A multi-scale context encoder for high quality restoration of facial images

1st Trung Dung Do
Faculty of Information Technology
Le Quy Don Technical University
Hanoi, Vietnam
dtdo@mta.edu

2nd Quoc Khanh Nguyen
Faculty of Information Technology
Le Quy Don Technical University
Hanoi, Vietnam
khanh29bk@mta.edu.vn

3rd Viet Hung Nguyen
Faculty of Information Technology
Le Quy Don Technical University
Hanoi, Vietnam
hungnv@mta.edu.vn

***Abstract*—In recent years, with the growth of data, storage and computing power many challenging problems can be solved. One of the most challenging and interesting tasks is to recover data in pixel level of image without using any additional information. Context encoder, a combination of auto-encoder and GAN, uses an unsupervised visual feature learning algorithm to predict missing pixel data in images. This paper presents a novel method to extract features in context encoder to restore high quality facial image. Instead of computing features from a single scale input image, the proposed method utilizes features from multi-scale input images to create features in the encoder step. The features are then fed into the decoder to create image with the needed information after feature aggregation in different ways. The restored images after using decoder and the original ones are then provided to a discriminator to make the network creating a looked-real image. Experimental results on some public data sets such as CELEBA-HQ and LFW show that the proposed method with features in multi-scale achieves a promising result compared with the original one.**

***Index Terms*—Context Encoder, multi-scale context encoder, image restoration**

I. Introduction

Deep learning has become the hottest trend in academic research as well as a great tool to make real-world products thanks to various deep neural network libraries such as Caffe [1], Theano [2], TensorFlow [3], CNTK [4], etc. Although, neural networks were created in the 1950s of the twentieth century, their power was only explored since the time Krizhevsky [5] published his very first work. With the advancement of a deep neural network, many challenges in computer vision, natural language processing, and speech processing are quickly resolved. Using deep learning, mankind is making a huge step and ability to redefine a new state-of-the-art in different areas more quickly. One of the most difficult and challenging problem in computer vision is to recover a missing data in a single image. Many researchers have tried to deal with this problem but the obtained results look very poor or unrealistic.

One of the most famous work to recover the missing data in an image is the Context Encoder [6] that utilizes the power of both Generative Adversarial Network [7] and Autoencoder [8] [9]. In that work, D. Pathak et al. use Autoencoder as a generator which creates a patch of missing data in the image. The recovered image patch then is fed to descriptor together with the original patch of image. The descriptor tries to distinguish the differences between the original image patch and the recovered image patch and let the Context Encoder learn to produce the realistic one. This early work, Context Encoder, is the huge step of humankind on recovering a missing data in the image. Compared with other works such as neighborhood pixel approximation, the results from Context Encoder is much better. However, it still has some limitations such as processing only on small image, the quality of recovered image to be far from applying on real-world problem, and training process to be difficult to converged.

This paper leverages the main concept of Context Encoder that encode the context information and decode the encoded feature to get the missing area in the image. In order to better capture context information, this work uses multiple encoders rather than a single encoder as demonstrated in the original work. The encoders are designed so that they produce the same size of feature map which then can be combined by using the adding or concatenating operation before feeding to the decoder.

The remainder of the paper is organized as follows: Section 2 introduces the proposed multi-scale context encoder that includes multiscale feature extraction and feature map aggregation. Section 3 discusses and shows the experimental results on some public datasets such as CELEBA-HQ and LWF data sets. Finally, Section 4 provides the conclusions and future works.

II. Multi-scale context encoder

Context encoder is a perfect combination of autoencoder [8] [9] and generative adversarial network [7]. The autoencoder is simple encoder-decoder which takes image with missing data as input and tries to encode the image to get feature representation. The decoder then uses the encoded feature representation to recover the information in the missing area. Without any correction, the result at first step looks unrealistic. Therefore, the generative adversarial network [7] needs to be used to guide the autoencoder learning the correct way to create more realistic result by simultaneously competing generator (autoencoder) and discriminator.

