Semantic inpainting with deep image classifier

Martin Buchacek **Amaris Chen** **Dejan Malesevic** **Rony Speck**

Abstract

We present a novel algorithm for completing large missing regions of corrupted car images. The missing content is generated by a generative adversarial network trained on a dataset. Our algorithm uses the content of hidden layers of an image classifier trained on an identical dataset, and aims to generate a new image completing the missing part, resulting in a similar layer activation as the corrupted image. Our method works particularly well on images with a large missing region. It visually outperforms the existing conventional methods as well as other deep learning methods based on generative adversarial networks.

1 Introduction

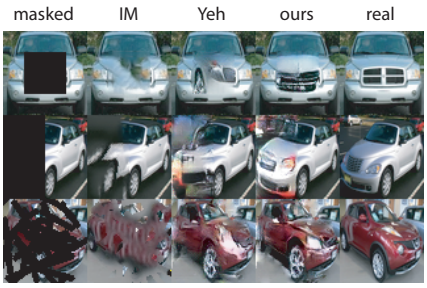
Inpainting refers to filling missing (masked) parts of a corrupted image and has numerous practical applications, such as restoring damaged paintings or removing unwanted objects from pictures. While it is often easy for humans to imagine the missing content, this task is challenging for computers since it requires the understanding of local patterns and the overall semantic meaning of the picture. Here we propose an innovative inpainting method based on generative adversarial networks [1, 2] that uses the hidden layers of an image classifier to learn specific features of cars. We aim to improve the inpainting of large connected masks on images of cars.

The conventional inpainting methods [3–6] usually do not fill large missing regions successfully. Figure 1a shows some failure examples for the ImageMelding [6], an improved method based on PatchMatch [5]. While it succeeds in producing smooth images, it clearly fails to complete the pictures in a meaningful way. It uses the patterns present in the car images in the wrong places, such as wheels on windows, or fills the missing region almost uniformly.

On the other hand, deep neural networks can learn to recognize and re-use features from unmasked parts of the picture. This can be accomplished by training the inpainting algorithm on a specific dataset with fixed mask shapes [7]. Some of the methods that have proved to be successful are training on rich image datasets and variable mask shapes (see e.g. [8–10] and the state-of-the art methods [11, 12]), and extracting features from a single image [13]. These methods are able to borrow patterns and features present in the images, and reproduce them in the missing part in a realistic way, thus completing various small and irregular masks.

Here we focus on the cases when a substantial part of the picture is missing and the present content does not provide enough information to complete the picture as seen in Fig 1a where the center of half of a car is missing. In this case, the focus of the existing algorithms is often not to reconstruct the ground truth, but to produce a visually plausible output.

We elaborate on the method by Yeh et. al. [2] by making use of generative adversarial networks (GANs). After a GAN is trained on a specific dataset, the pictures produced by the generator are used for filling the missing part. To enforce a meaningful image completion, Yeh defines the prior loss to penalize unrealistically-looking generated images and the content loss to penalize the differences between the generated image and the corrupted image near the mask boundary. The joint minimization of the two losses then enables the model to find the closest encoding of the corrupted image in the latent space. This is then used for image reconstruction with the generator.



(a) Examples of inpainting comparing the image melding [6] (IM), Yeh's method [2], our method and the ground truth.

(b) Failure examples of our method. Left: masked image, middle: our completition, right: ground truth.

Figure 1: Examples of inpainting

We notice, however, that these constraints may not be sufficient for a meaningful completion of a large missing region. Figure 1a shows some failure examples where the generator produces realistic wheels or window panes yet places them in incorrect positions. To address this issue we train the GoogLeNet [14] image classifier on Stanford's Cars Dataset [15] and construct a new *semantic* loss function that forces the generated and corrupted images to produce similar activation in the classifier's hidden layers. This approach assumes that the hidden layers carry information about features specific for the dataset [16]. We show that this method leads to more realistically-looking output than Yeh's method.

