Mastering Sketching: Adversarial Augmentation for Structured Prediction

EDGAR SIMO-SERRA, SATOSHI IIZUKA, and HIROSHI ISHIKAWA, Waseda University

We present an integral framework for training sketch simplification networks that convert challenging rough sketches into clean line drawings. Our approach augments a simplification network with a discriminator network, training both networks jointly so that the discriminator network discerns whether a line drawing is real training data or the output of the simplification network, which, in turn, tries to fool it. This approach has two major advantages: first, because the discriminator network learns the structure in line drawings, it encourages the output sketches of the simplification network to be more similar in appearance to the training sketches. Second, we can also train the networks with additional unsupervised data: by adding rough sketches and line drawings that are not corresponding to each other, we can improve the quality of the sketch simplification. Thanks to a difference in the architecture, our approach has advantages over similar adversarial training approaches in stability of training and the aforementioned ability to utilize unsupervised training data. We show how our framework can be used to train models that significantly outperform the state of the art in the sketch simplification task, despite using the same architecture for inference. We also present an approach to optimize for a single image, which improves accuracy at the cost of additional computation time. Finally, we show that, using the same framework, it is possible to train the network to perform the inverse problem, i.e., convert simple line sketches into pencil drawings, which is not possible using the standard mean squared error loss. We validate our framework with two user tests, in which our approach is preferred to the state of the art in sketch simplification 88.9% of the time.

CCS Concepts: • **Applied computing** → **Fine arts**; • **Computing methodologies** → *Neural networks*;

Additional Key Words and Phrases: Sketch simplification, pencil drawing generation, convolutional neural network

ACM Reference format:
Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. 2018. Mastering Sketching: Adversarial Augmentation for Structured Prediction. *ACM Trans. Graph.* 37, 1, Article 11 (January 2018), 13 pages.
https://doi.org/10.1145/3132703

1 INTRODUCTION

Sketching plays a critical role in the initial stages of artistic work such as product design and animation, allowing artists to quickly draft and visualize their thoughts. However, the process of digitizing and cleaning up rough pencil drawings involves a large overhead. This process is called *sketch simplification* and involves simplifying multiple overlapped lines into a single line and erasing superfluous lines that are used as references when drawing, as shown in Figure 1. Due to the large variety of drawing styles and tools, developing a generic approach to sketch simplification is a challenging task. In this work, we propose a novel approach to sketch simplification that is able to outperform current approaches. Furthermore, our proposed framework can also be used to do the inverse problem, i.e., pencil drawing generation, as shown in Figure 2.

Recently, we proposed to automatize the sketch simplification task with a fully convolutional network (Simo-Serra et al. 2016). In order to train this network, large amounts of supervised data were obtained, consisting of pairs of rough sketches and their corresponding sketch simplifications. To collect this data, the tedious process of inverse dataset construction was used. This involves hiring artists to draw a rough sketch, simplifying the sketch into a clean line drawing, and, finally, redrawing the rough sketch on top of the line drawing to ensure that the training data is aligned. This is not only time and money consuming, but the resulting data also differs from the true rough sketches drawn without clean line drawings as references, as shown in Figure 3. The resulting models trained with this data therefore generalize poorly to rough sketches found "in the wild," which are representative of true sketch simplification usage cases. Here, we propose a framework that can incorporate these unlabeled sketches found "in the wild" into the learning process and significantly improve the performance in real usage cases.

Our approach combines a fully convolutional sketch simplification network with a discriminator network that is trained to discern real line drawings from those generated by a network. The simplification network is trained to both simplify sketches and to deceive the discriminator network. In contrast to optimizing with only the standard mean squared error loss, which considers only individual pixels and not their neighbors, the discriminator loss considers the entire output image. This allows for significant improvement of the quality of the sketch simplifications. Furthermore, the same framework allows for augmentation of the supervised training dataset with unsupervised examples, leading to a hybrid supervised and unsupervised learning framework that we call *Adversarial Augmentation*. We evaluate our approach on a diverse set of challenging rough sketches, comparing with the state of the art and alternate optimization strategies. We also perform a user study in which our approach is preferred 88.9% of the time to the leading competing approach.

In addition, we evaluate our framework on the inverse sketch simplification problem, i.e., generating pencil drawings from line drawings. We note that, unlike the sketch simplification problem, converting clean sketches into rough sketches cannot be accomplished by using a straightforward supervised approach, such as a

Edgar Simo-Serra and Satoshi Iizuka assert equal contribution and joint first authorship. This work was partially supported by JST CREST Grant Number JPMJCR14D1, JST ACT-I Grant Number JPMJPR16UD, and JST ACT-I Grant Number JPMJPR16U3.

Authors' addresses: E. Simo-Serra, S. Iizuka, and H. Ishikawa, Waseda University, Okubo 3-4-1, Shinjuku, 169-8555 Tokyo; emails: {esimo, iizuka}@aoni.waseda.jp, hfs@waseda.jp.

https://doi.org/10.1145/3132703

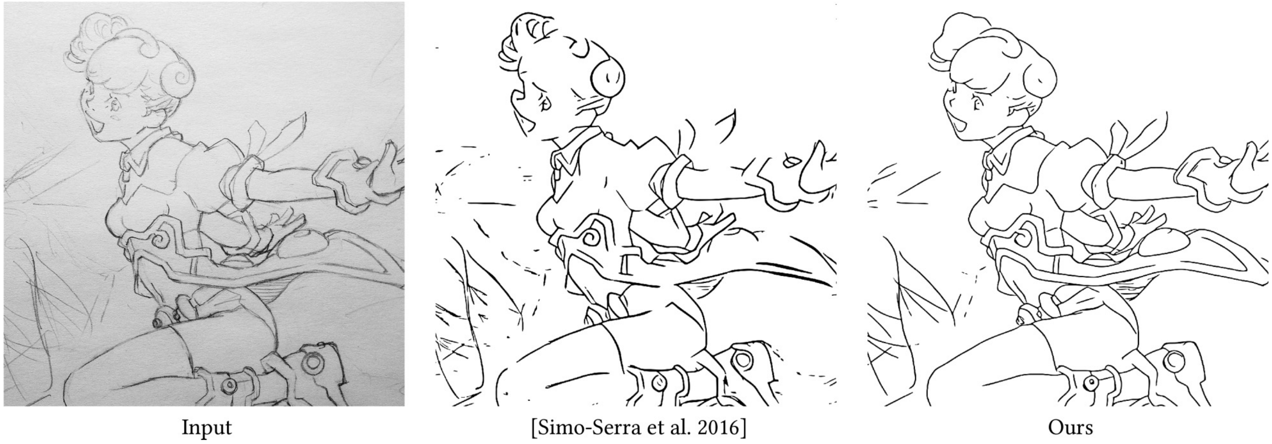


Fig. 1. Comparison of our approach for sketch simplification with the state of the art on a complex real-world pencil drawing. Existing approaches miss important lines and preserve superfluous details such as shading and support scribbles, which hampers further processing of the drawing, e.g., coloring. Output is shown with postprocessing vectorization. Image copyrighted by Eisaku Kubonoichi.

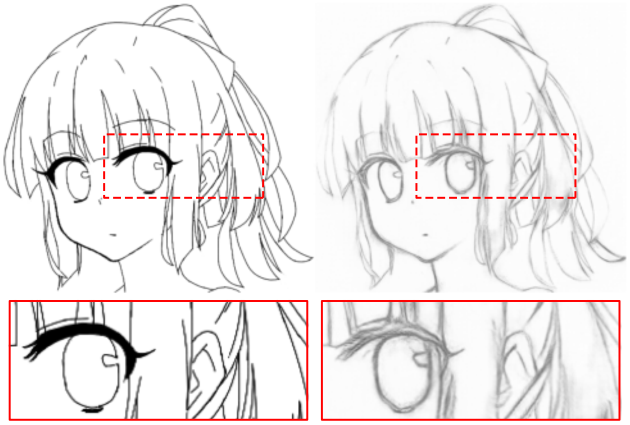


Fig. 2. Pencil drawing generation with our framework. The line drawing on the left is turned into the pencil drawing on the right. We show a zoomed-in area in which we can see that the network learns to add smudges, superfluous lines, and significantly changes the eye.

mean squared error loss: the model is unable to learn the structure of the output. Instead of producing rough pencil drawings, it produces a blurry representation of the input. However, by using our adversarial augmentation framework, we can successfully train a model to convert clean sketches into rough sketches.

Finally, as another usage case of our framework, we also consider the case of augmenting the training data with the test data input. We note that we use the test data in an unsupervised manner: no privileged information is used. By using the discriminator network and training jointly with the test images, we can significantly improve the results at the cost of computational efficiency.

In summary, we present:

- A unified framework to jointly train from supervised and unsupervised data.
- Significant improvements over the state of the art in sketch simplification quality.
- A method to further improve the quality by single image training.
- Results for pencil drawing generation.

