# Neural Techniques For Pose Guided Image Generation

Ayush Gupta
Department of EE
Stanford University
ayushg@stanford.edu

Shreyash Pandey
Department of EE
Stanford University
shreyash@stanford.edu

Vivekkumar Patel
Department of CS
Stanford University
vivek14@stanford.edu

## Abstract

*Generating realistic-looking images conditioned on user expectations is a hugely potent field of possibilities for neural techniques based computer vision models. With multiple applications ranging from movie editing to data augmentation, creating images that fulfill the user intentions and desired manipulations is however a difficult task. Inspired by the PG$^2$ model in [11], we work on one such representative problem of synthesizing person images based on a reference image and desired pose. Embedding explicit pose information with the reference image in 2 stages, the U-Net-like architecture in Stage-1 learns the pose which is further refined through adversarial training of a U-Net-like generator in Stage-2. To improve performance, the model is modified to incorporate Wasserstein GAN and a triplet classification based discriminator architecture. The modified model gives a better score on the used metrics: Structural Similarity, Inception Score, and mean of absolute image gradients. The models have been trained, analyzed and tested on the DeepFashion dataset [10]. Experimental results show that the model carries the capacity to perform pose modification of an image with fine details.*

## 1. Introduction

The task of image generation conditioned on user expectations has found immense traction with the research community, both for its interesting techniques and the immense industrial promise it carries. Consequently, a wide variety of models have been employed in the recent past to address this problem of estimating a probability distribution of the latent code implicitly or explicitly [11] [4] . However, Generative Adversarial Networks (GANs) have become hugely popular for their ability to generate high quality sharp images through adversarial training. Coupled with an architecture that embed the intended transformation of pose, GANs can be significantly handy in solving the problem at hand.

In this project, we focus on transferring a person's image in a certain pose to an intended pose specified by pose points. Since this involves transferring a reference image to a defined pose with detailed appearances simultaneously, splitting the task into two stages has been found more successful than an end-to-end learning framework [11]. Therefore, the process is divided into two stages: 1) Stage-1: The correct pose is modeled; 2) Stage-2: The quality of the generated images are enhanced by filling in appearance details through adversarial training.

For training, the input to our model is a 3-channel condition image (the reference image) concatenated by 18 channels of target pose points obtained using a state-of-the-art pose estimator [3] from the target image. The pose points, which correspond to different joints of the human body estimating pose, are also used to generate a morphological pose mask for supervised training of Stage-1 by a Pose-Mask loss. The obtained coarse results are further concatenated with the condition image and fed through a conditional DCGAN. The conditional DCGAN is adversarially trained to produce a difference map for improving high-frequency details in the image by a triplet classification. The expected output is a trained image that bears identical resemblance to the person in the condition image but carries the target pose.

The report is organized as follows. After conducting a brief review of the recent works associated with pose guided image generation in Section 2, we discuss the used methods in Section 3. Further, the specifications about the dataset are provided in Section 4 followed by the implementation and experimentation details in Section 5. The obtained performance results are also discussed. Finally, conclusions and scope for future work are noted in Section 6.

## 2. Related Work

Employing deep learning approaches to perform generative image modeling falls into two categories: One which follows an unsupervised setting; the other that conditions image generation on attributes, categories, text, or images.

Some of the popular methods from the former approach are variational autoencoders (VAEs), autoregressive models, and GANS. VAEs do re-parameterization to maximize the lower bound of the data likelihood and solve for ef-

ficient inference learning in presence of continuous latent variables with intractable posterior distributions [7] [15]. Another popular technique in this field is that of autoregressive models. These estimate the joint distribution of pixels in an image as a product of conditional distribution of pixels in a pixel-by-pixel manner. Architectures like PixelRNN [14] and PixelCNN [18] belong to this category of works. The most popular among these, though, are GANs which generate realistic images but don't use any explicit density function. Instead, they take a game-theory approach and learn to generate from training distribution through a 2-player performance game between the generator and discriminator network.

Many researchers have also explored generating images conditioned on various parameters and choices. Yan *et al.* worked on generating images from visual attributes in their proposed model of conditional variational autoencoders [20]. In another work [13], a conditional version of generative adversarial nets were introduced which were constructed by feeding the condition data on to both the generator and discriminator. In a more related work that used a two-stage pipeline for conditioned image generation, an image-based generative model of people in clothing for the full body was proposed [9]. The first stage learned to generate a semantic segmentation of the body and clothing, and the second stage learned a conditional model on the resulting segments to create realistic images.