2 Models and Methods

2.1 Inpainting model

Our inpainting method uses a deep convolutional generative adversarial network (DCGAN) [17] and a deep image classifier (we use the GoogLeNet [14] architecture). Both of them are trained on the uncorrupted car images [15]. Drawing a vector $\mathbf{z}$ from the latent space, the generator of the DCGAN produces images $G(\mathbf{z})$ mimicking the training images. The aim of the inpainting algorithm is to find the best encoding $\hat{\mathbf{z}}$ such that the generated image will meaningfully complete the corrupted image. We formulate this as the following minimization problem,

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z}} \{\mathcal{L}_c(\mathbf{z}|\mathbf{y},\mathbf{M}) + \mathcal{L}_p(\mathbf{z}) + \mathcal{L}_{\text{inp}}(\mathbf{z}|\mathbf{y},\mathbf{M}) + \mathcal{L}_{\text{sem}}(\mathbf{z}|\mathbf{y},\mathbf{M}) + \mathcal{L}_{\text{cat}}(\mathbf{z}|\mathbf{y},\mathbf{M})\}. \quad (1)$$

where $\mathbf{y}$ is the corrupted image and $\mathbf{M}$ is the mask (here $\mathbf{M}_{ij} = 0$ for the occluded regions and $\mathbf{M}_{ij} = 1$ otherwise). $\mathcal{L}_c$, $\mathcal{L}_p$, $\mathcal{L}_{\text{inp}}$, $\mathcal{L}_{\text{sem}}, \mathcal{L}_{\text{cat}}$ denote the content, prior, inpainting, semantic and categorical loss, respectively. The content and prior loss are identical to the Yeh's architecture [2]: the content loss constraints the generated image to resemble the corrupted image near the mask boundary, $\mathcal{L}_c(\mathbf{z}|\mathbf{y},\mathbf{M}) = ||\mathbf{W} \odot (G(\mathbf{z}) - \mathbf{y})||^2$ with $\mathbf{W}$, the weighting term, assigning more importance to the pixels near the holes. The prior loss penalizes unrealistically-looking generated images and is defined through the training loss of the discriminator, $\mathcal{L}_p(\mathbf{z}) = \lambda_p \log[1 - D(G(z))]$. Here λ_p is a tunable parameter balancing the importance of losses.

With $\mathbf{y}' = \mathbf{y} + (1-\mathbf{M}) \odot G(\mathbf{z})$ representing the inpainted image, we define additional loss functions. We use the inpainted rather than masked images, assuming they sufficiently resemble samples drawn from the real image distribution. This would not be the case for corrupted images with one or several dark regions. Motivated by this assumption, we define the *inpainting* loss, $\mathcal{L}_{\text{inp}} = \lambda_{\text{inp}} \log(1 - D(\mathbf{y}'))$, to penalize unrealistically-looking completions.

The remaining two losses are based on image representation in the high-level feature space [16]. The mapping to the feature space is obtained by feeding the inpainted and generated images into the trained classifier. When convolutional neural networks are trained on image classification, they develop a feature representation of the images in their hidden layers. Such representation reflects the image content rather than its pixel representation. We define the *semantic* loss function through the output I_ℓ of the ℓ^{th} GoogLeNet inception module. For a module with N_ℓ distinct filters, each of

size $M_\ell \times M_\ell$, I_ℓ is a tensor of shape $M_\ell \times M_\ell \times N_\ell$ and we define

$$\mathcal{L}_{\text{sem}}(\mathbf{z}|\mathbf{y}, \mathbf{M}) = \lambda_{\text{sem}} \sum_{\ell} w_\ell ||I_\ell(G(\mathbf{z})) - I_\ell(\mathbf{y}')||_2^2 / ||I_\ell(\mathbf{y}')||_2^2 \tag{2}$$

with w_ℓ representing relative weights of the modules and λ_{sem} a tunable parameter.

To enforce the inpainted image to lie in the same class of the cars dataset as the real one, we further define the *categorical* loss function through the output $\mathbf{p}$ of the softmax layer of the classifier ($\mathbf{p}$ is a N-dimensional vector representing probabilities corresponding to the N distinct classes),

$$\mathcal{L}(\mathbf{z}|\mathbf{y}, \mathbf{M}) = \lambda_{\text{cat}} \{KL[\mathbf{p}(\mathbf{G}(\mathbf{z}))||\mathbf{p}(\mathbf{y}')] + KL[\mathbf{p}(\mathbf{y}')||\mathbf{p}(\mathbf{G}(\mathbf{z}))]\} \tag{3}$$

with $KL(\mathbf{p}, \mathbf{q})$ the Kullback-Leibler divergence between the probability distribution $\mathbf{p}$ and $\mathbf{q}$.

