AutoRetouch: Automatic Professional Face Retouching

Alireza Shafaei
Skylab Technologies Inc.†
The University of British Columbia
alireza@skylabtech.ai

James J. Little
The University of British Columbia
little@cs.ubc.ca

Mark Schmidt
The University of British Columbia
CCAI Affiliate Chair (Amii)
schmidtm@cs.ubc.ca

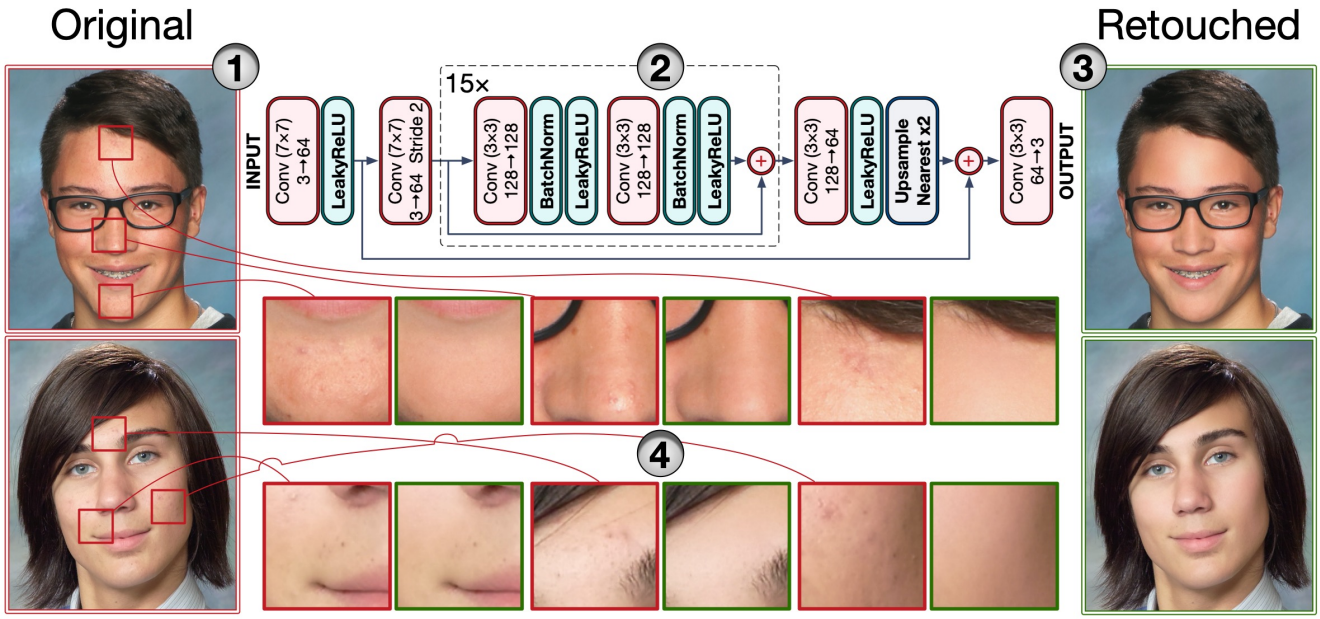

Figure 1: Overview of AutoRetouch: (1) input portraits cropped to the head, (2) the images pass through the fully convolutional architecture of AutoRetouch, (3) the output portraits are retouched while preserving fine textures. (4) Sample before-after patches scaled up. We empirically show that our final models have desirable generalization properties.

Abstract

Face retouching is one of the most time-consuming steps in professional photography pipelines. The existing automated approaches blindly apply smoothing on the skin, destroying the delicate texture of the face. We present the first automatic face retouching approach that produces high-quality professional-grade results in less than two seconds. Unlike previous work, we show that our method preserves textures and distinctive features while retouching the skin. We demonstrate that our trained models generalize across datasets and are suitable for low-resolution cellphone images. Finally, we release the first large-scale, professionally retouched dataset with our baseline to encourage further work on the presented problem.

1. Introduction

Phone cameras have reached an unprecedented level of image quality and resolution [1]. With the advancement of these consumer-level products, demand for image enhancement software has also increased significantly. Software features such as DeepFusion, Night Sight, and Bokeh now exist in virtually all high-end consumer phone cameras. Further improvement of phone cameras will only increase the demand for high-quality software enhancement.

