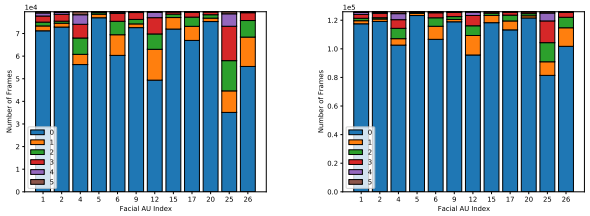
AUEditNet: Dual-Branch Facial Action Unit Intensity Manipulation with Implicit Disentanglement

Supplementary Material



(a) (b)

Figure S1. Distribution of AU intensities in DISFA. (a) Including samples with at least one non-zero AU intensity. (b) Whole DISFA dataset. As shown in (a), the distribution remains highly imbalanced after filtering out samples with zero intensities for all AUs.

S1. DISFA Dataset

The DISFA dataset [13, 14] is the only public dataset that contains intensity labels for 12 action units (AUs). It serves as the benchmark for AU intensity estimation tasks [15, 19]. Current AU intensity manipulation methods [11, 16, 21] often rely on large public datasets with predicted AU intensities as ground truth. This preference arises due to DISFA's limitations: it comprises only 27 subjects, notably fewer than the extensive subject pools of 337, 98, and over 1000 subjects used in these methods [11, 16, 21], respectively. Additionally, the intensity distribution within DISFA is highly imbalanced, as depicted in Fig. S1. Nevertheless, to the best of our knowledge, we are the first work to leverage such imbalanced datasets with limited subject counts for achieving AU intensity manipulation.

S2. Level-wise Architecture in AUEditNet

Supplementing the description of fitting our proposed AUEditNet to the multi-level structure of latent vectors in W^+ space [1] introduced in Sec. 3.3, here, we delve into the multi-level architecture and the encoding-decoding process for labels in AUEditNet.

S2.1. Multi-Level Architecture of AUEditNet

Given a source image I_{src}, e4e [20] encodes it into the W^+ latent space, producing corresponding latent vectors $W_{src} \in \mathcal{R}^{18\times512}$. These latent vectors can be used directly in StyleGAN2 [9] for high-quality image generation. The first dimension represents the level index, denoted by j in the main paper. Rather than reintegrating disentangled level-wise features in W^+ using a single editing module, we opt for multiple independent editing modules $\{\, \mathcal{P}^j(\cdot) \mid j \in [1, M] \,\}$, each responsible for editing a specific level of the latent vectors, shown in Fig. S2. Here, M denotes the number of levels we aim to edit, set to 11 in our task. The rest of latent vectors maintain invariant during editing.

S2.2. Encoding and Decoding of Labels

Various works explored incorporating input conditions into multi-level latent vectors within the W^+ space for editing purposes. StyleFlow [2] empirically found optimal level index ranges linked to specific facial attributes, like expression $(4-5)$, yaw $(0-3)$, and gender $(0-7)$. However, their focus was primarily on smiling expressions, which didn't satisfy our requirements for editing multiple AUs. Moreover, searching such optimal index ranges demands substantial datasets. ReDirTrans [8] proposed to apply the same conditions universally across levels and use error-based weights to determine each level's contribution to the target facial attribute. However, they assumed that their aimed attribute (gaze directions) could be estimated from a single level of the latent vectors in W^+, which might not suit other attribute manipulations.

Given these limitations, instead of focusing on which level (or levels) controls the target attribute, we propose encoding labels to align with the multi-level structure. This approach avoids mixing multiple facial attribute labels when inputted into individual levels. Specifically, we propose to first encode the target labels of multiple facial attributes $(\{\, (c^i_{tar}, a^i_{tar}) \mid 1 \in [1, N] \,\})$ into multi-level embeddings for fitting the multi-level structure. Then, we feed the j-th level embedding $(\{\, (\hat{c}^{i,j}_{tar}, \hat{a}^{i,j}_{tar}) \mid 1 \in [1, N] \,\})$ into the corresponding editing module $\mathcal{P}^j$ to perform editing. Given the level-wise estimated label embeddings from the source image, we decode them back to the original label space to get the estimated source labels $\{\, (\hat{c}^i_{src}, \hat{a}^i_{src}) \mid 1 \in [1, N] \,\}$. Eq. 4 supervises this training process. The loss values for level-wise label embeddings and final decoded labels are computed independently to prevent the label encoder from learning the same mapping as the input labels. On the other hand, due to the highly disentangled nature of the latent vectors in the W^+ space across different levels, predicting all 12 AU intensities from each level of the latent vectors can be challenging. Thus, an identical mapping by Ψ_{enc} could result in increased loss.

