Automatic Semantic Content Removal by Learning to Neglect

Siyang Qin
siqin@soe.ucsc.edu
Jiahui Wei
jwei19@ucsc.edu
Roberto Manduchi
manduchi@soe.ucsc.edu

University of California, Santa Cruz
Santa Cruz, CA, USA

Abstract

We introduce a new system for automatic image content removal and inpainting. Unlike traditional inpainting algorithms, which require advance knowledge of the region to be filled in, our system automatically detects the area to be removed and infilled. Region segmentation and inpainting are performed jointly in a single pass. In this way, potential segmentation errors are more naturally alleviated by the inpainting module. The system is implemented as an encoder-decoder architecture, with two decoder branches, one tasked with segmentation of the foreground region, the other with inpainting. The encoder and the two decoder branches are linked via *neglect nodes*, which guide the inpainting process in selecting which areas need reconstruction. The whole model is trained using a conditional GAN strategy. Comparative experiments show that our algorithm outperforms state-of-the-art inpainting techniques (which, unlike our system, do not segment the input image and thus must be aided by an external segmentation module.)

1 Introduction

Automatic removal of specific content in an image is a task of practical interest, as well as of intellectual appeal. There are many situations in which a part of an image needs to be erased and replaced. This may include text (whether present in the scene or overimposed on the image), which may have to be removed, for example to protect personal information; people, such as undesired passersby in the scene; or other objects that, for any reasons, one may want to wipe off the picture. While these operations are typically performed manually by skilled Photoshop editors, substantial cost reduction could be achieved by automatizing the workflow. Automatic content removal, though, is difficult, and an as yet unsolved problem. Removing content and inpainting the image in a way that it looks "natural" entails the ability to capture, represent, and synthesize high-level ("semantic") image content. This is particularly true of large image areas infilling, an operation that only recently has been accomplished with some success [14, 27, 29, 30].

Image inpainting algorithms described in the literature normally require that a binary "mask" indicating the location of the area to be synthesized be provided, typically via manual input. In contrast, an automatic content removal system must be able to accomplish two

tasks. First, the pattern or object of interest must be segmented out, creating a binary mask; then, the image content within the mask must be synthesized. The work described in this paper is born from the realization that, for optimal results, these two tasks (segmentation and inpainting) should *not* be carried out independently. In fact, only in few specific situations can the portion of the image to be removed be represented by a binary mask. Edge smoothing effects are almost always present, either due to the camera's point spreading function, or due to blending, if the pattern (*e.g.* text) is overimposed on the image. Although one could potentially recover an alpha mask for the foreground content to be removed, we believe that a more appropriate strategy is to *simultaneously* detect the foreground *and* synthesize the background image. By doing so, we do not need to resort to hand-made tricks, such as expanding the binary mask to account for inaccurate localization.

Our algorithm for content removal and inpainting relies on conditional generative adversarial networks (cGANs) [8], which have become the tool of choice for image synthesis. Our network architecture is based on an encoder-decoder scheme with skip layers (see Fig. 1). Rather than a single decoder branch, however, our network contains two parallel and interconnected branches: one (*dec-seg*) designed to extract the foreground area to be removed; the other (*dec-fill*) tasked with synthesizing the missing background. The *dec-seg* branch interacts with the *dec-fill* branch via multiple *neglect nodes* (see Fig. 2). The concept of neglect nodes is germane (but in reverse) to that of mixing nodes normally found in attention networks [22, 26]. Mixing nodes highlight a portion of the image that needs to be attended to, or, in our case, neglected. Neglect nodes appear in all layers of the architecture; they ensure that the *dec-fill* branch is aware of which portions of the image are to be synthesized, without ever committing to a binary mask.

