Appendix

A. Detailed Experimental Settings

A.1. Architecture

We use the model based on the improved version of DDIM [20]. We use linear beta scheduling for β_t from 0.0001 to 0.02 with $T = 1000$. This model has UNet structure with the blocks that consist of the residual and the attention blocks. With the number of base channels as 128, the number of channels is multiplied by $[1, 1, 2, 2, 4, 4]$ for each block in downsampling layers respectively, and spatial size is down-scaled by half. It is reversed in the upsampling layers. Attention resolution is $[16]$. The dimension of $z_{\texttt{face}}$ is 512. Time is first embedded into 128 dimensional vector by positional encoding and projected to 512 dimensional vector using a 2-layer MLP with SiLU activation. In each residual block, the time embedding for diffusion modeling and the face feature $z_{\texttt{face}}$ are first transformed by their corresponding SiLU-Linear layers respectively and these conditions are applied by AdaGN. In more detail, after the input of the residual block is passed to GroupNorm(32)-SiLU-Conv3x3-GroupNorm(32), each channel is scaled and shifted using time embedding. Similarly, after SiLU-Conv3x3-GroupNorm(32), channels are scaled and shifted by the transformed $z_{\texttt{face}}$. Final block output is obtained after SiLU-Dropout-Conv3x3 following skip connection of the block input. We refer the readers to the implementation code for more details.

A.2. Training Configuration

We optimize the learnable parameters jointly on 77294 videos of VoxCeleb1 dataset [18]. The videos are aligned and cropped for interesting face regions as in Tzaban *et al*. [35]. We use 4 V100 GPUs and an Adam optimizer [13] with a learning rate of 1e-4. Total training steps are 1 million and 4 frames per video so a total of 16 frames for 4 videos are taken for a single training step.

A.3. Manipulation

Classifier-based editing The linear classifiers $C_{\texttt{attr}}$ are trained on CelebA-HQ with attribute annotations in the normalized identity feature space. The classifier is optimized for 10 epochs with the batch size of 32 with a learning rate of 1e-3. Before taken by the classifier, identity features are normalized by the mean and standard deviation of identity features of all samples in VoxCeleb1 test set. Therefore, normalization and denormalization are conducted before and after the identity features are moved by the desired direction $w_{\texttt{attr}}$ as $\ell_2\texttt{Norm}(\texttt{DeNorm}(\texttt{Norm}(z_{\texttt{id}}) + sw_{\texttt{attr}}))$ where s is the hyperparameter for the editing step size, $\texttt{Norm}/\texttt{DeNorm}$ is normalizing and denormalizing function with the statistics of identity features respectively, and $\ell_2\texttt{Norm}$ is the normalization function that makes the ℓ_2 norm of vectors equal to 1. We use ℓ_2 normalization because $E_{\texttt{id}}$ outputs vectors after normalizing their size to 1.

CLIP-based editing For CLIP-based editing, we use ViT-B/16 among different CLIP architectures. We optimize an identity feature of a single selected frame (usually the first frame of the video) with the Adam optimizer to minimize the CLIP loss. We consider $S = 5$ for the number of intermediate latent states. After conduction optimization, the learned editing direction $\Delta z_{\texttt{id}}$ multiplied by the editing step size is added to the representative identity feature of the video $z_{\texttt{id},\texttt{rep}}$. The final edited feature is obtained by applying $\ell_2\texttt{Norm}$. The search spaces of remaining hyperparameters are provided in Tab. 3.

Table 3. Hyperparameter search space

Parameter	Search space
Learning rate	[2e-3, 4e-3, 6e-3]
Weight of CLIP loss	[3]
Weight of ID loss	[1, 3, 5]
Weight of ℓ_1 loss	[1, 3, 5]
Editing step size	[0.1, 0.5, 1.0, 1.5, 2.0, 2.5]
Optimization steps	[2000]

B. Ablation of Noisy CLIP Loss

In this section, we conduct an ablation study of our noisy CLIP loss introduced in Sec. 4.2. We compare our CLIP-based editing method using noisy CLIP loss (See Fig. 3) with the way that uses estimated x_0 conditioned by $z_{\texttt{id}}^{\texttt{opt}}$ as the target image and the original x_0 as the neutral image for each time step t_s to compute the directional CLIP loss, which is similar to the method suggested by Kim *et al*. [12]. The results are presented in Fig. 8. For a fair comparison, we use the same weights for ℓ_1 loss and ID loss.

