ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

УДК 630

Отчет об исследовательском проекте на тему:

Замена лица с помощью генеративных моделей GANs

Выполнил студент:

группы #БПМИ211, 3 курса Дробышевский Илья Александрович

Принял руководитель проекта:

Мещанинов Вячеслав Павлович
Научный сотрудник
Факультет компьютерных наук НИУ ВШЭ

Соруководитель:

Аланов Айбек
Научный сотрудник
Факультет компьютерных наук НИУ ВШЭ

Консультант:

Лисов Роман

Москва 2024

Содержание

Аннотация **4**

Ключевые слова **4**

1 Введение **5**

1.1 Постановка задачи 5

1.2 Описание предметной области 5

2 Обзор литературы **5**

2.1 FaceShifter 5

2.1.1 Adaptive Embbedings Integration Network(AEI-Net) 6

2.1.2 Heuristic Error Acknowledging Refinement Network(HEAR-Net) 8

2.1.3 Эксперименты 9

2.2 MegaFS 9

2.2.1 Face Encoding 9

2.2.2 Latent Code Manipulation 10

2.2.3 Face Generation 11

2.2.4 Эксперименты 12

2.3 FSLSD(HiRes) 13

2.3.1 Class-Specific Attributes Transfer 13

2.3.2 Background Transfer 14

2.3.3 Эксперименты 15

2.4 SEAN 15

2.4.1 Per-Region Style Encoder 16

2.4.2 SEAN 16

2.4.3 Эксперименты 17

2.5 StyleSwap 18

2.5.1 Identity Information Encoding 18

2.5.2 Swapping-Driven Mask Branch 18

2.5.3 Эксперименты 20

2.6 E4S 20

2.6.1 Reenactment 20

2.6.2 Mask-guided Style Extraction 21

2.6.3 Mask-guided Style Injection 22
2.6.4 Эксперименты 24

3 Выбор методов для запусков **24**

4 Протокол тестирования и визуализации **25**

5 Результаты запусков методов **25**

6 Эксперименты с postprocessing **26**

7 Выдвижение гипотез **30**

8 Проверка гипотез **31**

8.1 Первая гипотеза 31
8.2 Вторая гипотеза 31
8.3 Третья гипотеза 32
8.4 Четвёртая гипотеза 33
8.5 Пятая гипотеза 33
8.6 Подведение итогов по гипотезам 33

9 Проверка на реальных примерах **35**

10 Заключение **36**

Список литературы **39**

Аннотация

С развитием Computer Vision и Deep Learning возможности генерации реалистичных лиц заметно расширились. В связи с этим появляется всё больше различных методов, пытающихся решить задачу FaceSwap.

В этой работе представляется улучшенный метод переноса лиц с source изображения на target. Большинству существующих методов не удаётся в полной мере перенести идентичность (identity) source изображения, в то время как данный метод решает эту проблему.

За основу был взят E4S [1], в котором были исправлены все ключевые проблемы: улучшен метод получения сегментационной маски по изображению, пересмотрен метод построения сегментационной маски для итогового изображения, добавлен модуль для перекрашивания изображений, а так же последующего сглаживания.

Метод был протестирован не только на подготовленных изображениях, которые содержатся в датасетах (то есть выровненных лицах, находящихся в центре фотографии), но и на обычных фотографиях из интернета.

Сравнения с текущими state of the art методами показали, что данный метод лучше переносит цвет кожи, освещение, а также позу и выражение.

Имплементация метода со всеми весами доступна по этой ссылке:

https://github.com/free001style/FaceSwapViaGans

Ключевые слова

FaceSwap, GAN, StyleGAN, DeepFake, High Resolution

1 Введение

1.1 Постановка задачи

Постановка задачи следующая: по данным source и target изображениям получить новое, в котором от target будет задний фон, освещение, поза и выражение лица, а от source будет то, что называют identity, т.е. то, что делает человека индивидуальным: брови, глаза, нос, губы.

1.2 Описание предметной области

Первая и основная проблема, связанная с переносом лиц, – это сохранение идентичности, т.е. чтобы после переноса лица, на изображении остались уникальные черты source изображения. Как мы увидим дальше, не у всех методов получается получить латентные представления так, чтобы не потерять важные детали.

Второй проблемой является устранение так называемых facial occlusions. Из-за того, что на изображении кроме самих лиц есть ещё и различные атрибуты, такие как очки, головные уборы, руки и волосы, которые закрывают некоторую часть лица, финальный результат может содержать большое количество артефактов.

2 Обзор литературы

2.1 FaceShifter

FaceShifter [2] состоит из двух этапов. Первый этап это Adaptive Embbedings Integration Network(AEI-Net). Авторы сделали два нововведения:

1. Создан Multi-level attributes encoder, благодаря которому можно извлекать target атрибуты в разных пространственных разрешениях, а не сжимать в один вектор;

2. В generator добавлен Adaptive Attentional Denormalization layer(AAD), который адаптивно запоминает, куда стоит вставлять атрибуты.

Вторым этапом является Heuristic Error Acknowledging Refinement Network(HEAR-Net) – модуль, позволяющий справляться с facial occlusions.

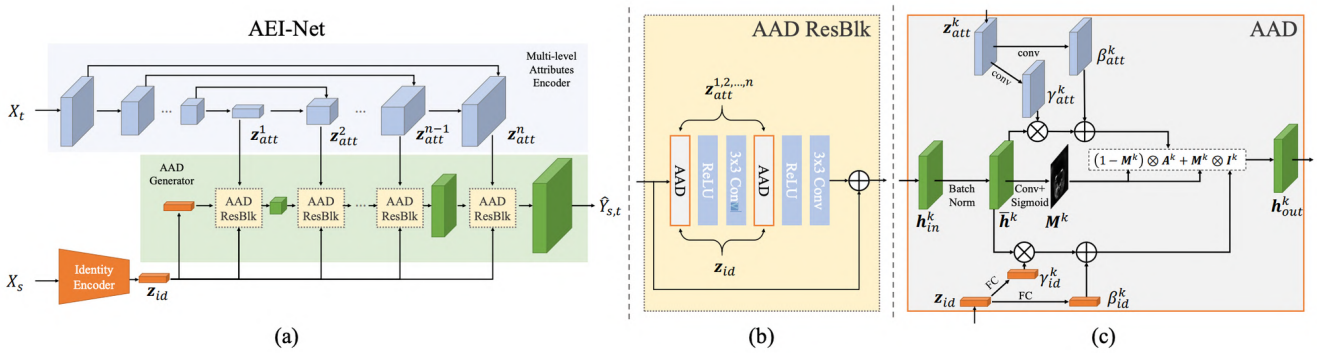

Рис. 1: Первый этап FaceShifter.

2.1.1 Adaptive Embbedings Integration Network(AEI-Net)

Состоит из 3 частей: Identity Encoder(для X_s), Multi-level attributes Encoder(для X_t) и AAD Generator(генерирует $\hat{Y}_{s,t}$)

- Identity Encoder

  Была взята предобученная SOTA [3]. В качестве identity embeddings $z_{id}(X_s)$ взяли last feature vector, полученный перед final FC layer. Авторы верят, что модель, обученная на 2D картинках, может давать more represetive embbedings, чем 3D-based models.

- Multi-level attributes Encoder(Рис. 1(a))

  X_t пропускают через U-Net-like model и в качестве attributes embbedings берут feature maps, сгенерированные U-Net-like decoder. Более формально

$$z_{attr}(X_t) = \{z^1_{attr}(X_t), \dots, z^n_{attr}(X_t)\} \tag{1}$$

  , где $z^k_{attr}(X_t)$ — это k-th level feature map(т.е. выходы на этапе upsamle).

  Этот encoder учился self-supervised.

- AAD Generator(Рис. 2)

  На вход каждому блоку(Рис. 1(c)) приходит выход предыдущего блока, он же $h^k_{in} \in \mathbb{R}^{C^k \times H_k \times W^k}$

  (z_{id}, если это первый блок), $z_{id} \in \mathbb{R}^{C_{id}}$ и $z^k_{attr} \in \mathbb{R}^{C^k \times H_k \times W^k}$. Сначала пропускаем вход через batch normalization [4] (получаем $\tilde{h}^k$) и пропускаем z_{id} с z^k_{attr} по два раза через FC. Далее результат умножаем и суммируем с $\tilde{h}^k$ и получаем A^k и I^k(attributes и identity соответственно). Ещё $\tilde{h}^k$ пропускаем через свёртку и sigmoid и получаем M^k. Тогда выходом будет $h^k_{out} = (1 - M^k) \otimes A^k + M^K \otimes I^k$

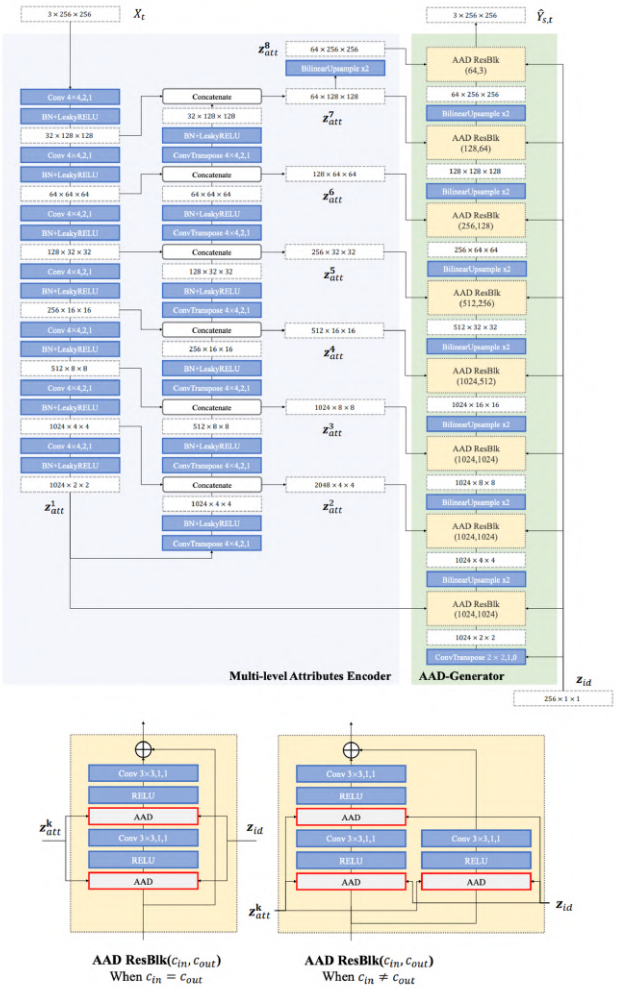

Рис. 2: Полная реализация generator.