On the other side, photography studios process thousands of high-quality portraits a day. In the USA alone and within the niche market of school photography, studios process over 56 million photos annually. One of the most time-consuming steps in a typical studio production pipeline is face retouching. On average, professional skin retouching

† The author produced the results at Skylab Technologies Inc.

requires two minutes of laborious work per portrait. Surprisingly, face retouching is still not automated in North America's largest studios that we were able to reach. Most studios that we surveyed were not happy with the quality of existing automation products, instead, relying on human labor. We show that the previous work on retouching does not adequately address the practical requirements of the domain. Furthermore, the lack of publicly-available datasets has stifled the research and development of working solutions. We take a close look at the skin retouching problem, discuss our findings, present our novel solution inspired by the most recent developments in computer vision and machine learning, and show that average consumers and professionals can use our model to enhance their photos. Finally, we release the first large-scale face retouching dataset with our baseline.

Figure 1 shows an overview of our AutoRetouch pipeline. Since we are developing a practical solution for real-world applications, we require the pipeline's output to be impeccable – the output image should not be deformed or degraded in any way. Furthermore, since we process high-resolution images, we need to design resource-sensitive models: a 24-megapixel image uncompressed in floating-point precision occupies over 250 MB memory. We first present the problem and discuss the subtleties that prevent a direct application of the existing methods. Then, we present our new strategy to address the specific challenges of the application. Our contributions are:

1. We present a carefully-designed and validated fast face retouching neural architecture with a low memory footprint that preserves the skin's distinctive features while removing blemishes. Our solution enables quick and high-quality automated face retouching for the first time, producing output superior to the groundtruth data. We compare our method with previous related work qualitatively and quantitatively. We show that our method generalizes across datasets and is useful for mobile phone applications.

2. We release the first large-scale professional face retouching dataset Flickr-Faces-HQ-Retouched (FFHQR), with our baseline to encourage more research on this problem.

2. Related Work

To the best of our knowledge, very little prior work exists for face retouching. Lin *et al*. [13] present an exemplar-based *freckle* retouching technique using a small dataset of cosmetic laser therapy. In the commercial domain, there is Visage Lab, BeautyPlus, Meitu, and Facetune, to mention a few. All of the available products provide face retouching and beautification tools, but none of them are fully automatic. Furthermore, none of these tools can provide high-quality, high-resolution face retouching even when all the parameters are carefully tuned. The existing methods

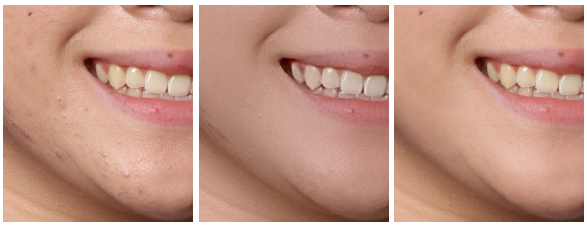

Figure 2: Automatic face retouching. The original image on the left. The middle image is the AI-assisted retouching of BeautyPlus. The result of our method is on the right.

Figure 3: Three samples from the Flickr-Faces-HQ-Retouching (FFHQR) Dataset. The left image is the original image from FFHQ [10], and the right image is the retouched version in our dataset. The figure is best viewed on a screen.

apply "blind" smoothing, where moles and freckles disappear, and fine skin-texture is destroyed (see Fig. 2). Our method preserves the identifying features of skin while removing blemishes at a high-resolution. We also release the first large-scale retouching dataset with our baseline to encourage further work on this problem (see Fig. 3).

A natural starting point for skin texture synthesis during retouching is thorough generative adversarial networks (GANs) [5]. GANs are a broad class of generative models within which two neural networks, a generator and a discriminator, compete against each other. During the training, the generator G produces samples from a *fake* distribution

$\tilde{\mathcal{D}}$ while the discriminator D tries to detect whether the incoming samples are from the target distribution $\mathcal{D}$, *i.e.*, if the image is *real* or *fake*. The objective of GANs is to learn a generator G such that the *real* distribution and the *fake* distribution are approximately equal, *i.e.*, $\tilde{\mathcal{D}} \approx \mathcal{D}$. Over the previous years, the research community has focused on improving the training time, stability, quality, and analysis of samples generated by GANs [2, 6, 3], while simultaneously identifying new applications [10, 7, 17].

One of the major developments based on GANs is the image-to-image translation method Pix2Pix [7]. Pix2Pix uses a conditional GAN loss plus ℓ_1 loss to learn a mapping $x \rightarrow y$. The authors use a U-Net [16] generator and a PatchGAN [12] discriminator. The PatchGAN discriminator model assumes independence between pixels separated by more than the patch window width. The authors present several use cases of Pix2Pix applications with impressive results. Our approach is similar to Pix2Pix [7] in the way that we learn a mapping function from original images to retouched images. However, our (i) network architectures, (ii) training loss function, and (iii) data augmentation schemes are different and demonstrably necessary to perform professional face retouching. We will further discuss the differences in Sec. 4 and will provide comparisons in Sec. 5.