After the best solution of (1) is found, the Poisson blending [18] is used to fine-tune the colours of the generated image $G(\mathbf{z})$ to match the colours in the known part of the corrupted image $\mathbf{y}$. This is defined as the minimization problem for the final inpainted picture $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} ||\nabla \mathbf{x} - \nabla G(\hat{\mathbf{z}})||_2^2$.

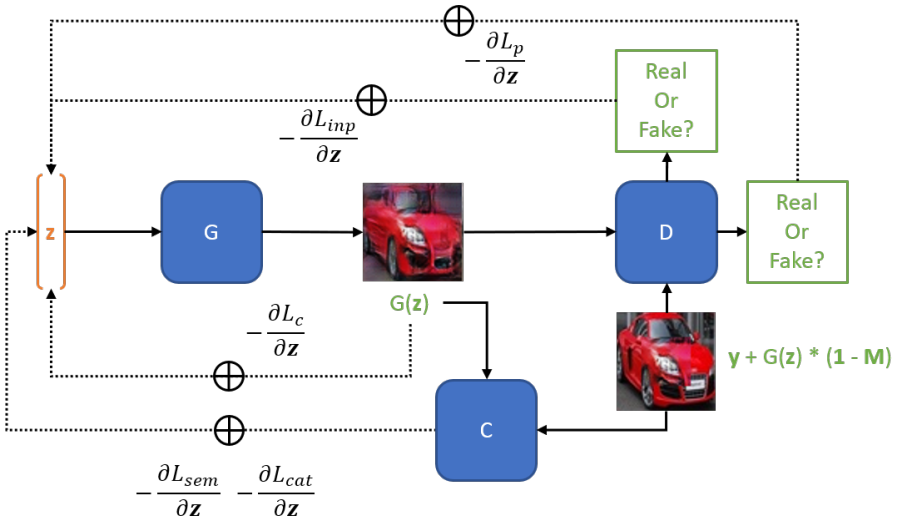


Figure 2: Overview of our inpainting model. It shows the generator G, the discriminator D, the classifier C and all the loss functions used for computing the optimal $\hat{\mathbf{z}}$. Note that $\mathbf{y}+G(\mathbf{z})\odot(\mathbf{1}-\mathbf{M})$ is the result of combining the original image $\mathbf{y}$ with the generated image $G(\mathbf{z})$, where $\mathbf{M}$ is the mask defining the missing part of the original image.

2.2 Generative adversarial network

Our GAN implementation is taken from ChengBinJin's [19] git repository. It implements DCGAN [17] and the inpainting method as described in Yeh [2]. The DCGAN model is trained with images taken from Stanford's Cars Dataset [15]. The dataset contains 8144 training images, 8041 testing images, and 196 different classes. For our training purposes, the images are cropped according to the provided bounding boxes and then resized to 64×64 pixels. We augmented the images using the horizontal flipping method, and thus our training set for DCGAN has a total of 16 288 images. The model is trained for 13 000 iterations using the batch size of 64, learning rate of 0.0002, and decay rate of 0.5 with Adam optimizer.

2.3 Classifier

Our classifier is created using GoogLeNet architecture and pre-trained weights from the ImageNet model. The implementation is taken from Qian Ge's git repository [20]. Our training result shows that both training the entire GoogLeNet model, and training only the fully connected layers (freeze all convolutional layers) do not yield satisfactory results. The validation accuracy for both types of methods are below 0.60. In order to train our model specifically for the classification of cars, we use

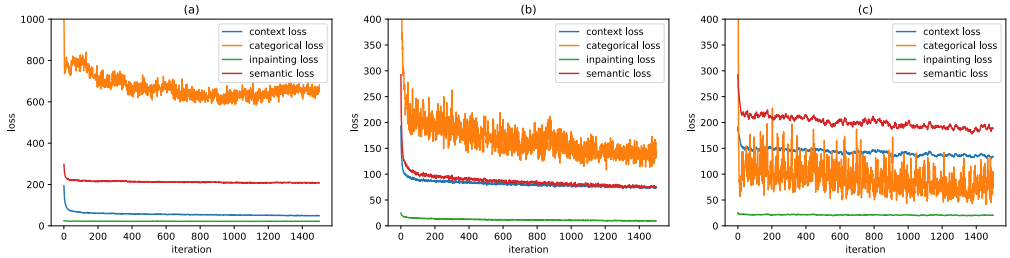


Figure 3: Evolution of the loss function averaged over 100 samples for different training strategies. In Yeh's implementation (a), only the context and prior loss are minimized and the categorical and semantic loss remain high. In our implementation, we further minimize the inpainting and semantic loss (b) or all terms in the loss function (1) including the categorical loss (c). The prior loss remains low during the optimization and is not shown in the figures.