Для обучения Attributes Encoder и AAD Generator использовались следующие лоссы:

- $\mathcal{L}_{adv}$ — adversarial loss
- $\mathcal{L}_{id} = 1 - \cos(z_{id}(\hat{Y}_{s,t}), z_{id}(X_s))$ — смотрим, чтобы identity embbedings у source картинки и у полученной были близки по cosine similarity.
- $\mathcal{L}_{attr} = \frac{1}{2}\sum_{k=1}^{n} \left\| z_{attr}^{k}(\hat{Y}_{s,t}) - z_{attr}^{k}(X_t) \right\|_2^2$ — смотрим, чтобы feature maps были близки.
- $\mathcal{L}_{rec} = \begin{cases} \frac{1}{2}\left\| \hat{Y}_{s,t} - X_t \right\|_2^2, & X_t = X_s, \\ 0, & \text{otherwise} \end{cases}$ — то есть модель должна идеально воссоздавать исходное изображение.

Итоговый лосс выглядит

$$\mathcal{L}_{AEI-Net} = \mathcal{L}_{adv} + \lambda_{id}\mathcal{L}_{id} + \lambda_{attr}\mathcal{L}_{attr} + \lambda_{rec}\mathcal{L}_{rec}, \tag{2}$$

где $\lambda_{rec} = \lambda_{attr} = 10$, $\lambda_{id} = 5$(были эмперически получены).

Рис. 3: Аугментации с синтетическими искажениями.

2.1.2 Heuristic Error Acknowledging Refinement Network(HEAR-Net)

Авторы заметили, что если в AEI-Net подать одну и ту же картинку дважды, то facial occlusions также останутся. Тогда heuristic error можно определить как

$$\Delta Y_t = X_t - \text{AIENet}(X_t, X_t). \tag{3}$$

Тогда весь второй этап сводится к пропуску ΔY_t и $\hat{Y}_{s,t}$ через HEAR-Net, которая имеет структуру U-Net.

Так как данных с facial occlusions мало, то авторы использовали некоторые аугментации(Рис. 3): брали и накладывали на существующие лица различные изображения с помощью поворотов и кропов.

Обучали это на следующие лоссы:

- $\mathcal{L}_{id} = 1 - \cos(z_{id}(Y_{s,t}), z_{id}(X_s))$
- $\mathcal{L}_{chg} = \left\| \hat{Y}_{s,t} - Y_{s,t} \right\|_1$ — сохранить консистентность между сгенерированным и улучшенным изображениями.
- $\mathcal{L}_{rec} = \begin{cases} \frac{1}{2} \left\| Y_{s,t} - X_t \right\|_2^2, & X_t = X_s, \\ 0, & \text{otherwise} \end{cases}$

Итоговый лосс выглядит

$$\mathcal{L}_{HEAR-Net} = \mathcal{L}_{rec} + \mathcal{L}_{id} + \mathcal{L}_{chg} \tag{4}$$

2.1.3 Эксперименты

Перед подачей картинки в модель, её сначала выравнивают(using five point landmarks) и изменяют размер до 256x256, чтобы почти всё изображение состояло из лица и атрибутов. Количество attributes embeddings in AIE-Net равно 8, количество downsample/upsample in HEAR-Net равно 5. AIENet учили на датасетах CelebA-HQ [5], FFHQ [6], VGGFace [7]. HEAR-Net учился только на лицах с Top-10% heuristic error.

В качестве метрик авторы использовали следующее:

- ID retrieval — для каждого swapped face из test set ищется ближайшая картинка по датасету FaceForensics++ [8](ближайшая в смысле получения identity vector с помощью Cosface [9] и используя cosine similirity), далее считается процент совпадения ближайшей картинки с правильной.
- Pose — картинка проходит через модель [10], получают вектора и берут ℓ_2 растояние с target изображением.
- Expression — тоже самое, только модель другая [11].

method	ID retrieval ↑	pose ↓	expression ↓
DeepFakes [1]	81.96	4.14	2.57
FaceSwap [2]	54.19	**2.51**	2.14
Nirkin *et al.* [31]	76.57	3.29	2.33
IPGAN [5]	82.41	4.04	2.50
Ours	**97.38**	2.96	**2.06**

Рис. 4: Сравнение на FaceForensics++.

На Рис. 4 приведены результаты сравнения данного метода с другими популярными на тот момент.

2.2 MegaFS

MegaFS [12] состоит из 3 этапов: Face Encoding, Latent Code Manipulation и Face Generation.

2.2.1 Face Encoding

На этом этапе изображения проецируются на латентное пространство $\mathcal{W}^{++}$ с помощью HieRFE(Рис. 5(stage 1)).

Изображения пропускают через resnet50; после второго и каждого последующего слоя возвращается feature map, который пропускают через conv(3, 1, 0), BN и ReLU, а далее суммируют, где это надо(чтобы размерности совпадали, применяют bilinear upsamling).

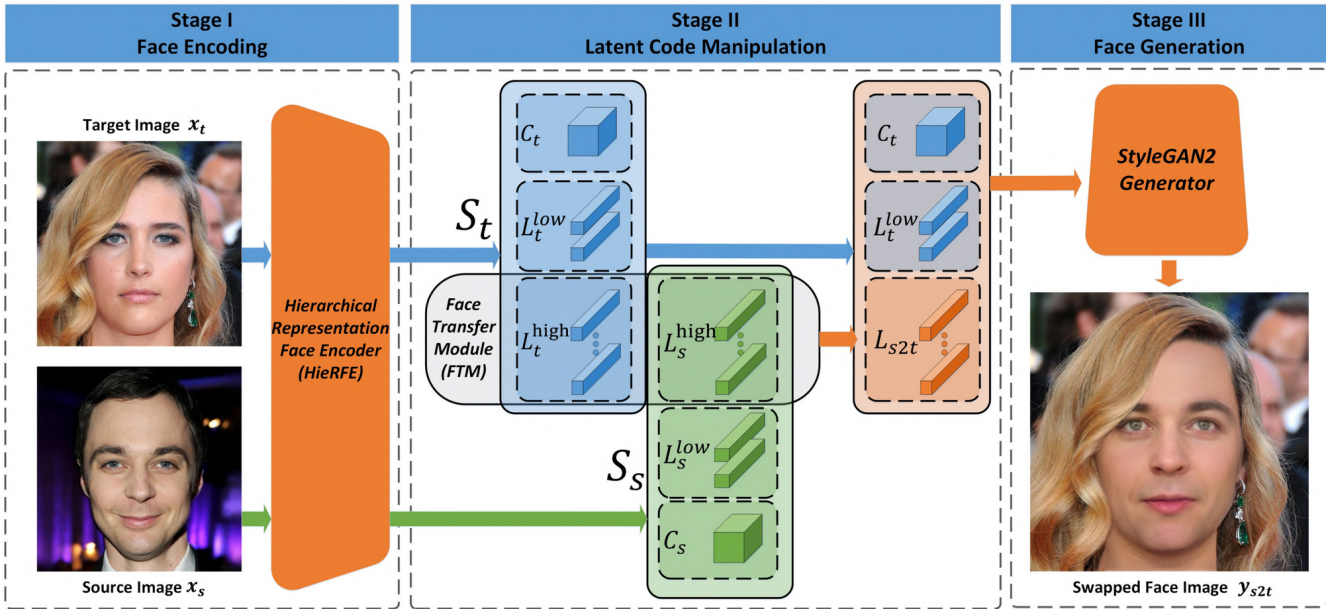

Рис. 5: Структура MegaFS.

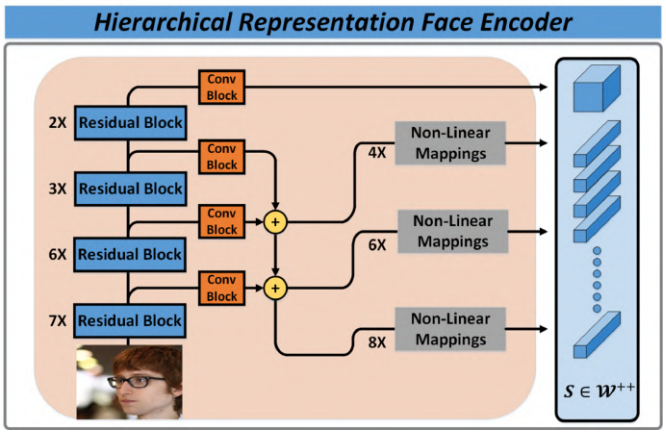

Рис. 6: Структура HieRFE.

Non-Linear Mappings — conv, BN и LeakyReLU применяются до тех пор, пока тензор не сожмётся до вектора размера 512. В конце применяется Tanh и отправляется на следующий этап вместе с самой верхней стрелочкой на картинке, оно же C на Рис. 6.

2.2.2 Latent Code Manipulation

На данном этапе получают латентное представление для финальной картинки с помощью латентных представлений исходных картинок. Для этого используют Face Transfer Module(FTM). На вход ему подаётся l^{high}. Далее идёт 3 одинаковых блока, как на Рис. 7: сначала латентные представления конкатинируют, прогоняют через FC, ReLU, FC и sigmoid, далее поэлементно перемножают; потом получившийся тензор прогоняют через FC, ReLU, FC и Tanh, далее суммируют.

Т.к. sigmoid переводит числа в значения (0,1), то, делая поэлементное произведение, мы

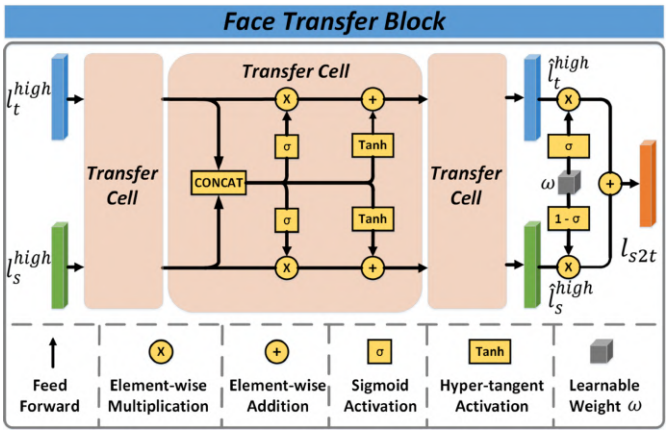

Рис. 7: Структура FTM.

отбрасывает нерелевантную информацию и оставляем только то, что нужно. В конце берётся обучаемый вектор размера 512 w и

$$l_{s2t} = \sigma(w) \times \hat{l}_t + (1 - \sigma(w)) \times \hat{l}_s \tag{5}$$

2.2.3 Face Generation

Тут всё просто: берётся предобученный StyleGAN2 [13] и на вход ему подаются C_t, L_t^{low}, L_{s2t}. В качестве аргументов, почему взяли именно StyleGAN2, авторы пишут: лица получаются более детальными, потому что работаем в $\mathcal{W}^{++}$, к тому же тренировочный процесс заметно упрощается.