A remarkable feature of the proposed system is that the multiple components of the network (encoder and two decoder branches, along with the neglect nodes) are all trained at the same time. Training seeks to minimize a global cost function that includes a conditional GAN component, as well as L_1 distance components for both foreground segmentation and background image. This optimization requires ground–truth availability of *foreground* segmentation (the component to be removed), *background* images (the original image without the foreground), and *composite* images (foreground over background). By jointly optimizing the multiple network components (rather than, say, optimizing for the *dec-seg* independently on foreground segmentation, then using it to condition optimization of *dec-fill* via the neglect nodes), we are able to accurately reconstruct the background inpainted image. The algorithm also produces the foreground segmentation as a byproduct. We should emphasize that this foreground mask is *not* used by the *dec-fill* synthesizing layer, which only communicates with the *dec-seg* layer via the neglect nodes.

To summarize, this paper has two main contributions. First, we present the first (to the best of our knowledge) truly automatic semantic content removal system with promising results on realistic images. The proposed algorithm is able to recover high-quality background without any knowledge of the foreground segmentation mask. Unlike most previous GAN–based inpainting methods that assume a rectangular foreground region to be removed [14, 27, 29], our system produces good result with any foreground shape, even when it extends to the image boundary. Second, we introduce a novel encoder-decoder network structure with two parallel and interconnected branches (*dec-seg* and *dec-fill*), linked at multiple levels by mixing (neglect) nodes that determine which information from the encoder should be used for synthesis, and which should be neglected. Foreground region segmentation and background inpainting is produced in one single forward pass.

2 Related Work

Semantic image content removal comprises two different tasks: segmentation of the foreground region (which in some contexts represents a "corrupted" image region to be removed), and synthesis of the missing background after foreground removal. We briefly review the literature for these two operations in the following.

Pixel-level semantic image segmentation has received considerable attention over the past few years. Most recently published techniques are based on fully convolutional networks (FCN) [12], possibly combined with fully connected CRF [3, 9, 17, 34]. The general architecture of a FCN includes a sequence of convolution and downsampling layers (*encoder*), which extract multi–scale features, followed by a sequence of deconvolution layers (*decoder*), which produce a high–resolution segmentation (or "prediction"). Skip layers are often added, providing shortcut links from an encoder layer to its corresponding decoder layer. The role of skip layers is to provide well-localized information to a decoder layer, in addition to the semantic-rich but poorly resolved information from the prior decoder layer. In this way, skip layers enable good pixel-level localization while facilitating gradient flow during training. Similar architectures have been used in various applications such as text segmentation [15, 17, 28, 32], edge detection [25], face segmentation [16], and scene parsing [33]. Although these algorithms could be used for the foreground segmentation component of a content removal system, unavoidable inaccuracies are liable to dramatically decrease the quality of the recovered background region.

Image inpainting [2] has a long history. The goal of inpainting is to fill in a missing region with realistic image content. A variety of inpainting methods have been proposed, including those based on prior image statistics [20, 35], and those based on CNNs [19, 31]. More recently, outstanding results have been demonstrated with the use of GANs to inpaint even large missing areas [14, 27, 29, 30]. While appropriate for certain domains (*e.g.* face inpainting), methods in this category often suffer from serious limitations, including the requirement that the missing region have fixed size and shape.

All of the inpainting methods mentioned above assume that the corrupted region mask is known (typically as provided by the user). This limits their scope of application, as in most cases, this mask is unavailable. This shortcoming is addressed by blind inpainting algorithms [11, 24], which do not need access to the foreground mask. However, prior blind inpainting work has been demonstrated only for very simple cases, with constant-valued foreground occupying a small area of the image.

3 Proposed Algorithm

Our system for automatic semantic content removal comprises an encoder-decoder network with two decoder branches, tasked with predicting a segmentation mask (*dec-seg*) and a background image (*dec-fill*) in a single forward pass. Neglect nodes (an original feature of this architecture) link the two decoder branches and the encoder at various levels. The network is trained along with a discriminator network in an adversarial scheme, in order to foster realistic background image synthesis.