2 BACKGROUND

Sketch Simplification. As sketch simplification is a tedious manual task for most artists, many approaches attempting to automate it have been proposed. A common approach is to assist the user by adjusting the strokes by, for instance, using geometric constraints (Igarashi et al. 1997), fitting Bézier curves (Bae et al. 2008), or merging strokes based on heuristics (Grimm and Joshi 2012). These approaches require all the strokes and their drawing order as input. Along similar lines, many works have focused on the problem of simplifying vector images (Fišer et al. 2015; Lindlbauer et al. 2013; Liu et al. 2015; Orbay and Kara 2011; Shesh and Chen 2008). However, approaches that can be used on raster images have been unable to process complex real-world sketches (Chen et al. 2013; Noris et al. 2013). Most of these approaches rely on a preprocessing stage that converts the image into a graph that is then optimized (Favreau et al. 2016; Hilaire and Tombre 2006; Noris et al. 2013); however, they generally cannot recover from errors in the preprocessing stage. Recently, we proposed a fully automatic approach for simplifying sketches directly from raster images of rough sketches by using a fully convolutional network (Simo-Serra et al. 2016). However, this approach still needs large amounts of supervised data, consisting of pairs of rough sketches and their corresponding sketch simplifications, tediously created by the process of inverse dataset construction. More important, training sketches differ from the true rough sketches, as shown in Figure 3. The resulting models trained with this data therefore generalize poorly to real rough sketches. We build upon this work and propose a framework that can incorporate unlabeled real sketches into the learning process and significantly improve the performance in real usage cases. Thus, in contrast to all previous works, we consider challenging real-world scanned data that is significantly more complex than previously attempted.

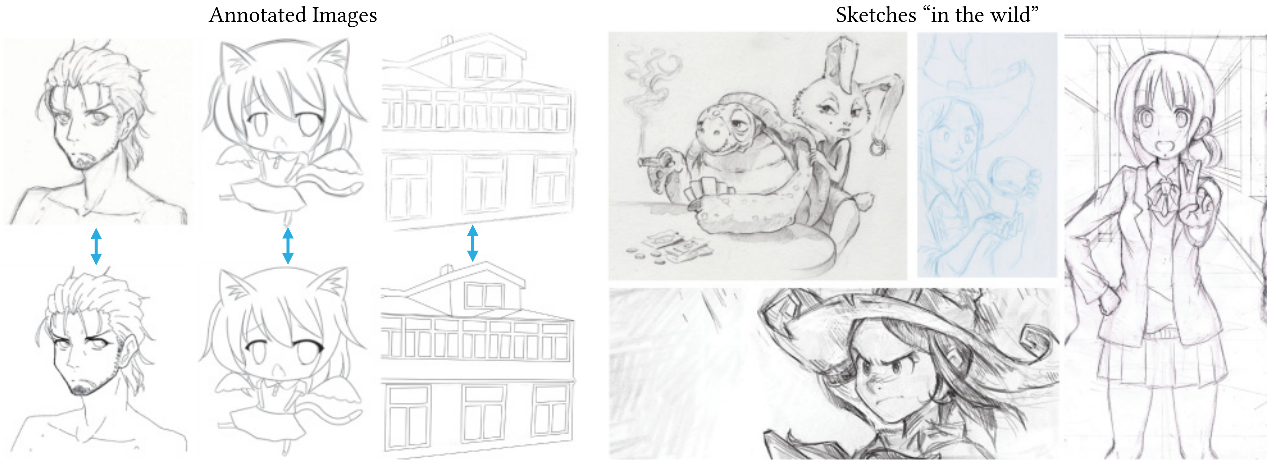


Fig. 3. Comparison between the supervised dataset of Simo-Serra et al. [2016] and rough sketches found in the wild. The difficulty of obtaining high-quality and diverse rough sketches and their corresponding simplified sketches greatly limits performance on rough sketches “in the wild” that can be significantly different from the annotated data used for training models. The three images on the left of the sketches “in the wild” are copyrighted by David Revoy (www.davidrevoy.com) and licensed under CC-by 4.0.

Generative Adversarial Networks. In order to train generative models using unsupervised data with back-propagation, Goodfellow et al. (2014) proposed the Generative Adversarial Networks (GANs). In the GAN approach, a generative model is paired with a discriminative model and trained jointly. The discriminative model is trained to discern whether an image is real or artificially generated, while the generative model is trained to deceive the discriminative model. By training both jointly, it is possible to train the generative model to create realistic images from random inputs (Radford et al. 2016). There is also a variant, Conditional GAN (CGAN) that learns a conditional generative model. This can be used to generate images conditioned on class labels (Mirza and Osindero 2014). In a concurrent work, using a CGAN for the image-to-image synthesis problem was recently proposed in Isola et al. (2017), in which the authors use a CGAN loss and apply it to tasks such as image colorization and scene reconstruction from labels. However, the CGAN is unable to use unsupervised data, which helps improve performance significantly. Recently, variants have been proposed for learning image-to-image synthesis problems, such as the one we tackle in this work, by combining standard losses with a GAN-based discriminator loss for applications such as image completion (Pathak et al. 2016), generating images from surface maps (Wang and Gupta 2016), clothing from images (Yoo et al. 2016), autoencoders (Dosovitskiy and Brox 2016), style transfer (Li and Wand 2016), or super-resolution (Ledig et al. 2017). In this article, we use a similar approach and show that it can be easily extended to use unsupervised data as large amounts of data augmentation, which has not been explored in other works. Unlike recent pure unsupervised approaches (Taigman et al. 2017; Zhu et al. 2017), the supervised data allows for maintenance of the fidelity to the original data by explicitly forcing a correspondence between the input and output. Our approach generates significantly better sketch simplification than existing approaches. An overview of different approaches is illustrated in Figure 4.

Pencil Drawing Generation. To our knowledge, the inverse problem of converting clean raster sketches to pencil drawings has not been tackled before. Making natural images appear like sketches has been widely studied (Kang et al. 2007; Lu et al. 2012), as natural images have rich gradients that can be exploited for the task. However, using clean sketches that contain very sparse information as inputs is an entirely different problem. In order to produce realistic strokes, most approaches rely on a dataset of examples (Berger et al. 2013), which heuristically matches input strokes to those in the database, and are limited to vector input images, while our approach can directly create novel realistic rough-sketch strokes that differ significantly from the training data from raster clean sketches. In comparison with recent style transfer approaches (Gatys et al. 2016), our approach can preserve more details and produce more convincing pencil drawings.

2.1 Deep Learning

We base our work on deep multilayer convolutional neural networks (Fukushima 1988; LeCun et al. 1989), which have seen a surge in usage in the past few years and have seen application in a wide variety of problems. Just restricting our attention to those with image input and output, there are such recent works as super-resolution (Dong et al. 2016), semantic segmentation (Noh et al. 2015), and image colorization (Iizuka et al. 2016). These networks are all built upon convolutional layers of the form

$$y^c_{u,v} = \sigma\left(\sum_k \left(b^{c,k} + \sum_{i=u-M}^{u+M} \sum_{j=v-N}^{v+N} w^{c,k}_{i+M,j+N}\, x^k_{i,j}\right)\right), \qquad (1)$$

where, for a $(2M+1) \times (2N+1)$ convolution kernel, each output channel c and coordinates (u, v), the output value $y^c_{u,v}$ is computed as an affine transformation of the input pixel $x^k_{u,v}$ for all input channels k with a shared weight matrix formed by $w^{c,k}$ and bias value $b^{c,k}$ that is run through a nonlinear activation function $\sigma(\cdot)$.

The most widely used nonlinear activation function is the Rectified Linear Unit (ReLU) where $\sigma(x) = \max(0, x)$ (Nair and Hinton 2010).

These layers are a series of learnable filters with w and b being the learnable parameters. In order to train a network, a dataset consisting of pairs of input and their corresponding ground truth (x, y^*) are used in conjunction with a loss function $L(y, y^*)$ that measures the error between the output y of the network and the ground truth y^*. This error is used to update the learnable parameters with the backpropagation algorithm (Rumelhart et al. 1986). In this work, we also consider the scenario in which not all data is necessarily in the form of pairs (x, y^*) but can also be in the form of single samples x and y^* that are not corresponding pairs.

Our work is based on fully convolutional neural network models that can be applied to images of any resolution. These networks generally follow an encoder-decoder architecture in which the first layers of the network have an increased stride to lower the resolution of the input layer. At lower resolutions, the subsequent layers are able to process larger regions of the input image: for instance, a 3 × 3-pixel convolution on an image at half resolution is computed with a 5 × 5-pixel area of the original image. Additionally, by processing at lower resolutions, both the memory requirements and computation times are significantly decreased. In this article, we base our network model on that of Simo-Serra et al. (2016) and show that we can greatly improve the performance of the resulting model by using a significantly improved learning approach.

3 ADVERSARIAL AUGMENTATION

We present *adversarial augmentation*, which is the fusion of unsupervised and adversarial training focused on the purpose of augmenting existing networks for structured prediction tasks. An overview of our approach compared with different training approaches can be seen in Figure 4. Similar to GANs (Goodfellow et al. 2014), we employ a discriminator network that attempts to distinguish whether an image comes from real data or is the output of another network. Unlike in the case of standard supervised losses, such as the Mean Squared Error (MSE), with the discriminator network, the output is encouraged to have a global consistency similar to the training images. Although similar approaches have been proposed (Ledig et al. 2017; Pathak et al. 2016; Zhou and Berg 2016), they have focused exclusively on supervised training. In this work, we focus on training with additional unlabeled examples and show that this both increases performance and generalization. An overview of our framework can be seen in Figure 5.