Researchers have also worked on performing photorealistic frontal view synthesis from a single face image and rotating an arbitrary pose and illumination image to a target-pose face image while preserving identity [6] [21]. In a similar problem as ours, Zhao *et al.* proposed VariGANs combining variational inference GANs to generate multi-view cloth images from only a single view input. To make the conditioning more expressive, Ma *et al.* proposed a Pose Guided Person Generation Network (PG$^2$) to synthesize person images in arbitrary poses based on an image of that person and a novel pose [11]. Using poses in the format of keypoints to model diverse human appearance, they are able to make use of pose information in a more explicit and flexible way.

The (PG$^2$) model has shown promising results generating realistic images with correct poses. However, GANs are seeing interesting developments, particularly in the direction to improve its training. One such recent development is Wasserstein GAN (WGAN) [1]. Using Wasserstein distance as GAN loss functions, WGAN is claimed to improve the stability of learning and get rid of problems like mode collapse. Incorporated with the (PG$^2$) model, the extended model provides promise to a more stable training. Thus, this work reimplements the (PG$^2$) model to incorporate WGAN and a triplet classification based discriminator architecture for better performance.

# 3. Method

Given an image of a person and a pose, we want to generate an image of that person in the given pose. Hence, the input will necessarily contain a condition image $I_A$ and pose information for the target as $P_B$.

As mentioned before, we extend the PG$^2$ model to incorporate WGAN and a triplet classification based discriminator architecture to improve the model's performance. The first part of the pipeline includes a pose estimator that extracts pose key-points for all images in our dataset. We then use a two-stage framework to generate an image conditioned on the appearance of the reference image and the desired target pose. Stage-1 combines the target pose with the condition image to produce a coarse result. This coarse output is refined by Stage-2 to incorporate finer details in order to make the generated image appear more realistic. The following sub-sections describe the model pipeline in detail. We first discuss the building blocks of the PG$^2$ model as used by us (shown in Figure 1), followed by a note about the experimented modifications.

## 3.1. Pose Generation

The poses for each target image are generated using a state-of-the-art pose estimator [3]. The pose-estimator takes an image as input and outputs set of 18 keypoints corresponding to different joints of the human body that estimate the pose. These keypoints are used to create 18 channels of pose input where each channel specifies a pose point. Since one pixel is not enough to adequately specify the pose point, the pose points are denoted by a value of 1 around a radius of 8 to each keypoint and 0 otherwise. This gives us the target pose $P_B$. For the downstream tasks, we also create a pose mask by connecting these points to form a skeleton, and then using morphological transformations such as dilation and erosion to form a rough estimate of a human body mask $M_B$.

## 3.2. Stage-1

The main purpose of this stage (denoted as $G_1$) is to generate a coarse result $\hat{I}_{B1}$ by combining the appearance of the person in condition image $I_A$ with the target pose $P_B$. We use a U-Net like architecture [12] similar to an auto-encoder. The input to this architecture is the concatenated result of $I_A$ and $P_B$. This makes sure that we can directly use convolutional layers to combine the pose and appearance information. The encoder consists of 6 residual blocks of convolutional, ReLU and max pool layers that each downscale an image by a factor of 2, followed by a fully connected layer. The decoder consists of nearest-neighbour upsampling layers followed by convolutional and ReLU layers that are symmetric to the encoder. According to [11] transposed convolution layers lead to similar performance. The skip connections in the architecture help in

