Controllable Face Generation via Conditional Latent Models

Grigorii Sotnikov[†]
Skoltech, Moscow, Russia

Vladimir Gogoryan[†]
Skoltech, Moscow, Russia

Dmitry Smorchkov[†]
Skoltech, Moscow, Russia

Ivan Vovk[†]
Skoltech, Moscow, Russia

[†] Guru Approach Team
Deep Learning Course Project
June 2020

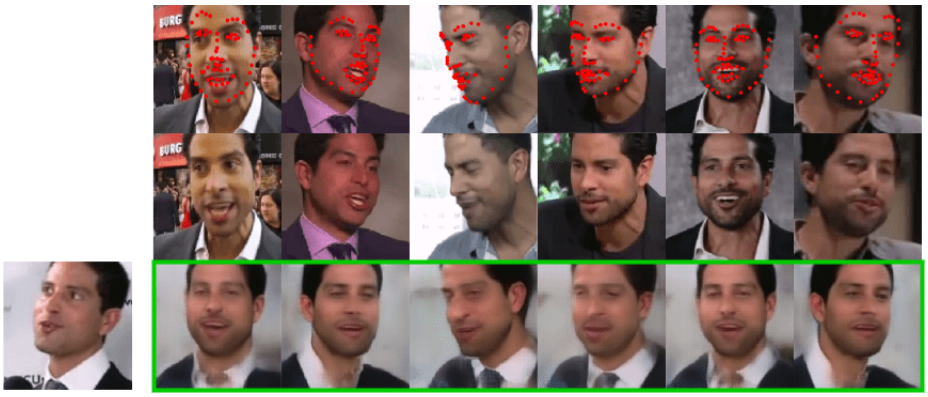

Figure 1: Results on facial keypoints transfer from target image to the source image using our solution, which via condition just manipulates the latent space of pretrained Adversarial Latent Autoencoder.

1 Introduction

In this paper we consider the photo-realistic controllable face generation task with transfer of target image attributes to the source image. As instance, for face attributes one can consider facial keypoints, hair or skin color, gender, age, glasses, etc. Generally, manipulation of these characteristics has a huge practical potential in such areas as photo editing, entertainment and video effects industry, etc. Nonetheless, obtaining plausible results with recent generative modelling approaches has a lot of problems connected with architecture limitations.

Current field of generative modelling comprises decades of investigations. Nonetheless, a lot of issues still remain being unsolved. Generative models zoo counts dozens of powerful architectures, but most of the them are divided into two major groups. First one is endowed with strong representative properties and follows autoencoding (AE) training strategies. This group explicitly models data and it's latent distribution [1, 2], which makes it possible to perform controllable generation and modify real data points. Nonetheless, such networks obtain weak restoration capabilities due to a pixel-wise losses optimization. Generative Adversarial Networks (GANs) [3] and their derivatives [4, 5, 6, 7] represent the second popular group of adversarially training models gaining the state-of-the-art generation power

(up to 1024×1024 image resolution), which withdraw pixel-wise losses optimization and play min-max game instead. Unfortunately, despite the existence of several promising attempts [8], generally, due to the concept restrictions such models face difficulties with real data reconstructions.

In order to obtain both reconstruction quality and manifold learning in lower-dimensional space we decided to use a novel approach called Adversarial Latent Autoencoders (ALAE) [9], which consolidated advantages of both AE and GAN approaches. While the model has state-of-the-art generation quality comparable to StyleGAN [7] also it is able to map real data points into the latent space and is able to sample from it. In more details the architecture will be reviewed in the next section. All in all, contributions of this work are as follows:

- We trained a baseline model using default AE for facial keypoints transfer and concluded its imperfection and limitations for this task.
- We implemented our own ALAE architecture and trained it on CelebA128 dataset with progressive generation strategy. The model has been successfully trained up to 64×64 resolution after which we stopped the procedure due to the resource-consuming process and switched to the publicly available pretrained solution.
- We tried to reach the results of [10] on "talking heads" task. Comparing to the authors' approach based on conditioning generator on given keypoints and pose-invariant face embedding, we show a working simple solution with CelebA128-finetuned ALAE, which manipulates pose estimation in the latent space directly. Training was accomplished on the part of VoxCeleb2 dataset.
- We successfully implemented Adversarially Constrained Autoencoder Interpolation (ACAI) [11]. The training was done on CelebA64 dataset with modified loss.
- We investigated the performance of conditional generation (gender and hair color) cGAN, AC-GAN, prGAN trained on top of ACAI-based latent codes.