We assume in this work that the foreground region to be removed occupies a large portion of the image (or, equivalently, that the image is cropped such that the foreground region takes most of the cropped region). In practice, this can be obtained using a standard object detector. Note that high accuracy of the (rectangular) detector is not required. In our experiments, the

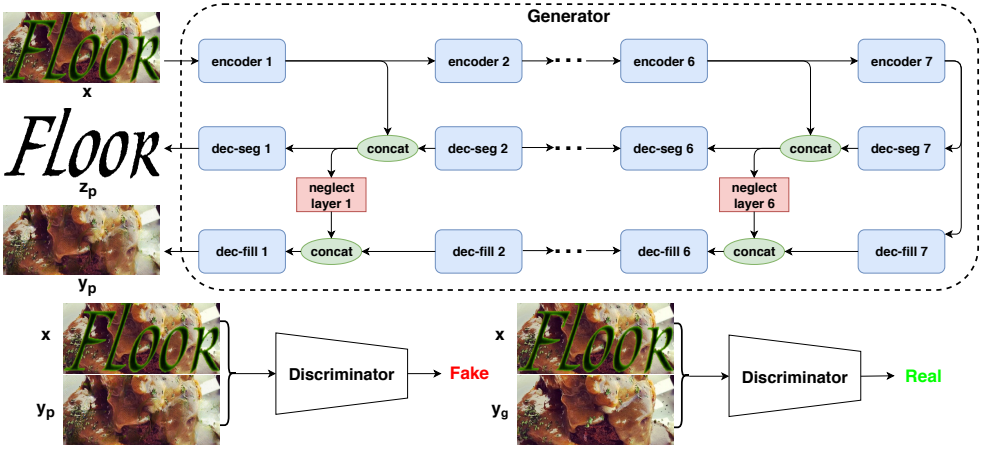

Figure 1: System architecture.

margin between the contour of the foreground region and the edges of the image was let to vary between 0 (foreground touching the image edge) to half the size of the foreground mask.

3.1 Loss Function

Following the terminology of GANs, the output z_p of *dec-seg* and y_p of *dec-fill* for an input image x are taken to represent the output of a generator $G(x)$. The generator is trained with a dual task: ensuring that z_p and y_p are similar to the ground–truth (z_g and y_g), and deceiving the discriminator $D(x,y)$, which is concurrently trained to separate y_p from y_g given x. The cost function L_G for the generator combines the conditional GAN loss with a linear combination of the L_1 distances between prediction and ground–truth for segmentation and inpainted background:

$$L_G = \mathbb{E}[\log(1 - D(x, y_p))] + \lambda_f \mathbb{E}[||y_g - y_p||_1] + \lambda_s \mathbb{E}[||z_g - z_p||_1] \tag{1}$$

The discriminator D is trained to minimize the following discriminator loss L_D:

$$L_D = -(\mathbb{E}[\log D(x, y_g)] + \mathbb{E}[\log(1 - D(x, y_p))]) \tag{2}$$

3.2 Network Architecture

Generator: Segmentation and background infilling is generated in a single pass by an encoder–decoder network architecture, with multiple encoder layers generating multi-scale features at decreasing resolution, and two parallel decoder branches (*dec-seg* and *dec-fill*) producing high–resolution output starting from low–resolution features and higher–resolution data from skip layers. Each encoder stage consists of a convolution layer with kernel of size 4×4, stride 2, followed by instance normalization [21] and ReLU. Each stage of *dec-seg* contains a deconvolution (transpose convolution) layer (kernel of 4×4, stride 2), followed by