While this work focuses on sketch simplification, the presented approach is general and applicable to other structured prediction problems, such as semantic segmentation or saliency detection.

3.1 The GAN Framework

The purpose of the GAN (Goodfellow et al. 2014) is, given a training set of samples, estimating a generative model that stochastically generates samples similar to those drawn from a distribution represented by the given training set ρ_y. Thus, a trained generative model produces, for instance, pictures of random cars similar to a given set of sample pictures. A generative model is described as a neural network model $G : z \mapsto y$ that maps a random vector z to an output y. In the GAN framework, we train two network models: (1) the generative model G above and (2) a discriminator model $D : y \mapsto D(y) \in \mathbb{R}$ that computes the probability that a structured input (e.g., image) y came from the real data, rather than the output of G. We jointly optimize G and D with respect to the expectation value:

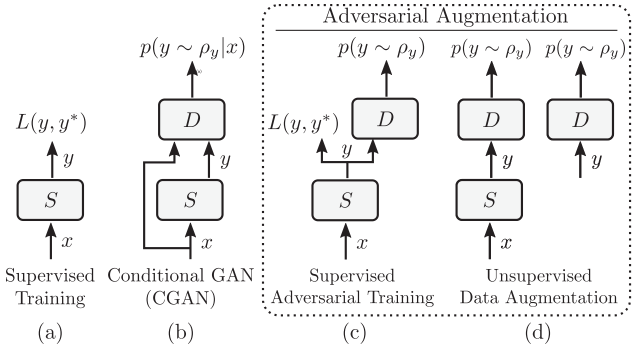


Fig. 4. Overview of our approach, Adversarial Augmentation, compared with different network training approaches. Our approach consists of the combination of Supervised Adversarial Training with Unsupervised Data Augmentation. (a) To train a prediction network $S : x \mapsto y$, supervised training trains by using a specific loss $L(y, y^*)$ that encourages the output y of the network to match that of the ground truth data y^*. (b) The Conditional Generative Adversarial Network (CGAN) introduces an additional discriminator network D that attempts to discern whether an image is from the training data or a prediction by the prediction network S, while S is trained to deceive D. The discriminator network D takes two inputs x and y and estimates the conditional probability that y comes from the training data given x. (c) The supervised training and the adversarial training can be combined, using a loss $L(y, y^*)$ to force the output to be consistent with the input while also using a discriminator network D, similar to a CGAN, but not conditioned on x (Supervised Adversarial Training). (d) The fact that S only takes x as input and that D only takes y enables us to use training data x and y that are not related, i.e., in an unsupervised manner, to further train S and D (Unsupervised Data Augmentation).

$$\min_G \max_D \; \mathbb{E}_{y \sim \rho_y} [\, \log D(y) \,] + \mathbb{E}_{z \sim p_z} [\, \log(1 - D(G(z))) \,], \qquad (2)$$

by alternately maximizing the classification log-likelihood of D and then optimizing G to deceive D by minimizing the classification log-likelihood of $1 - D(G(z))$. By this process, it is possible to train the generative model to create realistic images from random inputs (Radford et al. 2016). In practice, however, this min-max optimization is unstable and hard to tune so that desired results can be obtained, which led to some follow-up work (Radford et al. 2016; Salimans et al. 2016). This model is also limited in that it can handle only relatively low, fixed resolutions.

This generative model was later extended to the CGANs (Mirza and Osindero 2014), which models $G : (x, z) \mapsto y$ that generates y conditioned on some input x. Here, the discriminator model $D : (x, y) \mapsto D(x, y) \in \mathbb{R}$ also takes x as an additional input to evaluate the *conditional* probability given x. Thus, G and D are optimized with the objective

$$\begin{aligned}\min_G \max_D \; & \mathbb{E}_{(x,y^*) \sim \rho_{x,y}} [\, \log D(x, y^*) \,] \\ & + \mathbb{E}_{(x,y^*) \sim \rho_{x,y}, z \sim p_z} [\, \log(1 - D(x, G(x, z))) \,]. \qquad (3)\end{aligned}$$

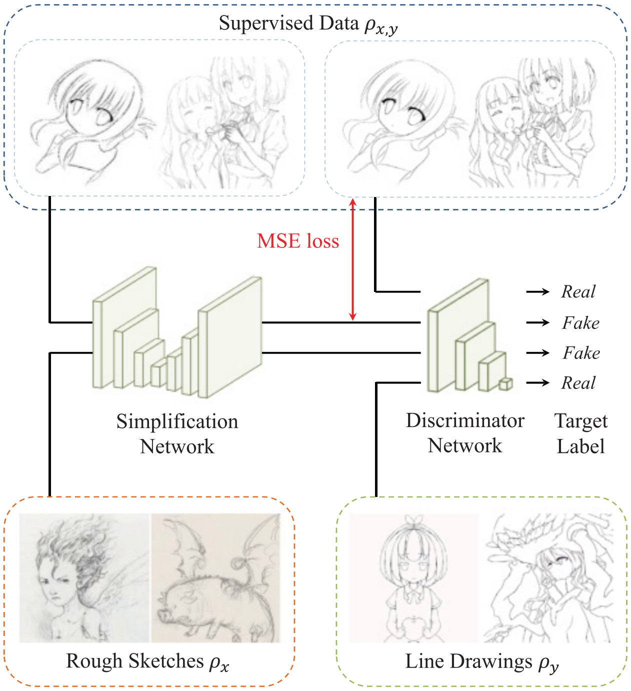


Fig. 5. Overview of our adversarial augmentation framework. We train the simplification network using both supervised pairs of data $\rho_{x,y}$, and unsupervised data ρ_x and ρ_y. The discriminator network is trained to distinguish real line drawings from those output by the simplification network, while the simplification network is trained to deceive the discriminator network. For the data pairs, we also use the MSE loss, which forces the simplification network outputs to correspond to the input rough sketches. The two images forming Rough Sketches σ_x are copyrighted by David Revoy (www.davidrevoy.com) and licensed under CC-by 4.0.

Note that the first expectation value is the average over supervised training data $\rho_{x,y}$ consisting of input-output pairs. Because of this, the CGAN framework is only applicable to supervised data.

The CGAN can be used for the same purpose as ours, *prediction*, rather than sample generation. For that, we just omit the random input z so that $S : x \mapsto y = S(x)$ is now a deterministic prediction given the input x. In this case, we thus optimize:

$$\min_S \max_D \mathbb{E}_{(x,y^*)\sim\rho_{x,y}} \left[\log D(x,y^*) + \log(1 - D(x,S(x))) \right] . \quad (4)$$

3.2 Supervised Adversarial Training

In order to adapt the training to image-to-image problems, recent approaches (Pathak et al. 2016) combine a supervised loss with a discriminator for improved performance. A prediction model $S : x \mapsto y$ is trained jointly with a discriminator model $D : y \mapsto D(y) \in \mathbb{R}$ that is not conditioned on the input x while using a supervised training loss $L(S(x), y^*)$, where y^* is the ground truth output corresponding to the input x. This can be called *supervised adversarial training* and is written as optimizing

$$\min_S \max_D \mathbb{E}_{(x,y^*)\sim\rho_{x,y}} \left[\alpha \log D(y^*) \right.$$
$$\left. + \alpha \log(1 - D(S(x))) + L(S(x), y^*) \right] , \quad (5)$$

where α is a weighting hyper-parameter and the expectation value is over a supervised training set $\rho_{x,y}$ of input-output pairs. This is a weighted combination of the loss $L(S(x), y^*)$ and a GAN-like adversarial loss, trained on pairs of supervised samples. The difference from GAN is that, here, supervised data is used. The difference from CGAN is that the coupling between the input x and the ground truth output y^* is through the conditional discriminator D in the case of CGAN (Equation (4)), while in the case of the supervised adversarial training (Equation (5)), they are coupled directly through the supervised training loss $L(S(x), y^*)$.

The training consists of jointly maximizing the output of the discriminator network D and minimizing the loss of the model with structured output by S. For each training iteration, we alternately optimize D and S until convergence. The hyper-parameter α controls the influence of the adversarial training on the network and is critical for training. Setting α too low gives no advantage over training with the supervised training loss, while setting it too high causes the results to lose coherency with the inputs, i.e., the network generates realistic outputs, however, they do not correlate with the inputs.

3.3 Adversarial Augmentation Framework

Our objective function above is motivated by the desire for unsupervised training. In most structured prediction problems, creating supervised training data by annotating the inputs is a time-consuming task. However, in many cases, it is much easier to procure nonmatching input and output data; thus, it is desirable to be able to somehow use them to train our prediction model. Note that in our objective function (Equation (5)), the first term inside the expectation value only needs y, whereas the second only takes x. This suggests that we can train using these terms separately with unsupervised data. It turns out that we can indeed use the supervised adversarial objective function to also incorporate the unsupervised data into the training, by separating the first two terms in the expectation value over supervised data in Equation (5) into new expectation values over unsupervised data.

