



Image-to-Image Translation with Conditional Adversarial Networks

Phillip Isola

Jun-Yan Zhu

Tinghui Zhou

Alexei A. Efros

Berkeley AI Research (BAIR) Laboratory

26 Nov 2018



CONTENT

目錄

01 /

INTRODUCTION

02 /

RELATED WORKS

03 /

PROPOSED METHOD

04 /

EXPERIMENTAL RESULT
&CONCLUSION



n1

INTRODUCTION




Abstract

We investigate **conditional adversarial networks** as a **general-purpose** solution to image-to-image translation problems.

These networks not only learn the mapping from input image to output image, but also learn a loss function to train this mapping. This makes it possible to apply the same generic approach to problems that traditionally would require very different loss formulations.

We demonstrate its **wide applicability and ease of adoption without the need for parameter tweaking.** As a community, we no longer hand-engineer our mapping functions, and this work suggests we can achieve reasonable results without hand-engineering our loss functions either

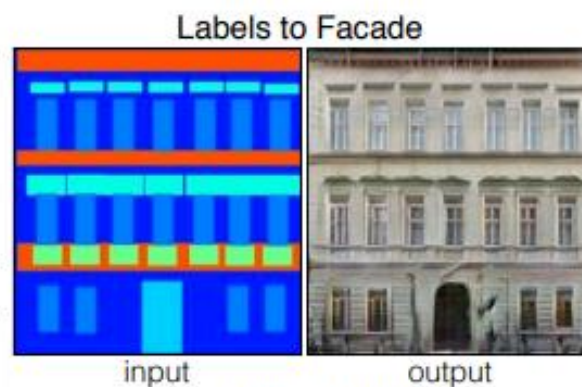


Problems in image processing

Labels to Street Scene

Labels to Facade

BW to Color



Aerial to Map

Day to Night


Edges to Photo



Our goal in this paper

we define automatic image-to-image translation **as the task of translating one possible representation of a scene into another**, given sufficient training data.

Traditionally, each of these tasks has been tackled with **separate, special-purpose machinery**, despite the fact that the setting is always the same: predict pixels from pixels. Our goal in this paper is to **develop a common framework for all these problems**






CNNs

Although the learning process is automatic, a lot of manual effort still goes into designing effective losses.

In other words, we still **have to tell the CNN what we wish it to minimize**. But, just like King Midas, we must be careful what we wish for!

If we take a naive approach and ask the CNN to minimize the Euclidean distance between predicted and ground truth pixels, it will tend to produce blurry results. This is because **Euclidean distance is minimized by averaging all plausible outputs, which causes blurring**.






GANs

It would be highly desirable if we could instead specify only a high-level goal, like “make the output indistinguishable from reality”, and then automatically learn a loss function appropriate for satisfying this goal. Fortunately, this is exactly what is done by the recently proposed **Generative Adversarial Networks (GANs)** .

GANs learn a loss that tries to classify if the output image is real or fake, while simultaneously training a generative model to minimize this loss. Blurry images will not be tolerated since they look obviously fake. Because **GANs learn a loss that adapts to the data, they can be applied to a multitude of tasks that traditionally would require very different kinds of loss functions**



cGANs

In this paper, we explore **GANs in the conditional setting**. Just as GANs learn a generative model of data, conditional GANs (cGANs) learn a conditional generative model. This makes cGANs suitable for image-to-image translation tasks, where we condition on an input image and generate a corresponding output image.