3. Flickr-Faces-HQ-Retouching (FFHQR) Dataset

Before discussing our method, we first introduce a new face retouching dataset based on the Flickr-Faces-HQ (FFHQ) dataset [10]. The original FFHQ dataset consists of $70{,}000$ $1\,\mathrm{MP}$ face-aligned images that are collected from Flickr. We chose FFHQ as the basis of our new dataset because of the variety of ages, ethnicity, lighting conditions, and the large number of images that could benefit from face retouching. To create the new dataset, we hired a team of professional image editors to retouch the images. See Fig. 3 for samples of our dataset. To the best of our knowledge, FFHQR is the first publicly available retouching dataset.

One of the key features of professional retouching is that the image updates are sparse – most of the pixels do not change. This attribute is unlike the result of commonly used blurring in the commercial applications where the entire skin is smoothed to remove blemishes. The FFHQR dataset reflects this professional retouching style. Figure 4 shows a scatter plot of the percentage of pixel change versus the mean absolute value of pixel updates (range $[0,\ 255]$). In the majority of images, less than 40% of the pixels are edited. Furthermore, most pixel value changes are in the $[100,\ 200]$ range. This observation has implications in model design: the retouching methods should facilitate learning a function that preserves the input values strictly.

We also experiment with studio data and measure cross-dataset performance. We present more information about the studio in the next section. The FFHQR dataset and the evaluation scripts are publicly available.[1]

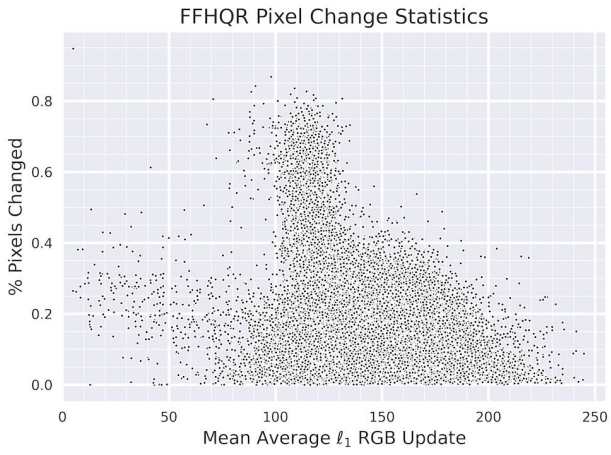

Figure 4: The scatter plot of FFHQR pixel update statistics. The y-axis is the percent of updated pixels per image. The x-axis is the mean absolute value of pixel updates. For the majority of images, less than 40% of pixels change.

4. AutoRetouch

Professionally retouched images must retain moles, freckles, and other distinctive features. Furthermore, the output must preserve the color tone of the original image (color-correction is a separate task in professional photography). The model must distinguish the features of the image that must be retained and the features that must be removed. Skin smoothing, which is the common algorithm in existing applications, will not preserve the identifying features. Furthermore, a blind smoothing alters the delicate texture of the original skin (see Fig. 2). Therefore, a desirable and practical retouching model needs to:

i Preserve the image structure.

ii Preserve the image texture while also performing skin retouching.

iii Generalize to a variety of skin tones and lightings.

To address these concerns, we carefully build a fully convolutional neural network that filters the input image. For (ii), a safe strategy is to train models that only retouch known blemishes. Since we need to process high-resolution images, even the slightest error will be visible. We empirically verify that our final model only removes the known skin blemishes and preserves everything else. Furthermore, we show that our method performs automatic face retouching while preserving the natural quality of the image. Interestingly, we present empirical results that indicate that our

[1] https://github.com/skylab-tech/ffhqr-dataset

final model produces a higher quality skin retouching than the training data.

4.1. Architecture

In our experiments, we observed that downscaling the image tensors to a factor smaller than half leads to slight pixel displacement and loss of the original texture in the output space. This observation eliminates most of the typical architectural choices such as U-Net [16] in Pix2Pix [7] since there are usually several levels of down-sampling in the architecture. We limited the downscaling in our network architecture search to preserve fine details.

Furthermore, we observed that incorporating additive skip-connection shortcuts decreases the random jitters in the output. Since we expect the majority of the pixels to remain unchanged, the skip connections also allow for more direct transfer of the input to the output.