The proposed encoding-decoding pipeline for labels doesn't restrict the estimation of aimed attributes to a single level of latent vectors. Fig. S2 presents the overall multi-level architecture of AUEditNet. The encoder-decoder pair, ψ_{enc}

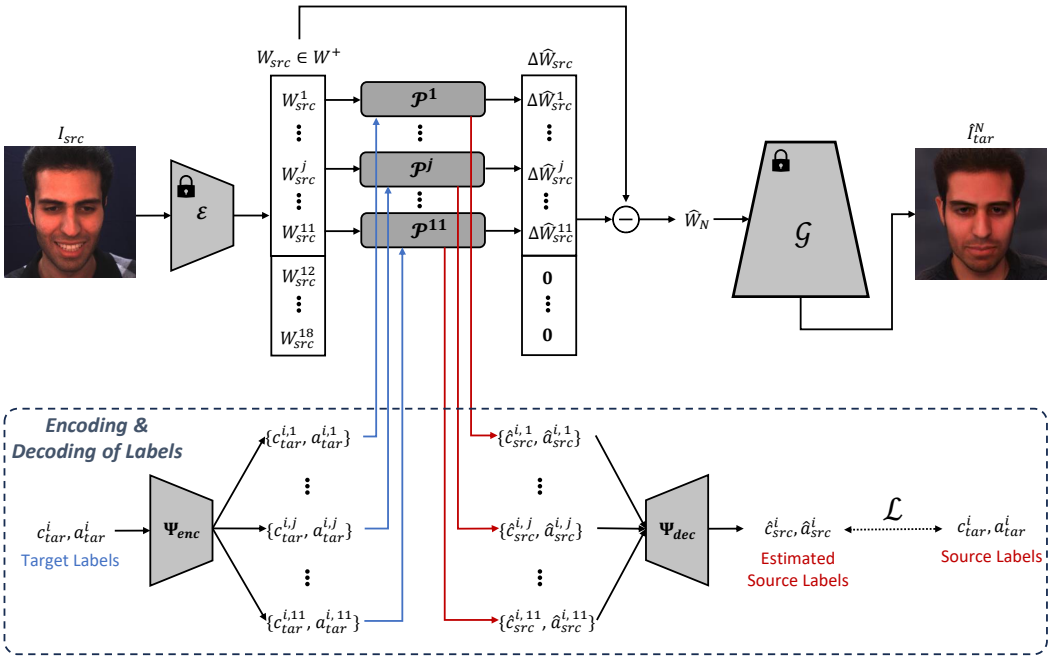


Figure S2. Multi-level architecture of AUEditNet. We only focus on editing the first 11 levels of latent vectors in W^+. Each level has one corresponding editing module $\mathcal{P}^j$, whose detailed structure is described in Fig. 1. Given a sequence of target labels for 12 AUs, we first use Ψ_{enc} to encode them into embeddings and feed these embeddings into each $\mathcal{P}^j$ for editing purposes. Meanwhile, each $\mathcal{P}^j$ estimates the label embeddings from the source latent vectors. Subsequently, we use Ψ_{dec} to decode these estimated embeddings back to the label space and compare them with the actual source labels for supervision. For simplicity, we only include one target attribute with the index i. In the real implementation, the input target labels should include labels for all 12 AUs. We only include the *source branch* in this figure for better description. The pipeline is the same and the weights are shared in the *target branch*.

	Manipulation Accuracy		ID Preservation	Image Similarity	
	ICC ↑	MSE ↓	Distance ↓	L2 ↓	LPIPS ↓
W/O $\psi(\cdot)$	0.617	0.288	0.471	**0.026**	**0.173**
W/ $\psi(\cdot)$	**0.628**	**0.283**	**0.468**	**0.026**	0.174

Table S1. Ablation Study on encoding-decoding processes for labels. $\psi(\cdot)$ represents the pair of encoder ψ_{enc} and decoder ψ_{dec}.

and ψ_{dec} are trained based on the **Label Loss** introduced in Sec. 3.4. Table S1 presents the comparison with and without the label encoding-decoding processes in AUEditNet.