2 Notation

In this section we set notation for different abbreviations, network architecture types, losses, etc. and will follow it during the whole paper:

- x - data point
- $\hat{x}$ - restored data point
- x_{real} - real data point
- x_{fake} - fake data point produced by generator
- x_s - source object
- x_t - target object
- x_{CI} - center identity object
- $z \in \mathbb{R}^d$ - d-dimensional latent code
- $c \in \mathbb{R}^k$ - k-dimensional condition
- $E(\cdot)$ - encoder network
- $G(\cdot)$ - generator network
- $D(\cdot)$ - discriminator network
- $C(\cdot)$ - critic network

– $F(\cdot)$ - mapping network

– $\mathcal{L}(\cdot)$ - loss

– $\mathcal{L}^G(\cdot)$ - loss optimizing w.r.t. network G

– $\mathbb{E}$ - expectation

– $\phi(\cdot, \cdot, n)$ - perceptual loss w.r.t. n levels of feature maps of VGG19

3 Related work

There is a big choice diversity of generative models. In this section we provide details on existing solutions and architectures, describe and discuss their training strategies, benefits and drawbacks.

3.1 Adversarially Constrained Autoencoder Interpolation

Generally, AE reconstruction quality highly benefits from well-structured latent space. In [11] authors build connection between the quality of latent space and ability to linearly interpolate between it's latent codes and provide quantitative evidence. In this case, mixing of two codes is defined in a way that $\hat{x}_\alpha = G(\alpha z_1 + (1-\alpha)z_2)$ for some $\alpha \in [0, 1]$ and $z_1 = E(x_1)$, $z_2 = E(x_2)$, where x_1 and x_2 are original data points. Authors consider two valuable characteristics of interpolations: the intermediate points along the interpolation should be indistinguishable from real data and provide a semantically smooth morphing between the latent codes of endpoints.

Here stands out an idea to use a regularizer which forces reconstructions of interpolated points to look more realistic or even indistinguishable from reconstructions of real images. To accomplish this issue a critic network (Adversarial Regularizer) is introduced. It's inputs are interpolations of existing data points (i.e. $\hat{x}_\alpha$ as defined above). This can be formulated as follows:

$$\mathcal{L}^C = \|C(\hat{x}_\alpha) - \alpha\|^2 + \|C(\gamma x + (1-\gamma)G(E(x))\|^2 \tag{1a}$$

$$\mathcal{L}^{E,G} = \|x - G(E(x))\|^2 + \lambda \cdot \|C(\hat{x}_\alpha)\|^2. \tag{1b}$$

However we observed that the application of this AE to the domain of facial images provides very blurry reconstructions, "averaging out" all personal information. To overcome this issue we have included perceptual loss in optimization objective:

$$\phi(x, G(E(x)), N) = \sum_{i=1}^{N} \frac{1}{C_i H_i W_i} \|VGG_{19}^i(x) - VGG_{19}^i(G(E(x)))\|_2 \tag{2a}$$

$$\hat{\mathcal{L}}^{E,G} = \mathcal{L}_{E,G} + \phi(x, G(E(x)), 3). \tag{2b}$$

3.2 Auxiliary Classifier GAN

In some tasks data distribution of images has several classes, so each sample from data is distributed according to its class. So, training procedure of Auxiliary Classifier GAN (AC-GAN) introduced in [12] takes this setup into account. Authors proposed additional terms to the loss of generator and discriminator to make the latter correctly classify classes of real samples. It forces generator to produce such samples under conditions so the discriminator will classify them correctly, corresponding to the condition. With regard to losses optimization process, if we denote x_{real} as original data point with its class c and $x_{fake} = G(z, c)$ as generated sample under condition class c then in terms of distributions we can formally

introduce two parts: $\mathcal{L}_s$ - log-likelihood of the correct predicted source of the sample S - whether it is real or fake, and $\mathcal{L}_c$ - log-likelihood of correct predicted class c:

$$\mathcal{L}_s = \mathbb{E}\log P(S = real|x_{real}) + \mathbb{E}\log P(S = fake|x_{fake}) \tag{3a}$$

$$\mathcal{L}_c = \mathbb{E}\log P(C = c|x_{real}) + \mathbb{E}\log P(C = c|x_{fake}). \tag{3b}$$

Discriminator maximizes $\mathcal{L}_s + \mathcal{L}_c$, while generator tries to fool it and maximizes $-\mathcal{L}_s + \mathcal{L}_c$.

3.3 cGAN and cGAN with projection discriminator

The majority of conditional GAN frameworks, for example, Conditional GAN (cGAN) [4], handles additional information by merely concatenating the encoded label representation to the feature vectors. In [13], projection discriminator was proposed to modify the conventional approach for conditioning. The key idea here is to replace the concatenation in a projection based way. The intuition here is that incorporation of labels by concatenation can work somewhat arbitrary and can find odd sets of functions, which cannot model desired probability distribution. For the sake of clarity, given input vector x and the conditional information y, we can express the standard adversarial loss for the discriminator in the following way:

$$\mathcal{L}^D = -\mathbb{E}_{q(y)}\left[\mathbb{E}_{q(x|y)}[\log(D(x,y))]\right] - \mathbb{E}_{p(y)}\left[\mathbb{E}_{p(x|y)}[\log(1 - D(x,y))]\right], \tag{4}$$

where q and p stand for true distribution and generator model respectively. A conventional approach of feeding y to D is a naive concatenation of the conditional information to the feature vector x, either at the input layer or at some hidden layer. Now, considering mapping D as some function $f(x, y)$, we can decompose optimal solution in the following way:

$$f^*(x,y) = log\frac{q(x|y)q(y)}{p(x|y)p(y)} = log\frac{q(y|x)}{p(y|x)} + log\frac{q(x)}{p(x)} = r(y|x) + r(x). \tag{5}$$