the weights from the pre-trained ImageNet model, and train our classifier according to the following changes and hyperparameters:

1. The inception modules 4a, 4b, and 4c are using the pre-trained weights (freeze modules) while the rest are trainable;
2. The first two max pooling layers are removed;
3. Two auxiliary classifiers are used, each with 512 hidden units;
4. Dropout with keep probability of 0.7 is applied to two fully connected layers;
5. The network is trained through Adam optimizer. Batch size is 32. The initial learning rate is 1e-3, decays to 1e-4 after 35 epochs, and finally decays to 1e-5 after 50 epochs.

Using the same car dataset as described in the previous section, our model takes roughly 2 hours to train on NVIDIA GeForce GTX 1080 gpu. Our model has a training accuracy of 0.998 and validation accuracy of 0.815. Our accuracy matches the results obtained by Liu [21].

3 Experiments and results

Experimental implementation Due to time and computational power constraints, we ran the experiments on the first 100 pictures from the testing set of Stanford cars dataset. We fixed the parameter $\lambda_p = 0.003$ based on reference [2] and subsequently tuned the remaining parameters such that all the loss terms in Eq. (1) have similar orders of magnitude. We used $\lambda_{\text{inp}} = 3$, $\lambda_{\text{sem}} = 10^5$ and $\lambda_{\text{cat}} = 10^6$ for inpainting center and left masks. Images with random masked regions have more pixels near the holes which resulted in higher context loss. Hence, we used higher values $\lambda_{\text{inp}} = 30$, $\lambda_{\text{sem}} = 10^6$ and $\lambda_{\text{cat}} = 10^7$ for inpainting random masks. Experimenting with different layers defining the semantic loss (2), we found the most visually plausible results with using $w_\ell = 1$ for the GoogLeNet inception-3b and inception-4b layers and $w_\ell = 0$ otherwise.

The loss function is minimized using the Nesterov accelerated gradient [22] with learning rate 0.01 and momentum 0.9. The vector $\mathbf{z}$ is restricted to the interval $[-1, 1]$ after each iteration [2]. Figure 3 shows the dependence of the sample-averaged value of various losses on the number of iterations. If only the context and prior loss are optimized as seen in Yeh's method, the newly-introduced semantic and categorical loss remain high. In our method, the simultaneous optimization of the inpainting and semantic loss leads to the drop of the latter. Additionally, categorical loss decreases even without direct optimization. Optimizing all terms in Eq. (1) pushes categorical loss to even lower values, and this leads to somewhat higher saturation of the semantic loss. Our inpainting results look in many cases more plausible than those by Yeh. Our observation suggests that this is directly related to the optimization of semantic and categorical loss.

Qualitative Results Figure 1a shows some cases in which our method outperforms the method of Yeh [2] and Image Melding [6]. Image Melding, which is based on PatchMatch, semantically infills incorrect content in the occluded parts. This can be seen in Fig 1a where it inpaints an additional

	IM [6]	Yeh [2]	ours w/o logits	ours with logits
PSNR - center	14.21	12.58	12.76	12.87
SSIM - center	0.81	0.79	0.80	0.79
PSNR - left	8.09	6.85	6.42	6.73
SSIM - left	0.56	0.55	0.55	0.55
PSNR - random	8.41	11.69	11.41	11.21
SSIM - random	0.54	0.65	0.63	0.60

Table 1: Quantitative analysis of the performance of algorithms: image melding (IM), Yeh's method and our implementation. With and w/o logits refers to whether the categorical loss has been optimized or not.

wheel instead of the front of a car. In addition, the method of Yeh et al. [2] fails in some cases where our method completes the image much more realistically. Nevertheless, our method also fails in some cases as seen in Fig. 1b. More results can be found in the appendix section.