S3. AU Intensity Estimator

In our work, pretrained AU intensity estimators are required at two stages: when utilizing the *Pretrained Function Loss* in Sec. 3.4 during training and when evaluating manipulation performance quantitatively during inference.

S3.1. Network Structure

We utilize a Siamese network for AU intensity estimation, shown in Fig. S3. The input is a pair of images from the same subject. One is viewed as the target image, and the other one is viewed as the anchor image. The output is the difference of AU intensities between the target image and

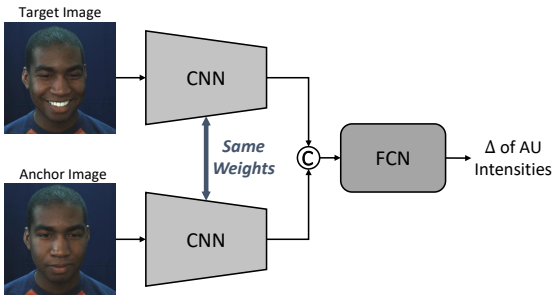


Figure S3. Structure of the AU intensity estimator. This Siamese network takes a pair of images from the same subject as inputs and estimates the difference of AU intensities between these two images (the target and anchor images). We use convolutional neural network (CNN) to extract features. After concatenating two features, we use fully-connected network (FCN) to regress the final output.

	Method	AU1	AU2	AU4	AU5	AU6	AU9	AU12	AU15	AU17	AU20	AU25	AU26	Avg
ICC	AUEditNet (*Real*)	**.848**	**.559**	.874	**.600**	.577	.230	**.890**	.276	.669	.511	**.950**	**.548**	.628
	AUEditNet (*Syn*)	.853	.551	**.885**	**.600**	**.586**	**.235**	.888	**.283**	**.685**	**.514**	.948	.533	**.631**
MSE	AUEditNet (*Real*)	.191	**.445**	.309	**.029**	.492	.579	**.228**	**.080**	.188	.322	**.169**	**.367**	.283
	AUEditNet (*Syn*)	**.186**	.452	**.291**	.030	**.483**	**.574**	.230	**.080**	**.181**	**.321**	.171	.377	**.281**

Table S2. Comparison of AU intensity manipulation performance when using different types of anchor images. '(*Syn*)' means using synthetic face images with deactivating all AUs as the anchor image. '(*Real*)' means using real images with zero intensities of all AUs from the test subject as the anchor image. The results under the 'Real' case are copied from Table 1.

Figure S4. Comparison of eyebrow positions and shapes on the DISFA dataset. All of these four images have deactivated (zero-intensity) AU 1 (Inner Brow Raiser), AU 2 (Outer Brow Raiser) and AU 4 (Brow Lowerer). We can observe that the different eyebrow positions and shapes could affect the performance given a unified AU intensity estimator.

the anchor image. This design could help to reduce personal facial attributes' influences, such as eyebrow positions and shapes affecting the eyebrow-related AU movements, illustrated in Fig. S4. If all AU intensities in the anchor image are at zero, the output represents the absolute intensities of AUs in the target image.

S3.2. Estimator in Training

During training, the pretrained convolutional part of VGG-16 [18] serves as the backbone in the AU intensity estimator, trained on the DISFA training subset. It functions as F_{pre} to detect AU intensities in synthesized images during AUEditNet's training. The anchor image is randomly chosen from the same subject's data with all AUs deactivated (zero intensity).

S3.3. Estimator in Inference

During testing, we use another external AU intensity estimator to quantify the manipulation performance, which is unseen during training. We use the pretrained convolutional part of ResNet-50 [7] as the backbone to build the AU intensity estimator H_{est}, trained on the DISFA training subset.

When evaluating the performance of AUEditNet, the target image for the AU intensity estimator is the generated image with the provided target conditions. The anchor image can be either a real image with zero intensities of all AUs from the test subject or the generated one with deactivating all AUs. Table S2 presents the comparison using real or synthetic images with deactivated AUs as the anchor images. When using synthetic images as the anchor images, the final performance is further improved even if the external AU intensity estimator H_{est} is only trained with the real images in the training subset. Additionally, it proves AUEditNet's effectiveness in AU intensity manipulation when deactivating all AUs.