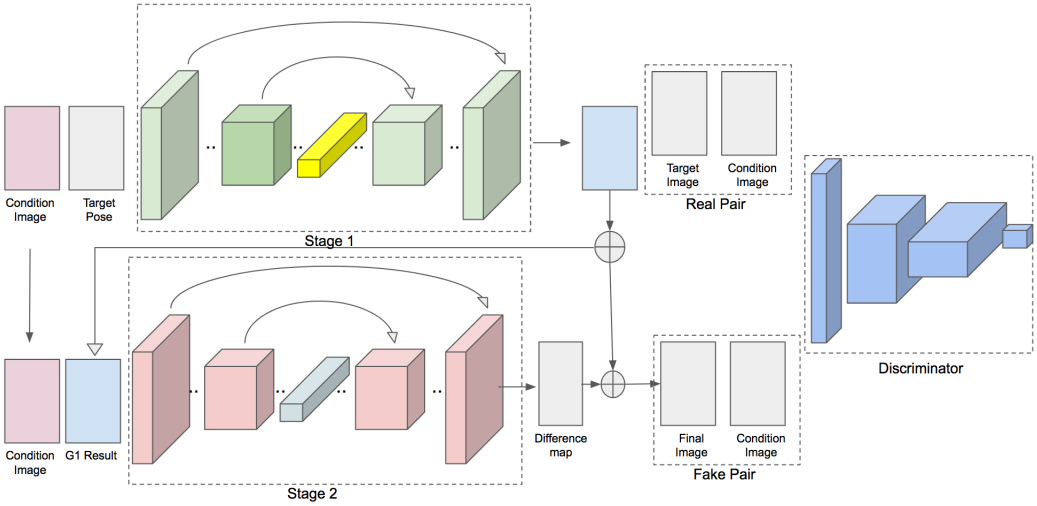


Figure 1. Model architecture from [11]

passing appearance information from the condition image $I_A$ to the target image $\hat{I}_{B1}$.

#### 3.2.1 Pose-Mask Loss

The first stage undergoes supervised training, meaning that for a condition image $I_A$ and target pose $P_B$, we have a target image $I_B$. The input to $G_1$ are condition image $I_A$ and target pose $P_B$ and the output is $\hat{I}_{B1}$. Similar to [11] we adopt L1 distance as a metric of similarity between the generated image $\hat{I}_{B1}$ and target image $I_B$. To ensure that stage 1 models the pose and is unaffected by background changes in the target image, we use a pose mask $M_B$ that is 1 for the foreground and 0 for the background. Mathematically,

$$L(G_1) = ||(G_1(I_A, P_B) - I_B) * (1 + M_B)||_1$$

However, this loss function leads to blurry outputs as L1 loss encourages the result to be an average of all data points. This necessitates another stage to model the fine details of clothing and other bodily features.

### 3.3. Stage-2

$G_1$ successfully captures the global structure of the human body in the target pose but leads to blurry outputs. To improve this, we use another stage $G_2$ which consists of a conditional DCGAN. GANs, with their adversarial training, are known to generate sharp, realistic looking images [5]. As we already have a coarse version of the output $\hat{I}_{B1}$, we use this image and the condition image $I_A$ as the input to the generator, which outputs a difference map $I_D$. The final output is then the sum of the difference map and the coarse result $\hat{I}_B = \hat{I}_{1B} + I_D$. The advantage of generating the difference map as the output of $G_2$ is that now we only need to learn to capture few missing features in the image, rather than generating the whole image from scratch.

$G_2$ consists of encoder-decoder architecture where the encoder includes 6 residual blocks of convolutional, ReLU and max pool layers that each downscale an image by a factor of 2. However, unlike $G_1$, it does not have a fully connected layer. The decoder consists of symmetric nearest-neighbour upsampling layers followed by convolutional and ReLU layers.

### 3.4. Discriminator

In commonly used GAN architectures, the discriminator $D$ differentiates between real-looking images and fake-generated images. In our case, we want the image to be real and also similar to the target image. Moreover, it's possible that the network can learn identity mapping since the condition image is always real. To prevent this from happening, we follow [11] and provide pairs of images for the discriminator to differentiate. The real pair is made up of the condition image and the target image $(I_A, I_B)$, whereas the fake pair is made up of the condition image and the generated image, i.e $(I_A, \hat{I}_B)$. This makes sure that the generated image should not only look real but also be similar to the target image $I_B$. Mathematically, the loss functions for the stage 2 generator and discriminator can be written as a function of Binary Cross Entropy loss $L_{BCE}$ in the following way :

$$L(G_2) = L_{BCE}(D(I_A, \hat{I}_B), 1)$$

$$L(D) = L_{BCE}(D(I_A, \hat{I}_B), 0) + L_{BCE}(D(I_A, I_B), 1)$$

### 3.5. Modifications

While the above mentioned architecture and training regime works well, we noticed that it takes a long time to

converge and is highly sensitive to correct hyperparameters and weight initializations. Therefore, we have also implemented two modifications to the $PG^2$ model with an aim to improve the model performance. The following subsections provide more detail about the tried modifications.