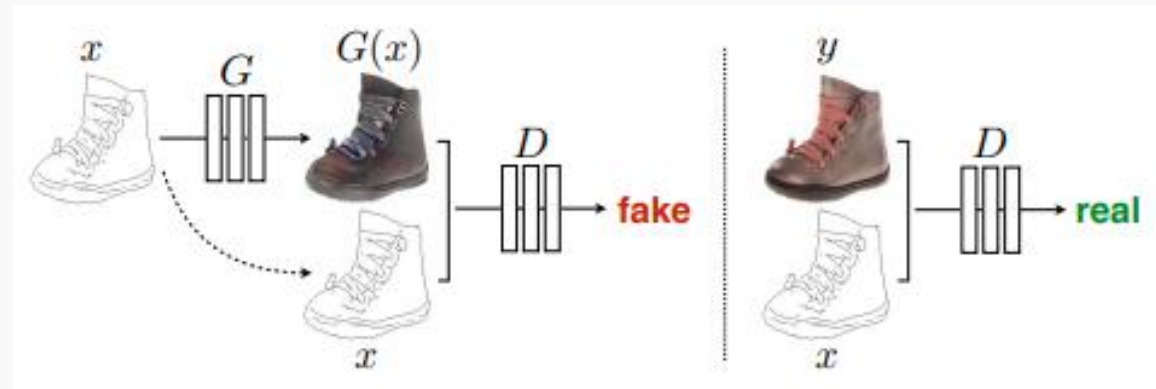


Figure 2: Training a conditional GAN to map **edges**→**photo**. The discriminator, D , learns to classify between fake (synthesized by the generator) and real {edge, photo} tuples. The generator, G , learns to fool the discriminator. Unlike an unconditional GAN, both the generator and discriminator **observe the input edge map**.



RELATED WORKS




Structured Losses

Structured losses for image modeling Image-to-image translation problems are often formulated as **per-pixel classification or regression**.

These formulations treat the output space as “**unstructured**” in the sense that **each output pixel is considered conditionally independent from all others** given the input image. Conditional GANs instead **learn a structured loss**. Structured losses **penalize the joint configuration of the output**.

A large body of literature has considered losses of this kind, with methods including **conditional random fields, the SSIM metric , feature matching , nonparametric losses , the convolutional pseudo-prior , and losses based on matching covariance statistics .**

The conditional GAN is different in that the loss is learned, and can, in theory, **penalize any possible structure that differs between output and target**






Conditional GANs & U-Net & patchGAN

We are not the first to apply GANs in the conditional setting. Each of the prior methods was tailored for a specific application. Our framework differs in that nothing is **applicationspecific**. This makes our setup considerably simpler than most others.

Our method also differs from the prior works in several architectural choices for the generator and discriminator. Unlike past work, for our generator we use a “**U-Net**”-based architecture, and for our discriminator we use a convolutional “**PatchGAN**” classifier, which only penalizes structure at the scale of image patches. A similar PatchGAN architecture was previously proposed in to **capture local style statistics**. Here we show that this approach is effective on a wider range of problems, and we investigate the effect of changing the patch size.





nr

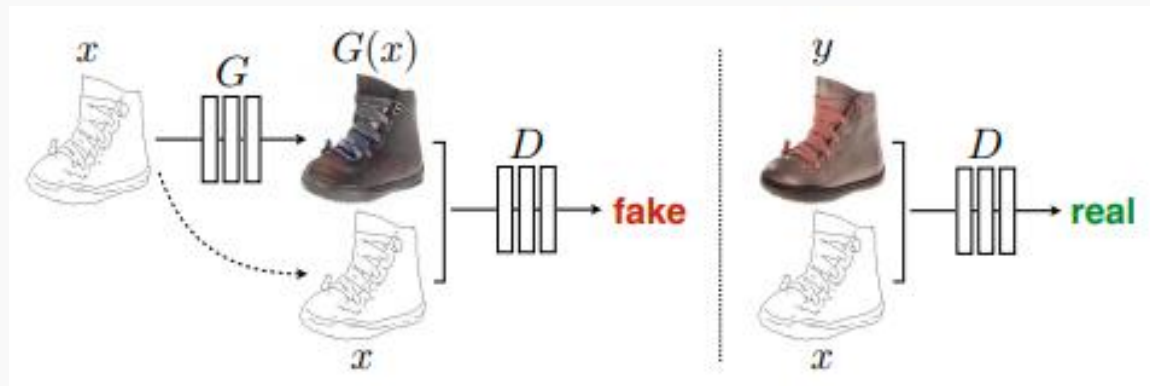
PROPOSED METHOD

GAN v.s cGAN

GANs are generative models that learn a mapping from **random noise vector z to output image y** , $G : z \rightarrow y$.

In contrast, conditional GANs learn a mapping from **observed image x and random noise vector z** , to y , $G : \{x, z\} \rightarrow y$.