Чтобы учить HieRFE, авторы используют следующие лоссы:

- $\mathcal{L}_{rec} = \|x - \hat{x}\|_2$
- $\mathcal{L}_{LPIPS} = \|\mathrm{F}(x) - \mathrm{F}(\hat{x})\|_2$, где F — это feature map
- $\mathcal{L}_{id} = 1 - \cos(\mathrm{R}(x), \mathrm{R}(\hat{x}))$, где R — ArcFace [3]
- $\mathcal{L}_{lmd} = \|\mathrm{P}(x) - \mathrm{P}(\hat{x})\|_2$, где P — facial landmark predictor [14] (нужен для того, чтобы сравнивать позу и выражения target лица)

Общий лосс выглядит как

$$\mathcal{L}_{inv} = \lambda_1 \mathcal{L}_{rec} + \lambda_2 \mathcal{L}_{LPIPS} + \lambda_3 \mathcal{L}_{id} + \lambda_4 \mathcal{L}_{lmd}, \tag{6}$$

и во время экспериментов авторы использовали $\lambda_1 = 1, \lambda_2 = 0.8, \lambda_3 = 1, \lambda_4 = 1000$.

Чтобы учить FTM, авторы используют следующие лоссы:

- $\mathcal{L}_{rec} = \|x_s - \hat{x}_s\|_2 + \|x_t - \hat{x}_t\|_2$
- $\mathcal{L}_{LPIPS} = \|\mathrm{F}(x_t) - \mathrm{F}(y_{s2t})\|_2$
- $\mathcal{L}_{id} = 1 - \cos(\mathrm{R}(x_s), \mathrm{R}(y_{s2t}))$
- $\mathcal{L}_{lmd} = \|\mathrm{P}(x_t) - \mathrm{P}(y_{s2t})\|_2$
- $\mathcal{L}_{norm} = \|L_s^{high} - L_{s2t}\|_2$, нужен для стабилизации обучения

Общий лосс тут выглядит как

$$\mathcal{L}_{swap} = \varphi_1 \mathcal{L}_{rec} + \varphi_2 \mathcal{L}_{LPIPS} + \varphi_3 \mathcal{L}_{id} + \varphi_4 \mathcal{L}_{lmd} + \varphi_5 \mathcal{L}_{norm}, \tag{7}$$

и во время экспериментов авторы использовали $\varphi_1 = 8, \varphi_2 = 32, \varphi_3 = 24, \varphi_4 = 100000, \varphi_5 = 32$.

2.2.4 Эксперименты

Учили на следующих датасетах:

- для экспериментов на FaceForensics++ [8] — CelebA [15], CelebA-HQ [5], FFHQ [6];
- для эксперимeтнов на CelebA-HQ — FFHQ.

В качестве метрик авторы использовали следующее:

- ID retrieval — то же самое, что и FaceShifter.
- Pose — то же самое, что и у FaceShifter.
- Expression — тоже самое, но модель другая [16]

Method	ID retrieval ↑	pose ↓	expression ↓
DeepFakes [11]	88.39	4.64	3.33
FaceSwap [15]	72.69	2.58	2.89
Face2Face [53]	-	2.68	**2.09**
Neural Textures [52]	-	**2.21**	**1.64**
FaceShifter [28]	**90.68**	**2.55**	2.82
Ours	**90.83**	2.64	2.96

Рис. 8: Сравнение на FaceForensics++.

На Рис. 8 приведены результаты сравнения данного метода с другими популярными на тот момент.

2.3 FSLSD(HiRes)

Авторы FSLSD [17] выдвинули гипотезу о том, что на самом деле существуют structure attributes и appearance attributes. К первым относятся поза и выражение лица, ко вторым освещение и цвета. Также упоминание MegaFS: пишут, что путать латентные предствления target и source неэффективно, и в этой статье будет показано, как надо делать. Также пишут, что передавать атрибуты лица с помощью MLP – не самая лучшая идея, и у них есть другой способ: научится извлекать structure and appearance attributes.

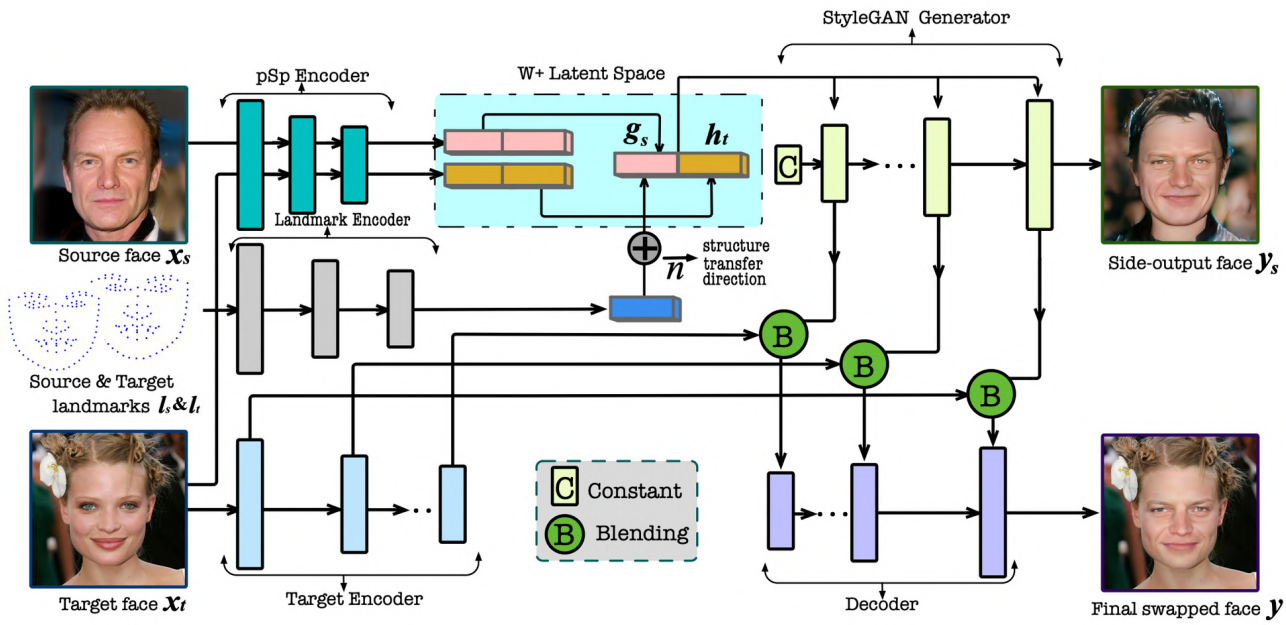

Рис. 9: Структура HiRes: на вход подаются две картинки в высоком разрешение; сначала генерируется a side-output swapped face путём смешивания structure attributes source и target изображений, переиспользования appearance attributes target и подачи этого на вход генератору StyleGan; далее для переноса заднего фона исопользуется encoder, выход которое смешивается с выходом блоков StyleGAN и генерируется финальное изображение.

2.3.1 Class-Specific Attributes Transfer

В StyleGAN [13] подаются 18 векторов размерности 512, первые 7 из них будут отвечать за structure attributes. Чтобы получить латентные представления в $\mathcal{W}^+$, source и target пропускаются через предобученный pSp Encoder [18]. Далее чтобы лучше воспроизвести позу и выражение target face, используется structure transfer latent direction $\vec{n}$, который получается путём прогона source and target landmarks через обучаемый Landmark Encoder[1]. Тогда конечное латентное предствление — это $\hat{w} = \mathrm{Cat}(g_s + \vec{n}, h_t)$, g_s — это первые 7 векторов латентного представления source, а h_t — 11 последних векторов латентного представления

[1]По сути это тот же pSp Encoder, но с уменьшенным количеством возращаемых векторов.

target. Далее это подаётся в предобученный generator StyleGAN для получения y_s.

2.3.2 Background Transfer

Target пропускается через Target Encoder, который предстваляет из себя 8 downsample блоков со свёрточными слоями. Пусть $F_t = \{f_t^1, \ldots, f_t^8\}$, где f_t^i — это выход i-ого слоя target encoder. Пусть $F_s = \{f_s^1, \ldots, f_s^8\}$, где f_s^i — это выход i-ого слоя generator StyleGAN. Пусть m — это маска лица таргета: высокие значения стоят там, где находится человек, а низкие, там где его нет. На каждом слое декодера берутся f_t^i и f_s^i и складываются

$$f_t^i * (1 - m) + f_s^i * m \tag{8}$$

Таким образом decoder[2] не только переносит информацию из target (фон), но и работает с лицом с помощью выходов из StyleGAN.

Авторы учат Landmark Encoder, Target Encoder, Decoder и discriminator. Для этого используются следующие лоссы:

- $\mathcal{L}_{adv} = \mathbb{E}[-\log(D_f(y_f))]$
- $\mathcal{L}_{id} = 1 - \cos(\mathrm{R}(x_s), \mathrm{R}(y_f))$, где y_f — финальный результат, R — ArcFace [3]
- $\mathcal{L}_{lmk} = \|E(y_s) - E(x_t)\|_2 + \|E(y_f) - E(x_t)\|_2$, где y_s — side-output face, E — предобученный landmark estimator [14]
- $\mathcal{L}_{rec} = \begin{cases} \|y_s - x_t\|_2 + 0.8\|\mathrm{R}(y_s) - \mathrm{R}(x_t)\|_2 + \|y_f - x_t\|_2 + 0.8\|\mathrm{R}(y_f) - \mathrm{R}(x_t)\|_2, & x_t = x_s, \\ 0, & \text{otherwise} \end{cases}$,
  где R — perceptual feature extractor.
- $\mathcal{L}_{st} = \|y_f - HM(y_f, x_t)\|_2$, где HM — histogram mapping [19]

Общий лосс тут выглядит как

$$\mathcal{L} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{id}\mathcal{L}_{id} + \lambda_{lmk}\mathcal{L}_{lmk} + \lambda_{rec}\mathcal{L}_{rec} + \lambda_{st}\mathcal{L}_{st}, \tag{9}$$

, где $\lambda_{adv} = 1, \lambda_{id} = 2, \lambda_{lmk} = 0.1, \lambda_{rec} = 2, \lambda_{st} = 0.2$.

[2] он имеет зеркальную структуру.

2.3.3 Эксперименты

StyleGAN2 generator и pSp Encoder были предобучены на FFHQ [6]. Для оценивания были использованы CelebA-HQ [5] и FaceForensics++ [8].