#### 3.5.1 Triplet Classification

The architecture in Figure 1 involves feeding pairs of images $(I_A, I_B)$ and $(I_A, \hat{I}_B)$ to the discriminator to make sure that the generated image is not only real looking but also similar to the target image. A simplification to this architecture is to feed in a triplet $(I_A, \hat{I}_B, I_B)$ to the discriminator instead of the pairs of images. Under this regime, shown in Figure 2, the discriminator no longer differentiates between "real" and "fake" examples but classifies whether a given image is the target image or not. Therefore, $I_B$ has a label of 1, and both $I_A$ and $\hat{I}_B$ have their labels as 0. The loss functions are then written as follows:

$$L(G_2) = L_{BCE}(D(I_A, \hat{I}_B), 1)$$
$$L(D) = L_{BCE}(D(I_A, \hat{I}_B), 0) + L_{BCE}(D(I_B), 1)$$

Note that unlike the previous architecture where the pairs were concatenated along the channels and constituted as a single example, here the images are concatenated along the batch and constitute different examples.

This formulation makes sure that in order to fool the discriminator, the generator has to come up with an image that is not only real (competing with $I_A$ and $I_B$), but also close to the target image. Since the condition image $I_A$ always has a label of 0, our model learns to stay away from the identity function. This simplifies the learning process considerably and our model converges in much fewer iterations.

#### 3.5.2 Wasserstein GAN (WGAN)

WGAN is a new technique for training GANs introducted by Arjovsky et al [1]. Its algorithm is as follows:

Parameters: $G$, $D$
while G has not converged:
  for $t$=0...$n_critic$ do:
    $X_r \leftarrow$ Batch of real data of size m
    $z \leftarrow$ Batch of samples from prior $p_z$ of size m
    $dD = \nabla_w \frac{1}{m}(\sum_{i=1}^{m} D((X_r)_i) - \sum_{i=1}^{m} D(G(z_i)))$
    $D \leftarrow D + lr * RMSProp(D, dD)$
    $D \leftarrow clip(D, -c, c)$
  end for
  $z$ = Batch of samples from prior $p_z$
  $dG \leftarrow \nabla \sum_{i=1}^{m}(D(G(z_i)))$
  $G \leftarrow G - lr * RMSProp(G, dG)$
end while

The WGAN algorithm has a few differences from the traditional GAN algorithm. It involves training the discriminator to convergence for the current generator and also clips the coefficients of the discriminator after every iteration. Not going into the mathematical details, we mention some claimed benefits of this algorithm. The WGAN algorithm is said to remove the requirement of maintaining balance between the generator and discriminator as in normal GANs, along with removing the need of properly designed architectures. The authors also argue that this approach reduces the problem of mode collapse (the problem where if the true data distribution has multiple modes, the generator learns to capture only one or few and the discriminator learns the rest but both of them never learn the entire distribution together) to a large extent.

## 4. Dataset and Features

The dataset used is the DeepFashion (In-shop Clothes Retrieval Benchmark) dataset [10]. It consists of 52,712 number of in-shop clothes images and around 200,000 cross-pose/scale pairs. Each image has a resolution of 256 x 256.

The dataset was filtered for non-single and complete images and split into 32,031 images for training and 7,996 images for test. The training set was augmented by performing left-right flip while removing the generated non-different images. Thus, a total of 57,520 training images were available. The 18 pose keypoints from all the training and test images were estimated using a state-of-the-art pose estimator [3] and stored as pickle files. These keypoints were also used to generate a human body mask by morphological transformations such as dilation and erosion. Finally, we obtained 127,022 training examples composed of a condition image and target image of the same person but in different poses. Each example also contained 18 channel pose information from the target image, with value 1 around a radius of 8 to each keypoint and 0 otherwise; and a single channel pose mask of the target image with value 1 for the filled human pose and 0 otherwise. Similarly, 18,568 test examples were generated for comparing model performances. Both, the training and test examples, were saved and processed as HDF5 files. Figure 4 shows one such example with the 18 channel pose point input shown as a single image for visualization.

Further, the inputs were normalized before feeding into the model. The image pairs were normalized by a mean of 127.5 and a standard deviation of 127.5. The poses were normalized to be between the range of -1 and +1.