The generator G is trained to produce outputs that cannot be distinguished from “real” images by an adversarially trained discriminator, D , which is trained to do as well as possible at detecting the generator’s “fakes”. This training procedure is diagrammed in Figure 2.



Objective

The objective of a conditional GAN can be expressed as

$$\mathcal{L}_{cGAN}(\mathbf{G}, \mathbf{D}) = \mathbb{E}_{x, y}[\log \mathbf{D}(x, y)] + \mathbb{E}_{x, z}[\log(1 - \mathbf{D}(x, \mathbf{G}(x, z)))] \quad (1)$$

where \mathbf{G} tries to minimize this objective against an adversarial \mathbf{D} that tries to maximize it, i.e.

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D).$$

Objective

To test the importance of conditioning the discriminator, we also compare to an unconditional variant in which the **discriminator does not observe x** :

$$\mathcal{L}_{GAN}(\mathbf{G}, \mathbf{D}) = \mathbb{E}_y[\log \mathbf{D}(y)] + \mathbb{E}_{x, z}[\log(1 - \mathbf{D}(\mathbf{G}(x, z)))] \quad (2)$$

the generator is tasked to not only fool the discriminator but also to be near the ground truth output in an L2 sense. We also explore this option, **using L1 distance rather than L2 as L1 encourages less blurring**:

$$\mathcal{L}_{L1}(\mathbf{G}) = \mathbb{E}_{x, y, z}[\|y - \mathbf{G}(x, z)\|_1] \quad (3)$$

Objective

Final objective is

$$\mathbf{G}^* = \arg \min_{\mathbf{G}} \max_{\mathbf{D}} \mathcal{L}_{cGAN}(\mathbf{G}, \mathbf{D}) + \lambda \mathcal{L}_{L1}(\mathbf{G}) . \quad (4)$$

Without z , the net could still learn a mapping from x to y , but would produce **deterministic outputs**, and therefore **fail to match any distribution other than a delta function**. Past conditional GANs have acknowledged this and provided **Gaussian noise z as an input to the generator**, in addition to x . In initial experiments, we did not find this strategy effective – the generator simply learned to ignore the noise – which is consistent with Mathieu et al. Instead, for our final models, we provide noise **only in the form of dropout, applied on several layers of our generator at both training and test time**. Despite the dropout noise, we observe only **minor stochasticity** in the output of our nets. Designing conditional GANs that produce highly stochastic output, and thereby capture the full entropy of the conditional distributions they model, is an important question left open by the present work




Network architectures

Both generator and discriminator use modules of the form **convolution-BatchNorm-ReLu**

A defining feature of image-to-image translation problems is that they **map a high resolution input grid to a high resolution output grid**.

In addition, for the problems we consider, the input and output differ in surface appearance, but both are renderings of the same underlying structure. Therefore, structure in the **input is roughly aligned with structure in the output**. We design the generator architecture around these considerations.






Network architectures

For many image translation problems, there is a great deal of **low-level information** shared between the input and output, and it would be desirable to shuttle this information directly across the net.