Since we do not perform much downscaling, the depth of the architecture and the number of kernels would typically become limited by the available GPU memory. However, face retouching is mostly a local operation – the correction of any pixel does not depend on the far away pixels. This feature is unlike typical Pix2Pix-like applications, where the network takes the global context of the input image into account to generate the output. This problem structure enables a sliding-window strategy that we exploit with convolutional architectures [14]. Without loss of generality, we utilize more kernels and deeper architectures by limiting the input to a large-enough $w \times w$ sub-window.

Given that we limit our network's receptive field, the resulting models are not statistically dependent on the *entire image* – this provides more robustness to out-of-distribution test images, which enables a broader application than just the training image distribution. Combined with our data augmentation schemes, we show that our models can generalize across datasets with substantial domain shift. See Fig. 1 for our final generator architecture. We use the same discriminator architecture as Pix2Pix [7].

4.2. Training

The loss function that we use for training consists of (i) Relativistic Average GAN loss [9, 5], (ii) perceptual loss [8], and (iii) direct mean squared-error (MSE) loss:

$$\mathcal{L}_{\mathcal{G}} = \alpha \cdot \mathcal{L}^{\mathcal{G}}_{\text{ragan}} + \beta \cdot \mathcal{L}_{\text{perceptual}} + \gamma \cdot \mathcal{L}_{\text{mse}} \qquad (1)$$

We set $\alpha = 10^{-3}, \beta = 6 \cdot 10^{-3}, \gamma = 0.5$ in our experiments. For the Relativistic Average GAN, we use a running estimate of 300 previous predictions. For the perceptual loss, we use mean squared-error on the output features of the 14th layer in the 19-layer VGG model [18].

We experimented with the original GAN [5] as well as other variants such as WGAN [2]. Overall, we were able to achieve the best results using RAGAN [9]. We also experimented with ℓ_1 loss, but we found that the model with mean-squared-error loss converges much faster. Similar to Pix2Pix [7] we combine traditional loss functions with GAN-type loss functions. However, we (i) specifically use RAGAN, we (ii) also include a *perceptual loss* function, and (iii) use mean-squared-error instead of ℓ_1 loss. In Sec. 5, we do an ablation study to show how much each term of the loss function contributes to the output retouching quality. Our results indicate that each term contributes to the overall performance, and the synergy of the loss functions creates a strong baseline for automated professional face retouching. We show that our final model yields retouching results that are qualitatively better than the groundtruth data.

We repeatedly perform one gradient descent step on D and one gradient descent step on G to train our models. We train the studio data model for approximately two weeks on three 2080 Ti GPUs and one 1080 Ti. We train a model on the FFHQR dataset in just five days. We run the training for 500 epochs and set $w = 150\,\text{px}$. We set the batch-size to 75, the learning rate of Adam [11] to $2 \cdot 10^{-4}$ for our model and 10^{-4} for RAGAN discriminator.

4.3. Data Preprocessing

We use before-after professionally retouched image pairs for the training. We sample $w \times w$ image patches from the training data and perform mirror augmentation and color perturbation during the training. To encourage scale-invariance, we randomly downscale the images before sampling. Since there are quadratically more $w \times w$ windows in larger images than the spatially smaller ones, uniformly random scaling under-represents the patches in the larger images. To fix the under-representation, we first take a random scaling factor $s \sim \text{Unif}(s_{\min}, 1)$, where $s_{\min}$ is the smallest acceptable image scale cubed, and downscale the image to $s^{\frac{1}{3}}$. This change of variable ensures all windows in all image scales are equally likely (see appendices).

We evaluate our model using both the newly presented FFHQR dataset and also studio data. The FFHQR data consists of $70{,}000$ image pairs of original and retouched images. We use $56{,}000$ images for training, 7000 images for validation, and 7000 images for testing. The studio data contains $203{,}725$ image pairs of original and retouched images. We split the studio data into $158{,}683$, $22{,}409$, and $22{,}633$ for train, validation, and test. Following the same procedure as Karras *et al*. [10] (FFHQ), we pre-process the studio data by cropping the images to the head. While FFHQR is free-form in-the-wild photos of faces, the studio data is captured in strictly controlled, well-lit conditions with DSLR cameras.