Both log likelihood ratios can be modelled by some parametric functions f_1 and f_2, which are considered as simple distributions. Therefore, we can modify the above Equation 5 as:

$$f(x,y;\theta) = f_1(x,y;\theta) + f_2(x;\theta) = y^T V h(x;\theta_h) + \psi\left(h(x,\theta_h);\theta_\psi\right), \tag{6}$$

where V is the embedding matrix of y. Discriminator now requires to take an inner product between the embedded condition vector y and the feature vector x, which is referred by authors as *projection*. Thus, we will further refer to it as prGAN in order to avoid confusion with ordinary cGAN. All parameters $\theta = \{V, \theta_h, \theta_\psi\}$ are trained to optimize the adversarial loss.

3.4 StyleGAN

To address the problem of producing high resolution realistic images the authors in [7] made several important contributions by adaptation of well-known style transfer technique in domain of GANs. It was the first paper to use Adaptive Instance Normalization (AdaIN) [14] as a core element of generator's architecture. It plays a role of scaling and shifting intermediate activations according to a given style, however, the term style itself is special too. Unlike previous works, authors avoid using noise vector z as the only input to the generator. They introduce a special mapping network F, which is implemented as MLP, taking a vector of noise z and returning a latent code w, interpreted as style. The main claim is that intermediate latent space $w \in \mathbb{W}$ is less entangled than original $z \in \mathbb{Z}$ which is measured using Perceptual Path Length. Also the authors suggest injection of noise into intermediate activations to add some stochastic variation into generated samples. To model more complex interactions between styles, a novel regularization called Style Mixing is used which adds generator additional flexibility by producing realistic samples dependent on multiple style vectors. This work builds upon previous best practices in training GANs

like progressive growing [15], allowing generating at 1024×1024 resolution and also produce samples of mixed styles which have high correlation with human perception in terms of identity attributes mixing, developing a theory about coarse, middle and fine styles in Style Generator's framework.

3.5 Adversarial Latent Autoencoders

A problem of designing autoencoder architecture which would be able to match generative potential of GANs, but maintain encoding-decoding properties is actively researched for the last several years. On the one hand, per-pixel-trained models can show good autoencoding capabilities and suffer from the lack of quality because of explicit mean squared error minimization. On the other hand, adversarially trained models demonstrate high-end sampling, but have a lot of issues with encoding capability of turning real datapoints into latent codes, thus having troubles with entangled representations. Adversarial Latent Autoencoder (ALAE), introduced in [9], makes a step towards the ultimate architecture that combines both features. The authors rely on three main observations:

1. Unlike the prior works where latent space is represented by probability distribution that is fixed and should be matched by AE, authors build their solution upon a strategy introduced in [7]. An intermediate latent space improving disentanglement properties is used. In other words, the model allows to learn latent distribution from data.

2. Produced image distribution is learned via adversarial strategy, allowing not to use pixel-wise ℓ_2-loss, often leading to more blurry reconstructions (and does not reflect human visual perception, also being very sensible even to small shifts of images).

3. Reciprocity in the latent space is imposed to match the distributions of intermediate latent codes, produced by mapping network F and the outputs of encoder E which is obtained by using ℓ_2-distance between them.

$$\mathcal{L}^{E,D} = \text{softplus}(-D(E(x))) + \text{softplus}(D(E(G(F(z), \eta)))) + \frac{\gamma}{2}\mathbb{E}_{p_{\mathbb{D}}(x)}\|\nabla_x D(x)\|_2 \tag{7a}$$

$$\mathcal{L}^{F,G} = \text{softplus}(-D(E(G(F(z), \eta)))) \tag{7b}$$

$$\mathcal{L}^{E,G} = \|F(z) - E(G(F(z), \eta))\|_2. \tag{7c}$$

First two losses represent a classical min-max game written in ALAE terminology (generator is split into two parts: fully-connected mapping network F and generator G; discriminator is also split into two parts: encoder E and fully-connected discriminator D). The third loss is novel and can be interpreted as "inverse autoencoding" as it is used not in the pixel space as usual, but in the latent space.

