-4} at epoch 98; the validation loss went down from 6.73×10^{-4} at epoch 1, to 6.34×10^{-4} at epoch 98[10]. The losses only decrease by around 5%, so that there is still much room to further train the model. The losses decrease more slowly as epochs increase. As can be seen in Figure 3, compared to the pre-trained model, the fine-tuned model tends to predict all similar joints indiscriminately and start to confuse left and right body parts. This suggests that the dataset may not be suitable for training.

The validation accuracies (on drawings dataset) of the 1-stack hourglass network fine-tuned on our synthetic dataset are show in Table 1. This fine-tuing took 240 epochs. The training loss went from 5.88×10^{-4} to 5.12×10^{-4}, and the validation loss on our synthetic dataset went from 5.65×10^{-4} at epoch 1 to 4.95×10^{-4} at epoch 240[10]. There is still much room for further training. We note that after fine-tuning, the validation accuracies dropped below that of the pre-trained model. This could be due to our small pool of anime characters, sequences and backgrounds.

[10]We do not include training and validation accuracies because an error in our code caused incorrect numbers to be recorded.

6 CONCLUSION

We demonstrate that knowledge from human pose estimation can be transferred to the domain of anime characters. We apply a 1-stack hourglass model, pre-trained on MPII human pose dataset, to the anime drawings. We find that the performance was competitive against the model trained by Khungurn et al. on 1,000,000 synthetic images [2].

We also demonstrate the use of our prototype script which exports joint annotations for Maya. A similar tool could be developed and released to Maya users to aid anime pose estimation tasks.

Acknowledgements. We would like to thank Professor Dinesh K. Pai for his support throughout our project, and the robotics lab for access to GPU.

REFERENCES

[1] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. *CoRR*, abs/1603.06937, 2016.

[2] Pramook Khungurn and Derek Chou. Pose estimation of anime/manga characters: A case for synthetic data. In *Proceedings of the 1st International Workshop on coMics ANalysis, Processing and Understanding*, MANPU '16, pages 3:1–3:6, New York, NY, USA, 2016. ACM.

[3] https://www.gwern.net/Danbooru2017, 2017.

[4] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V. Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. *CoRR*, abs/1511.06645, 2015.

[5] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. *CoRR*, abs/1312.4659, 2013.

[6] Bearpaw. bearpaw/pytorch-pose. https://github.com/bearpaw/pytorch-pose.

[7] Dragonmeteor. dragonmeteor/animedrawingsdataset, Jun 2015.

[8] Mikumikuflex. https://archive.codeplex.com/?p=mmflex.

[9] Pramook Khungurn. Approximation of mmd ik. https://github.com/dragonmeteor/AnimePoseProject/blob/master/yumyai_jg/src/yumyai/mmd/pmx/PmxIkSolver.java.

PCK@0.5 Scores by Joints on Drawings Dataset			
Joint Name	Pre-trained Model	Drawings Trained	Synthetic Trained
r ankle	0.5686	0.5758	0.4747
r knee	0.5392	0.5253	0.4141
r hip	0.5490	0.5859	0.4343
l hip	0.5784	0.5859	0.5152
l knee	0.5882	0.5859	0.5354
l ankle	0.5588	0.5960	0.5556
pelvis	0.6078	0.6364	0.5657
t thorax	0.0784	0.0606	0.1414
upper neck	0.8333	0.8384	0.8182
head top	0.3333	0.3434	0.3030
r wrist	0.3333	0.3232	0.2626
r elbow	0.4608	0.4545	0.3535
r shoulder	0.6765	0.7071	0.6263
l shoulder	0.7059	0.7071	0.5657
l elbow	0.3725	0.4040	0.3535
l wrist	0.4235	0.4444	0.3131
average	0.5130	0.5234	0.4520

Table 1: Accuracy by joints on drawings validation dataset. Gray entries are less reliable but included for completeness. In particular, pelvis, thorax and head top are joints added when annotations were converted into the Mpii format. The pelvis location was the arithmetic average of left and right hips, thorax was "body upper", and head top was offsetting upwards the neck location by a fixed amount. The generated synthetic dataset has invalid left wrist annotations, so that the performance of synthetic trained model was negatively affected.