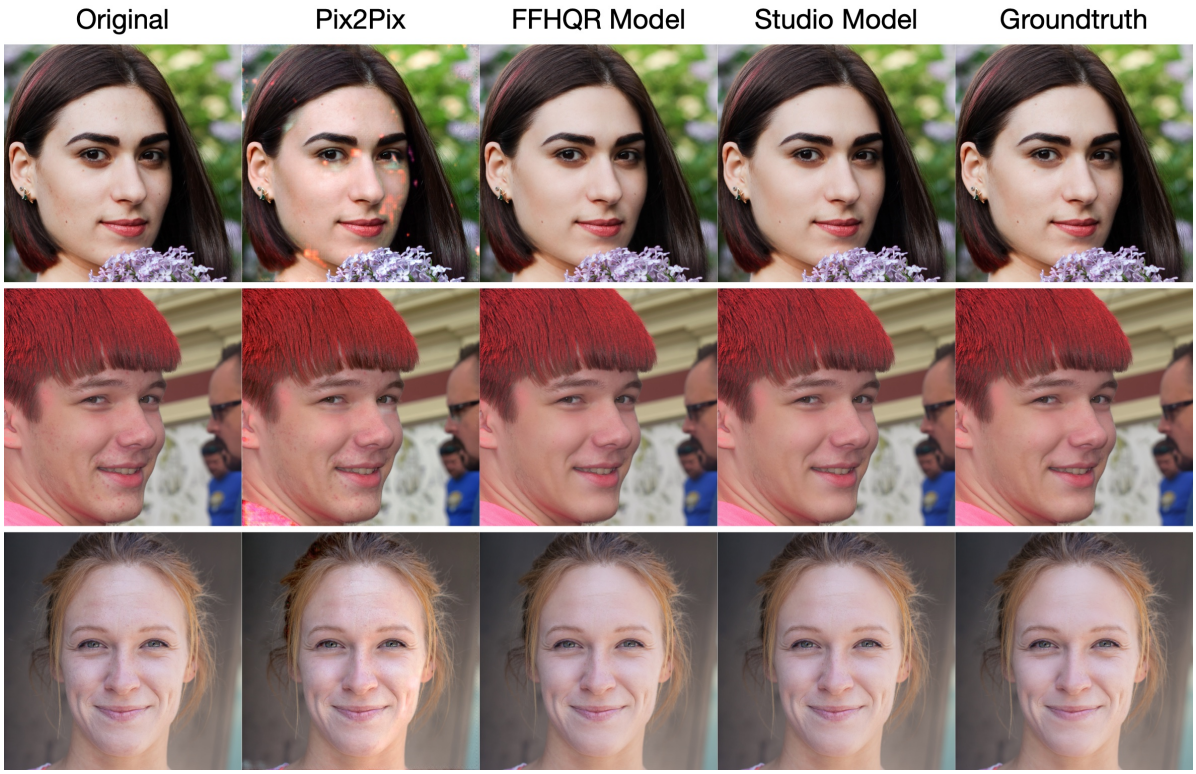

Figure 5: Outputs of Pix2Pix, FFHQR model, and the studio model on the FFHQR test set. The studio model generalizes well to FFHQR data even though there is a considerable domain shift. The Pix2Pix results often contain artifacts.

5. Experimental Results

Face retouching evaluation presents a similar set of challenges as the super-resolution (SR) literature. While the assessments in the SR literature primarily rely on reproducible measures such as PSNR and SSIM [19], Blau *et al*. [4] show that none of these metrics translate into a *better perceptual result* beyond a certain accuracy. In face retouching, this issue is more severe. Since professional retouching mostly produces sparse image updates, the before and the after images are already very similar before any processing. As a result, both of these metrics yield very high values, with only a slight variation across experiments. Our evaluation results on face retouching also confirm the findings of Blau *et al*. [4]: neither PSNR nor SSIM are reliable performance indicators for high-quality face retouching methods.

An alternative assessment in the SR literature is the subjective mean-opinion-score (MOS), where human subjects compare the perceptual quality of images generated with multiple algorithms. The MOS score is generally regarded as not reproducible and expensive to obtain. However, this approach can perceptually validate and compare different methods to produce reliable rankings.

To the best of our knowledge, there is no scalable and objective measure of quality for our application at the time of writing. We provide quantitative PSNR and SSIM comparisons as well as user-oriented studies. To perform the user study, we present professional retouchers with two methods' outputs and ask them to either choose the best output or skip if the images are too similar. We capture their choices and the time that it takes to make a decision. See Supplementary Material for more details of our setup.

For evaluation, we use a single RTX 2080Ti GPU. For Pix2Pix [7], we use the provided code and the recommended configuration by the authors. Figure 5 shows the output of our method when trained on FFHQR and the studio data. The studio model is only trained with the controlled studio data, whereas the FFHQR model is directly trained on FFHQR. Both of our models perform retouching well. However, Pix2Pix [7] fails to perform retouching and degrades the image. Figure 6 shows the test samples of retouched image patches based on our trained model on the studio data. Our model consistently preserves the moles and other identifying features while removing the blemishes.