В качестве метрик авторы использовали следующее:

- ID retrieval – то же самое, что и FaceShifter.
- ID similarity – косинусная близость в ID retrieval.
- Pose – то же самое, что и FaceShifter.
- Expression – то же самое, что и FaceShifter.
- Frechet Inception Distance (FID) — Wasserstein-2 расстояние между распределениями настоящих и swapped images.

Table 1. Quantitative evaluation of face swapping on CelebA-HQ dataset with four metrics. ↓ denotes the lower the better and vice versa, the best results are marked in **bold**.

Methods	ID Simi.↑	Pose Err.↓	Exp. Err.↓	FID↓
MegaFS [55]	0.5214	3.498	2.95	11.645
Ours	**0.5688**	**2.997**	**2.74**	**9.987**

Table 2. Quantitative evaluation of face swapping on FaceForensics++ dataset with four metrics. ↓ denotes the lower the better and the better and vice versa, the best results are marked in **bold**.

Methods	ID Retri.(%)↑	Pose Err.↓	Exp. Err.↓
FaceSwap [2]	72.69	2.58	2.89
Deepfakes [1]	88.39	4.64	3.33
FaceShifter [26]	90.68	2.55	2.82
MegaFS [55]	**90.83**	2.64	2.92
Ours	90.05	**2.46**	**2.79**

Рис. 10: Сравнение на разных датасетах.

На Рис. 10 приведены результаты сравнения данного метода с другими популярными на тот момент.

2.4 SEAN

За основу ресёрча был взят SPADE [20], но были найдены две проблемы: 1) хотелось для каждой зоны картинки подавать нужный стиль; 2) идея подавать стилевую информацию только на вход – не самый лучший выбор, потому что существуют блоки по типу AdaIN [21](то есть информация о стиле подаётся как нормализующий параметр в разные слои).

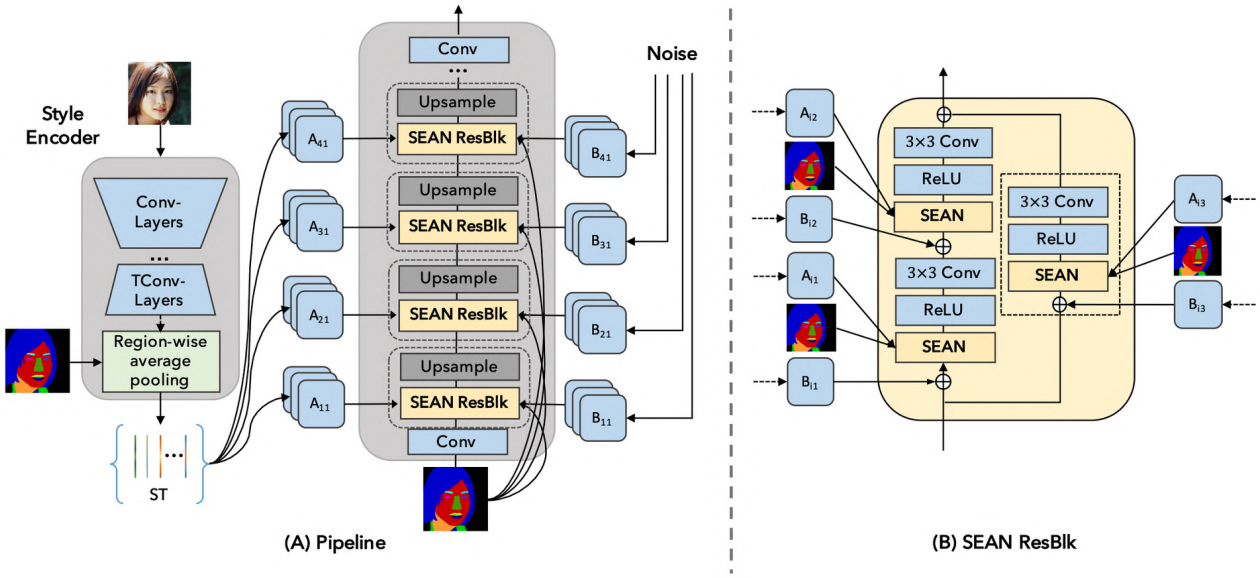

Рис. 11: Общая структура модели

2.4.1 Per-Region Style Encoder

Этот модуль(Рис 12) по входному стилю и сегментационной маске выдаёт матрицу(которую тут называют ST) размера $512 \times s$, где s — это количество семантических зон.

Region-wise average pooling — для каждой семантической зоны считаем среднее по выходу, на которого наложена данная маска.

2.4.2 SEAN

Generator представляет собой StyleGAN, в котором AdaIN был заменён на SEAN ResBlk (Рис 11 B). А сам SEAN выглядит так(Рис. 13)

Авторы учили каждый модуль с помощью следующих лоссов:

- $\mathcal{L}_{adv}$ – adversarial loss
- $\mathcal{L}_{FM} = \mathbb{E} \sum_{i=1}^{T} \frac{1}{N_i} \|\mathrm{D}^i - \mathrm{D}^i(\mathrm{G})\|_1$
- $\mathcal{L}_{perc} = \|\mathrm{F}(x) - \mathrm{F}(\hat{x})\|_2$

Общий лосс выглядит как

$$\mathcal{L} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{FM}\mathcal{L}_{FM} + \lambda_{perc}\mathcal{L}_{perc}, \tag{10}$$

, где $\lambda_{adv} = 1, \lambda_{FM} = 10, \lambda_{perc} = 10$.

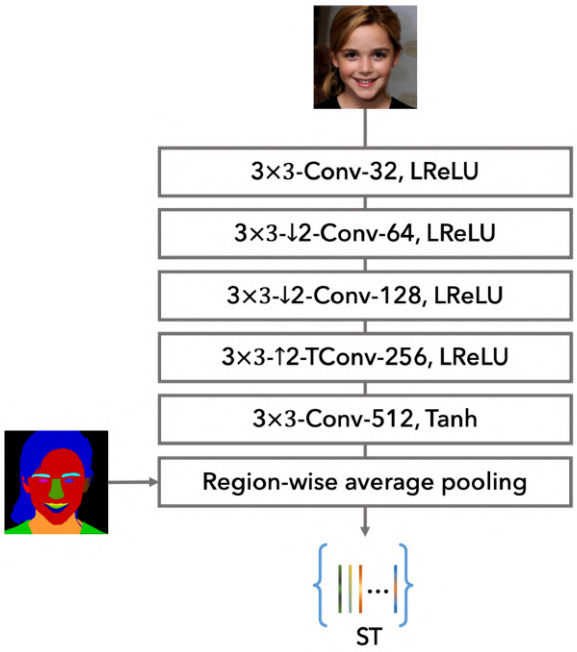

Рис. 12: Устройсво Style Encoder

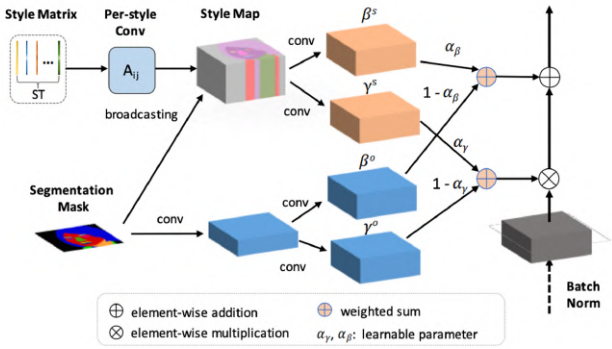

Рис. 13: Устройство блока SEAN

2.4.3 Эксперименты

Учили на следующих датасетах: CelebAMask-HQ [5], ADE20K [22], Cityscapes [23], Facades dataset [24].

В качестве метрик авторы использовали следующее:

- segmentation accuracy measured by mean Intersection-over-Union (mIoU)
- pixel accuracy (accu)
- FID

На Рис. 14 приведены результаты сравнения данного метода с другими популярными на тот момент.

Method	CelebAMask-HQ			CityScapes			ADE20K			Façades
	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID	FID
Ground Truth	73.14	94.38	9.41	66.21	93.69	32.34	39.38	78.76	14.51	14.40
Pix2PixHD [47]	76.12	95.76	23.69	50.35	92.09	83.24	22.78	73.32	43.0	22.34
SPADE [39]	**77.01**	**95.93**	22.43	56.01	93.13	60.51	**35.37**	**79.37**	34.65	24.04
Ours	75.69	95.69	**17.66**	**57.88**	**93.59**	**50.38**	34.59	77.16	**24.84**	**19.82**

Рис. 14: Сравнение на разных датасетах.

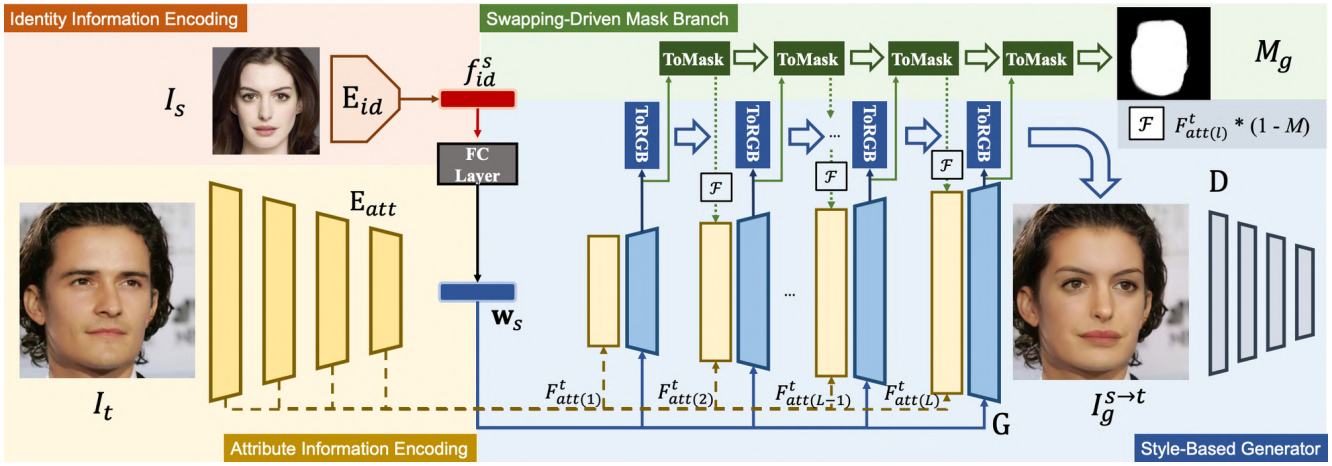

Рис. 15: Структура StyleSwap

2.5 StyleSwap

StyleSwap [25] состоит двух частей: Identity Information Encoding и Swapping-Driven Mask Branch.