instance normalization and ReLU. *dec-fill* replaces each deconvolution layers with a nearest–neighbor upsampling layer followed by a convolution layer (kernel sized 3×3, stride 1). This strategy, originally proposed by Odena *et al*. [13] to reduce checkerboard artifacts, was found to be very useful in our experiments (see Sec. 4.3). The total number of convolution kernels at the i-th encoder layer is $\min(2^{i-1} \times 64, 512)$. The number of deconvolution kernels at the i-th *dec-seg* layer or convolution kernels at the i-th *dec-fill* layer are the same as the number of kernels at the $(i-1)$-th encoder layer. The output of the first *dec-seg* layer and *dec-fill* layer has one channel (foreground segmentation) and three channels (recovered background image) respectively. All ReLU layers in the encoder are leaky, with slope of 0.2. In the *dec-seg* branch, standard skip layers are added. More precisely, following the layer indexing in Fig. 1, the input of the i-th layer of *dec-seg* is a concatenation of the output of the $(i+1)$-th layer in the same branch and of the output of the i-th encoder layer (except for the 7-th decoder layer, which only receives input from the 7-th encoder layer.) As mentioned earlier, skip layers ensure good segmentation localization.

The layers of *dec-fill* also receive information from equi-scale encoder layers, but this information is modulated by *neglect masks* generated by neglect nodes. Specifically, the i-th neglect node receives in input data from the i-th encoder layer, concatenated with data from the $(i+1)$-th *dec-seg* layer (note that this is the same as the input to the i-th *dec-seg* layer). A 1×1 convolution, followed by a sigmoid, produces a neglect mask (an image with values between 0 and 1). The neglect mask modulates (by pixel multiplication) the content of the i-th encoder layer, before it is concatenated with the output of the $(i+1)$-th *dec-fill* layer and fed to the i-th *dec-fill* layer. The process is shown in Fig. 2 (a). In practice, neglect nodes provide *dec-fill* with information about which areas of the image should be erased and infilled, while preserving content elsewhere. Visual inspection of the neglect masks shows that they faithfully identify the portion of the image to be removed at various scales (see *e.g.* Fig. 2).

Discriminator: The input to the discriminator is the concatenation of the input image x and of the predicted background y_p or background ground–truth y_g. The structure of the discriminator is the same as the first 5 encoder layers of the generator, but its output undergoes a 1×1 convolution layer followed by a sigmoid function. In the case of 128×128 input dimension, the output size is 4×4, with values between 0 and 1, representing the decoder's confidence that the corresponding region is realistic. The average of these 16 values forms the final discriminator output.

4 Experiments

4.1 Datasets

Training our model requires, for each image, three pieces of information: the original background image (for *dec-fill*); the foreground region (mask) to be removed (for *dec-seg*); and the composite image. Given that this type of rich information is not available in existing datasets, we built our own training and test sets. Specifically, we considered two different datasets for our experiments: one with synthetic text overimposed on regular images, and one with real images of pedestrians.

Text dataset: Images in this dataset are generated by pasting text (generated synthetically) onto real background images. In this way, we have direct access to all necessary data (fore-

Figure 2: (a) Neglect node architecture. Green blocks: encoder output. Blue block: output of previous *dec-seg* layer. (b) Top row: input images. Second and third row: associated neglect masks generated by the neglect node in layer 1 and 2, respectively.

ground, background, and composite image). Text images come from two resources: (1) the word synthesis engine of [17], which was used to generate 50K word images, along with the ground–truth associated segmentation masks; (2) the ICDAR 2013 dataset [9], which provides pixel-level text stroke labels, allowing us to extract 1850 real text regions. Random geometry transformations and color jittering were used to augment the real text, obtaining 50K more word images. Given a sample from the 100K word image pool, a similarly sized background image patch was cropped at random positions from images randomly picked from the MS COCO dataset[10]. More specifically, the background images for our training and validation set come from the training portion of the MS COCO dataset, while the background images for our test set come from the testing portion of the MS COCO dataset. This ensures that the training and testing sets do not share background images. In total, the training, validation and testing portions of our synthetic text dataset contain 100K, 15K and 15K images, respectively.