Figure 7 compares the retouching output of our model with the groundtruth patches on the studio data. Notice that our model preserves the fine texture better than the groundtruth images. We suspect this happens because, in real-life retouching, the professionals may use large brushes for correction that would inadvertently affect areas of the image that do not require retouching. Our model, how-

Figure 6: Sample outputs of our retouching model. The figure is best viewed on a screen.

Figure 7: The output of our model versus the groundtruth retouching. The left column is the input, the middle column is groundtruth retouching, and the right column is our output. Our model preserves the fine details more than the groundtruth. See Supplementary Material for more images.

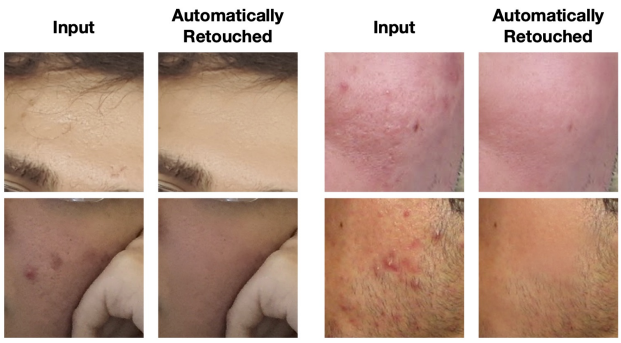

Figure 8: Sample input/output of retouched images captured with cellphones. The figure is best viewed on a screen. See Supplementary Material for more images.

ever, operates at the pixel level and can preserve details at no extra cost. Furthermore, in Sec. 5.1, we perform ablation studies on the effect of each term in the loss function. We observe that adding the RAGAN loss term encourages the model to preserve the input as much as possible. The preservation of the details produces retouching models that produce better images than the groundtruth retouching data. We observe the same trends in our user study.

We also test our studio retouching model on lower-resolution smartphone camera images. The images are captured with iPhone 6 or iPhone X. Figure 8 shows sample outputs from our tests. Although the studio model is not trained on cellphone images, it generalizes to lower-resolution and noisy images.

In Fig. 9, we test if the retouching network would correct unfamiliar patterns in the image. We manually add brush strokes of different colors and brush sizes with varying opacity to an image and process it using our model. Fig. 9 shows that our model did not change the added brush strokes. This behaviour is a desirable feature in professional face retouching since we wish to preserve the distinctive features as much as possible – it is safer for the model to retouch only known blemishes that require correction.

Figure 10 shows some of the failure cases of our retouching model. The images that our model fails to correct usually contain severe skin blemishes. Although our model improves the output image, it does not eliminate the blemishes.

5.1. Ablation Study

We perform a qualitative and quantitative ablation study to compare the effect of each term in Eq. 1. We run the following experiments on the loss function $\mathcal{L}$:

1. Only mean squared error (MSE) ($\alpha = \beta = 0$).
2. MSE and Perceptual ($\alpha = 0$).
3. MSE, Perceptual, and RAGAN.

For this evaluation, we run the experiment on a subset of 7905, 950, and 988 studio images for train, validation, and

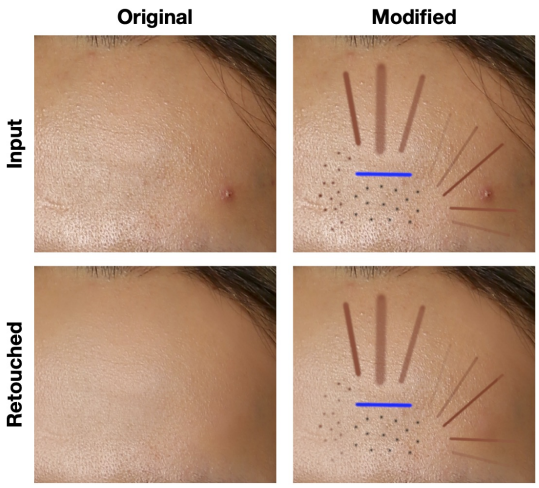

Figure 9: Our retouching model does not change unfamiliar patterns in the image. The model appears to be only responding to known skin blemishes. This behaviour is desirable in professional retouching to ensure distinctive features are preserved.