978-1-7281-6555-4/20/$31.00 ©2020 IEEE

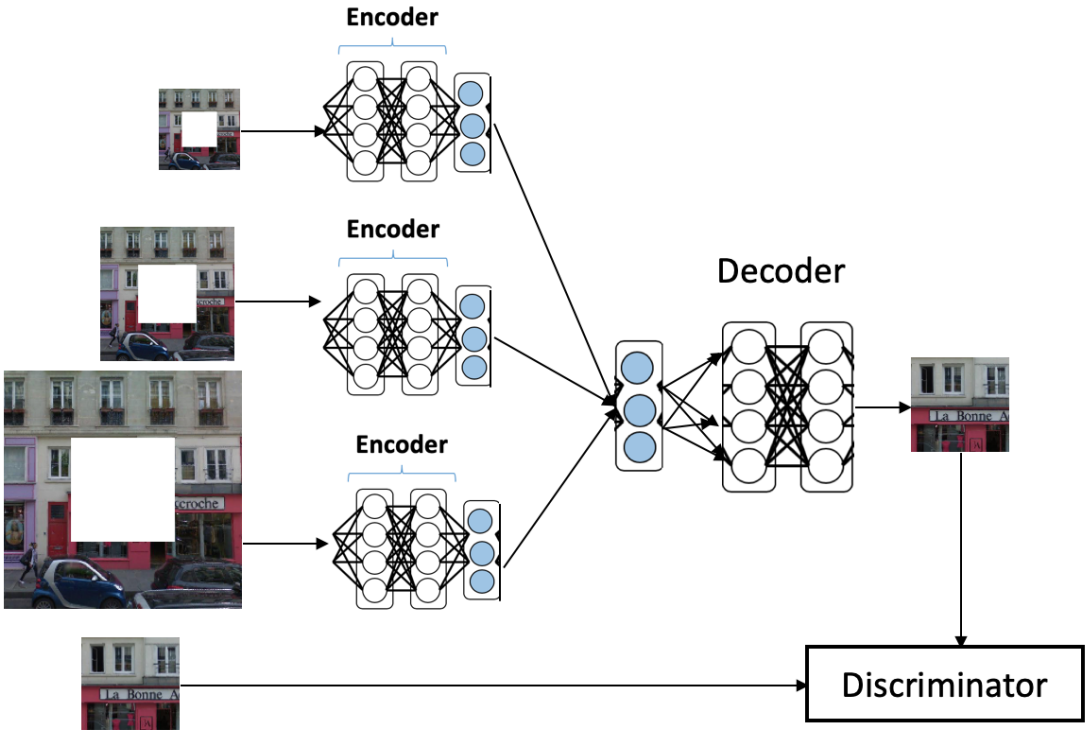

Fig. 1. Model of multi-scale context encoder for feature extraction

A. *Multiscale feature extraction*

In this paper, the multiscale feature extraction consists of many encoders at different scales, namely sub-encoder, decoder, and discriminator. The number of encoders depends on the scale number of the input image. The overall structure is shown in Fig. 1.

Given a single original image, denoted as I_0, multiple images are generated to create multiscale input image. Assume the number of scale to be the value of n, the corresponding multiscale input images are denoted as $\{I_k\}$ with $k = 1..n$. At each scale, the size of image is reduce by the factor of 2. Therefore, as the number of k growing, we obtain images from fine to coarse level.

As shown in the overall flowchart, Fig. 1, this paper uses input image at three different scales ($n = 3$). At each scale level the corresponding image mask is generated for the training process. Remember that, our goal is to reconstruct the missing data at the pixel level of image without any additional information but the image itself. In order to reconstruct the missing data successfully, one need to understand context of whole image and provide a plausible information of the missing part. Moreover, other researchers in [17] [18] suggest that multiscale images provide valuable context information that the image in a single scale does not.