2.5.1 Identity Information Encoding

Для получения латентного представления source пропускают через предобученный ArcFace [3], далее через 9 линейных слоёв, тем самым на выходе имеем w_s.(Рис. 15)

Target же в зависимости от разрешения пропускают либо через 4(если 256), либо через 5(если 512) свёрточных слоя, понижая размерность.

2.5.2 Swapping-Driven Mask Branch

За основу generator был взят StyleGAN2 [13].

Последний выход target encoder конкатинируется с первым входом в generator, они проходят через него, далее идёт слой ToRGB и ToMask. ToMask представляет из себя одноканальную свёртку с модуляцией и sigmoid(в формулах ' обозначает отсутствие sigmoid). На слоях >1 маска считается таким образом:

$$\tilde{M}_{g(l)} = upsample(\tilde{M}_{g(l-1)}) + M'_{g(l)} \tag{11}$$

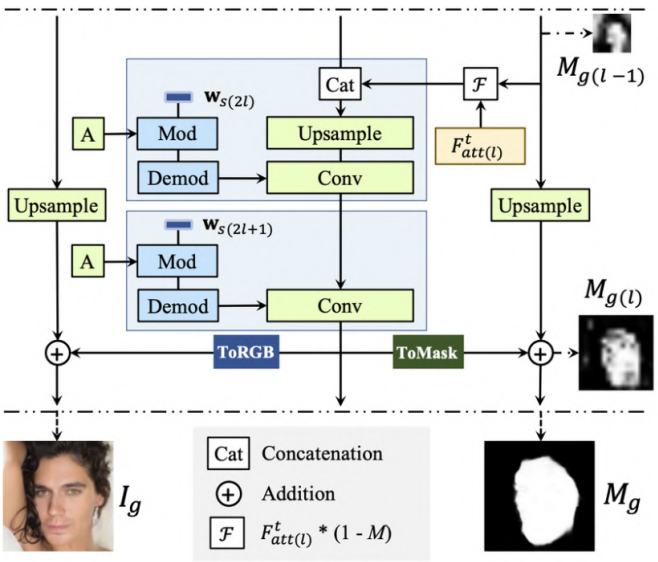

Рис. 16: Структура одного блока generator

Далее берётся предпоследний выход из target encoder, он маскируется по следующей формуле

$$\hat{F}^t_{att(l+1)} = F^t_{att(l+1)} * (1 - M_{g(l)}) \tag{12}$$

и дальше всё повторяется. Финальная картинка получается, как

$$\hat{I}^{s\to t}_g = M_g * I^{s\to t}_g + (1 - M_g) * I_t, \tag{13}$$

где M_g — это финальная маска, а $I^{s\to t}_g$— это финальный выход ToRGB.

Дискриминатор идентичен оригинальной реализации в [13].

Учатся target encoder, FC, ToMask, generator и discriminator на следующие лоссы:

- $\mathcal{L}_{adv}$ — для обучение GAN.
- $\mathcal{L}_{id} = 1 - cos(f^s_{id}, E_{id}(I_g))$ — смотрим, чтобы identity совпадала.
- $\mathcal{L}_{FM} = \sum_{m=nd}^{Nd}(\|D_m(I_g) - D_m(I_t)\|_1)$ — чтобы атрибуты совпадали (D — discriminator layer).
- $\mathcal{L}_{rec} = \begin{cases} \|I_g - I_t\|_1 + \sum_{m=1}^{N_{vgg}}(\|VGG_m(I_g) - VGG_m(I_t)\|_1), & I_t = I_s, \\ 0, \quad \text{otherwise} \end{cases}$, где VGG — это pre-trained VGG19 layer.
- $\mathcal{L}_{mask} = BCELoss(M_g, M_j)$ — нужен для того, чтобы маски становились всё лучше и лучше.

Method \ Metric	ID Retrieval ↑	ID Cosine ↑	Pose error↓	Exp. error ↓	FID ↓
Deepfakes [10]	86.43%	0.438	3.96	8.98	4.07
FaceShiter [25]	90.04%	0.510	2.19	6.77	3.50
SimSwap [7]	93.07%	0.578	**1.36**	**5.07**	3.04
MegaFS [60]	89.12%	0.497	3.69	10.12	4.62
InfoSwap [14]	95.82%	0.635	2.54	6.99	4.74
StyleSwap (Ours)	**97.05%**	**0.677**	1.56	5.28	**2.72**
StyleSwap w/ $\mathcal{W}^+$	**97.87%**	**0.706**	1.51	5.27	**2.58**

Рис. 17: Сравнение на FaceForensics++.

Общий лосс выглядит как

$$\mathcal{L} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{id}\mathcal{L}_{id} + \lambda_{FM}\mathcal{L}_{FM} + \lambda_{rec}\mathcal{L}_{rec} + \lambda_{mask}\mathcal{L}_{mask}, \tag{14}$$

, где $\lambda_{adv} = 1, \lambda_{id} = 10, \lambda_{FM} = 100, \lambda_{rec} = 100, \lambda_{mask} = 1$.

2.5.3 Эксперименты

Учились на VGGFace [7], VoxCeleb2 [26], GPEN [27](первые два 256x256, последний 512x512). Тестировали на FaceForensics++ [8] и CelebA-HQ [5](первый 256x256, второй 512x512).

В качестве метрик авторы использовали следующее:

- ID retrieval
- ID cosine
- Pose error
- Expression error

На Рис. 17 приведены результаты сравнения данного метода с другими популярными на тот момент.

2.6 E4S

2.6.1 Reenactment

На вход подаются source и target. Они пропускаются через pre-trained FaceVid2Vid [28] и на выходе получается drive image I_d. Далее из I_d и I_t получаются сегментационные маски M_d, M_t с помощью pre-trained face-parser [34]. Эти маски содержат 12 категорий: задний фон, брови, глаза, нос, рот, губы, кожа лица, шея, волосы, уши, солнечные очки и серьги.

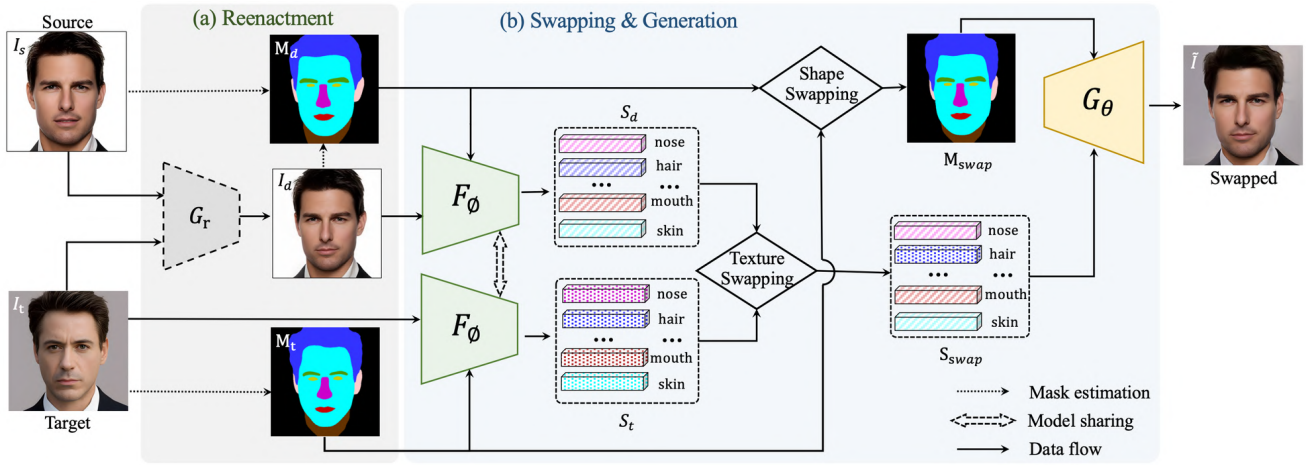

Рис. 18: Структура E4S [1]

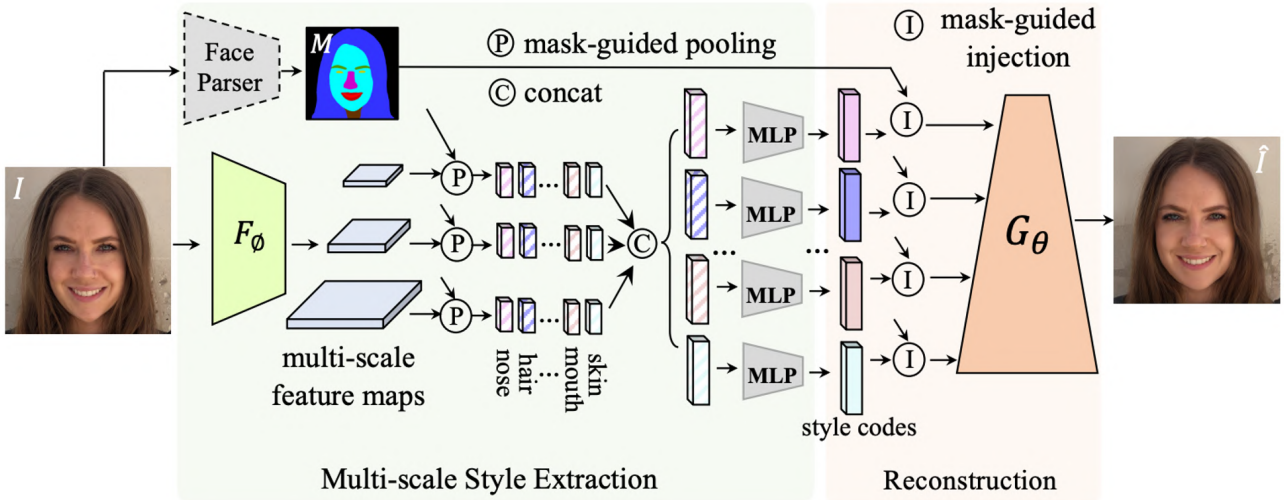

Рис. 19: Работа RGI

2.6.2 Mask-guided Style Extraction

На этом этапе I_d и I_t по очереди пропускаются через F — свёрточная сеть. В результате для каждой картинки имеем multi-scale feature maps $[F_1, \ldots, F_N]$, где N — количество этих маштабов. Далее для каждого изображения, маштаба и стиля можем получить следующий вектор

$$v_{ij} = AVG(F_i \odot ([M]_i == j)), \forall j \in \{1, 2, \ldots, C\} \tag{15}$$

, где C — это количество категорий(12), $[M]_i$ — downsample mask под размер F_i.