Methods	Body	Head	N.root	Elbows	Wrists	Knees	Ankles
SYNTH	65.0	85.4	78.4	56.8	39.6	49.3	54.5
DRAW	60.0	68.6	60.4	35.2	17.6	39.6	33.1
SYNTH-DRAW	**78.6**	**89.0**	**79.4**	**64.1**	**50.3**	**61.0**	**59.6**

Figure 4: PDJ results by Khungurn et al. [2] on the drawings test set. Their network was a modified GoogLeNet mostly initialized with pre-trained weights. This network trained on their 1,000,000 synthetic images gives SYNTH; the network trained on the 2,000 drawings data gives DRAW; SYNTH fine-tuned with drawings data gives SYNTH-DRAW.

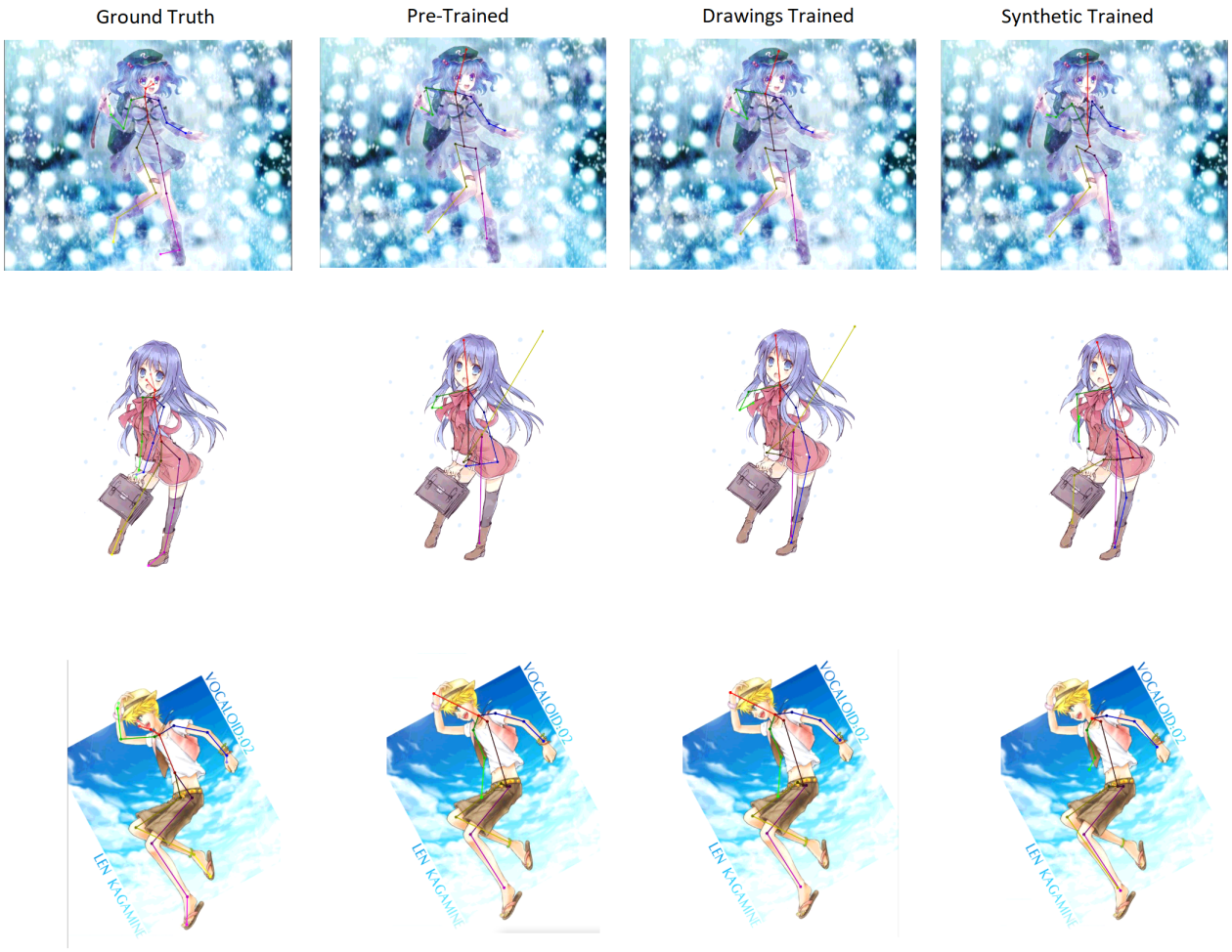

Figure 5: Visualization of pose predictions from differently trained 1-stack hourglass model. The leftmost column is the ground truth with annotation in the drawings dataset format. The conversion of ground truth to MPII format is not shown here. The other 3 columns use the MPII format.