Suppose that we have large amounts of both input data ρ_x and output data ρ_y, in addition to a dataset $\rho_{x,y}$ of fully annotated pairs. We modify the optimization function to:

$$\begin{aligned}\min_S \max_D \; & \mathbb{E}_{(x,y^*)\sim\rho_{x,y}} \left[L(S(x), y^*) + \alpha \log D(y^*) \right. \\ & \qquad \left. + \alpha \log(1 - D(S(x))) \right] \\ & + \beta\, \mathbb{E}_{y\sim\rho_y} \left[\log D(y) \right] \\ & + \beta\, \mathbb{E}_{x\sim\rho_x} \left[\log(1 - D(S(x))) \right] , \end{aligned} \quad (6)$$

where β is a weighting hyper-parameter for the unsupervised data term.

Optimization is done on both S and D jointly using supervised data $\rho_{x,y}$ and unsupervised data ρ_x and ρ_y. If learning from only ρ_x and ρ_y, the model S will not necessarily learn the mapping $x \mapsto y$, but only to generate realistic outputs y, which is not the objective of structured prediction problems. Thus, using the supervised dataset $\rho_{x,y}$ is still critical for training as well as the model loss $L(S(x), y^*)$. The supervised data can be seen as an anchor that forces the model to generate outputs coherent with the inputs, while the unsupervised data is used to encourage the model to generate realistic outputs for a wider variety of inputs.

See Figure 5 for a visualization of the approach. As we note above, it is not possible to train CGAN models (Equation (4)) using unsupervised data, as the discriminator network D requires both the input x and output y of the model as input.

One interesting benefit of being able to use unsupervised data is that we can treat the test data as additional unsupervised data, and train the network on the fly to improve the prediction results. Thus, the results are improved by encouraging the prediction network S to deceive the discriminator network D for testing data. This does, however, incur a heavy overhead as it requires using the entire training framework and optimizing the network.

4 MASTERING SKETCHING

We focus on applying our approach to the sketch simplification problem and its inverse. Sketch simplification consists of processing rough sketches, such as those drawn by pencil, into clean images that are amenable to vectorization. Our approach is also the first that can handle the inverse problem, that is, converting clean sketches into pencil drawings.

4.1 Simplification Network

In order to simplify rough sketches, we rely on the same model architecture as Simo-Serra et al. (2016). The model consists of a 23-layer, fully convolutional network that has three main building blocks: down-convolutions, 3 × 3 convolutions with a stride of 2 to halve the resolution; flat-convolutions, 3 × 3 convolutions with a stride of 1 that maintain the resolution; and up-convolutions, 4 × 4 convolutions with a stride of 1/2 to double the resolution. In all cases, 1 × 1 padding to compensate for the reduction in size caused by the convolution kernel as well as the ReLU activation functions are employed. The general structure of the network follows an hourglass shape, that is, the first seven layers contain three down-convolution layers to decrease the resolution to one eighth. Afterward, seven flat-convolutions are used to further process the image, and, finally, the last nine layers contain three up-convolution layers to restore the resolution to that of the input size. There are two exceptional layers: the first is a down-convolution layer with a 5 × 5 kernel and 2 × 2 padding and the last layer uses a sigmoid activation function to output a grayscale image in which all values are in the [0, 1] range. In contrast with Simo-Serra et al. (2016), which used a weighted MSE loss, we use the MSE loss as the model loss

$$L(S(x), y^*) = \| S(x) - y^* \|^2 , \tag{7}$$

where $\| \cdot \|$ is the Euclidean norm.

Note that the MSE loss itself is not a structured prediction loss, i.e., it is oblivious of any structure between the component pixels. For each output pixel, neighboring output pixels have no effect. However, by also using the supervised adversarial loss, as done in Equation (6), the resulting optimization does take into account the structure of the output.

4.2 Discriminator Network

The objective of the auxiliary discriminator network is not that of high performance but rather to help train the simplification network. If the discriminator network becomes too strong with respect to the simplification network, the gradients used for training generated by the discriminator network tend to vanish, causing the optimization to fail to converge. To avoid this issue, the network is kept "shallow," uses large amounts of pooling, and employs dropout (Srivastava et al. 2014). This also allows for reduction of overall memory usage and computation time, speeding up the training itself.

Table 1. Architecture of the Discriminator Network

	Kernel	Activation	
Layer Type	Size	Function	Output
input	-	-	1 × 384 × 384
convolutional	5 × 5	ReLU	16 × 192 × 192
convolutional	3 × 3	ReLU	32 × 96 × 96
convolutional	3 × 3	ReLU	64 × 48 × 48
convolutional	3 × 3	ReLU	128 × 24 × 24
convolutional	3 × 3	ReLU	256 × 12 × 12
dropout (50%)	-	-	512 × 6 × 6
convolutional	3 × 3	ReLU	512 × 6 × 6
dropout (50%)	-	-	512 × 6 × 6
fully connected	-	Sigmoid	1

Note: All convolutional layers use padding to avoid a decrease in output size and a stride of 2 to halve the resolution of the output.

We base our discriminator network on a small CNN with seven layers, the last being fully connected. Similar to the simplification network, the first layer uses a 5 × 5 convolution and all subsequent convolutional layers use 3 × 3 convolutions. The first convolutional layer has 16 filters and all subsequent convolutional layers double the number of filters. We also incorporate 50% dropout (Srivastava et al. 2014) layers after the last two convolutional layers. All fully connected layers use Rectified Linear Units (ReLU) except the final layer, which uses the sigmoid activation function to have a single output that corresponds to the probability that the input came from the real data ρ_y instead of the output of S. An overview of the architecture can be seen in Table 1.

4.3 Training

Adversarial networks are notoriously hard to train; this has led to a series of heuristics for training. In particular, for GANs, the balance between the learning of the discriminative and generative components is critical, i.e., if the discriminative component is too strong, the generative component is unable to learn and vice versa. Unlike the GAN, which has to rely entirely on the adversarial network for learning, we also have supervised data $\rho_{x,y}$, which partially simplifies the training.

Training of both networks is done with backpropagation (Rumelhart et al. 1986). For a more fine-grained control of the training, we balance the supervised training loss $L(S(x), y^*)$ and the adversarial loss so that the gradients are roughly the same order of magnitude.

An alternate training scheme is used for both networks: in each iteration, we first update the discriminator network with a mini-batch, and then proceed to update the generative network using the same mini-batch.

During the training, we use batch normalization layers (Ioffe and Szegedy 2015) after all convolutional layers for both the simplification network and the discriminator network, which