S4. Smile Attribute Manipulation

To further validate AUEditNet's effectiveness, we assess the facial expression editing performance by manipulating intensities over some AUs. We modify the intensities of AU 6 (Cheek Raiser) and AU 12 (Lip Corner Puller) across eight levels (shown in Fig. S5) simultaneously to enable smile intensity editing [6]. Following the evaluation proposed in [3], we utilize a pretrained face recognition model [5] for identity preservation assessment and utilize Face++ [4] to evaluate the smile attribute intensity values in generated images.

S5. Data Annotation

In comparison to the abundance of publicly available datasets containing various expressions, datasets with detailed AU intensity labels are relatively scarce. Introducing new experts for AU intensity annotation could lead to subjective discrepancies. To address this, we have devised an annotation pipeline while maintaining the architecture of the editing network. During the annotation process, we keep all the network parameters fixed and solely iterate the conditions until the generated image closely resembles the input image. Evaluation criteria consist of pixel-wise loss and a pretrained function loss. Visualized results are used to assess performance by comparing them with the input images, making the process more accessible to data annotators who do not require expertise in AU intensity, but rather focus on comparing image similarity. This pipeline ensures that the provided target conditions fully control AU intensities in the final image, supported by intermediate results with deactivating all AUs and the final generation is based on this intermediate result. Fig. 2 shows the intermediate results with all intensity values set to zero, alongside final generated images with target AU intensities. Given the multitude of

Figure S5. Smile attribute manipulation achieved by the AU intensity manipulation. a_{smile} denotes the target smile intensity, ranged $[0, 1]$

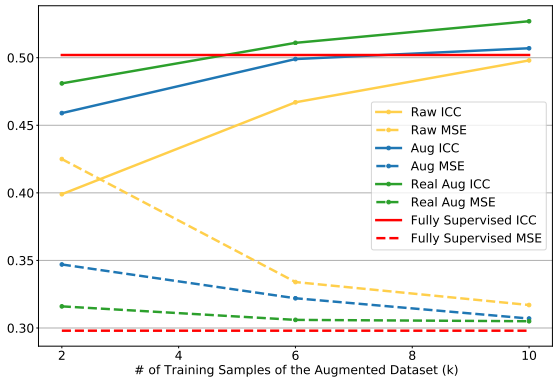


Figure S6. AU intensity estimation performance based on the data augmentation from the BU-4DFE dataset annotated via AUEditNet. 'Real Aug' denotes the augmentation performance with real images from BU-4DFE. With new information (new images from BU-4DFE) included in the AU intensity estimation task, the final estimation performance is further improved.

intensity combinations across 12 AUs with six ordinal levels (let alone float intensities), we iteratively adjust one AU's intensity with six levels, selecting the best level based on loss values before moving to the next AU. We loop the above pipeline with each image twice and it takes around 12s on a single NVIDIA RTX 3090. To prove the effectiveness of this annotation pipeline, we utilize the same augmentation pipeline, introduced in Sec. **??**. However, instead of using synthetic images, we utilize the real images from BU-4DFE [22] accompanied with the estimated AU intensities for training an AU intensity estimator. Fig. S6 presents the results when we augmented the raw data with the same number of real images and estimated annotations. With including new information instead of synthetic images generated from the same training samples, the estimator's performance is further improved. This pipeline offers a pathway for conditional synthesis networks to establish pairs between real images and pseudo regressed labels. Furthermore, it offers the possibility of a manual evaluation step, which does not demand specialized expertise, to validate the accuracy of the pseudo labels.

S6. State-of-the-Art Baselines

We reproduce two StyleGAN-based facial attribute editing methods: ReDirTrans [8] and DeltaEdit [12] to achieve AU intensity manipulation.

ReDirTrans. Jin *et al.* [8] proposed ReDirTrans, focusing on redirecting gaze directions and head orientations based on the provided yaw and pitch angles. They edited the gaze-related and head-related embeddings through rotation matrices built by the target conditions to make the whole transformation process interpretable. We adapt this method, using translation processes scaled with aimed AU intensities to replace the rotation processes while maintaining other modules unchanged.

DeltaEdit. Lyu *et al.* [12] proposed DeltaEdit, which is a text-driven facial attribute editing method. Instead of using interpretable editing processes as ReDirTrans did, they fed both editing conditions and source latent vectors into a network to estimate the editing directions for desired attribute editing. Because they utilized the pretrained CLIP [17] to extract image features, they didn't require any labels for facial attributes. In our case, we replace the text prompts with AU intensities as conditions. We use an extra fully-connected network (FCN) to bridge the dimension gap between text features from CLIP and AU intensities. During training, we employ a pair of images from the same subject as input instead of different subjects as DeltaEdit did because we no longer utilized the CLIP image encoder, which has the ability to capture different identity information. Instead, we use the ground truth of AU intensities as the input during training.