To help readers understand, we first briefly explain the directional CLIP loss [7] and DiffusionCLIP [12] before we address the ablation results. The directional CLIP loss compares the direction from neutral image embedding to target image embedding with the direction from neutral text embedding to target text embedding in the CLIP space to edit the target image to match the target text. Kim *et al*. [12] propose an image manipulation method, named DiffusionCLIP, that optimizes the unconditional diffusion model ϵ_θ with the directional CLIP loss [7]. To preserve the original images to some extent, Kim *et al*. [12] consider the latent states of the original images in not the whole but just a partial range such as $[0, T/2]$ obtained by sparsely passing through the range with the deterministic forward process of DDIM. In the GPU-efficient version of DiffusionCLIP, they take an es-

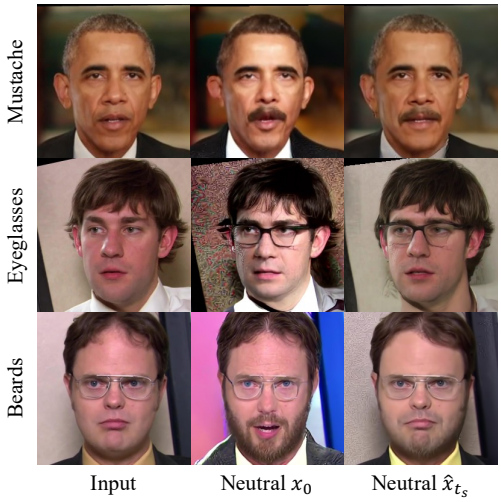

Figure 8. Ablation of using clean image x_0 or intermediate noisy image $\hat{x}_{t_s}$ for a neutral image in directional CLIP loss. As the target image, the former uses estimated x_0 at the intermediate time step like [12] and the latter uses $x_{t_s}^{\texttt{edit}}$.

timated x_0 from the latent states in considered intermediate time steps as the target images and compare them with the clean original image x_0 as the neutral image.

Going back to the ablation study, unlike Kim *et al*. [12], we consider the entire range $[0, T]$ to start conditional sampling from the noise that only has background information and split the range into total S steps (*e.g*. $S = 5$) for computational efficiency. In this case, applying CLIP loss between the original image (neutral) and the estimated x_0 (target) makes identity to be altered as in the second column of Fig. 8 because the difference between the estimated x_0 at time t and the clean image x_0 becomes larger as t goes larger. To overcome this phenomenon, we apply CLIP loss between the intermediate outputs $\hat{x}_{t_s}$ (conditioned on original $z_{\texttt{id}}$ for the neutral images) and $x_{t_s}^{\texttt{edit}}$ (conditioned on trainable $z_{\texttt{id}}^{\texttt{opt}}$ for the target images). In the last column of Fig. 8, the original identity is well preserved with the desired features edited properly. From these results, we conclude that applying the CLIP loss between the images on the same level of uncertainty as our method leads to relatively stable editing results.

C. Additional Quantitative Results

C.1. User Study

Current video editing tasks still lack metrics to measure how well the video is edited or whether the edited video is temporally consistent. Although quantitative results have already been reported with metrics used by the prior work [35], we further conduct the user study for sufficient evaluation of editing quality and temporal consistency. Since we use both classifier-based and CLIP-based editing methods, we choose Tzaban *et al*. [35] which allows both predefined attributes and CLIP-based editing as a baseline. For a fair comparison, the hyperparameters of baseline, α and β, are carefully determined among the $4, 8, ..., 24$, and $0.1, 0.2, 0.3$ respectively.

52 volunteers were asked to select the superior result between the edited output of prior work [35] (GAN-based) and ours (with $T = 100$ for fairness in time-cost) on 24 videos. The evaluation covers two aspects; 1) *quality*: the given target attribute should be properly reflected in the video, and 2) *consistency*: consecutive frames continue naturally after editing.