Unlike generator, which is totally adopted from StyleGAN, the design of encoder network is unique. The idea is that every layer of G is injected by a style input w and E is designed symmetrically. Then from a corresponding layer the style information is extracted. As style is represented in generator by scaling and shifting of feature maps, in encoder it is the same:

$$w = \sum_i^N C_i \begin{bmatrix} \mu(y_i^E) \\ \sigma(y_i^E) \end{bmatrix}, \tag{8}$$

where y_i^E is the output of i-th encoder layer encoder, μ and σ represent channel-wise calculation of mean, standard deviation and C_i is a learnable affine transformation. Mapping network F and discriminator network D are implemented as MLPs, which operate on latent codes and contain 8 and 3 layers respectively.

3.6 Neural Talking Heads Task

In [10] authors presented the first few-shot learning system for generating realistic human head movements, where they get rid of previous limitations of a huge dataset size for finetuning. A key ingredient here is that the system is able to initialize the parameters of both generator and discriminator in a person-specific way, such that just a few images are required to perform the update. Furthermore, this model generalizes even on portraits, when only one photo with implicit style is available.

The model consists of 3 essential networks. Embedder allows to extract information about identity of the person given the image and a set of corresponding landmarks. The generator network considers the prediction of embedder, which is plugged in as parameters of AdaIN, allowing conditional generation defining a particular person. Then given a set of landmarks, the pose of head for a fixed identity can be generated. To establish adversarial training a discriminator network is introduced, which compares realism between ground truth facial expression and one, produced by generator.

The training process is split into two stages: meta-learning, when the main goal is to get familiar with a diverse set of expressions and facial positions across a huge dataset, and a finetuning stage, devoted to refinement the predictions of generator for a defined user to overcome identity gap.

Finetuning process is organised in such way that it involves the direct optimization of parameters, which were computed by affine transformation during the training guaranteeing extra flexibility of a model. The loss for this procedure consists of perceptual (w.r.t. ImageNet&VGGFace-pretrained VGG networks) and adversarial parts.

4 Methods

4.1 Baseline on facial keypoints

We started our investigation of inserting information of the pose (face landmarks) into latent space on the example of our own autoencoder - pair (E, G). We trained it with the simple landmark classifier C in the bottleneck. Thus our loss consisted of two terms: $\mathcal{L} = \mathcal{L}^{E,G} + \mathcal{L}^{C}$ where for reconstruction loss $\mathcal{L}^{E,G}$ we used VGG perceptual loss - extracted features from first two blocks of pretrained VGG-19 network [16]. Next we explored naive arithmetics for generation from the latent space. The experiment was to take one sourse image x_s and one target image x_t with completely different landmarks l_s and l_t. Let z_s be a latent code of the sourse image. Then we considered the collection of top-100 closest landmarks to l_t by MSE and took the mean of their latent codes $\bar{Z}_t$ and also the mean latent code across all images $\bar{Z}$. Next we generated an output image $G(\hat{z})$ where:

$$\hat{z} = z_s + (\bar{Z}_t - \bar{Z}). \tag{9}$$

This added difference suppose to give kind of hidden representation of target landmark l_t to the sourse latent code.

4.2 Conditional sampling with GANs

Our preliminary goal is to build several baseline models, which will transfer simple domain-specific attributes of a dataset and demonstrate the performance of considered approaches. After that, we can switch to more sophisticated tasks like keypoints transfer. We first consider using Generative Adversarial Networks, acting on the latent codes of images. As there is a wide variety of conditional architectures in GANs, we stop at the following models:

- Conditional GAN (cGAN)
- Auxiliary Classifier GAN (AC-GAN)
- cGAN with Projection Discriminator (prGAN)

Despite the distinctions in architecture and adversarial objectives, the main difference is how every model wires the conditional information into the network. We enlightened particular details were in Section 3. We apply considered GANs to the CelebA dataset. The main difference with typical adversarial learning examples is that we work only in the space of latent codes obtained from ACAI model. Thus, both generator and discriminator can be viewed as a composition of linear layers with non-linearities. We sample from the distribution of embeddings of real pictures to transfer distinctive attributes of a human, represented in the dataset. As mentioned approaches does not imply any, we constrain the generator with cycle-consistency loss, identity preserving outputs. Thus its overall adversarial objective transforms in the following way:

$$\mathcal{L}^G = \mathcal{L}_{adv} + \lambda \mathcal{L}_{cycle}, \tag{10}$$

where $\mathcal{L}_{adv}$ stands for conventional adversarial loss of generator and λ controls the effect of penalizing samples with bad identity presence. We will further discuss the effect of adding this additional term to the optimization process.

4.3 Our ALAE implementation

The paper is very much inspired by StyleGAN [7] as well as the training procedure. To begin with, ALAE is trained in progressive fashion [15], adding new blocks allowing to generate images at higher resolution after processing fixed number of training samples. It is proved to be not only an effective way of resources utilization but also has an intuitive interpretation of giving network more freedom to add more details in the image and preserve several levels of abstractions. For training we took CelebA dataset and used a progressive training at scales 4, 8, 16, 32, 64, 128 (with batch sizes 128, 128, 64, 64, 32, 16 to fit into a single GPU) using Style Mixing starting from scale 8 with p=0.9. At each scale training took 12 epochs (2.2 millions of images): 6 to smoothly blend a new block and 6 to stabilize training. However we followed proposed training techniques, the transitions from scale 16 to scale 32 (same as from 32 to 64) often caused high instability (as in Figure 10) of loss and sometimes immediately led to mode collapse. But sometimes the outcome of this transitions were not that straightforward however significantly harmed the diversity of generated samples (e.g. produced only females, Figure 9).