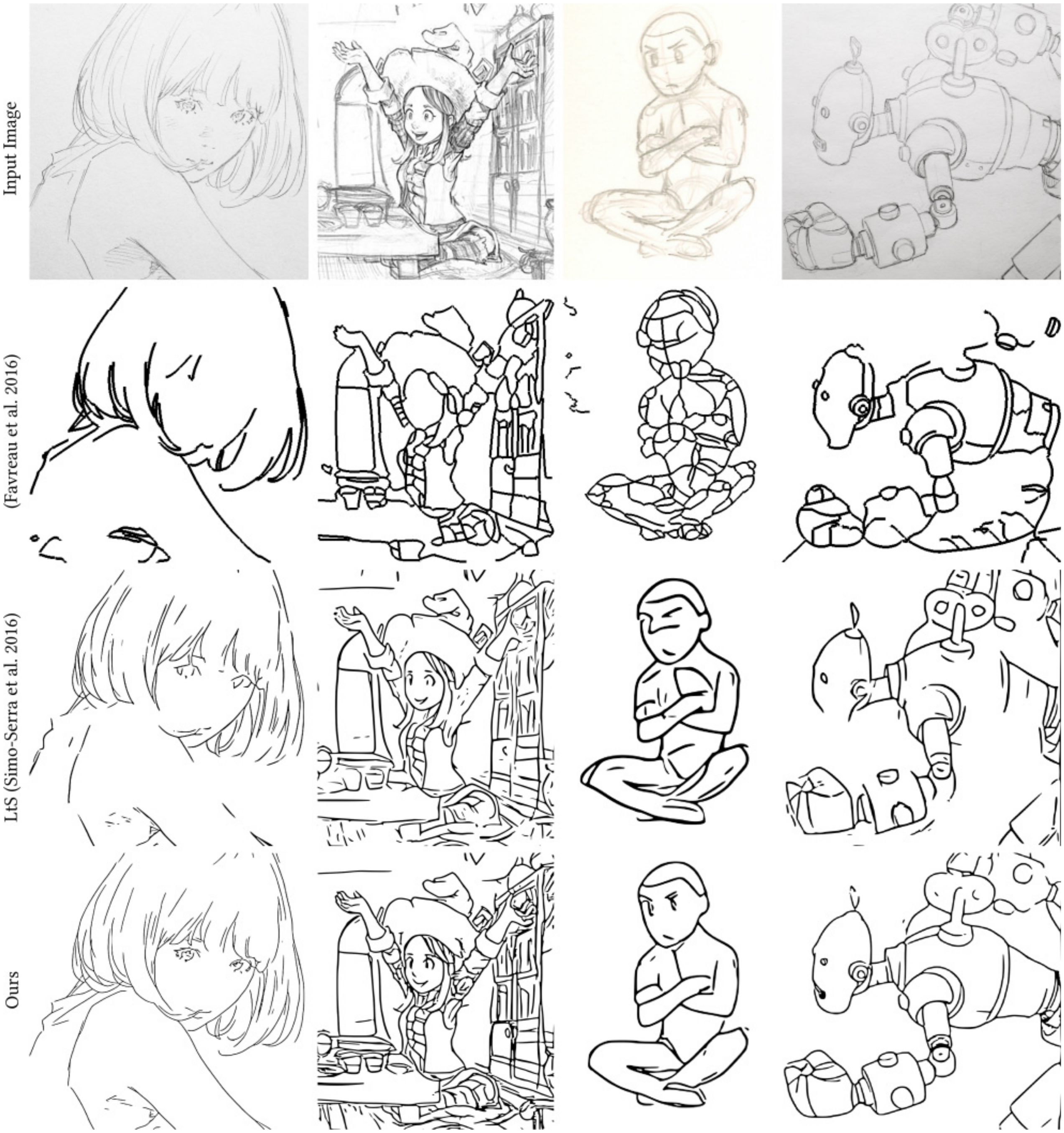


Fig. 6. Comparison with the state-of-the-art methods of Favreau et al. (2016) and LtS (Simo-Serra et al. 2016). We note that these images are significantly more challenging than those tackled in previous works. For the approach of Favreau et al. (2016), we had to also preprocess the image with a tone curve and tune the default parameters in order to obtain the shown results. Without this manual tweaking, recognizable outputs were not obtained. For both LtS and our approach, we did not preprocess the image but rather postprocessed the output with simple vectorization techniques (Selinger 2003). While Favreau et al. (2016) manage to capture the global structure somewhat, many different parts of the image are missing due to the complexity of the scene. LtS fails to simplify most regions in the scene and fails to preserve important details. Our approach can simplify all of the images, both preserving details and obtaining crisp and clean outputs. The first- and fourth-column images are copyrighted by Eisaku Kubonoichi. The second-column image is copyrighted by David Revoy (www.davidrevoy.com) and licensed under CC-by 4.0.

are then folded into the preceding convolutional layers during the testing. Optimization is done using the ADADELTA algorithm (Zeiler 2012), which abolishes the need for tuning hyper-parameters, such as the learning rate, adaptively setting a different learning rate for all weights in the network.

We follow a similar data augmentation approach as Simo-Serra et al. (2016), namely, training with eight additional levels of downsampling: $7/6$, $8/6$, $9/6$, $10/6$, $11/6$, $12/6$, $13/6$, and $14/6$, while using the constant-size 384×384 image patch crops. All training output images are thresholded so that pixel values below 0.9 are set to 0 (pixels are in the $[0, 1]$ range). All the images are randomly rotated and flipped during the training. Furthermore, we sample the image patches with more probability from larger images such that patches from a 1024×1024 image will be four times more likely to appear than patches from a 512×512 image. We subtract the mean of the input images of the supervised dataset from all images. Finally, with a probability of 10%, the ground truth images are used as both input and output with the supervised loss to teach the model that sketches that are already simplified should not be modified.

5 EXPERIMENTS

We train our models using the supervised dataset from Simo-Serra et al. (2016), consisting of 68 pairs of rough sketches with their corresponding clean sketches ($\rho_{x,y}$), in addition to 109 clean sketches (ρ_y) and 85 rough sketches (ρ_x) taken from Flickr and other sources. Note that the additional training data ρ_y and ρ_x require no additional annotations, unlike the original supervised dataset. Some examples of the data used for training are shown in Figure 5. We set $\alpha = \beta = 8 \times 10^{-5}$ and train for 150,000 iterations. Each batch consists of 16 pairs of image patches sampled from the 68 pairs in $\rho_{x,y}$, 16 image patches sampled from the 109 clean sketches in ρ_y, and 16 image patches sampled from the 85 rough sketches in ρ_x. We initialize the model weights for all models by training exclusively in a supervised fashion on the supervised data $\rho_{x,y}$ and in particular use the state-of-the-art model (Simo-Serra et al. 2016). We note that this pretraining is critical for learning and that without it training is greatly slowed down and in the case of CGAN, it does not converge.

5.1 Comparison with the State of the Art

We compare with the state of the art of Favreau et al. (2016) and Learning to Sketch (LtS) (Simo-Serra et al. 2016) in Figure 6. For the comparison, we use the postprocessing approach of Simo-Serra et al. (2016), which consists of applying a high-pass filter and binarizing the output images before vectorizing them. We use the same parameters as Simo-Serra et al. (2016) for all output images. We can see that, in general, the approach of Favreau et al. (2016) fails to preserve most fine details while preserving unnecessary details. On the other hand, LtS has low confidence on most fine details, resulting in large blurry sections that become lost during the vectorization postprocessing, as shown in Figure 7. Our proposed method preserves these fine-grained details, producing clean outputs without need of postprocessing.

5.2 Perceptual User Study

We perform two perceptual user studies for a quantitative analysis on additional test data that is not part of our unsupervised training set. For both studies, we process 99 images with both our approach and LtS. Like the comparison with the state of the art, for a fair comparison, we perform the postprocessing of Simo-Serra et al. (2016) with the default parameters for all output images. In the first study, we randomly show the output of both approaches side by side to 15 users and ask them to choose the better result of the two. In the second study, we show both the input rough sketch and a simplified sketch randomly processed by one of the approaches and ask them to rate the conversion on a scale of 1 to 5. The participants were university students between 21 and 36 years of age with a 1:2 female-to-male ratio. Five of the participants did illustration as a hobby and all participants had familiarity with illustration.

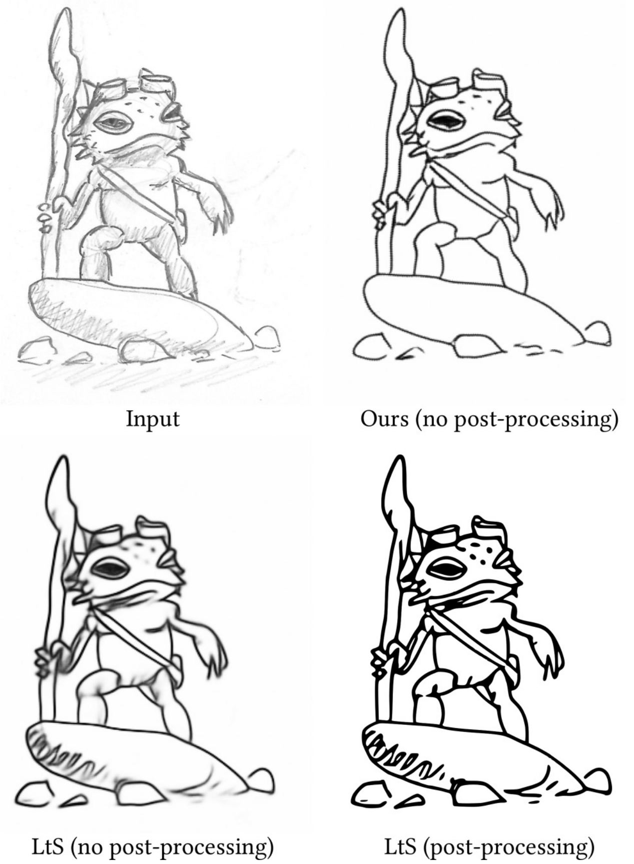
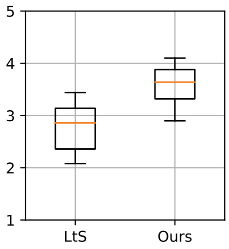

Input
Ours (no post-processing)
LtS (no post-processing)
LtS (post-processing)

Fig. 7. We compare the output of our model with that of LtS (Simo-Serra et al. 2016) with and without post-processing. We can see that LtS uses postprocessing to attempt to clean the heavily blurred image, while our approach can directly output clean line drawings.

The order of the images shown is randomized for every user and in the case of the side-by-side comparison, the presentation order is also randomized. Users are told to take their time deciding and specifically to look for multiple overlapping strokes being properly simplified into single strokes, loss of detail in the output, and noise in the output images. We note that 94 of the 99 images used for evaluation come from artists neither in our supervised nor unsupervised set. Furthermore, 60 of the images come from Twitter and are representative of challenging images “in the wild,” many taken with cellphone cameras. Evaluation results are shown in Figure 8.



(a) Absolute rating.

	LtS	Ours
absolute	2.77	3.60
vs LtS	-	88.9%
vs Ours	11.1%	-

(b) Mean results for all users.

Fig. 8. Results of the user study in which we evaluate the state of the art of LtS (Simo-Serra et al. 2016) and our approach on both absolute (1-5 scale) and relative (which result is better?) scales. The red line in the box plot indicates the median of the data, the box spans from the lower quartile to the upper quartile, and the whiskers show the range of the data.