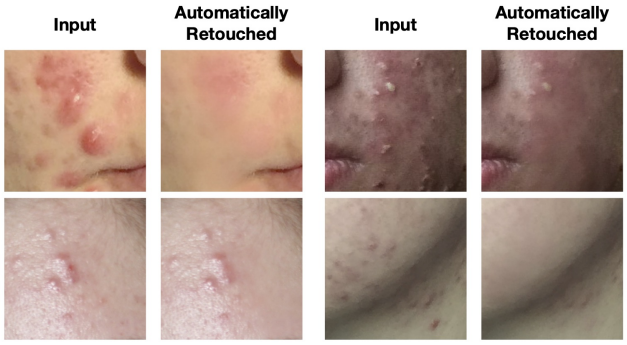

Figure 10: Failure cases when the blemishes are severe.

Table 1: Quantitative results using PSNR and SSIM. (1) is trained on studio data, (2) is trained on FFHQR.

		Studio Data		FFHQR	
		PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑
	No edit (input → output)	39.12	0.9807	45.58	0.9938
(1)	MSE	39.58	0.9858	40.53	0.9876
	MSE + Perc.	39.41	**0.9875**	39.04	0.9889
	MSE + Perc. + RAGAN	**41.04**	0.9865	44.82	0.9944
	Pix2Pix [7]	28.12	0.8893	29.58	0.9224
(2)	MSE	40.01	0.9844	**45.88**	0.9949
	MSE + Perc.	39.88	0.9851	45.00	**0.9952**
	MSE + Perc. + RAGAN	39.65	0.9849	44.92	**0.9952**
	Pix2Pix [7]	30.29	0.9152	33.45	0.9585

test, respectively. On FFHQR, we train on the entire training data. We run the training for 300 epochs and leave the other parameters as the original experiment. The training of each configuration takes 19 hours on studio data and three days on FFHQR. For each configuration, we use the model with the lowest validation loss.

Figure 11: The effect of loss function on retouching. Using MSE loss alone leads to smooth images, which is improved by adding a perceptual loss. However, a perceptual loss still does not preserve fine details. Adding an adversarial loss encourages the network to make as few changes on the image as possible. The figure is best viewed on a screen.

Table 1 shows the quantitative results. Since the before and after images are almost identical (except for the sparsely retouched regions), both PSNR and SSIM metrics will be too high. While in the super-resolution literature the best PSNR ≈ 25, and SSIM ≈ 0.75 [15], the "no edit" baseline already achieves PSNR $= 45$ and SSIM $= 0.99$ on FFHQR. Except for one case, the numerical results indicate that adding more terms to the mean-squared error (MSE) hurts the performance in terms of PSNR and SSIM. However, our user study reveals that the addition of perceptual loss and RAGAN improves the quality of retouching by a large margin (see Tab. 2 and Fig. 11). This observation is consistent with Blau *et al*. [4] – a higher PSNR or SSIM does not necessarily translate into higher perceptual quality.

Figure 11 shows the qualitative results of our ablation study. While MSE yields impressive results in retouching, it is easy to see the output of this network is smoothed, and sometimes fine textures (*e.g*. freckles) are removed. Although the images may appear pleasant at a low resolution,

Table 2: Human perceptual test. In each evaluation, we asked the subjects to chose their favourite method between two options. The participants could skip if the images were too similar. We measure how often a method is chosen and how long it took to make a decision. (1) Models are trained and tested on FFHQR; (2) models are trained and tested on studio data.

			Voted 1 ↑		Voted 2 ↑		Undecided		
#	Method 1	Method 2	%	Median Time (s)	%	Median Time (s)	%	Median Time (s)	n
(1)	MSE+Perc.+RAGAN	MSE	**50.1%**	28.0	17.9%	24.0	32.0%	26.3	730
	MSE+Perc.+RAGAN	MSE+Perc.	**28.3%**	30.9	22.9%	30.3	48.8%	26.1	791
	MSE+Perc.+RAGAN	Pix2Pix [7]	**100.0%**	1.9	0%	-	0%	-	508
(2)	MSE+Perc.+RAGAN	MSE	**94.6%**	12.2	4.4%	19.5	1.0%	19.6	802
	MSE+Perc.+RAGAN	MSE+Perc.	**95.7%**	9.8	3.4%	17.5	0.9%	15.7	898
	MSE+Perc.+RAGAN	Pix2Pix [7]	**99.9%**	1.3	0.1%	0.9	0%	-	1008
	MSE+Perc.+RAGAN	Groundtruth	**76.3%**	8.4	18.7%	12.0	5.0%	19.0	672

they do not possess the level of quality that is essential to professional high-resolution production. Adding perceptual loss improves the output by sharpening the image, but the network still does not preserve the input's delicate texture at all times. When we add the RAGAN loss, the network (seemingly) learns to minimize the output space changes, which results in minimal retouching that preserves the input details as much as possible. This quality leads to a better retouching than the groundtruth data (see Fig. 7).