Zero-centered gradient penalty was used with the coefficient $\gamma = 10$. In all experiments with model we were using Adam with $\beta_1 = 0$ and $\beta_2 = 0.99$. The default learning rate (lr) was set to 0.0015, however some scheduling techniques were tested. One of them was decreasing learning rate after adding a new block and linearly scaling it to default values but in some scenarios it involved imbalance into training process, giving a huge advantage to Discriminator. As in [7], learning rate of mapping network F (lr^F) is defined as $lr^F = 0.01 \cdot lr$.

We would like to mention that despite the fact that the paper is written in a beautiful way, we assume that some implementation details are intentionally hidden by the authors and could be reproduced only by descent amount of time devoted to original implementation in GitHub. The most confusing we found the fact that actually discriminator contains no activations, meaning that it is just a linear mapping. Learning rate for discriminator is also set to $lr^D = 0.1 \cdot lr$.

4.4 Conditional ALAE

Taken ALAE pretrained model, we decided to perform controllable generation by modifying latent codes. For conditional input we can take anything: landmarks, face class labels, etc. In order to train such a model, we used several popular approaches, which are discussed in more details in this section.

4.4.1 Source-target facial keypoints transfer

One of the problems of ALAE is it's identity gap during generation. This means that given real face image, it's restoration quality would be great, but the human identity would

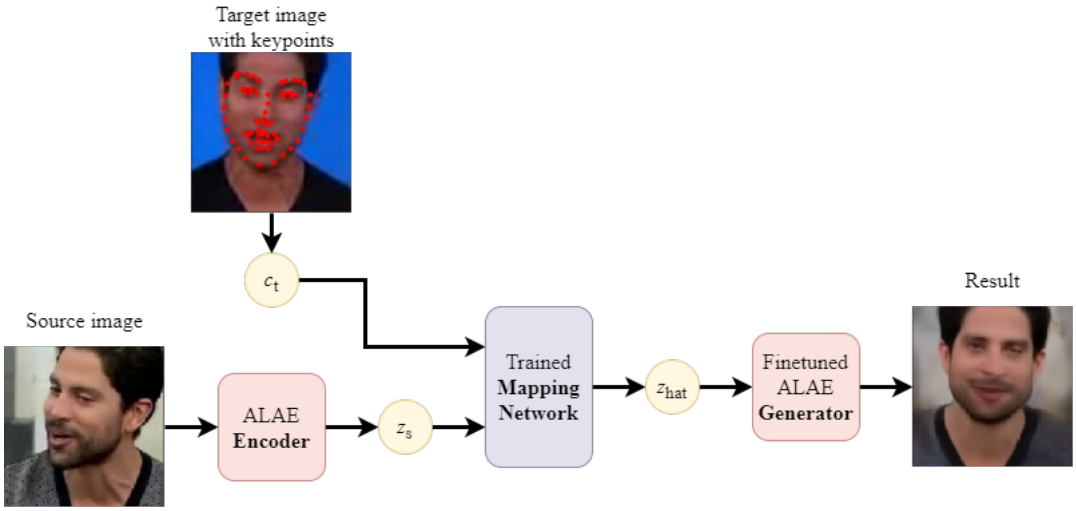

Figure 2: Inference of proposed ALAE-based solution on "talking heads" task.

change. In order to overcome this issue given a new face from VoxCeleb2 dataset we have tried two approaches:

1. Non-parametric finetunning on a fixed set of unseen face latent codes:

$$z^* = \underset{z}{\operatorname{argmin}} \left[\phi\left(G(z), x, 3\right)\right]. \tag{11}$$

2. Parametric finetunning of generator by minimizing perceptual VGG19 loss, which we, eventually, have settled down with as it provided better results:

$$G = \underset{G}{\operatorname{argmin}} \left[\phi\left(G(z), x, 3\right)\right]. \tag{12}$$

Also, during experiments we have captured the clear dependence between the detailed restoration quality (including eyes, mouth, etc.) of unseen face and the final quality of keypoints transfer.

Approach proposed in [10] uses Embedder, which produces pose-invariant face embedding and feeds it into the generator, conditioned on the target facial keypoints. Now, in new terms of having mapping ALAE, we decided to simplify the generation process, investigated and show that face pose estimation can be done directly in the latent space of pretrained model. So, potentially, once you have a dataset with a new face, you can solve the "talking heads" task on already well-trained generative model having autoencoding properties.