Далее каждый атрибут мы конкатинируем и пропускаем через линейный слой, то есть

$$s_i = MLP([v_{i1}, \ldots, v_{iN}]) \in \mathbb{R}^{18 \times 512}. \tag{16}$$

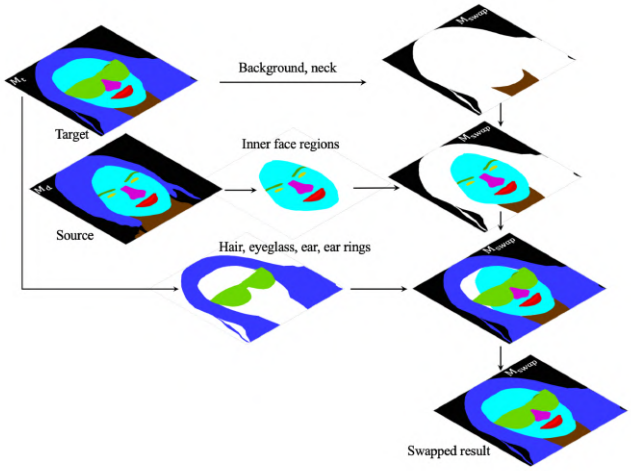

Рис. 20: Процесс создания итоговой маски

По итогу для каждой картинки получается тензор $s \in \mathbb{R}^{C \times 18 \times 512}$.

Но перед дальнейшей отправкой из двух тензоров получают один S_{swap}. От S_d берутся брови, глаза, нос, рот, губы, кожа лица, шея and уши; а от S_t всё остальное.

Последнее, что происходит на данном этапе — это получение итоговой маски M_{swap}. Делают это следующим образом(Рис. 20): берут пустую маску, от target туда переносят background и neck, далее от drive mask inner face regions, потом волосы, очки, уши and серьги берутся от target. В конце пропуски между лицом и волосами заполняются лицевой категорией.

2.6.3 Mask-guided Style Injection

За основу generator авторы взяли pre-trained StyleGAN2 [13](Рис. 21). Но теперь на каждый блок подаётся не один вектор, а C векторов(для каждого атрибута). После применения свёртки каждый выход поэлементно умножается на маску своей категории, а потом все выходы суммируются, то есть

$$F_l = \sum_{i}^{C}((F_{l-1} * W_{il}) \odot ([M]_l == i)), \forall l \in \{1, 2, \ldots, K\} \tag{17}$$

$$W_{il} = demod(mod(W_l, s_{il})) \tag{18}$$

Такой mask inject происходит только на первый K слоях. Эмпирически авторы установили K равным 13. Оставшиеся 5 слоёв это просто не pre-trained StyleGAN2.

По итогу получается изображение 1024x1024. Финальная картинка — это блендиг её с target, то есть

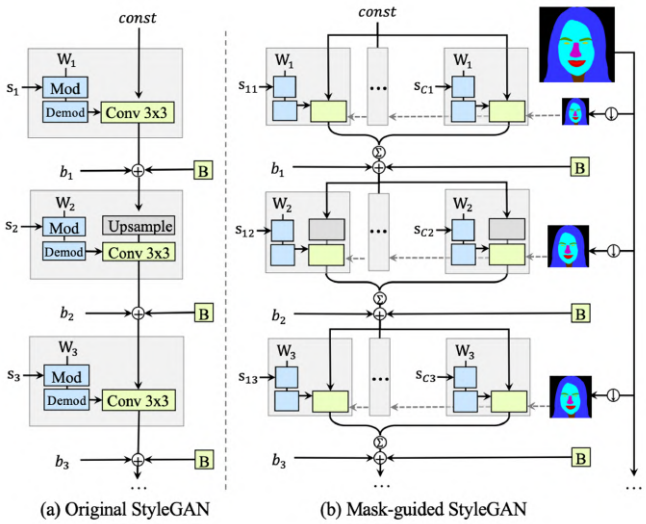

Рис. 21: Структура Mask-guided Style Injection

$$\hat{I} = M \odot I + (1 - M) \odot I_t \tag{19}$$

В качестве дискриминатора был взят pre-trained StyleGAN2 discriminator.

В данной моделе учатся лишь F(который multi-scale feature maps), MLP для каждого атрибута и 5 слоёв StyleGAN. Очень удобно, что в отличие от других фреймворков, тут на вход в качетсве target и source подаётся одно и то же изображение:

- $\mathcal{L}_{rec} = \|\hat{I} - I\|_2^2$
- $\mathcal{L}_{LPIPS} = \sum_s \|V[\hat{I}]_s - V[I]_s\|_2^2$, где V — это предобученный на ImageNet AlexNet [29], $[\cdot]_s$ — downsample to $s \in \{256, 512, 1024\}$.
- $\mathcal{L}_{id} = \sum_{i=1}^{5}(1 - cos(R_i(\hat{I}), R_i(I)))$, где R_i — i-ый выход из pre-trained ArcFace [3].
- $\mathcal{L}_{parsing} = \sum_{i=1}^{5}(1 - cos(P_i(\hat{I}), P_i(I)))$, где P_i — это pre-trained face parser [30].
- $\mathcal{L}_{adv} = \mathbb{E}[1 - \log D(\hat{I})] + \mathbb{E}[\log D(I)]$

Общий лосс выглядит как

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{LPIPS}\mathcal{L}_{LPIPS} + \lambda_{id}\mathcal{L}_{id} + \lambda_{parsing}\mathcal{L}_{parsing} + \lambda_{adv}\mathcal{L}_{adv}, \tag{20}$$

где $\lambda_{rec} = 1, \lambda_{LPIPS} = 0.8, \lambda_{id} = 0.1, \lambda_{parsing} = 0.1, \lambda_{adv} = 0.01$.

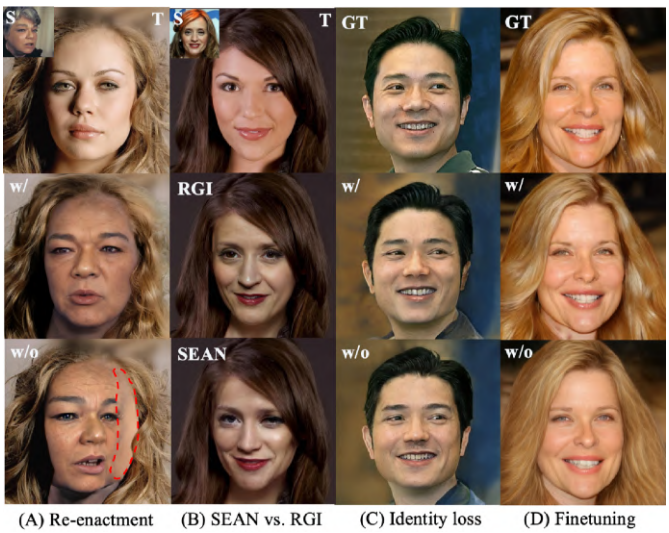

Рис. 22: Проверка важности разных модулей модели.

2.6.4 Эксперименты

Обучали на CelebAMask-HQ [5](сегментационные маски содержатся в датасете), FFHQ [6](маски были получены с помощью [34]).

Провели эксперименты для оценки важности некоторых составляющих модели(Рис. 22). Как видно, с ними значительно лучше.

В качестве метрик авторы брали всё то же самое, что и остальные. Оценивали на 500 рандомных картинках из CelebAMask-HQ(при этом учились на FFHQ).

Method	ID retrieval↑		Pose↓	Expr.↓
	Top-1	Top-5		
FSGAN † [25]	0.17	0.32	2.33	2.45
SimSwap ‡ [9]	0.12	0.32	2.89	2.84
FaceShifter ‡ [24]	0.06	0.26	**1.73**	**2.35**
HifiFace ‡ [38]	0.15	0.37	2.77	2.82
MegaFS * [48]	0.29	0.45	3.03	3.05
StyleFusion * [15]	0.35	0.18	5.37	2.94
HiRes ‡ * [41]	0.05	0.51	2.71	2.83
Ours † *	**0.38**	**0.54**	3.29	3.05

Рис. 23: Сравнение на CelebAMask-HQ.

На Рис. 23 приведены результаты сравнения данного метода с другими популярными на тот момент.

3 Выбор методов для запусков

В выборе статей для запуска приходилось отталкивался от того, есть ли чекпоинты для моделей и является ли данная модель SOTA. В статьях для обзора есть 2 SOTA, но лишь

для E4S [1] есть все необходимые чекпоинты. С другой же стороны нет смысла запускать что-то, что, возможно, уже не очень актуально, поэтому был выбран FSLSD [17], так как это предыдущая SOTA, поэтому это будет хорошим дополнением для сравнения полученных результатов; для неё так же есть все необходимые чекпоинты.

4 Протокол тестирования и визуализации

В качестве датасета для обучения и тестирования взят CelebA-HQ [5]: он был поделен на на train (28k изображений) и test (2k); гиперпараметры были такие же, что и в статьях; в качестве метрик взяты ID similarity(косинусная похожесть латентных представлений source и swap, полученные с помощью предобученного Arcface, pose error (ℓ_2 расстояние между векторами target и swap, полученные с помощью предобученного hopenet), expression error (ℓ_2 расстояние между 68 landmarks target и swap) и FID (Fréchet inception distance). Для визуализации source, target и swap конкатинируются и отсматриваются.

5 Результаты запусков методов

Все метрики считались по тестовым изображениям, т.е. половина была в качестве source, половина в качестве target. В таблице 1 указаны полученные результаты.

	ID similarity ↑	Pose error ↓	Expr. error ↓	FID ↓
FSLSD	0.19	3.41	0.12	1.8
E4S	0.39	4.07	0.18	2.03

Таблица 1: Метрики запущенных методов.

Если сравнивать со статьями, то pose error совпал с результатами авторов, id similarity для E4S так же совпал, а вот expression получился другим. Подсчёт метрик очень связан с тем, какие изображения подаются на вход, например, результаты на нормализованных и ненормализованных изображениях заметно отличаются. Так как в статье авторы подробно не расписывают такие моменты, то досконально повторить их результаты не представляется возможным.

Если сравнивать изображения(Рис. 1) между собой, то E4S явно лучше справился с задачей: картинка приятнее, меньше артефактов, source identity сохранилась в большей мере.