S7. Training Details

To expedite training and mitigate the influence of numerous samples with zero intensities of all AUs, shown in Fig. S1, we always use one sample with at least one non-zero AU intensity as the source image. The target and random images are chosen randomly from the rest data without any special requirements. We utilize the cycle pipeline [23] to input the generated target image with source image conditions back to the network to achieve cycled image reconstruction.

We opt for a batch size of 2, utilizing Adam optimizer [10] with default momentum values ($\beta_1 = 0.9, \beta_2 = 0.999$). The training process, consuming around $18,123$ MiB on a single NVIDIA RTX 3090, iterates for $30,000$ iterations. The loss weights in Eq. 4 are set as $\lambda_R = 8, \lambda_P = 1, \lambda_F = 125, \lambda_{ID} = 20, \lambda_L = 20$.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4432–4441, 2019. S1

[2] Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (ToG)*, 40(3):1–21, 2021. S1

[3] Hoseok Do, EunKyung Yoo, Taehyeong Kim, Chul Lee, and Jin Young Choi. Quantitative manipulation of custom attributes on 3d-aware image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8529–8538, 2023. S3

[4] Face++. Face detection api. https://www.faceplusplus.com/, Accessed: 2023-11-15. S3

[5] Adam Geitgey. Github – face recognition. https://github.com/ageitgey/face_recognition/, 2021. S3

[6] Jeffrey M Girard, Gayatri Shandar, Zhun Liu, Jeffrey F Cohn, Lijun Yin, and Louis-Philippe Morency. Reconsidering the duchenne smile: indicator of positive emotion or artifact of smile intensity? In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 594–599. IEEE, 2019. S3

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. S3

[8] Shiwei Jin, Zhen Wang, Lei Wang, Ning Bi, and Truong Nguyen. Redirtrans: Latent-to-latent translation for gaze and head redirection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5547–5556, 2023. S1, S4, S5

[9] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. S1

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. S5

[11] Jun Ling, Han Xue, Li Song, Shuhui Yang, Rong Xie, and Xiao Gu. Toward fine-grained facial expression manipulation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 37–53. Springer, 2020. S1

[12] Yueming Lyu, Tianwei Lin, Fu Li, Dongliang He, Jing Dong, and Tieniu Tan. Deltaedit: Exploring text-free training for text-driven image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6894–6903, 2023. S4, S5

[13] S Mohammad Mavadati, Mohammad H Mahoor, Kevin Bartlett, and Philip Trinh. Automatic detection of non-posed facial action units. In *2012 19th IEEE International Conference on Image Processing*, pages 1817–1820. IEEE, 2012. S1

[14] S Mohammad Mavadati, Mohammad H Mahoor, Kevin Bartlett, Philip Trinh, and Jeffrey F Cohn. Disfa: A spontaneous facial action intensity database. *IEEE Transactions on Affective Computing*, 4(2):151–160, 2013. S1

[15] Ioanna Ntinou, Enrique Sanchez, Adrian Bulat, Michel Valstar, and Yorgos Tzimiropoulos. A transfer learning approach to heatmap regression for action unit intensity estimation. *IEEE Transactions on Affective Computing*, 2021. S1

[16] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of the European conference on computer vision (ECCV)*, pages 818–833, 2018. S1

[17] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. S5

[18] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR 2015)*. Computational and Biological Learning Society, 2015. S3

[19] Tengfei Song, Zijun Cui, Yuru Wang, Wenming Zheng, and Qiang Ji. Dynamic probabilistic graph convolution for facial action unit intensity estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4845–4854, 2021. S1

[20] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4): 1–14, 2021. S1

[21] Soumya Tripathy, Juho Kannala, and Esa Rahtu. Icface: Interpretable and controllable face reenactment using gans. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3385–3394, 2020. S1

[22] Xing Zhang, Lijun Yin, Jeffrey F Cohn, Shaun Canavan, Michael Reale, Andy Horowitz, and Peng Liu. A high-resolution spontaneous 3d dynamic facial expression database. In *2013 10th IEEE international conference and workshops on automatic face and gesture recognition (FG)*, pages 1–6. IEEE, 2013. S4

[23] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. S5