In the first place, we added auxiliary (see inference process on the Figure 2) 68×2-dimensional keypoints conditional MLP network $R(z_s, c_t) : \mathbb{R}^{d+k} \to \mathbb{R}^d$ in order to manipulate latent codes of source images. Further, we added MLP critic network $C(z, c)$ and set the training procedure to be tied on adversarial optimization game of the following losses:

$$\mathcal{L}\left(z_s, c_t\right) = C\left(R\left(z_s, c_t\right), c_t\right) - C\left(z_s, c_s\right) \to \min_C \tag{13a}$$

$$\mathcal{L}\left(z_s, c_t\right) = -C\left(R\left(z_s, c_t\right), c_t\right) + \mathcal{L}_{cycle} \to \min_R, \tag{13b}$$

where $\mathcal{L}_{cycle}$ is cycle loss from CycleGAN [6] and follows the formula:

$$\mathcal{L}^R_{cycle}\left(z_s, c_s, c_t\right) = \left\|z_t - R\left(z_s, c_t\right)\right\|_1 + \left\|z_s - R\left(R\left(z_s, c_t\right), c_s\right)\right\|_1. \tag{14}$$

It is worth to mention, that for training stability we took 8 consequent frames x^i_{CI} of speaker video to compute an average latent code $z_{CI} = \frac{1}{8}\sum_{i=1}^{8} G(x^i_{CI})$ of pose-invariant center identity and treated it as z_s.

5 Results

5.1 GAN attributes manipulation via latent codes

The experiments were conducted using the latent space provided by ACAI, trained on CelebA64. They revealed that given a properly trained encoder, the complexity of generator and discriminator can be fairly low. Precisely, we have experimented with MLPs of depth 1-6 and haven't observed much difference in performance. However there were some insights that we have outlined:

- For models, dependent on concatenating latent codes with labels a principal aspect is proper mixing of modalities. Most experiments involved the strategy of projecting vector of labels to $\nu \cdot |z|$, where $\nu \in [0, 1]$. Empirically we have observed that taking ν close to 1 will result in disregarding information from latent codes themselves, producing almost constant output depending on the label.
- As latent code is already a compressed representation of a given object, projecting it to lower dimensional subspace led to meaningless results, where the manifold structure was lost.
- $\mathcal{L}_{cycle}$ proved to be essential to preserve common features between original latent code and it's reconstruction after cycle completion.
- Dropout regularization (of input/intermediate activations) performed surprisingly well and we consider two reasons of it's effectiveness. The first one is that it prevents generator and discriminator from exploiting some patterns in the input ignoring the others. Another important fact is that data is already efficiently compressed, so each coordinate in the latent code contains some facial description and zeroing it has a natural interpretation of excluding particular attributes from person's face. It drastically increases stochastic variation during training and as a result the model is able to explore interactions between latent codes better.
- A negative aspect of such approach was found while transferring hair color (between black and blond). CelebA dataset is certainly biased: hair color is highly correlated with gender. Then obtained latent space imposes a strong prior on a joint distribution of gender and hair color. Without direct access to image pixels GANs can not overcome this issue, though becoming biased too.

We strive to change the hairstyle and gender of a person. cGAN performs at both condition tasks reasonably well. In Figure 4, we can see hairstyle transfer learned on the latent codes of the images. Here the network learned blond hair to associate with women, which is certainly a bias of CelebA dataset. In Figure 5 we depict gender transfer. We applied dropout here because, without it, the generator sticks to conditioning and did not use latent codes to preserve the person's identity. Setting dropout to $p = 0.4$ lead to the best results, so the model started generating realistic mappings of a human. However, performing transformations on the samples, which are narrowly represented in the data distribution, is expected to fail, as with the black person.

We also gained similar consistent results with AC-GAN and prGAN, which are illustrated in Figure 6 and Figure 7, respectively. Therefore, our results prove the concept of manipulating attributes in the latent space and generating realistic images with given conditional information.

5.2 Facial keypoints transfer

Figure 1 clearly shows that our mapping network successfully learned how to rotate speaker's head directly in the latent space according to conditioned facial landmarks and is able to slightly transfer the mimic of target, although the reconstruction quality is far from ideal and photo-realistic. Blurry images and bad mouth transfer we refer to several problems, which our solution experiences:

- Identity gap minimization. Although we succeeded in overcoming this issue at some level, we noticed that: firstly, sometimes ALAE generator restores another mouth and eyes poses, which inevitably affects the keypoints transfer, and, secondly, VGG19 loss injects grid texture into real image reconstructions. So, one way to improve our solution is to pay more attention to finetunning process.
- ALAE is pretrained on CelebA128 dataset, which is not so much pose-diverse as it required. One possible solution is to train ALAE from scratch on a part of VoxCeleb2 dataset.
- Facial landmarks labelling sometimes dramatically fails in unexpected situations (check out the first column on the Figure 1). Thus, it should be payed more time to face alignment model hyperparameters tunning (e.g. number of keypoints). Also, we suppose that such effects could appear after images downsampling to 128×128 resolution.