Since we use images at three different scales ($n = 3$), three corresponding encoders need to be used. From this point, we denote three encoders as sub-encoders E_k with $k = 0..2$. The first sub-encoder E_0 corresponds to the original image I_0. The sub-encoder E_0 at first scale take three channels (RGB) of image I_0 with the size of 250×250 as an input followed by 4 convolutional layers to create $16 \times 16 \times 512$ output feature. Note that, in the input image, image mask with the size of 64×64 is automatically generated to represent the missing data in the original image. After each convolutional layer, Leaky RELu [11] [12] layer and batch normalization layer [10] are used to create a nonlinear relation in the network. The number of kernels is increase twice when the pooling layer [14] is applied to compensate the missing information in pooling step. The detailed structure of first sub-encoder is presented in the first four lines of Table 1. Analogy, the other two input images with the size of $125 \times 125 \times 3$ and $63 \times 63 \times 3$ are fed into 3 convolutional layers and 2 convolutional layers, respectively. The details of entire network structure are shown in Table 1.

TABLE I
DETAILS STRUCTURE OF SUB-ENCODERS

Input	Convolutional layer	Output
$250 \times 250 \times 3$	$32, 3 \times 3 \times 3$	$125 \times 125 \times 32$
$125 \times 125 \times 32$	$64, 3 \times 3 \times 32$	$63 \times 63 \times 64$
$63 \times 63 \times 64$	$128, 3 \times 3 \times 64$	$32 \times 32 \times 128$
$32 \times 32 \times 128$	$512, 1 \times 1 \times 128$	$16 \times 16 \times 512$
$125 \times 125 \times 3$	$64, 3 \times 3 \times 3$	$63 \times 63 \times 64$
$63 \times 63 \times 64$	$128, 3 \times 3 \times 64$	$32 \times 32 \times 128$
$32 \times 32 \times 128$	$512, 1 \times 1 \times 128$	$16 \times 16 \times 512$
$63 \times 63 \times 3$	$128, 3 \times 3 \times 64$	$32 \times 32 \times 128$
$32 \times 32 \times 128$	$512, 1 \times 1 \times 128$	$16 \times 16 \times 512$

B. Feature map aggregation

In the deep convolutional network, the feature maps from single original input image encode the fine features. In order to recover high quality missing area, a coarse level feature maps are needed as well. Therefore, our method tries to extract feature map from different scales. As in the preview work, only one decoder is used to decode information from encoder. Note that the outputs of encoders are the feature map with the size of 16×16. Since feature maps are designed to be the same dimension, the concatenation and addition operators can be applied to aggregate feature map for decoder which is used to create missing area image with the size of $64 \times 64 \times 3$ (upsampling). The RELu [11] [12] and batch normalization [10] also are applied after each convolutional layer. The detailed structure of decoder is shown in Table 2.

TABLE II
DETAILS STRUCTURE OF DECODERS

Input	Convolutional layer	Output
$16 \times 16 \times 1536$	Upsampling	$32 \times 32 \times 1536$
$32 \times 32 \times 1536$	$128, 3 \times 3 \times 1536$	$32 \times 32 \times 128$
$32 \times 32 \times 128$	Upsampling	$64 \times 64 \times 128$
$64 \times 64 \times 128$	$64, 3 \times 3 \times 128$	$64 \times 64 \times 64$
$64 \times 64 \times 64$	$3, 3 \times 3 \times 64$	$64 \times 64 \times 3$

III. EXPERIMENTAL RESULTS

In this experiment, we use a PC with two GeForce GTX 1080 graphics cards and 32 GB of RAM running Ubuntu 18.04 LTS with an installed TensorFlow tool with a version of 1.15 [3]. We set the mini-batch size to 32 at a fixed momentum of 0.9. The total number of iterations is set to 300 epochs and the initial value of the learning rate is set to 0.001, which keeps reducing 10% after every 50 epochs. For a fair comparison with other networks, we use the standard data augmentation as [16].