In the absolute evaluation we can see that, while both approaches are scored fairly high, our approach obtains 0.84 points above the state of the art on a scale of 1 to 5. We compare distribution of scores with a dependent t-test and obtain a p-value of 1.42×10^{-9}, indicating that results are significantly different. In the relative evaluation, our approach is preferred to the state of the art 88.9% of the time. This highlights the importance of using adversarial augmentation to obtain more realistic sketch outputs, avoiding blurry or ill-defined areas. From the example images, we can see that the LtS model, in general, tends to miss complicated areas that it cannot fully parse while our approach produces more well-defined outputs. Note that both network architectures are exactly the same: only the learning process and thus the weight values change. Additional qualitative examples are shown in Figure 6.

5.3 Pencil-Drawing Generation

We also apply our proposed approach to the inverse problem of sketch simplification, that is, pencil-drawing generation. We swap the input and output of the training data used for sketch simplification and train new models. However, unlike sketch simplification, it turns out that it is not possible to obtain realistic results without supervised adversarial training: the output becomes a blurred version of the input. Only by introducing the adversarial augmentation framework is it possible to learn to produce realistic pencil sketches. We train three models: one with the MSE loss and two with adversarial augmentation for different artists. MSE loss and Artist 1 models are trained on 22 image pairs, while the Artist 2 model is trained on 80 image pairs. We do not augment the training data with unsupervised examples, as we only have training pairs for both artists. Results are shown in Figure 9. We can see how the adversarial augmentation is critical in obtaining realistic outputs and not just a blurred version of the input. Furthermore, by training on different artists, we seem to obtain models that capture each artist's personality and nuances. Additional results are shown in Figure 10.

We also provide a comparison with the approach of Gatys et al. (2016) for pencil-drawing generation in Figure 11, which optimizes the output image to match the style of a target image. We initialize (Gatys et al. 2016) with the input image and run the optimization until convergence. As the style target image, we use one of the images for training our approach. We note that this approach is unable to generate a realistic pencil drawing and takes 3 minutes for a single image while our approach generates convincing results and runs in well under a second.

5.4 Generalizing with Unsupervised Data

One of the main advantages of our approach is the ability to exploit unsupervised data. This is very beneficial, as acquiring matching pairs of rough sketches and simplified sketches is very time consuming and laborious. Furthermore, it is hard to obtain examples from many different illustrators to teach the model to simplify a wide variety of styles. We train a model using the supervised adversarial loss, i.e., without unsupervised data, by setting $\beta = 0$ and compare it with our full model using unsupervised data in Figure 12. We can see a clear benefit in images fairly different from those in the training data, indicating better generalization of the model. In contrast to our approach, existing approaches are unable to benefit from a mix of supervised and unsupervised data.

5.5 Single-Image Optimization

As another extension of our framework, we introduce single-image optimization. Since we are able to directly use unsupervised data, it seems natural to use the test set with the adversarial augmentation framework to optimize the model for the test data. Note that this is done in the test time and does not involve any privileged information, as the test set is used in a fully unsupervised manner. We test this approach using a single additional image and optimizing the network for this image. Optimization is done by using the adversarial augmentation from Equation (6) with $\alpha = 0, \rho_y \subset \rho_{x,y}$, with ρ_x consisting of the single test image. The other hyperparameters are set to the same values as used for sketch simplification. Results are shown in Figure 13. We can see how optimizing results on a single test image can provide a further increase in accuracy, particularly when considering very hard images. In particular, in the left image, using the pretrained model leads to a nonrecognizable output, as there is very little contrast in the input image. We do note, however, that this procedure leads to inference times a few orders of magnitude slower than using a pretrained network.

5.6 Comparison with Conditional GAN

We also perform a qualitative comparison with the recent CGAN approach as an alternative learning scheme. As in the other comparisons, the CGAN is pretrained using the model of Simo-Serra et al. (2016). The training data is the same as our model when using only supervised data, the difference lies in the loss. The CGAN model uses a loss based on Equation (4), while the supervised model uses Equation (5). The discriminator network of the CGAN model uses both the rough sketch x and the simplified sketch y as an input; in our approach, D uses only the simplified sketch y. We note that we found the CGAN model to be much more unstable during training, several times becoming completely unstable, forcing us to redo the training. This is likely caused by using only the GAN loss, in contrast with our model, which also uses the MSE loss for training stability.

Results are shown in Figure 14. We can see that the CGAN approach is able to produce nonblurry, crisp lines thanks to the GAN loss; however, it fails at simplifying the input image and adds

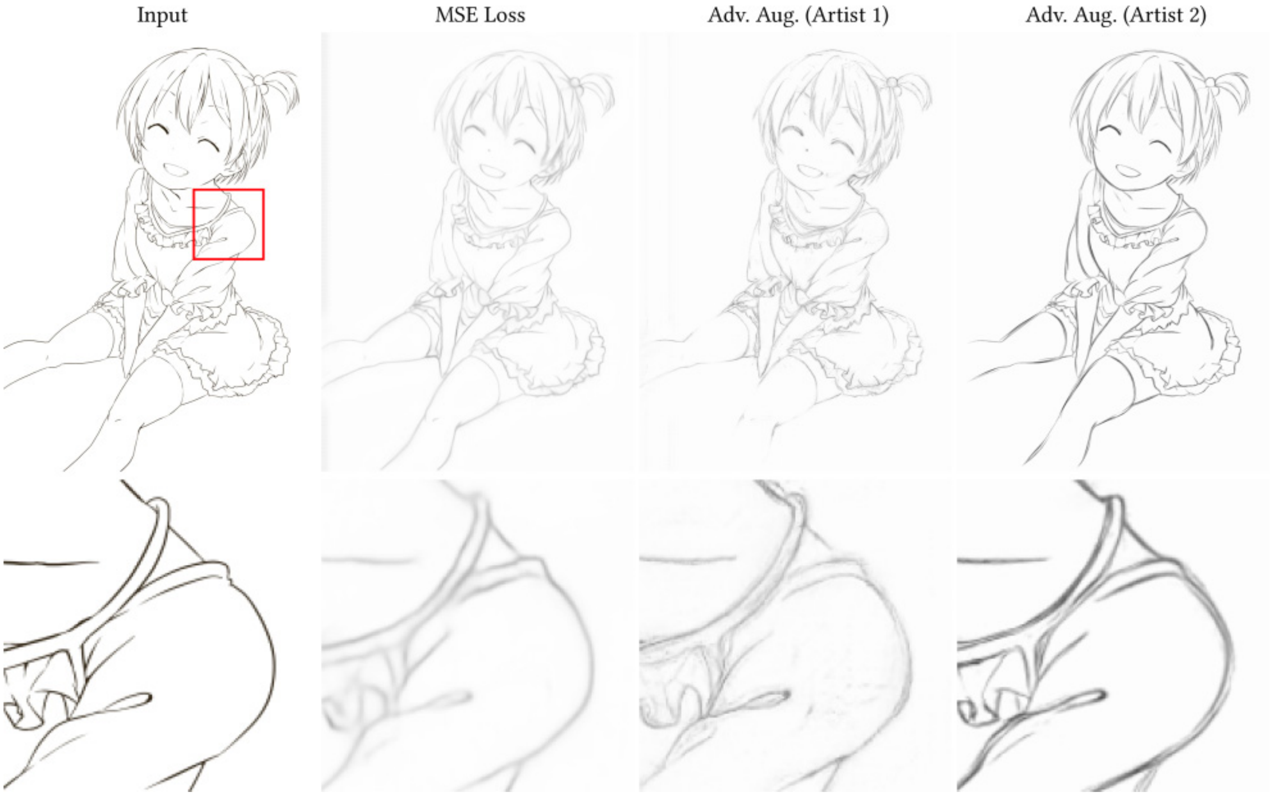


Fig. 9. Examples of pencil-drawing generation with our training framework. We compare three models: one trained with the standard MSE loss and two models trained with adversarial augmentation using data from two different artists. In the first column, we show the input to all three models, followed by the outputs of each model. The first row shows the entire image while the bottom row shows the area highlighted in red in the zoomed input image. We can see that the MSE loss succeeds only in blurring the input image while the two models trained with adversarial augmentation are able to show realistic pencil drawings. We also show how training on data from different artists gives significantly different results. Artist 1 tends to add lots of smudge marks even far away from the lines, while Artist 2 uses many overlapping lines to give the shape and form to the drawing.