Finally, we fed source image and target video sequence keypoints to the model in order to obtain "talking heads" result. Gained video image had stable smooth frame transitions and proper match with target face position. Nonetheless, the model showed weak capabilities in exact transferring mouth and eyes movements due to the problems discussed above.

6 Conclusions

In this work we have studied the question of taking the best from both worlds by combining models with autoencoding and generative properties, which are trained in adversarial fashion. Our key finding is that the application of appropriate adversarial strategy on top of disentangled representations learned by two types of AE: ACAI and ALAE can be used to successfully solve various attributes manipulation tasks. Such approach allows us using fairly simple architectures on top of carefully trained latent codes to perform non-trivial manipulations and emphasizes the importance of unsupervised disentangled representations learning.

References

[1] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013.

[2] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," 2017.

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[4] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014.

[5] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," 2016.

[6] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2017.

[7] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," 2018.

[8] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," 2016.

[9] S. Pidhorskyi, D. Adjeroh, and G. Doretto, "Adversarial latent autoencoders," 2020.

[10] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky, "Few-shot adversarial learning of realistic neural talking head models," 2019.

[11] D. Berthelot, C. Raffel, A. Roy, and I. Goodfellow, "Understanding and improving interpolation in autoencoders via an adversarial regularizer," 2018.

[12] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," 2016.

[13] T. Miyato and M. Koyama, "cgans with projection discriminator," 2018.

[14] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," 2017.

[15] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," 2017.

[16] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," 2016.

7 Code references

- We used publicly available ALAE implementation by paper authors: `https://github.com/podgorskiy/ALAE`.
- For landmark detection we used open source library face-alignment: `https://github.com/1adrianb/face-alignment`.
- The scripts of all our implementations and experiments are available on `https://github.com/ivanvovk/controllable-face-generation`.

8 Team contribution

8.1 Grigorii Sotnikov

Grigorii was responsible for ALAE implementation. This paper involved recent ideas in Adversarial learning like progressive growing, style generator, mapping network and others which were written by him from scratch. As ALAE itself is a novel idea, the only implementation is done by the authors and according to our knowledge it is the first complete reimplementation available. Also he was responsible for finetuning pretrained ALAE to solve the task of identity gap minimization for keypoints transfer.

8.2 Vladimir Gogoryan

Vladimir was the main contributor of ACAI. He implemented the algorithm, training procedure and debugged it to work properly. Then Vladimir implemented and trained cGAN with projection discriminator on the set of obtained latent codes. He checked its performance on transferring hair color attribute. He considered experiments with GAN architectures aka grid-search. Finally, Vladimir revealed the importance of dropout in GANs on latent codes and explained its motivation.

8.3 Dmitry Smorchkov

Dmitry explored feature arithmetics in the latent code of baseline AE in order to transfer facial keypoints. Next part of work was about conditional sampling with GANs trained on the set of latent codes produced by ACAI network. He implemented AC-GAN and cGAN choosing the proper architecture with training them on CelebA64 with labeled class of hair color and explored the quality of obtained transferring this attribute.

8.4 Ivan Vovk

Ivan also contributed in the implementation of ACAI as he inserted the perceptual loss from pretrained VGG19. He trained ACAI on the large dataset CelebA64 and obtained final latent space for experiments with GANs. Also Ivan prepared dataset VoxCeleb2 and proceeded landmarks labelling. Next part of his work was related to ALAE-based facial keypoints transfer where he has implemented Adversarial strategy to make realistic facial manipulations after several experiments. Whole team thanks Ivan for creating presentation video and for great work at organisation of GitHub repository.

Appendix

Figure 3: Smooth morphing between latent codes, obtained by ACAI.

Figure 4: Hair color transfer using cGAN. 2 lines above black line represent model without Dropout. For 2 last it is set to 0.5.

Figure 5: Gender transfer using cGAN. In the first 4 columns (before vertical red line), the source image (in green box) is mapped into female gender, on the right side - to the male. We can see five lines of images below the green box, each of them contains predictions of independently trained GAN (MLP, 4 layers ν = 0.5), with all hyperparameters being identical except Dropout of intermediate activations (0, 0.2, 0.4, 0.6, 0.8). We can see that above yellow box generations suffer from the lack of variation and below (especially in the case of Dropout=0.8) lack of details.

Figure 6: Hair color transfer using AC-GAN. Top line - reconstructions of source images, middle - source images mapped to dark hair, bottom - source images mapped to blond hair.