In Tab. 4, 61.9% and 66.3% of the volunteers favor our method in terms of editing quality and temporal consistency, respectively. In particular, certain attributes (such as a beard or eyeglasses; called 'fragile' in the Tab. 4) exhibit a noticeable lack of temporal consistency when edited using the baseline method [35]. When only considering these cases, 72.3% of the volunteers picked ours for better consistency.

Table 4. Results of a user study

	quality	consistency	
Method	all	all	fragile
Tzaban *et al*. [35]	38.1	33.7	27.7
Ours	**61.9**	**66.3**	**72.3**

C.2. Disentangled Editing

Although we verified that ours show better performance in terms of temporal consistency in Tab. 2, this metric cannot detect the identity-irrelevant attribute. Thus, we measured non-target attributes preservation as target attribute change in the way used by Yao *et al*. [41].

Here, we use videos and the corresponding target attributes prepared for the user study. Since we measured 24 edited videos with various target attributes, unlike Yao *et al*. [41] used 1K images for each target attribute, we averaged a non-target attributes preservation rate as corresponding target attribute change. The Fig. 9 shows that our method is slightly better at preserving non-target attributes compared to the baseline [35].

Figure 9. Attribute preservation vs. target attribute change (a higher curve is better).

C.3. Inference Time Comparison

While GAN-based methods require only a single pass to synthesize a new edited frame, they internally run PTI [27] for reconstruction, which takes more than a few seconds per frame. As shown in the Tab. 5 where we compare the elapsed time with a single RTX 3090 to edit per frame via ours and Tzaban *et al*. [35], we are rather slow with full iterations of T but slightly faster at $T = 100$ with sufficiently reasonable performance. Moreover, deterministic and faster ODE samplers [15, 16] can also be utilized to reduce time further to 2.9s (the 3rd order of DPM-solver with 15 steps) with comparable quality (refer to project page).

Table 5. Inference time comparison

	Ours		Tzaban *et al*. [35]
	$T = 1000$	$T = 100$	
Classifier	60.9s	5.8s	12.7s
CLIP	62.4s	7.3s	12.0s
+ sampler	**2.9s**		

D. Limitations and Further Discussion

The main limitations of our method come from exploiting the pretrained networks such as an identity encoder (ArcFace) and a landmark encoder: 1) Using these networks limits the domain to face video as other previous works including face editing, face swapping, and face reenactment. 2) Our method is difficult to edit poses or facial expressions that can not be fully captured by the identity encoder. We conjecture that this is the reason why the eyebrows are unnatural in the last row of Fig. 13. 3) Since the identity encoder is trained for face recognition tasks, the latent space may lack disentanglement for editing. For example, we observed a gender bias when attempting to apply a 'beard' to a woman. As a future direction, this weakness could be resolved by finding out the disentangled space analogous to the style space of StyleGAN or training a module to discover disentangled editing directions.

In addition, a higher resolution video is possible as a future direction. We apply our method to 256^2 resolution videos for the following reasons: 1) The implementation of diffusion autoencoders [23] on which we are based is for 256^2 images. 2) The dataset used, VoxCeleb1, includes many low-resolution videos, which we have resized to a suitable and balanced size of 256^2. For a higher resolution, our method can be seamlessly applied by exploiting a diffusion upsampler module as DALLE-2 [25] or latent diffusion model architecture as Stable Diffusion [28] with conditioning our semantic representation.

E. Comparison of Temporal Consistency

We upload the video file of Fig. 4 to the project page. In the video, the result of Yao *et al*. [41] shows an altered identity, the result of Tzaban *et al*. [35] shows temporal inconsistency that beards fade away as the mouth opens, and the result of Xu *et al*. [40] shows unnatural movements with the mouth not opening as much as the original and inconsistency of the beard. On the other hand, ours demonstrates much improvement in terms of the temporal consistency and identity preservation.

F. Additional Editing Results

We show additional video editing results with classifier-based editing in Figs. 10 and 11 and CLIP-based editing in Figs. 12 and 13. These results demonstrate that our video editing method has temporal consistency for other attributes as well.

Figure 10. Classifier-based video editing on the other videos not in VoxCeleb1.