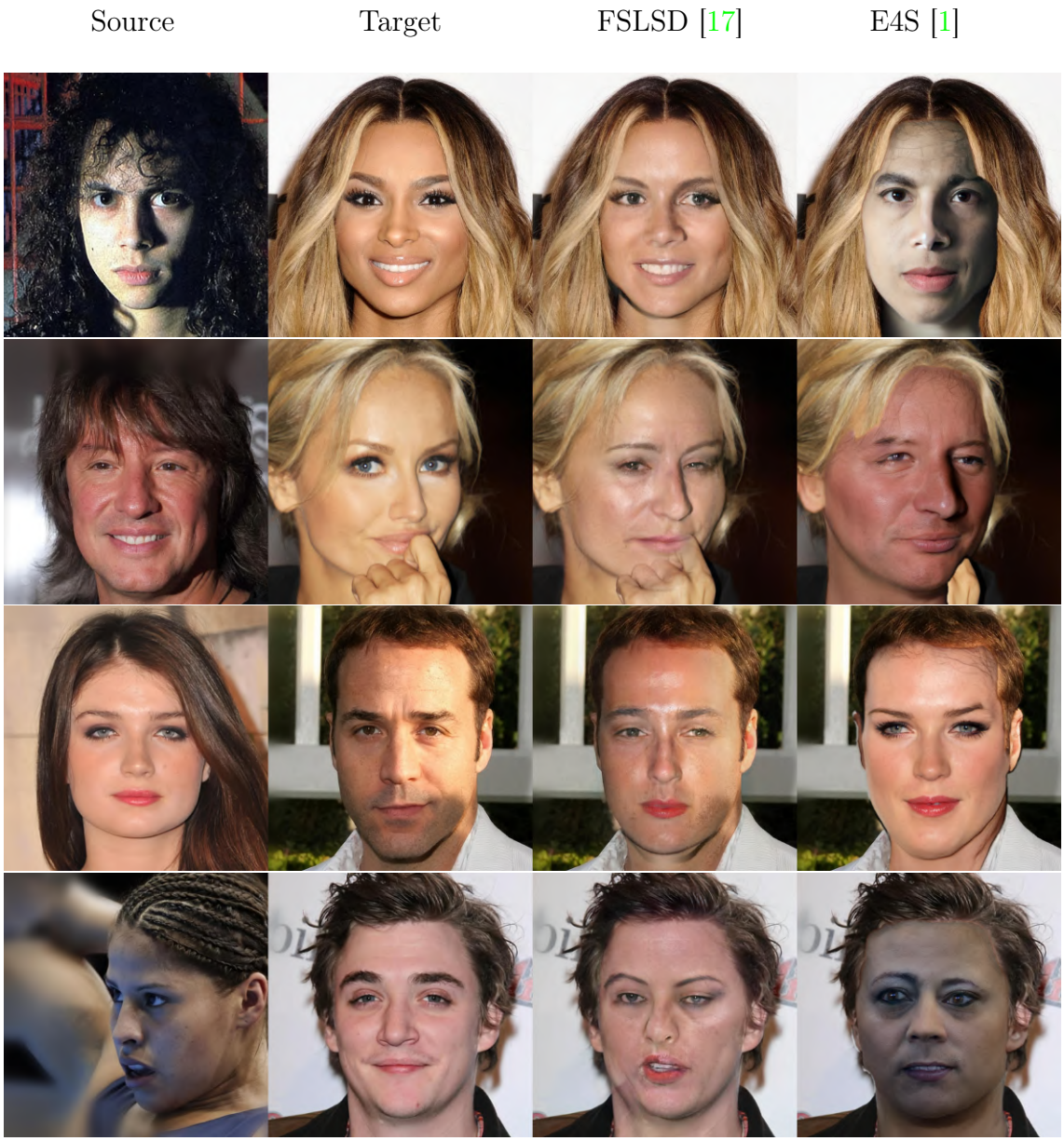

Рис. 1: Сравнение методов на CelebA-HQ.

На данный момент можно сделать следующий вывод: у FSLSD получаются более реалистичные картинки с точки зрения FID, поза и выражение у него сохраняется лучше; тем не менее E4S лучше сохраняет source identity и получает более качественные изображения.

6 Эксперименты с postprocessing

Первым делом был повторён эксперимент авторов. Получившиеся метрики указаны в Таблице 2. Результаты получились примерно такими же. Изображения (Рис. 1) так же получились похожими.

После детального осмотра изображений, были найдены следующие проблемы (Рис. 2):

	ID similarity ↑	Pose error ↓	Expr. error ↓	FID ↓
E4S(orig)	0.39	4.07	0.18	2.03
E4S(my train)	0.37	4.01	0.17	2.07

Таблица 1: Метрики запущенных методов.

Рис. 1: Сравнение результатов повторения запуска авторов (слева результат авторов, справа результат повтороного запуска).

1. На полученных изображениях можно заменить, что лицо находится как бы над волосами. Происходит так из-за ошибки в авторском коде.

2. Авторы считают, что цвет лица принадлежит source identity. В статье даже есть цитата: "Note that the skin is carefully considered here since it is identity-related, while it is neglected by most existing works". Как верно отмечено, в большенстве статьей это не так, поэтому в данной работе мы будем считать, что цвет лица должен быть от target изображения.

Чтобы исправить первую проблему, в коде было заменено следующее: сначала на маску добавлять лицо, а только потом волосы, перед добавлением очков (Рис. 3).

Чтобы исправить вторую проблему, на первом этапе была выдвинута следующая идея: брать латентные представления от target всего того, где содержиться кожа, т.е. лицо, шея, нос, уши (Рис. 4).

Понятно, что теперь изображения стали слишком target-обусловленными. Попробуем брать выпуклую комбинацию латентных представлений исходных изображений. На Рис. 5 показана линейная интерполяция между source и target с шагом 0.25.

На данный момент это все идеи, которые можно использовать при postprocessing, т.е. они не требуют переобучения моделей.

Рис. 2: Явные проблемы методов.

Рис. 3: Попытка устранения первой проблемы (слева проблемное фото, справа исправленное).

Рис. 4: Попытка устранения второй проблемы (слева проблемное фото, справа исправленное).

Рис. 5: Интерполяция от source к target.

	ID similarity ↑	Pose error ↓	Expr. error ↓	FID ↓
FSLSD	0.19	3.41	0.12	1.8
E4S(orig)	0.39	4.07	0.18	2.03
E4S(my train)	0.37	4.01	0.17	2.07
E4S(накладывание волос)	0.34	3.92	0.15	1.98
E4S(25% target)	0.31	3.5	0.15	1.93
E4S(50% target)	0.24	3.11	0.14	1.88
E4S(75% target)	0.16	2.82	0.14	1.77
E4S(100% target)	0.11	2.68	0.13	1.68

Таблица 2: Общие результаты для postprocessing экспериментов.

Результаты данных экспериментов находятся в Таблице 2. Что нужно заметить: делая изображение более реалистичным и улучшая pose и expression, мы ухудшаем id similarity, потому что заставляем swap быть больше похожим на target.

Последнее, что хочется отменить: авторы используют довольно старый парсер для лиц [34]; поэтому он был заменён на более новый [31]; глобально это ситуацию не улучшает, но теперь грани атрибутов становятся чётче. Также ко всем фотографиям был добавлен Laplacian blending, чтобы сглаживать полученное изображение.

7 Выдвижение гипотез

Основной проблемой на прошлом этапе было ухудшение id similarity, т.е. потеря source identity. На данном этапе мы попробуем исправить это.

Гипотеза 1: Увеличить значения коэффициента перед identity loss, тем самым больше штрафовать за то, что итоговое изображения содержет меньше информации об индетичности. Значения коэффициента будут взяты в 3 и в 10 раз больше.

Гипотеза 2: На данный момент метод обучается на реконструкцию, то есть в качестве source и target одно и то же изображение; хотя в остальных статьях из обзора литературы source и target были различными. Поэтому можно изменить обучение под разные source и target. В таком случае можно будет воспользоваться улучшениями, которые были предолжены для postprocessing.

Гипотеза 3: На данный момент из 18 слоёв StyleGAN учатся лишь 5 последних, причём mask injeciton выполняется на оставшихся 13. Поэтому можно попробывать несколько идей: обучить больше слоёв(7 вместо 5). Также попробуем делать mask injection на всех слоях StyleGAN, хотя достаточно сильно увеличит время обучения.

Гипотеза 4: На большинстве полученных изображений глаза смотрят в отличную от target точку. Для исправления этого можно добавить eye loss, который будет штрафовать за то,

Рис. 1: Проверка первой гипотезы. У 3 изображения коэффициент 0.1, у 4 0.3, у 5 1.0.

что положение глаз на исходном изображении отличается от полученного.

Теперь попробуем зайти с другой стороны. В методе авторов нас не устравивал полученный цвет кожи, в то время как identity переносилась в полной мере.

Гипотеза 5: Добавим дополнительный модуль, который будет перекрашивать лица. Попробуем покрасить driven (source изображение, которое было повёрнуто в ту же позу, что и target) в цвет target. Также попробуем сначала получить swap изображение, а потом красить его в цвет target.

8 Проверка гипотез

Так как метрики не очень передают качество изображений, ниже для каждой гипотезы будут приведены лишь получившиеся сгенерированные изображения, а метрики будут собраны в конце в одной таблице.

8.1 Первая гипотеза

На Рис. 1 приведены результаты проверки первой гипотезы. Визуально кажется, что результаты очень похожи, хотя на первом изображении зубы были лучше.

Гипотеза отвергается.

8.2 Вторая гипотеза

На Рис. 2 приведены результаты проверки второй гипотезы. Качество полученных фотографий сильно упало. Возникает вопрос: а почему другие методы учатся на разных изображениях и получают визуально неплохой результат? Ответа на этот вопрос нам найти не

Рис. 2: Проверка второй гипотезы.

удалось.

Гипотеза отвергается.

8.3 Третья гипотеза

На Рис. 3 приведены результаты проверки третьей гипотезы. Можно заметить, что увеличение обучаемых слоёв ухудшает качество, при этом постоянное применение mask injection улучшает. Особенно это заметно на фотографиях с зубами.

Таким образом принимаем гипотезу о том, что наличие mask injection на каждом слое

Рис. 3: Проверка третьей гипотезы. 3 изображение – 5 обучаемых слоёв, 4 изображение – 5 обучаемых слоёв и mask injection на всех слоях, 5 изображение – 7 обучаемых слоёв, 6 изображение – 7 обучаемых слоёв и mask injection на всех слоях.

делает изображения лучше.

8.4 Четвёртая гипотеза

Тут просто стоит отметить, что гипотеза достаточно бесполезная: если модель учится на реконструкцию изображений, глаза и так встанут куда нужно. Одако гипотезу при обучении на разных фотографиях проверять тоже нет смысла, так как в целом гипотеза об обучении на разных изображения была отвергнута.

8.5 Пятая гипотеза

В качестве модуля для изменения цвета был взят [33]. Для перекрашевания вместо всего лица брались лишь части лица, покрытые кожей, то есть кожа лица, уши, нос и шей. Далее полученная "маска"из новой кожи вставлялась в изображение. После перекрашивания к изображениям применялся блэндинг для сглажевания.