A. Dataset information and training parameters

Face image restoration is the important field in computer vision and image processing as well. Thank to the availability of cameras such as surveillance camera, smartphone camera, one can capture human face easily. However, the chance of obtaining the low quality image or image with a missing data is high. In order to recover the missing data in the facial image, this paper uses CELEBA-HQ and Labeled Faces in the Wild (LFW) dataset sicen they contain lot of facial images. The LFW data set contains 13234 images from 5749 people while the CELEBA-HQ provides 30000 high quality facial images. We use 80% images from total for training and remaining 20% for testing the proposed algorithm. The results of recovered images are compared to the original images based on multiscale structural similarity for image assessment (MS-SSIM) [15]. The details of data sets are shown in the Table III:

TABLE III
DATASETS INFORMATION

Dataset	# training	# testing	# total
CELEBA-HQ	$24,000$	$6,000$	$30,000$
LFW	$9,788$	$2,446$	$12,234$

B. Effect of feature aggregation

The proposed method is evaluated differently in term of aggregating operation. As shown in Table I, in order to combine feature map from all sub-encoder and feed them to the decoder, we need to use a aggregating operation. Since the all sub-encoder has the same size of $16 \times 16 \times 512$, we can apply addition or concatenation on them. In case of applying the additional operation, the number of parameters keeps the same $(16 \times 16 \times 512)$ while applying the concatenation operation makes the number of parameter dramatically rising $(16 \times 16 \times 1.536)$.

C. Effect of multiscale structure

In this subsection, we evaluate the effect of multiscale structure using the additional operation to combine feature maps of all sub-encoders. Compared to concatenation operation, the additional operation has similar performance but much faster both in training and testing phases . The original algorithm (Context Encoder) is compared the proposed method, Multiscale Context Encoder (MCE) in term of L1 loss and MS-SSIM [15]. The results on Table IV show on the CELEBA-HQ dataset that L1 loss from context encoder (2.27) is lower than MCE (2.43) but the overall similarity structure of MCE (75.43%) is higher than the one produced by context encoder (73.40%). The higher value of MS-SSIM the more realistic image is generated while the L1 loss taking the differences between two images does not reflect the realistic level of image. The performance of the proposed method is again stated in Table V when the evaluation is processed on LFW dataset. The value of MS-SSIM produced from MCE (80.35%) is much higher than the one from context encoder (77.18%). The reason here probaly the lower resolution of LFW compared to the CELEBA-HQ dataset. Figure 2 shows some results of original algorithm and the proposed method on the CELEBA-HQ dataset.

TABLE IV
COMPARISON RESULTS ON CELEBA-HQ DATASET OF CONTEXT ENCODER AND MULTISCALE CONTEXT ENCODER (MCE).

Method	L1 loss	MS-SSIM
Context encoder	2.27	73.40%
MCE	2.43	75.43%

IV. CONCLUSIONS AND FUTURE WORK

Missing area in an image can be restored by using combination of autoencoder and generative adversarical network, Context Encoder. Instead of encoding feature at a single scale, this paper tried to encode feature at multi-scale for better capturing the context information. The features after encoding

TABLE V
COMPARISON RESULTS ON LFW DATASET OF CONTEXT ENCODER AND MULTISCALE CONTEXT ENCODER (MCE).

Method	L1 loss	MS-SSIM
Context encoder	4.65	77.18%
MCE	4.78	80.35%

are aggregated and fed to the decoder to create the needed images. A comparison with the original context encoder, the proposed method achieve higher accuracy in term of MS-SSIM. However, due to multi-scale input image, the parameter number of whole network is higher than the original context encoder too. As a future work, we will attempt to apply the separable convolution reducing the number of parameters while keeping the performance.