## 5. Experiments/Results/Discussion

This section gives details about the conducted experiments, results and a discussion for the same. For brevity,

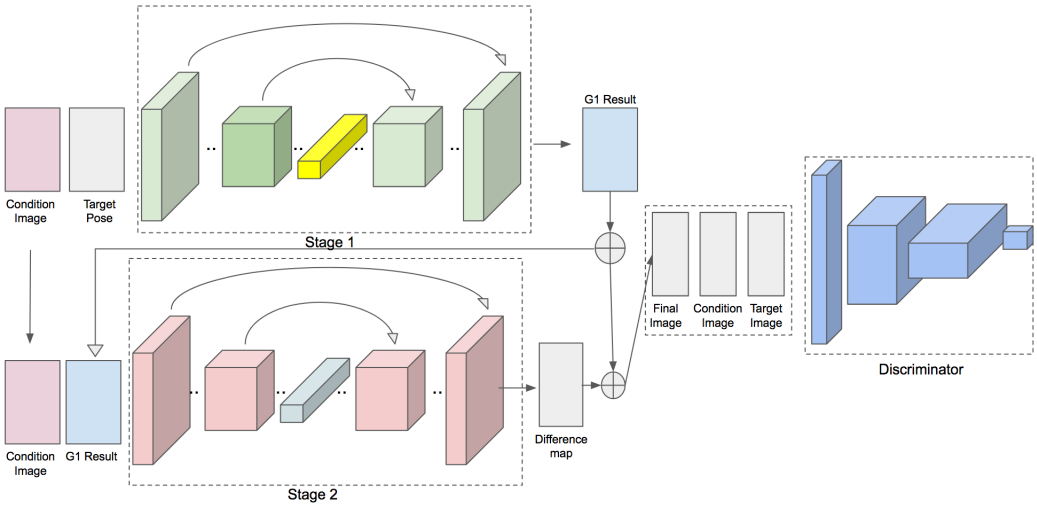


Figure 2. Model architecture with Triplet Classification

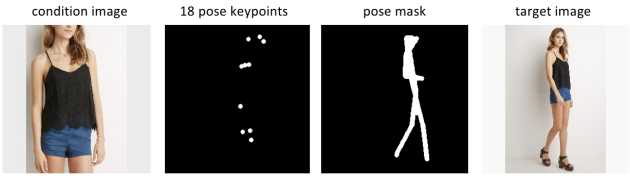


Figure 3. A sample example

stage-1 is denoted as G1, stage-2 is denoted as G2 and the discriminator is denoted as D. Also, the abbreviations as listed in Table 2 have been adopted while mentioning compared results. The coding platform used is Python with Pytorch 0.4 as the deep learning platform. A GPU from Google Cloud Services was also utilized to reduce training time.

Also, it must be noted here that the model requires a lot of time to train for which we lacked both, resources and time. Therefore, the results mentioned will be subpar to the fully trained results obtained from [11].

| Abbreviation | Context (Result from) |
|---|---|
| G1 | Stage-1 |
| G1+G2+D | Stage-2, disc. from [11] |
| Triplet | Stage-2 + triplet classification |
| WGAN-Triplet | Stage-2 WGAN + triplet classification |
| Ma *et al.* [11] | G1+G2+D from [11] |
| Target | Target image set |

Table 1. List of used abbreviations

### 5.1. Hyperparameters

GANs are notoriously difficult to train and it requires a right set of hyper-parameters to obtain good results. To find a starting point, we followed some commonly suggested training hacks [17] to achieve good training of our model. The stage-1 was trained for 40k steps, followed by 25k steps for stage-2 training. Intuitively, Stage-1 needs to be properly trained before stage-2 begins training. Otherwise, it becomes very difficult for the training to converge if both stages train together. Further, the learning rates for $G_1$, $G_2$ and $D$ were each set to 5e-5. The batch-size was kept at 4.

Optimizers play a crucial role in the training of the model. For the PG$^2$ model, Adam optimizer is used for all, $G_1$, $G_2$, and $D$. However, in the WGAN implementation, we use RMSProp for $G_2$ and $D$ as suggested in the original paper [1]. We still use Adam Optimizer for $G_1$.

### 5.2. Quantitative Results

This section discussed the used metrics and obtained results for performing quantitative comparison between the original PG$^2$ model and the modified model, namely: Structural Similarity, Inception Score, and Image Gradients for image sharpness.

#### 5.2.1 Structural Similarity (SSIM)

SSIM is a widely used metric to measure image quality, claimed to be a better evaluator than Peak Signal-to-Noise Ratio (PSNR) or mean-squared-error (MSE) [8] . This metric quantifies the similarity between two considered images calculated on windows of them as:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

Here $\mu_x, \mu_y$ are the means of the windows, $\sigma_x, \sigma_y$ are the standard deviations, $\sigma_{xy}$ is the covariance. $c_1, c_2$ are constants to maintain numerical stability during division.