Кроме этого ещё раз был пересмотрен алгоритм построения маски для итогового изображения. Так, теперь сначала от target на маску добавляются задний фон, кожа, уши, зубы и губы, далее от source глаза, брови и нос, затем от target добавляются серьги, очки и волосы, а в конце пустоты заполняются лицом. Важно соблюдать именно такой порядок, потому что если сначала добавить например кожу, а потом уши, то получится что-то странное (далее мы увидем такой результат у авторов и у нашей первой попытки).

Последним изменением стал дополнительный блэндинг swap изображением с target вместе с очками, чтобы они лучше прорисовывались (на Рис. 3 видно, что очки правая линза у очков была размыта).

На Рис. 4 приведены результаты проверки пятой гипотезы. Видно, что при перекраске swap, итоговое изображение лучше сохраняет цвет target. Также видно, что из-за нового построения маски, уши выглядят более естественно.

Стоит отметить, что под результатом авторов подразумевается результат, полученный с помощью метода авторов, для которого были исправлены проблемы с лицом и волосами, применён новый парсер для лиц и блэндинг. Сделано это для того, чтобы сравнения были честными.

8.6 Подведение итогов по гипотезам

В Таблице 1 представлены результаты проверки гипотез. По качеству фотографий и метрикам в качестве рабочего метода пока что берём Гипотеза 5 (перекраска swap).

Рис. 4: Проверка пятой гипотезы. 3 изображение – результат авторов, 4 изображение – перекраска применялась для driven, 5 изображение – перекраска применялась к swap и изменено построение маски.

	ID similarity ↑	Pose error ↓	Expr. error ↓	FID ↓
FSLSD	0.19	3.41	0.12	1.8
E4S (fixed)	0.35	3.83	0.16	1.89
Гипотеза 1 (коэф=0.3)	0.12	2.96	0.14	1.71
Гипотеза 1 (коэф=1.0)	0.13	2.9	0.14	1.73
Гипотеза 2	0.13	2.75	0.12	1.66
Гипотеза 3 (5 слоёв и MI)	0.11	2.59	0.13	1.63
Гипотеза 3 (7 слоёв)	0.12	2.65	0.13	1.66
Гипотеза 3 (7 слоёв и MI)	0.12	2.56	0.13	1.67
Гипотеза 4	0.1	3.03	0.14	1.68
Гипотеза 5 (перекраска driven)	0.29	3.26	0.12	1.91
Гипотеза 5 (перекраска swap)	0.28	3.26	0.12	1.89

Таблица 1: Результаты проверки всех гипотез

Рис. 1: Результат работы метода на реальных примерах.

9 Проверка на реальных примерах

Теперь перейдём к более интересным примерам. Протестируем текущий метод фотографиях, где лицо не занимает почти всю площадь, а напротив, всего небольшой кусок изображения; к тому же размеры этих изображений будут различными.

Работает это следующим образом: сначала на изображениях находятся лица, после чего эти участки вырезаются и выравниваются; далее запускается метод как для фотографий, с которыми мы работали раньше; в конце полученное изображение вставляется в target.

На Рис. 1 представлены результаты работы метода на таких фотографиях. И сразу видны проблемы:

- Очень грубый переход на шее.
- Уши получаются мыльными, т. е. плохая прорисовка.
- Направление взгляда target отличается от swap. На самом деле эта проблема приследует нас с самого начала, но на больших изображениях он стала более явной.

Первые две проблемы решаются довольно быстро: чтобы их исправить перед финальным

Рис. 2: Результат работы метода на реальных примерах при дополнительном блэндинге шеи и ушей.

Рис. 3: Результат при замене модуля для reenactment.

изображением будем делать блэндинг ещё и шеи с ушами. Результаты реализации этой идеи на Рис. 2. Как видно, первые две проблемы успешно исправлены.

Для решения последней проблемы откажемся от текущего модуля (FaceVid2Vid [28]) для re-enactment в пользу нового (LIA [32]). Результаты реализации этой идей на Рис. 3. Эту проблему также исправили.

10 Заключение

В Таблице 1 преведены итоговые метрики данного метода, а также для методов, с которыми проходило сравнение. На Рис. 1 преведены результаты генерации изображений для данного метода, а также других SOTA методов. Исходя из качества изображений и получен-

	ID similarity ↑	Pose error ↓	Expr. error ↓	FID ↓
FSLSD	0.19	<u>3.41</u>	**0.12**	<u>1.8</u>
E4S (fixed)	**0.35**	3.83	0.16	1.89
MyE4S	<u>0.3</u>	**3.26**	**0.12**	**1.76**

Таблица 1: Результаты проверки всех гипотез. Жирным выделен лучший результат, подчёркнутый – второй после лучшего.

ных метрик, можно сделать вывод, что данный метод может претендовать на звание SOTA.

Рис. 1: Сравнение методов на CelebA-HQ.

Список литературы

[1] Zhian Liu, Maomao Li, Yong Zhang, Cairong Wang, Qi Zhang, Jue Wang, Yongwei Nie. Fine-Grained Face Swapping Via Regional GAN Inversion. In Proceedings of the IEEE/CVF CVPR, pages 8578-8587, 2023.

[2] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen. Advancing High Fidelity Identity Swapping for Forgery Detection. In Proceedings of the IEEE Conference on CVPR, pages 5074–5083, 2020.

[3] Jiankang Deng, Jia Guo, Niannan Xue, Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In Proceedings of the IEEE Conference on CVPR, pages 4690–4699, 2019.

[4] Sergey Ioffe, Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015.

[5] Tero Karras, Timo Aila, Samuli Laine, Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, variation. arXiv preprint arXiv:1710.10196, 2017.

[6] Tero Karras, Samuli Laine, Timo Aila. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE Conference on CVPR, pages 4401–4410, 2019.

[7] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In bmvc, volume 1, page 6, 2015.

[8] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. arXiv preprint arXiv:1901.08971, 2019.

[9] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In Proceedings of the IEEE Conference on CVPR, pages 5265–5274, 2018.

[10] Nataniel Ruiz, Eunji Chong, James M Rehg. Finegrained head pose estimation without keypoints. In Proceedings of the IEEE Conference on CVPR Workshops, pages 2074–2083, 2018.

[11] Bindita Chaudhuri, Noranart Vesdapunt, Baoyuan Wang. Joint face detection и facial motion retargeting for multiple faces. In Proceedings of the IEEE Conference on CVPR, pages 9719–9728, 2019.

[12] Yuhao Zhu, Qi Li, Jian Wang, Cheng-Zhong Xu, Zhenan Sun. One shot face swapping on megapixels. In Proceedings of the IEEE/CVF Conference on CVPR, pages 4834–4844, 2021.

[13] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, Timo Aila. Analyzing и improving the image quality of stylegan. In Proceedings of the IEEE Conference on CVPR, pages 8110–8119, 2020.

[14] Jingdong Wang, Sun ke, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, Bin Xiao. Deep high-resolution representation learning for visual recognition. IEEE Transactions on Pattern Analysis и Machine Intelligence, pp:1–1, 04 2020.

[15] Ziwei Liu, Ping Luo, Xiaogang Wang, Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of the IEEE International Conference on Computer Vision, pages 3730–3738, 2015.

[16] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, Xin Tong. Accurate 3d face reconstruction with weakly-supervised learning: From single image to image set. In Proceedings of the IEEE Conference on Computer Vision и Pattern Recognition Workshops, pages 0–0, 2019.

[17] Yangyang Xu, Bailin Deng, Junle Wang, Yanqing Jing, Jia Pan, Shengfeng He. High-resolution face swappingvia latent semantics disentanglement. In Proceedings of the IEEE/CVF Conference on CVPR, pages 7642–7651, 2022.

[18] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In CVPR, pages 2287–2296, 2021.

[19] Tingting Li, Ruihe Qian, Chao Dong, Si Liu, Qiong Yan, Wenwu Zhu, Liang Lin. Beautygan: Instance-level facial makeup transfer with deep generative adversarial network. In ACM Multimedia, pages 645–653, 2018.

[20] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In Proceedings of the IEEE Conference on CVPR, 2019.

[21] Xun Huang, Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization, 2017.

[22] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, Antonio Torralba. Scene parsing through ade20k dataset. In Proceedings of the IEEE Conference on CVPR, 2017.

[23] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In Proc. of the IEEE Conference on CVPR (CVPR), 2016.

[24] Dragomir Anguelov, Carole Dulong, Daniel Filip, Christian Frueh, Stephane Lafon, Richard Lyon, Abhijit Ogale, Luc ´Vincent, Josh Weaver. Google street view: Capturing the world at street level. Computer, 43(6):32–38, 2010.

[25] Zhiliang Xu, Hang Zhou, Zhibin Hong, Ziwei Liu, Jiaming Liu, Zhizhi Guo, Junyu Han, Jingtuo Liu, Errui Ding, и Jingdong Wang.

[26] Chung, J.S., Nagrani, A., Zisserman, A.: Voxceleb2: Deep speaker recognition. arXiv preprint arXiv:1806.05622 (2018).

[27] Yang, T., Ren, P., Xie, X., Zhang, L.: Gan prior embedded network for blind face restoration in the wild. In: Proceedings of the IEEE/CVF Conference on CVPR. pp. 672–681 (2021).

[28] Ting-Chun Wang, Arun Mallya, Ming-Yu Liu. One-shot free-view neural talking-head synthesis for video conferencing. In Proceedings of the IEEE/CVF Conference on CVPR, pages 10039–10049, 2021.

[29] Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton. Imagenet classification with deep conv neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, и K. Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 25. Curran Associates, Inc., 2012.

[30] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, Ping Luo. Maskgan: Towards diverse и interactive facial image manipulation. In Proceedings of the IEEE/CVF Conference on CVPR, pages 5549–5558, 2020.

[31] Guo Meng-Hao, Lu Cheng-Ze, Hou Qibin, Liu Zhengning Cheng, Ming-Ming Hu, Shi-Min. SegNeXt: Rethinking Convolutional Attention Design for Semantic Segmentation. In arXiv preprint arXiv:2209.08575, 2022.

[32] Yaohui Wang, Di Yang, Francois Bremond, Antitza Dantcheva. Latent Image Animator: Learning to Animate Images via Latent Space Navigation. ICLR, 2022.

[33] iperov. DeepFaceLab. URL: https://github.com/iperov/DeepFaceLab/blob/master/core/imagelib/color_transfer.py.

[34] zllrunning. face-parsing.pytorch. URL: https://github.com/zllrunning/face-parsing.PyTorch.