Pedestrian dataset: This is built from the LASIESTA dataset[8], which contains several video sequences taken from a fixed camera with moving persons in the scene. LASIESTA provides ground–truth pedestrian segmentation for each frame in the videos. In this case, ground–truth background (which is occluded by a person at a given frame) can be found from neighboring frames, after the person has moved away. The foreground map is set to be equal to the segmentation mask provided with the dataset. We randomly selected 15 out of 17 video sequences for training, leaving the rest for testing. A sample of 1821 training images was augmented to 45K images via random cropping and color jittering. The test data set contains 198 images.

4.2 Implementation Details

Our system was implemented using Tensorflow and runs on a desktop (3.3Ghz 6-core CPU, 32G RAM, Titan XP GPU). The model was trained with input images resized to 384×128 (for the text dataset images) or 128×128 (for the pedestrian dataset images.) Adam solver was used during training, with learning rate set to 0.0001, momentum terms set to $\beta_1 = 0.5$ and $\beta_2 = 0.999$, and batch size equal to 8. We set $\lambda_f = \lambda_s = 100$ in the generator loss

Method	Text			Pedestrian			Time (s)
	L_1	PSNR	SSIM	L_1	PSNR	SSIM	
Exemplar (z_p)	5.44%	16.563	0.553	4.11%	17.99	0.74	15.6
Exemplar (z_g)	5.46%	16.769	0.554	3.87%	18.36	0.77	
Contextual (z_p)	3.59%	19.788	0.752	3.41%	20.916	0.879	0.23
Contextual (z_g)	3.28%	20.170	0.779	2.64%	21.997	0.891	
EPLL (z_p)	2.00%	19.123	0.732	2.9%	16.143	0.780	53.5
EPLL (z_g)	1.87%	24.417	0.823	2.61%	16.852	0.800	
IRCNN (z_p)	2.62%	21.282	0.773	2.87%	19.117	0.870	6.67
IRCNN (z_g)	**1.79%**	**25.767**	0.835	**2.39%**	20.225	0.884	
Baseline	2.07%	24.188	0.811	3.06%	23.064	0.883	0.011
Ours (deconv)	1.91%	24.879	0.831	2.63%	23.612	0.904	0.018
Ours (nn+conv)	1.85%	25.300	**0.845**	2.55%	**23.877**	**0.918**	0.018

Table 1: Quantitative comparison of our method against other state-of-the-art image inpainting algorithms (Exemplar [4], Contextual [30], EPLL[35], and IRCNN[31]). Competing inpainting algorithms are fed with a segmentation mask, either predicted by our algorithm (z_p), or ground–truth (z_g). The difference between the original and reconstructed background image is measured using L_1 distance, PSNR (in dB, higher is better) and SSIM (higher is better)[23]. Time measurements refer to a 128×128 input image.

(1), and followed the standard GAN training strategy [7]. Training is alternated between discriminator (D) and generator (G). Note that the adversarial term for the cost L_G in (1) was changed to $(-\log(D(x,y_p)))$ (rather than $\log(1-D(x,y_p))$) for better numerical stability, as suggested by Goodfellow *et al.* [7]. When training D, the learning rate was divided by 2.

4.3 Ablation Study

Baseline: In order to validate the effectiveness of the two-branches decoder architecture and of the neglect layers, we compared our result against a simple baseline structure. This baseline structure is made by the encoder and the *dec-fill* decoding branch, without input from the neglect nodes, but with skip layers from the encoder. This is very very similar to the architecture proposed by Isola *et al.* [8]. Tab. 1 shows that our method consistently outperforms the baseline structure with both datasets and under all three evaluation metrics considered (L_1 residual, PSNR, SSIM [23]). This shows that explicit estimation of the segmentation mask, along with bypass input from the encoder modulated by the neglect mask, facilitates realistic background image synthesis. An example comparing the result of text removal and of inpainting using the full system and the baseline is shown in the first row of Fig. 3.