Table 2 gives the SSIM scores for the obtained results. The target image is given a score of 1 for reference. As can be noticed, WGAN-triplet model obtains a better score for SSIM, followed by the model just using a triplet classification for DCGAN training.

| No. | Set | SSIM |
|---|---|---|
| 1 | G1 | 0.72 |
| 2 | G1+G2+D | 0.71 |
| 3 | Triplet | 0.727 |
| 4 | WGAN-triplet | 0.73 |
| 5 | Ma et. al [11] | 0.762 |
| 6 | Target | 1 |

Table 2. SSIM score for various models and target set.

#### 5.2.2 Inception Score (IS)

Inception Score is another commonly used metric for evaluating Generative Models. It was introduced by Salimans et al. [16] where they found that it correlated well with human judgment of the generated image's visual appearance and reality. The metric uses the Inception-v3 network pretrained on Imagenet Dataset and calculates the following quantity over the outputs of the network on the generated images:

$$IS(G) = \exp(\mathbb{E}_{x\sim p_g} D_{KL}(p(y|x)||p(y)))$$

.

Here $p_g$ is the distribution learnt by the generator. $x \sim p_g$ denoted the images generated by the generator, $D_{KL}(p||q)$ is the KL divergence between distributions $p$ and $q$, $p(y|x)$ is the class distribution conditioned on image $x$ and $p(y) = \int_x p(y|x)p(x)dx$ is the marginal class distribution.

This metric was proposed to capture two desirable qualities in images: 1. The images should be clear (not blurry); and 2. That the generator should be able to generate images from diverse classes and not just a few. It must be noted here that a recent work has suggested that this metric may not be useful in all cases and should be used with caution [2].

We used IS to compare the model performances and list the obtained scores in Table 3. The reported score is the $mean \pm std$ calculated across 10 splits.

| No. | Set | IS |
|---|---|---|
| 1 | G1 | $2.58 \pm 0.441$ |
| 2 | G1+G2+D | $2.993 \pm 0.319$ |
| 3 | Triplet | $3.01 \pm 0.383$ |
| 4 | WGAN-triplet | $3.02 \pm 0.326$ |
| 5 | Ma et. al. [11] | 3.091 |
| 6 | Target | $3.25 \pm 0.354$ |

Table 3. Inception scores for various models and target set.

As can be observed, the WGAN-triplet model achieves a good inception score compared to the target set and falls in the good range of generally reported scores [16]. However, replacing GAN with a WGAN shows little effect on this metric. Though, more extensive training and hyperparameter tuning may have widened the score gap.

#### 5.2.3 Image Gradients to measure sharpness

As mentioned before, the role of stage-2 is to improve image sharpness and appearance deriving from the output of stage-1. A very naive and simple metric to quantify this improvement is to view the gradients of the generated images. A real-number metric can then be the mean of the absolute gradients which can offer a relative comparison between the generated images and target images.

To compute the gradient, we use built-in functions of the skimage library [19] with a disk-shape element of size 5. Table 4 lists the obtained scores. Though the absolute values change with the size of the disc, the order of the values always remains the same: $G1 < G2 < Target$. This implies that although the generated images are not that sharp, but stage-2 still offers an improvement over the results of stage-1. It may also be noted here that the actual score values do not have much significance, but its the order that makes an inference base.

| No. | Set | Mean Absolute gradient |
|---|---|---|
| 1 | G1 | 27.37 |
| 2 | G1+G2+D | 41.61 |
| 3 | Target | 48.22 |

Table 4. Image gradients at different stages. The G1+G2+D stage is from WGAN-triplet model.

Figure 6 shows gradient images of some examples for visual comparison.

### 5.3. Qualitative Results and Discussion

The quantitative results confirm the benefits of modifying PG$^2$ model to use WGAN with a triplet classification training. This section gives some qualitative results from the WGAN-triplet model with a discussion about the obtained results. In our observations, other models generate