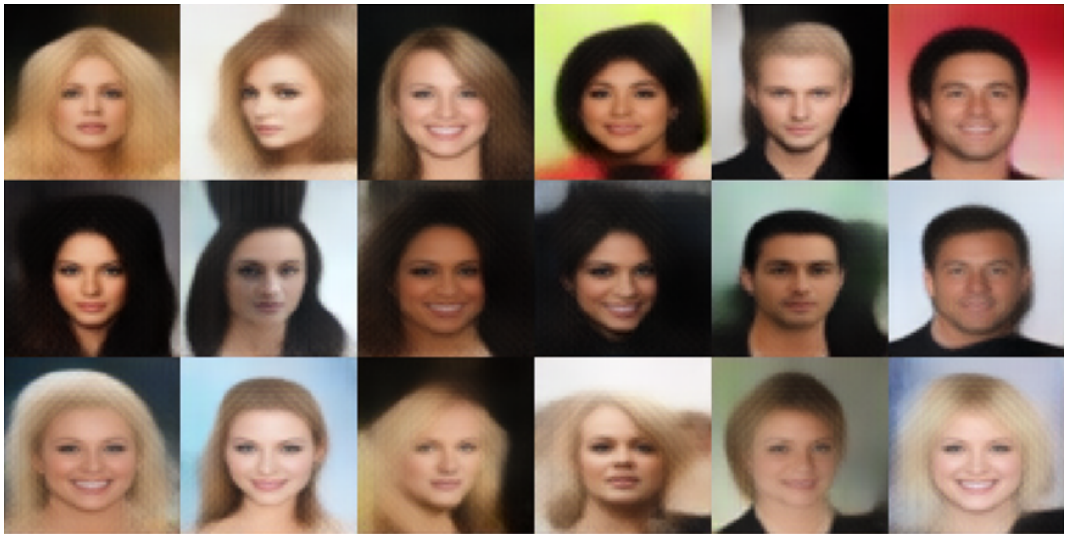

Figure 7: Hair color transfer using prGAN. Top line - reconstructions of source images, middle - source images mapped to dark hair, bottom - source images mapped to blond hair.

Figure 8: Generating with trained ALAE at 64x64 resolution. Cherry picked 25 samples after generation from 200 noise vectors distributed according $\mathcal{N}(0, 1)$.

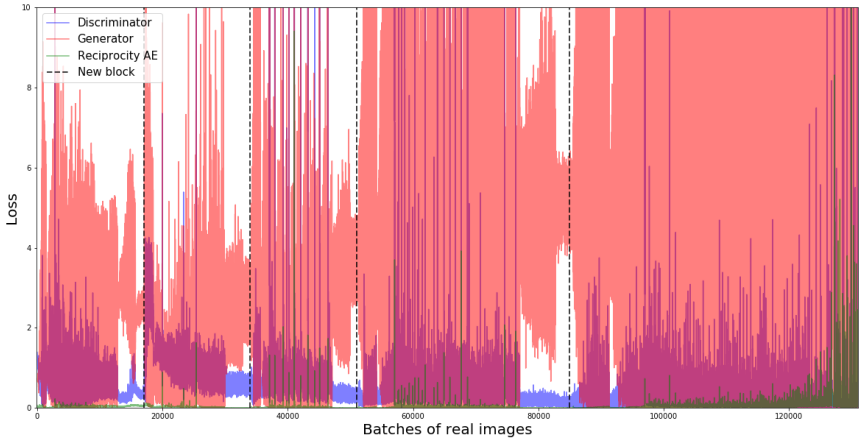

Figure 9: Example of training instability of ALAE. It is noticeable that at the end of each scale, loss becomes more stable as result of lr scheduling. However at scale 64x64 such strategy always fails.

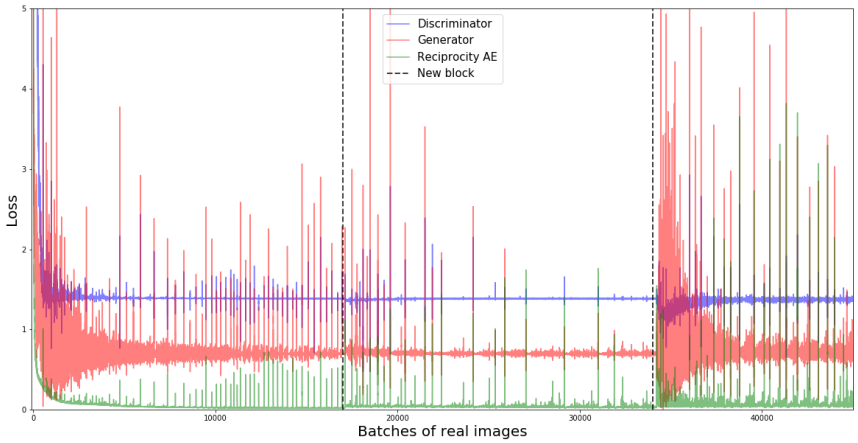

Figure 10: Taking into consideration the loss curve above several modifications were used. Namely, we found out that discriminator was linear. Here also zero-centered penalty is included. It is clearly seen that loss of generator and discriminator are balanced and reflect the authors results.