ACKNOWLEDGMENT

The authors would like to thank the Ministry of Science and Technology of Vietnam for their support provided through the project: A smart surveillance system based on facial recognition for border and immigration controls (code HNQT/SPDP 14.19) This research is funded by project: A smart surveillance system based on facial recognition for border and immigration controls (code HNQT/SPDP/14.19)

REFERENCES

[1] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., et al.: "Caffe: Convolutional architecture for fast feature embedding". Proc. Int. Conf. Proceedings of the 22nd ACM international conference on Multimedia, pp. 675-678, 2014.

[2] Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., et al.: "Theano: A CPU and GPU math compiler in Python". Proc. Int. Conf. Proc. 9th Python in Science Conf, pp., 2010.

[3] Girija, S.S.: "Tensorflow: Large-scale machine learning on heterogeneous distributed systems", arXiv preprint arXiv:1603.044672016.

[4] Yu, D., Eversole, A., Seltzer, M., Yao, K., Huang, Z., et al.: "An introduction to computational networks and the computational network toolkit", Microsoft Technical Report MSR-TR-2014–1122014.

[5] Krizhevsky, A., Sutskever, I., and Hinton, G.E.: "Imagenet classification with deep convolutional neural networks". Proc. Int. Conf. Advances in neural information processing systems, Lake Tahoe, Nevada, USA, pp. 1097-1105, Dec. 2012.

[6] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, Alexei A Efros, "Context encoders: Feature learning by inpainting", Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2536-2544, 2016.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, "Generative adversarial nets", Advances in neural information processing systems, pp. 2672-2680, 2014.

[8] Juergen Schmidhuber, "Deep Learning in Neural Networks: An Overview", arXiv preprint arXiv:1404.7828.

[9] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, Brendan Frey, "Adversarial autoencoders", arXiv preprint arXiv:1511.05644.

[10] Ioffe, S., and Szegedy, C.: "Batch normalization: Accelerating deep network training by reducing internal covariate shift", arXiv preprint arXiv:1502.031672015.

[11] Glorot, X., and Bengio, Y.: "Understanding the difficulty of training deep feedforward neural networks". Proc. Int. Conf. Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp. 249-256, 2010.

[12] Nair, V., and Hinton, G.E.: "Rectified linear units improve restricted boltzmann machines". Proc. Int. Conf. International Conference on Machine Learning, Haifa, Israel, pp. 807-814, Jun. 2010.

[13] Li, J., Cheng, J.-h., Shi, J.-y., and Huang, F.: "Brief introduction of back propagation (BP) neural network algorithm and its improvement", in Advances in Computer Science and Information Engineering (Springer, 2012), pp. 553-558.

[14] Nagi, J., Ducatelle, F., Di Caro, G.A., Cireşan, D., Meier, U., et al.: "Max-pooling convolutional neural networks for vision-based hand gesture recognition". Proc. Int. Conf. IEEE International Conference on Signal and Image Processing Applications (ICSIPA), pp. 342-347, 2011.

[15] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In IEEE ACSSC, vol. 2, pp. 1398–1402, 2003.

[16] Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y.: "Maxout networks", arXiv preprint arXiv:1302.43892013.

[17] Krizhevsky, A., and Hinton, G.: "Learning multiple layers of features from tiny images". Master's thesis, University of Toronto, 2009.

[18] Trung Dung Do, Cheng-Bin Jin, Van Huan Nguyen, Hakil Kim, "Mixture separability loss in a deep convolutional network for image classification", IET Image Processing, vol. 13, no. 1, pp. 135-141, 2019.