Deconvolution vs. upsampling + convolution: Deconvolution (or transpose convolution) is a standard approach for generating higher resolution images from coarse level features [6, 8, 18]. A problem with this technique is that it may produce visible checkerboard artifacts, which are due to "uneven overlapping" during the deconvolution process, especially when the kernel size is not divisible by the stride. Researchers [13] have found that by replacing deconvolution with nearest neighbor upsampling followed by convolution, these artifacts can be significantly reduced. In our experiments, we compared the results using these two techniques (see Fig. 3, second row). Specifically, upsampling + convolution was

implemented using a kernel sized 3×3 with stride 1 (as described earlier in Sec. 3.2), while deconvolution was implemented by a kernel with size of 4×4 and stride of 2. Even though the deconvolution kernel side is divisible by the stride, checkerboard artifacts are still visible in most cases using deconvolution. These artifacts do not appear using upsampling + convolution, which also achieves better quantitative results as shown in Tab. 1.

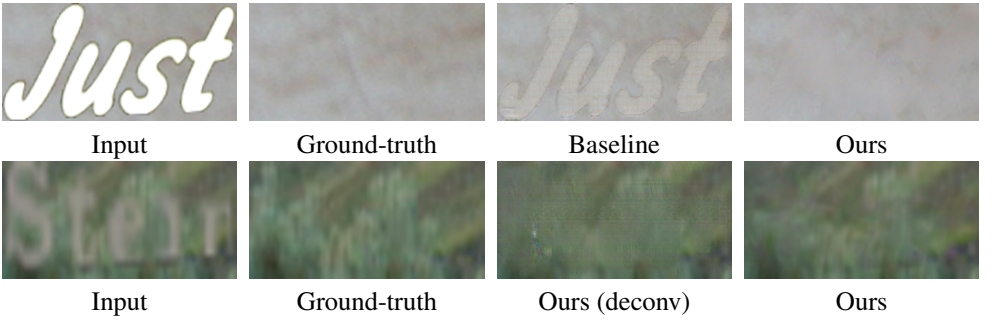

Figure 3: Experimental comparison between the baseline and our architecture (top), and between our architecture using deconvolution and using upsampling+convolution in *dec-fill* (bottom). See Sec. 4.3.

4.4 Comparative Results

Due to the lack of directly comparable methods for automatic content removal, we contrasted our technique with other state-of-the-art image inpainting algorithms, which were provided with a foreground mask. More specifically, we considered two setting for the foreground mask fed to these algorithms: (1) the segmentation mask obtained as a byproduct from our algorithm (z_p), and (2) the ground–truth mask (z_g). Note that the latter is a best-case scenario for the competing algorithms: our system never accessed this mask. In both cases, the masks were slightly dilated to ensure that the whole foreground region was covered.

Tab. 1 shows comparative results with two legacy (but still widely used) inpainting techniques (Exemplar [4] and EPLL[35]), as well as with two more recent CNN-based algorithms (IRCNN[31] and Contextual [30]). When fed with the z_p mask (setting (1)), all competing algorithms produced substantially inferior results with respect to ours under all metrics considered. Even when fed with the (unobservable) ground–truth mask z_g (setting (2)), these algorithms generally performed worse than our system (except for IRCNN, which gave better results than ours, under some of the metrics). We should stress that, unlike the competing techniques, our system *does not* receive an externally produced foreground map. Note also that our algorithm is faster (often by several orders of magnitude) than the competing techniques.

Fig. 4 shows comparative examples of results using our system, IRCNN, and Contextual (where the last two were fed with the ground–truth foreground mask, z_g). Note that, even when provided with the "ideal" mask, the visual quality of the results using these competing methods is generally inferior to that obtained with our content removal technique. The result of IRCNN, which is very similar to the result of EPLL, is clearly oversmoothed. This makes the object boundary visible due to the lack of high frequency details in the filled-in region. We also noted that this algorithms cannot cope well with large foreground masks, as can be

Figure 4: Sample inpainting results using Contextual [30] (third row), IRCNN [31] (fourth row), and our system (last row). Contextual and IRCNN were fed with the ground–truth segmentation mask, while our system automatically extracted and inpainted the foreground. Top row: input image. Second row: ground-truth background image.

seen in the last two columns of Fig. 4 (pedestrian dataset). Contextual [30] does a better job at recovering texture, thanks to its ability to explicitly utilize surrounding image features in its generative model. Yet, we found that our method is often better at completely removing foreground objects. Part of the foreground's boundary is still visible in Contextual's reconstructed background region. Furthermore, the quality Contextual's reconstruction drops significantly when the foreground region reaches the border of the image. This problem is not observed with our method.