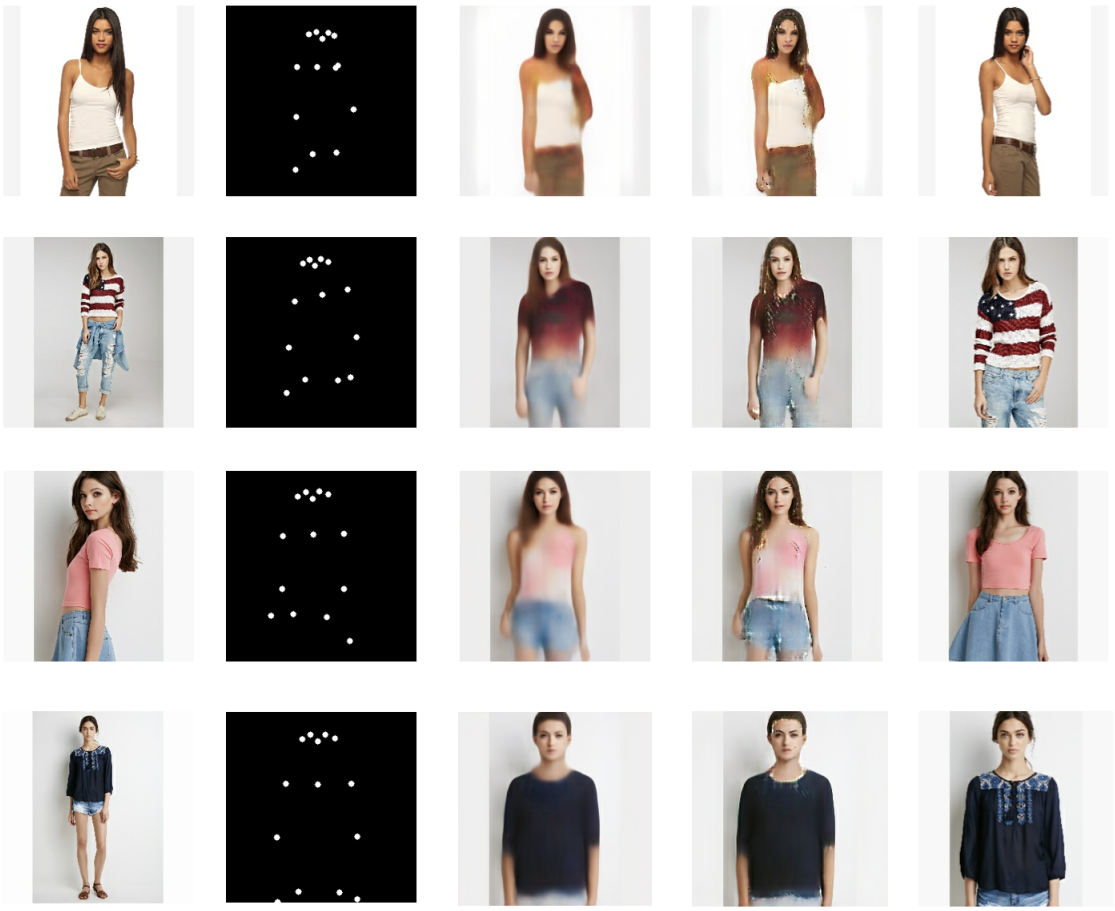

Figure 4. Results of WGAN-triplet model. From left to right: condition image, target pose, G1 output, G2 output, target image

similar results with the differences becoming pronounced only on zooming in.

Samples of the final results for this model are shown in Figure 4, where we compare the stage-1 and stage-2 results. The auto-encoder model in stage-1, $G_1$, is able to capture the pose information really well but leads to blurry outputs. The conditional DCGAN, $G_2$, refines the coarse result and sharpens the image. Facial features and clothing textures refinements are clearly visible.

For critical analysis, we also show the failure modes of our model in Figure 5. Target poses that lead to closeups of people is one mode in which our model fails to generate real looking images. This is because such poses are fewer in number in the training set. We also observed that attires such as jackets that look different from different angles are hard to generate and often lead to absurd outputs.

Moreover, due to large dataset size and heavy architecture, this model needs quite a lot of time to train after figuring out the right set of hyper-parameters. Though we demonstrated decent results with comparable metric scores in the previous section, they can be further improved if both the stages are trained more and with better hyper-parameters.

Comparing the results of the two stages also confirms the fact that adversarial training helps in generating realistic looking images. Also, the WGAN training process was found to be more robust than vanilla GANs. However, it may be noted here that the differences in the generated outputs are not significant to be perceivable by naked human eye.