Fig. 2. Facial image restoration on CELEBA dataset of context encoder and multiscale context encoder. The fist column - image with the mask , the second column - result from context encoder, the last column - result from MCE

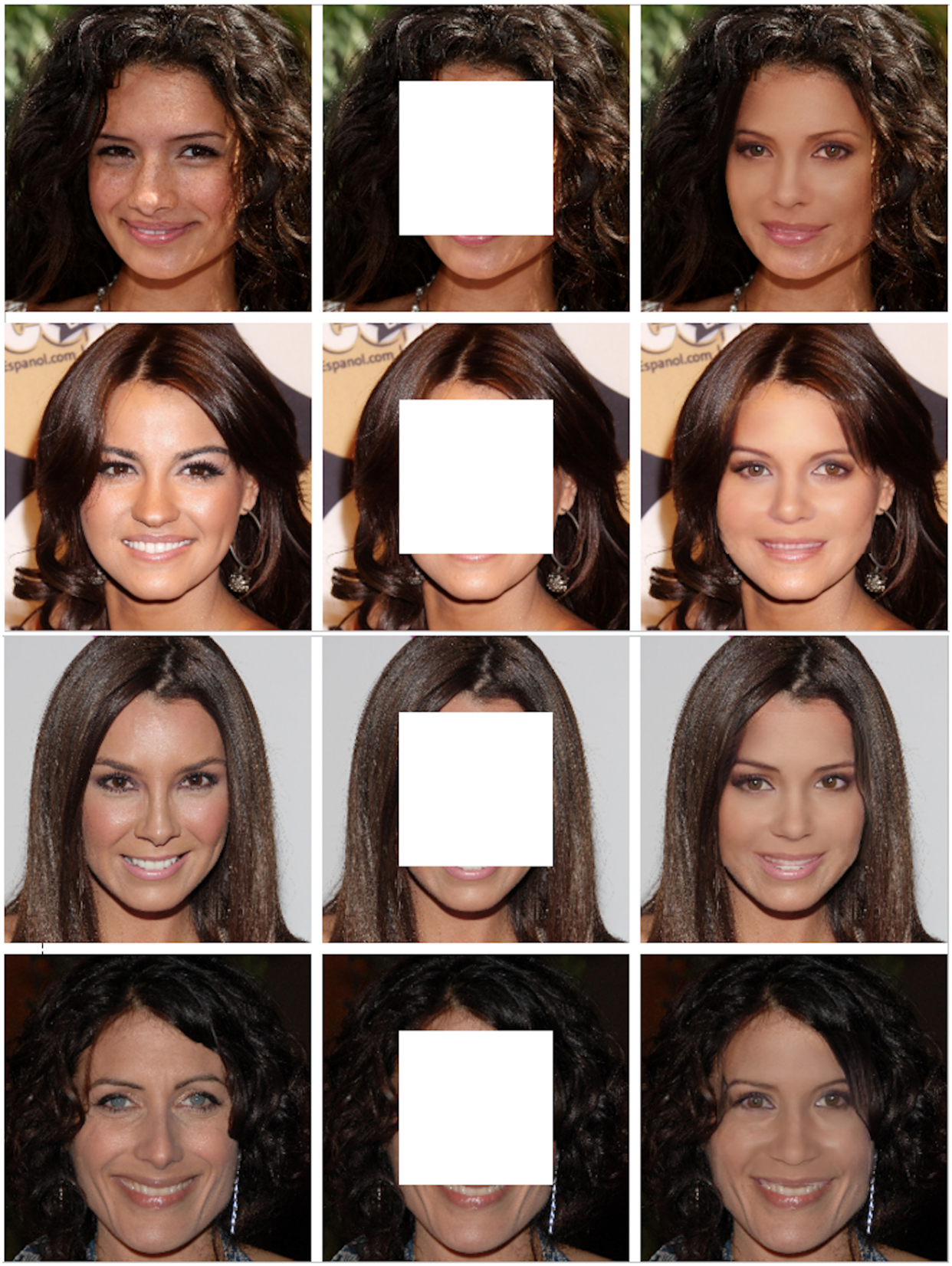

Fig. 3. More results of the proposed method on CELEBA-HQ dataset. The fist column - original image, the second column - image with the mask, the last column - result from MCE