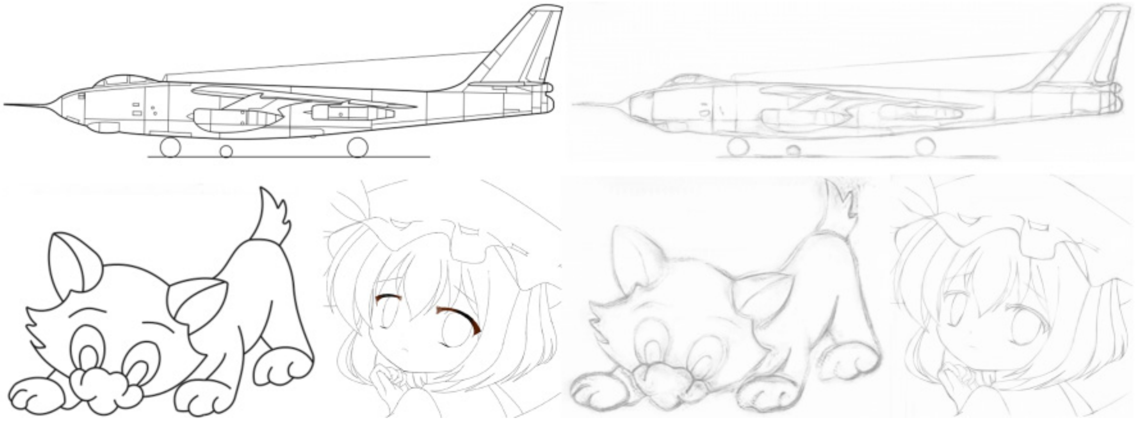


Fig. 10. More examples of pencil-drawing generation. The line drawings on the left are automatically converted to the pencil drawings on the right.

additional artifacts. This is likely caused by the GAN loss itself, as it is a very unstable loss prone to artifacting. Our approach, on the other hand, uses a different loss that allows training with unsupervised data while maintaining training stability and coherency to the output images.

5.7 Discussion and Limitations

While our approach can make great use of unsupervised data, it still has an important dependency on high-quality supervised data, without which it would not be possible to obtain good results. As an extreme case, we train a model without supervised data and

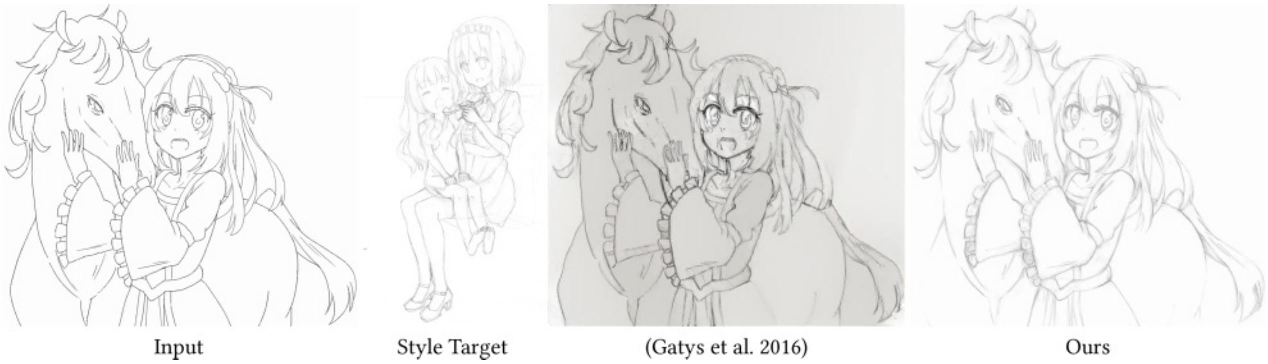


Fig. 11. Comparison with the approach of Gatys et al. (2016) for pencil-drawing generation. One of the images for training our approach is used as the target style image for the approach of Gatys et al. (2016).

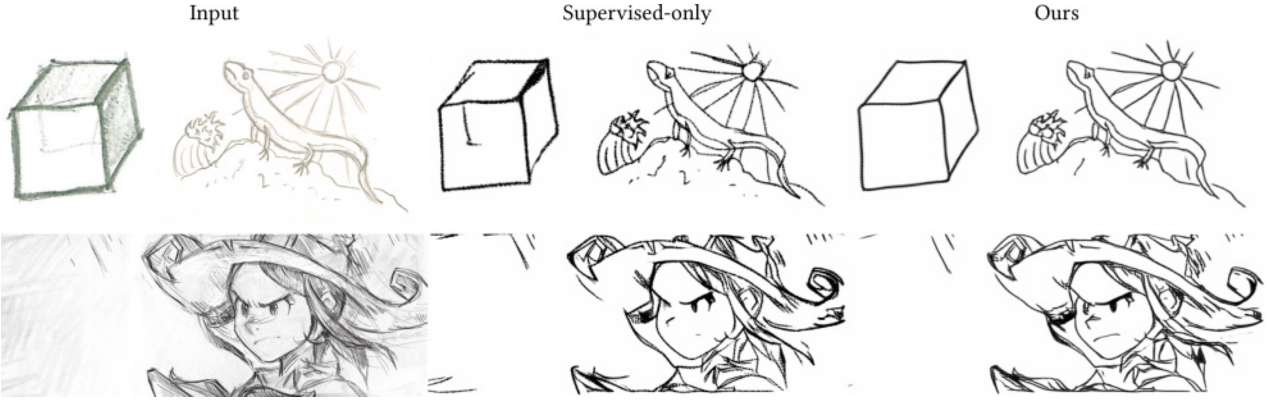


Fig. 12. Visualization of the benefits of using additional unsupervised data for training with our approach. For rough sketches fairly different from those in the training data, we can see a clear benefit when using additional unsupervised data. Note that this data, in contrast with supervised data, is simple to obtain. We note that other approaches, such as the CGAN, are unable to use unsupervised data in training. The bottom image is copyrighted by David Revoy (www.davidrevoy.com) and licensed under CC-by 4.0.

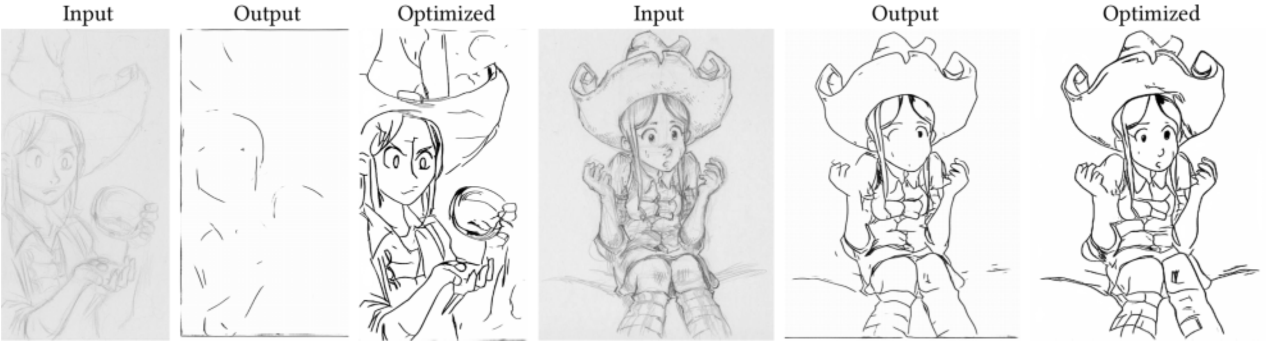


Fig. 13. Single-image optimization. We show examples of images in which our proposed model does not obtain very good results and optimize our model for these single images in an unsupervised fashion. This optimization process allows adaptation of the model to new data without annotations. Output is shown without postprocessing. The images are copyrighted by David Revoy (www.davidrevoy.com) and licensed under CC-by 4.0.

show results in Figure 15. Note that this model uses the initial weights of the LtS model, without which it would not be possible to train it. While the output images do look like line drawings, they have lost any coherency with the input rough sketch. Recent fully unsupervised approaches (Taigman et al. 2017) provide better results for certain problems in which the input and output have high degrees of similarity, but the lack of supervised guidance generally does not allow for preserving fine details.

Another limitation on the approach is that the model has difficulty removing shading from the input image, instead preserving

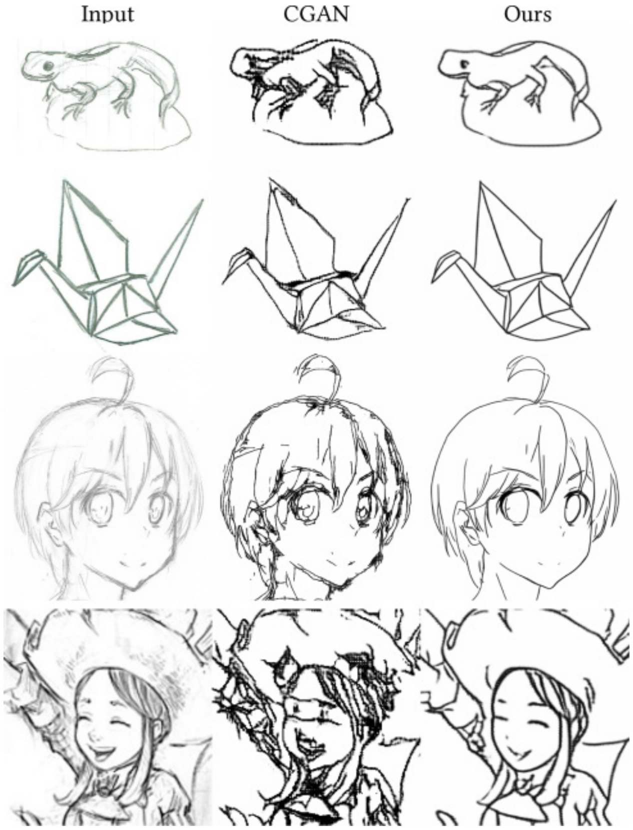


Fig. 14. Comparison of our approach with the Conditional GAN approach. Output is shown without postprocessing. The bottom image is copyrighted by David Revoy (www.davidrevoy.com) and licensed under CC-by 4.0.