Quantitative Results A corrupted image can be inpainted in many different ways, there exist various numerical methods for measuring the quality of inpainting results. In the past, classical inpainting methods have used PSNR and SSIM [23] to measure the quality of their results. Hence, we compare PSNR of our method to that of Image Melding [6] and Yeh et. al. [2]. Table 1 shows a comparison of these values. The provided values do not correlate with our increased visual appealing results as seen in the appendix, Figure 4. From the classical inpainting methods, higher PSNR ratio and SSIM values have shown to correspond to better visual quality. The PSNR value of our method is lower in most cases yet we obtained better visual quality.

4 Discussion

In certain cases, our method does not manage to fill the missing part in a satisfactory way. It either produces sharp colour transitions near the mask boundaries or fills in the incorrect content, see Fig. 1b. The first case may be caused by insufficient minimization of the content loss, such that the filled content does not smoothly follow the rest of the picture even after post-processing. The second case is possibly related to the limited ability of the generator to produce well-fitting completions, especially when inpainting the types of cars rarely presented in the training dataset. To improve our method further, it would be interesting to extend our method by implementing other recent and more advanced generator architectures, e.g. [24].

Our successful cases, nevertheless, show that the hidden and logits layers of the classifier provide valuable information about the structural and semantic contents of the image, and the minimization of the corresponding semantic and categorical losses yields better inpainting results. Our approach of including classifier hidden layer information is not restricted to classification tasks as seen in our work. A future work is to inpaint images using features stored in hidden layers of the autoencoder.

5 Conclusion

We presented an inpainting method based on generative adversarial networks and extending the Yeh's architecture [2] to take into account the contents of the hidden layers of an image classifier. It outperforms the classical approaches, and in many cases gives more convincing results than Yeh. It is particularly successful for images with a large region missing (e.g. the left mask). Our method shows a very satisfactory result for using a novel loss function for inpainting with GAN.

Appendix: inpainting examples

Figure 4: Randomly selected examples of inpainting comparing the conventional method (IM [6]), Yeh's method [2] and our method without or with optimizing the categorical loss (see columns w/o and w logits). The ground truth image is shown on the right.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 2672–2680, Curran Associates, Inc., 2014.

[2] R. A. Yeh, C. Chen, T. Lim, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with perceptual and contextual losses," *CoRR*, vol. abs/1607.07539, 2016.

[3] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Transactions on Image Processing*, vol. 20, pp. 681–695, March 2011.

[4] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He, "Fast and accurate matrix completion via truncated nuclear norm regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2117–2130, 2013.

[5] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, Aug. 2009.

[6] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen, "Image melding: Combining inconsistent images using patch-based synthesis.," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 82–1, 2012.

[7] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros, "Context encoders: Feature learning by inpainting," 2016.

[8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arxiv*, 2016.

[9] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li, "High-resolution image inpainting using multi-scale neural patch synthesis," *CoRR*, vol. abs/1611.09969, 2016.

[10] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," *CoRR*, vol. abs/1801.07892, 2018.

[11] G. Liu, F. A. Reda, K. J. Shih, T. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," *CoRR*, vol. abs/1804.07723, 2018.

[12] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," *arXiv preprint arXiv:1806.03589*, 2018.

[13] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Deep image prior," *CoRR*, vol. abs/1711.10925, 2017.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.

[15] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, (Sydney, Australia), 2013.

[16] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *CoRR*, vol. abs/1508.06576, 2015.

[17] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2015.

[18] P. Prez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Transactions on Graphics (SIGGRAPH'03)*, vol. 22, no. 3, pp. 313–318, 2003.

[19] C.-B. Jin, "semantic-image-inpainting." `https://github.com/ChengBinJin/semantic-image-inpainting`, 2018. commit e4e1a54.

[20] Q. Ge, "Googlenet for image classification." `https://github.com/conan7882/GoogLeNet-Inception`, 2018.

[21] D. Liu, "Monza : Image classification of vehicle make and model using convolutional neural networks and transfer learning,"

[22] Y. Nesterov, "A method for solving a convex programming problem with convergence rate o(1/k2)," *Soviet Mathematics Doklady*, vol. 27, pp. 372–367, 1983.

[23] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[24] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks,"