Fig. 4 also reveals an interesting (and unexpected) feature of our system. As can be noted in the last two columns, the shadow cast by the person was removed along with the image of the person. Note that the system was *not* trained to detect shadows: the foreground mask only outlined the contour of the person. The most likely reason why the algorithm removed the shadow region is that the background images in the training set data (which, as mentioned in Sec. 4.1, were obtained from frames that did not contain the person) did not contain cast shadows of this type. The system thus decided to synthesize a shadowless image, doubling up as a shadow remover.

5 Conclusion

We have presented the first automatic content removal and impainting system that can work with widely different types and sizes of the foreground to be removed and infilled. Com-

parison with other state-of-the-art inpainting algorithms (which, unlike are system, need an externally provided foreground mask), along with the ablation study, show that our strategy of joint segmentation and inpainting provides superior results in most cases, at a lower computational cost. Future work will extend this technique to more complex scenarios such as wider ranges of foreground region sizes and transparent foreground.

Acknowledgements

Research reported in this publication was supported by the National Eye Institute of the National Institutes of Health under award number R01EY029033. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

[1] Icdar focused scene text dataset. http://rrc.cvc.uab.es/?ch=2.

[2] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424. ACM Press/Addison-Wesley Publishing Co., 2000.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.

[4] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing*, 13(9):1200–1212, 2004.

[5] Carlos Cuevas, Eva María Yáñez, and Narciso García. Labeled dataset for integral evaluation of moving object detection algorithms: Lasiesta. *Computer Vision and Image Understanding*, 152:103–117, 2016.

[6] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

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

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

[9] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011.

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[11] Yang Liu, Jinshan Pan, and Zhixun Su. Deep blind image inpainting. *arXiv preprint arXiv:1712.09078*, 2017.

[12] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[13] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016.

[14] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016.

[15] Siyang Qin and Roberto Manduchi. Cascaded segmentation-detection networks for word-level text spotting. *arXiv preprint arXiv:1704.00834*, 2017.

[16] Siyang Qin, Seongdo Kim, and Roberto Manduchi. Automatic skin and hair masking using fully convolutional networks. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, 2017.

[17] Siyang Qin, Peng Ren, and Roberto Kim, Seongdo Manduchi. Robust and accurate text stroke segmentation. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2018.

[18] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[19] Jimmy SJ Ren, Li Xu, Qiong Yan, and Wenxiu Sun. Shepard convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909, 2015.

[20] Stefan Roth and Michael J Black. Fields of experts: A framework for learning image priors. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 860–867. IEEE, 2005.

[21] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint*, 2016.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[23] 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.

[24] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems*, pages 341–349, 2012.

[25] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1395–1403, 2015.

[26] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[27] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 3, 2017.

[28] Cong Yao, Xiang Bai, Nong Sang, Xinyu Zhou, Shuchang Zhou, and Zhimin Cao. Scene text detection via holistic, multi-channel prediction. *arXiv preprint arXiv:1606.09002*, 2016.

[29] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5485–5493, 2017.

[30] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. *arXiv preprint arXiv:1801.07892*, 2018.

[31] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. *arXiv preprint*, 2017.

[32] Zheng Zhang, Chengquan Zhang, Wei Shen, Cong Yao, Wenyu Liu, and Xiang Bai. Multi-oriented text detection with fully convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4159–4167, 2016.

[33] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017.

[34] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015.

[35] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 479–486. IEEE, 2011.