Table 2 shows a summary of our user study results. We present the users with randomly ordered images of two test methods and ask them to pick the best photo. See Supplementary Material for a complete description of the setup. We run this experiment on both studio data and FFHQR, comparing against Pix2Pix [7] and models produced by our ablation study. Our proposed loss combination is favoured over the alternative more frequently across all the experiments. The preference gap between our method and the two alternatives is much larger on the studio data than for the FFHQR dataset. Furthermore, it took the users much longer to pick the best output on FFHQR than studio data. This gap is due to the higher quality imaging of the studio data compared to the in-the-wild FFHQR. While our proposed method still outperforms the alternatives, the difference is much more evident in the higher quality photos.

We also compare the output of our method with the groundtruth on the studio data. Since the studio images contain more details in perfect lighting, it is much easier to compare and perceive the difference. The skin-quality of our method is more frequently favored over the groundtruth data. Pix2Pix is seldom chosen, mainly because of the visible artifacts in the image. It took the users an average of two seconds to determine which output is better on both of the experiments that involved Pix2Pix models. Our proposed method perceptually outperforms alternative design choices despite having a slightly lower PSNR and SSIM.

We also have validated our automatic retouching model in real-world applications. Over the past year, our workflow has assisted professional retouching of over $1{,}000{,}000$ portraits. Typical failures that we have encountered ($< 2\%$) are when the image patches require a substantial amount of change, or the skin blemish is atypical. This usually happens when an extended region of the skin is covered in hard blemishes, and the natural color of the skin is no longer apparent (see Fig. 10). In these cases, the model removes the blemishes most often, but the area remains slightly red.

Our retouching model is only *20 MB* and requires *2580 MB* GPU RAM to process an input image of 1024×1024 in *1.1* s on a Titan RTX 2080Ti, or *1.9* s on a 1080Ti. The processing window size could be optimized according to the available resources down to a $w \times w$ sub-window at a time, while still producing an identical image. In contrast, the Pix2Pix [7] model is *208 MB* and occupies *1340 MB* memory while processing each image in *1.82* s on a 1080Ti.

6. Conclusion

We presented the AutoRetouch pipeline for automatic professional-grade face retouching. We explained the subtleties of face retouching, discussed the shortcomings of the previous work, and introduced our novel solutions to address the application's specific challenges. We presented qualitative and quantitative comparisons with previous related work. We compared our retouching models with several baselines on two datasets and highlighted the improvements, the generalization power, and the desirable properties that produce high-quality retouched images. We release the first large-scale retouching dataset FFHQR and invite the community to improve upon the presented work. The code to replicate our results is publicly available online.[2]

Acknowledgments We would like to thank the PhotoRetouchOnline.com and the Artona Group Inc. staff for their valuable feedback and support. This research was partially supported by the Canada CIFAR AI Chair Program and the NSERC Discovery Grants RGPIN-2015-06068 and 2018-04830.

[2] https://github.com/skylab-tech/autoRetouch

References

[1] DXOMARK - Smartphone and Digital Camera Reviews.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

[3] David Berthelot, Thomas Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[4] Yochai Blau, Roey Mechrez, Radu Timofte, Tomer Michaeli, and Lihi Zelnik-Manor. 2018 PIRM Challenge on Perceptual Image Super-resolution. In *ECCV Workshops*, 2018.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *NIPS*, 2014.

[6] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein GANs. In *NIPS*, 2017.

[7] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[8] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual Losses for Real-time Style Transfer and Super-resolution. In *ECCV*, 2016.

[9] Alexia Jolicoeur-Martineau. The Relativistic Discriminator: a Key Element Missing from Standard GAN. *ICLR*, 2019.

[10] Tero Karras, Samuli Laine, and Timo Aila. A style-based Generator Architecture for Generative Adversarial Networks. In *CVPR*, 2019.

[11] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint 1412.6980*, 2014.

[12] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *ECCV*. Springer, 2016.

[13] Tsung-Ying Lin, Yu-Ting Tsai, Tsung-Shian Huang, Wen-Chieh Lin, and Jung-Hong Chuang. Exemplar-based Freckle Retouching and Skin Tone Adjustment. *Computers & Graphics*, 78:54–63, 2019.

[14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. In *CVPR*, 2015.

[15] Andreas Lugmayr, Martin Danelljan, and Radu Timofte. Ntire 2020 challenge on real-world image super-resolution: Methods and results. In *CVPR Workshops*, 2020.

[16] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*. Springer, 2015.

[17] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017.

[18] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-scale Image Recognition. *ArXiv e-prints*, 2014.

[19] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.