## 6. Conclusions & Future Work

This work extends the PG$^2$ model [11] to incorporate Wasserstein GAN for stable training and triplet loss classification based discriminator architecture for added model benefits. After a discussion about the model architectures, various model combinations are compared quantitatively based on Structural Similarity, Inception Score, and mean of absolute image gradients. Using WGAN with a triplet loss led to better performance. Further, some qualitative results for the success and fail cases were presented and the

Figure 5. Failures of WGAN-triplet model. From left to right: condition image, target pose, G1 output, G2 output, target image

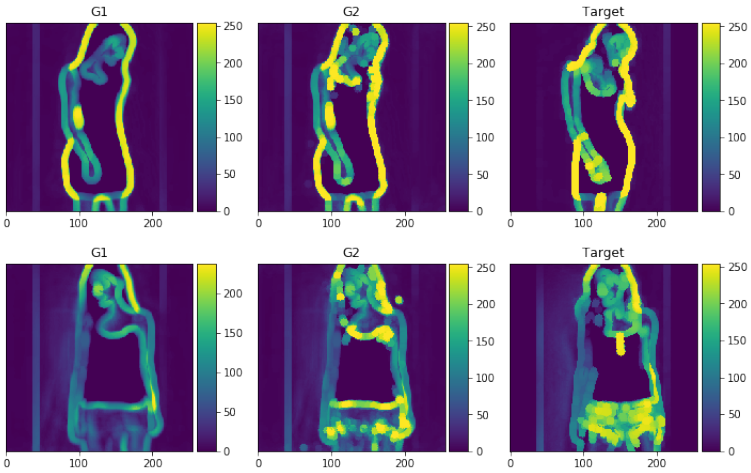


Figure 6. Visualization of image gradients of the WGAN-triplet model. Stage-2 captures more sharpness and is closer to the target images.

model performances were discussed.

An interesting extension to this project can be to convert this into a semi-supervised setting. VAEs usually learn a latent representation, a complex function of the pose and appearance of the person, that we generally cannot understand. Separating pose information (supervised) from image appearance (un-supervised) by using partial supervision can allow generation of images in arbitrary poses [4]. Combining them with a GAN can then lead to generation of real looking images.

Also, $G_1$ makes a crucial part of the model. It combines the pose and appearance information, strongly influencing the model's performance. Therefore, a natural improvement of this work can be to better $G_1$.

## 7. Contributions & Acknowledgements

AG implemented the data procuring, preparation, and loading pipeline, including generating poses from the pose estimator and writing the data loader. SP implemented the training and testing pipeline including the auto-encoder and conditional DCGAN model architectures. VP implemented

WGAN, experimented with various ways of training the model and worked on the quantitative metrics. AG, SP and VP contributed equally to the report.

We would like to thank the entire CS231N course staff and especially Amani for his encouragement in the initial stages of our project. We used the starter code provided in Assignment-3 for numerically stable BCE loss implementation.

The codebase along with all the relevant data links is provided in the repo: https://github.com/ayushgs/PoseGuided.

## References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. *CoRR*, abs/1701.07875, 2017.

[2] S. Barratt and R. Sharma. A Note on the Inception Score. *ArXiv e-prints*, Jan. 2018.

[3] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, volume 1, page 7, 2017.

[4] R. de Bem, A. Ghosh, T. Ajanthan, O. Miksik, N. Siddharth, and P. H. Torr. Dgpose: Disentangled semi-supervised deep generative models for human body analysis. *arXiv preprint arXiv:1804.06364*, 2018.

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. *ArXiv e-prints*, June 2014.

[6] R. Huang, S. Zhang, T. Li, R. He, et al. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. *arXiv preprint arXiv:1704.04086*, 2017.

[7] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[8] K. G. Larkin. Structural Similarity Index SSIMplified: Is there really a simpler concept at the heart of image quality measurement? *ArXiv e-prints*, Jan. 2015.

[9] C. Lassner, G. Pons-Moll, and P. V. Gehler. A generative model of people in clothing. *arXiv preprint arXiv:1705.04098*, 2017.

[10] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1096–1104, 2016.

[11] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool. Pose guided person image generation. In *Advances in Neural Information Processing Systems*, pages 405–415, 2017.

[12] T. Minh Quan and D. Hildebrand. Fusionnet: A deep fully residual convolutional neural network for image segmentation in connectomics. *ArXiv e-prints*, 2016.

[13] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[14] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

[15] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[16] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[17] M. A. M. M. Soumith Chintala, Emily Denton. How to train a gan. 2016.

[18] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems*, pages 4790–4798, 2016.

[19] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.

[20] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, pages 776–791. Springer, 2016.

[21] J. Yim, H. Jung, B. Yoo, C. Choi, D. Park, and J. Kim. Rotating your face using multi-task deep neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 676–684, 2015.