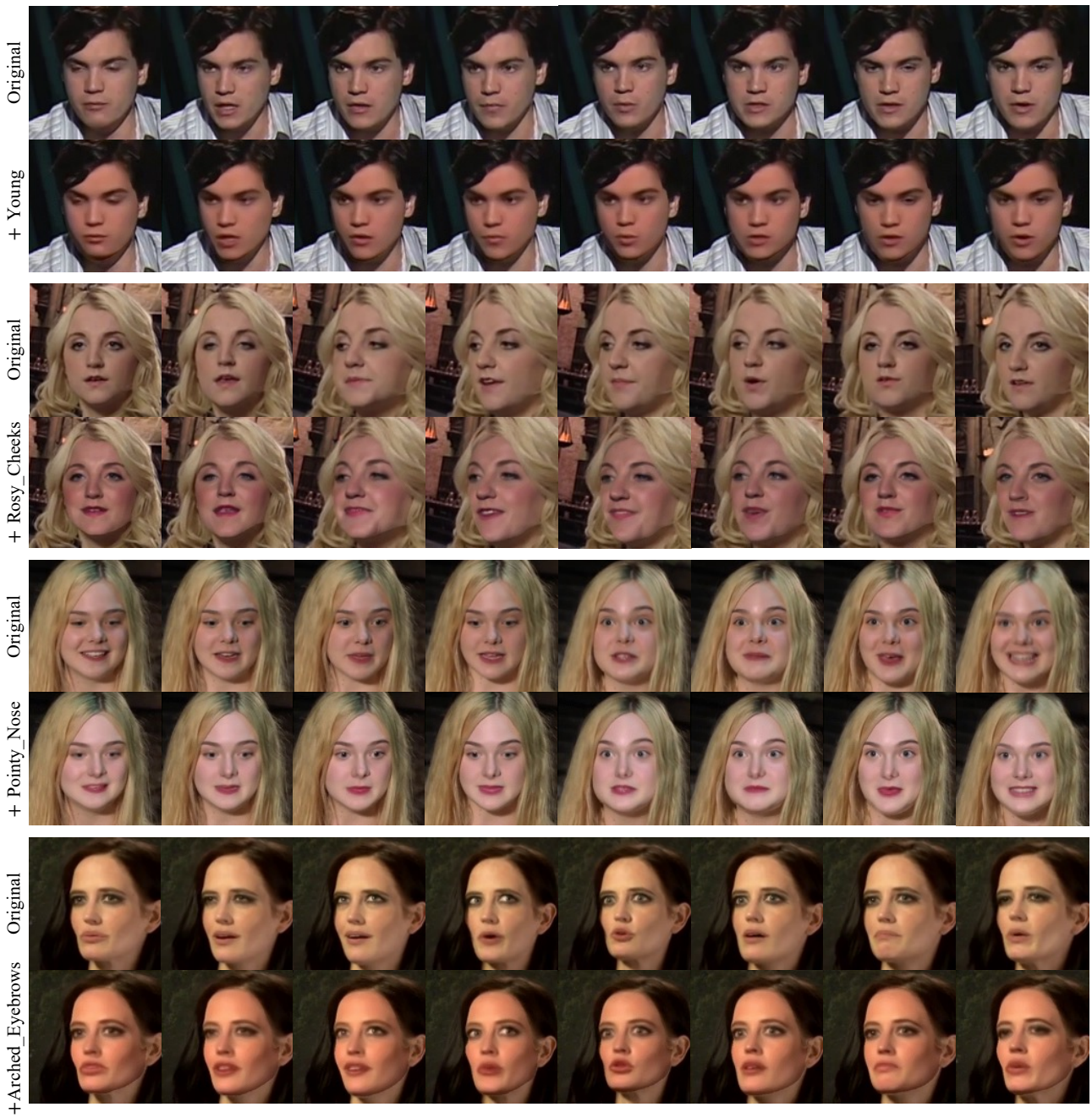

Figure 11. Classifier-based video editing on VoxCeleb1 test set.

Figure 12. CLIP-based video editing on the other videos not in VoxCeleb1.

Figure 13. CLIP-based video editing on VoxCeleb1 test set.