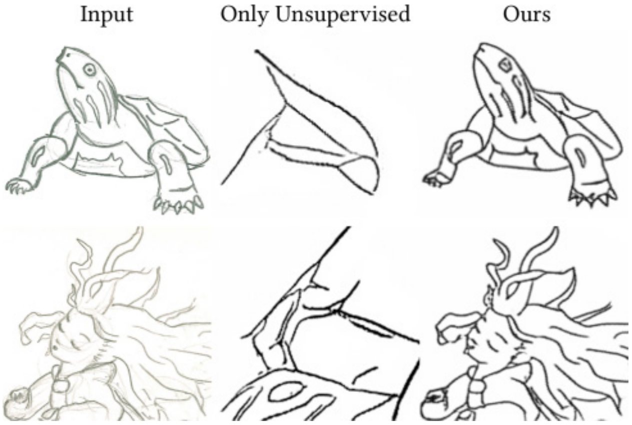


Fig. 15. Comparison of our approach with and without supervised data. Output is shown without postprocessing. With only unsupervised data, the output loses its coherency with the input and ends up looking like abstract line drawings.

the unnecessary lines in the output, as shown in Figure 16. Distinguishing between shading and lines is a complicated task that also depends heavily on the drawing style. However, it is likely that this can be mitigated or even eliminated by using additional training data containing large amounts of shading.

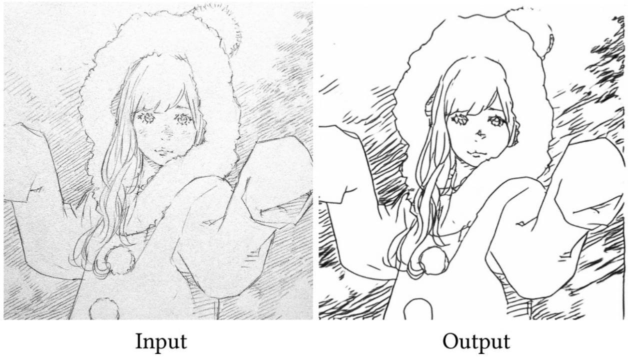


Fig. 16. Limitation of our approach when handling images with large amounts of pencil shading. Output is shown without postprocessing. The model is unable to distinguish between the shading and the lines and ends up preserving superfluous shading. Image copyrighted by Eisaku Kubonoichi.

While the network predicts each pixel by using a large area of the input image, roughly a 200×200 pixel area, it is unable to take advantage of information outside that area. Grouping of strokes can be explained by the gestalt phenomena of visual perception (Wertheimer 1923), such as the law of proximity and law of continuity, which depend on only a small local region. However, other laws, such as the law of closure, which suggests that humans tend to perceptually group strokes together when they form closed shapes, is based on nonlocal information that can be approximated only with a fully convolutional network. For large strokes or drawn regions, the model will be unable to use the full region information, which can lead to erroneous sketch simplifications of closed regions.

6 CONCLUSIONS

We have presented the adversarial augmentation for structured prediction and applied it to the sketch simplification task as well as its inverse problem, i.e., pencil-drawing generation. We show that augmenting the loss of a sketch simplification network with an adversarial network leads to more realistic outputs. Furthermore, our framework allows for unsupervised data augmentation, essential for structured prediction tasks in which obtaining additional annotated training data is very costly. As adversarial augmentation applies only to the training, the resulting models have exactly the same inference properties as the nonaugmented versions. As a further extension of the problem, we show that the framework can also be used to optimize for a single input for situations in which accuracy is valued more than quick computation. This can be used, for example, to personalize the model to different artists using only unsupervised rough and clean training data from each particular artist.

REFERENCES

Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. 2008. ILoveSketch: As-natural-as-possible sketching system for creating 3D curve models. In *ACM Symposium on User Interface Software and Technology.* 151–160.

Itamar Berger, Ariel Shamir, Moshe Mahler, Elizabeth Carter, and Jessica Hodgins. 2013. Style and abstraction in portrait sketching. *ACM Transactions on Graphics* 32, 4, 55.

Jiazhou Chen, Gal Guennebaud, Pascal Barla, and Xavier Granier. 2013. Non-oriented MLS gradient fields. *Computer Graphics Forum* 32, 8, 98–109.

Chao Dong, C. C. Loy, Kaiming He, and Xiaoou Tang. 2016. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 2, 295–307.

Alexey Dosovitskiy and Thomas Brox. 2016. Generating images with perceptual similarity metrics based on deep networks. In *Conference on Neural Information Processing Systems*.

Jean-Dominique Favreau, Florent Lafarge, and Adrien Bousseau. 2016. Fidelity vs. simplicity: A global approach to line drawing vectorization. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 35, 4.

Jakub Fišer, Paul Asente, Stephen Schiller, and Daniel Sýkora. 2015. ShipShape: A drawing beautification assistant. In *Workshop on Sketch-Based Interfaces and Modeling*. 49–57.

Kunihiko Fukushima. 1988. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks* 1, 2, 119–130.

Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Conference on Neural Information Processing Systems*.

Cindy Grimm and Pushkar Joshi. 2012. Just drawIt: A 3D sketching system. In *International Symposium on Sketch-Based Interfaces and Modeling*. 121–130.

Xavier Hilaire and Karl Tombre. 2006. Robust and accurate vectorization of line drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 6, 890–904.

Takeo Igarashi, Satoshi Matsuoka, Sachiko Kawachiya, and Hidehiko Tanaka. 1997. Interactive beautification: A technique for rapid geometric design. In *ACM Symposium on User Interface Software and Technology*. 105–114. http://doi.acm.org/10.1145/263407.263525

Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 2016. Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 35, 4.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2017. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Henry Kang, Seungyong Lee, and Charles K. Chui. 2007. Coherent line drawing. In *International Symposium on Non-Photorealistic Animation and Rendering*. 43–50.

Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Computation* 1, 4, 541–551.

Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Chuan Li and Michael Wand. 2016. Precomputed real-time texture synthesis with Markovian generative adversarial networks. In *European Conference on Computer Vision*.

David Lindlbauer, Michael Haller, Mark S. Hancock, Stacey D. Scott, and Wolfgang Stuerzlinger. 2013. Perceptual grouping: Selection assistance for digital sketching. In *International Conference on Interactive Tabletops and Surfaces*. 51–60.

Xueting Liu, Tien-Tsin Wong, and Pheng-Ann Heng. 2015. Closure-aware sketch simplification. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 34, 6, 168:1–168:10.

Cewu Lu, Li Xu, and Jiaya Jia. 2012. Combining sketch and tone for pencil drawing production. In *International Symposium on Non-Photorealistic Animation and Rendering*. 65–73.

Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. In *Conference on Neural Image Processing Deep Learning Workshop*.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *International Conference on Machine Learning*. 807–814.

Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. 2015. Learning deconvolution network for semantic segmentation. In *International Conference on Computer Vision*.

Gioacchino Noris, Alexander Hornung, Robert W. Sumner, Maryann Simmons, and Markus Gross. 2013. Topology-driven vectorization of clean line drawings. *ACM Transactions on Graphics* 32, 1, 4:1–4:11.

Günay Orbay and Levent Burak Kara. 2011. Beautification of design sketches using trainable stroke clustering and curve fitting. *IEEE Transactions on Visualization and Computer Graphics* 17, 5, 694–708.

Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. 2016. Context encoders: Feature learning by inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323, 533–536.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Conference on Neural Information Processing Systems*.

Peter Selinger. 2003. Potrace: A polygon-based tracing algorithm. *Potrace (online)*. Retrieved November 16, 2017 from http://potrace.sourceforge.net/potrace.pdf (2009-07-01)

Amit Shesh and Baoquan Chen. 2008. Efficient and dynamic simplification of line drawings. *Computer Graphics Forum* 27, 2, 537–545. DOI: https://doi.org/10.1111/j.1467-8659.2008.01151.x

Edgar Simo-Serra, Satoshi Iizuka, Kazuma Sasaki, and Hiroshi Ishikawa. 2016. Learning to simplify: Fully convolutional networks for rough sketch cleanup. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 35, 4.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15.1929–1958.

Yaniv Taigman, Adam Polyak, and Lior Wolf. 2017. Unsupervised cross-domain image generation. In *International Conference on Learning Representations*.

Xiaolong Wang and Abhinav Gupta. 2016. Generative image modeling using style and structure adversarial networks. In *European Conference on Computer Vision*.

Max Wertheimer. 1923. Untersuchungen zur Lehre von der Gestalt, II. *Psychologische Forschung* 4, 301–350.

Donggeun Yoo, Namil Kim, Sunggyun Park, Anthony S. Paek, and In So Kweon. 2016. Pixel-level domain transfer. In *European Conference on Computer Vision*.

Matthew D. Zeiler. 2012. ADADELTA: An adaptive learning rate method. *arXiv Preprint arXiv:1212.5701*.

Yipin Zhou and Tamara L. Berg. 2016. Learning temporal transformations from time-lapse videos. In *European Conference on Computer Vision*.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision*.

Received April 2017; revised October 2017; accepted October 2017