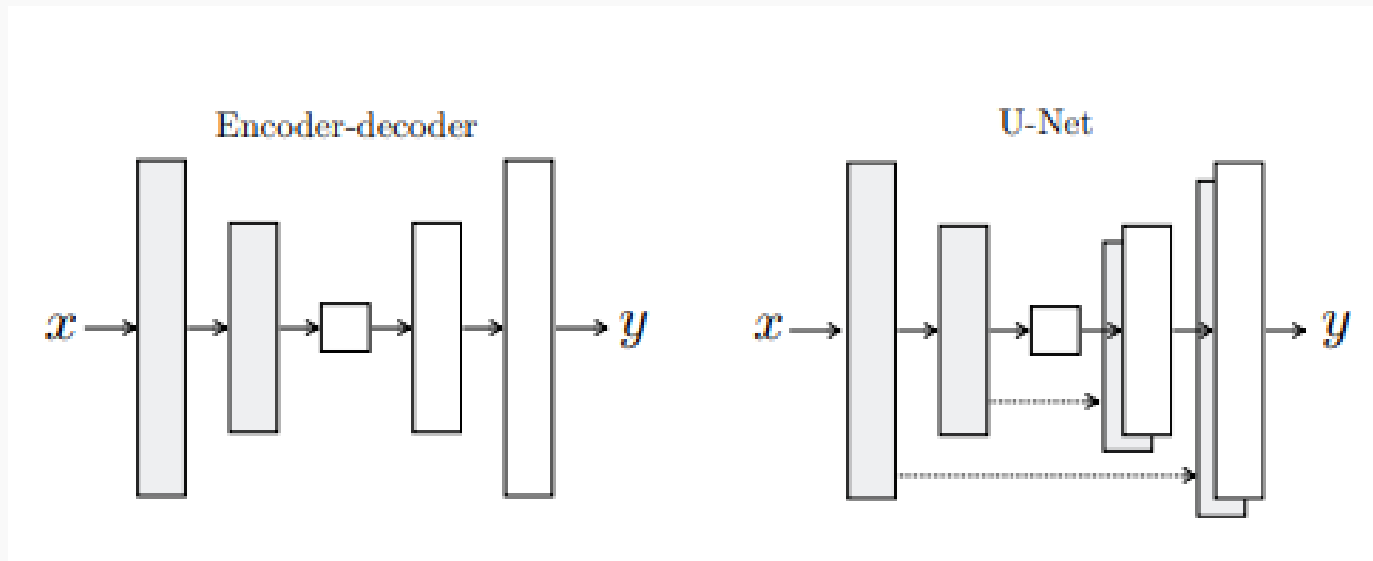
For example, in the case of image **colorization**, the input and output share the **location of prominent edges**

To give the generator a means to circumvent the bottleneck for information like this, we add skip connections, following the general shape of a “**U-Net**”. Specifically, we add **skip connections between each layer i and layer $n - i$** , where n is the total number of layers. Each skip connection simply concatenates all channels at layer i with those at layer $n - i$



Network architectures

Adding skip connections between each layer i and layer $n - i$, where n is the total number of layers. Each skip connection simply concatenates all channels at layer i with those at layer $n - i$



Two choices for the architecture of the generator.

The “U-Net” is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.

Network architectures

It is well known that the L2 loss – and L1, see Figure 4 – produces blurry results on image generation problems. Although these losses fail to encourage high-frequency crispness, in many cases they nonetheless accurately capture the low frequencies. For problems where this is the case, we do not need an entirely new framework to enforce correctness at the low frequencies. L1 will already do.



Network architectures



Figure 4 shows the qualitative effects of these variations on two labels \rightarrow photo problems. L1 alone leads to reasonable but blurry results. The cGAN alone (setting $\lambda = 0$ in Eqn. 4) gives much sharper results but introduces visual artifacts on certain applications. Adding both terms together (with $\lambda = 100$) reduces these artifacts

$$\mathbf{G}^* = \arg \min_{\mathbf{G}} \max_{\mathbf{D}} \mathcal{L}_{cGAN}(\mathbf{G}, \mathbf{D}) + \lambda \mathcal{L}_{L1}(\mathbf{G}) . \quad (4)$$

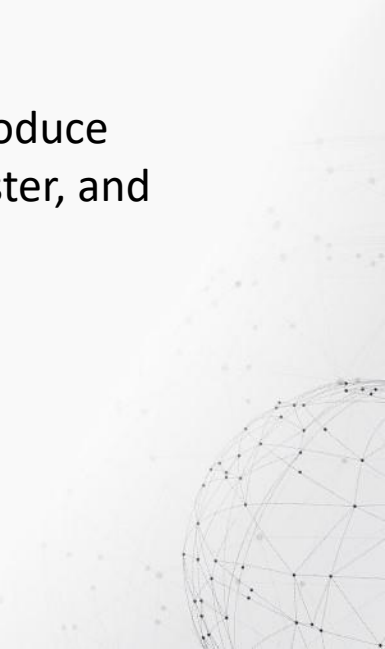


Network architectures

This motivates restricting the GAN discriminator to only model high-frequency structure, relying on an L1 term to force low-frequency correctness (Eqn. 4). In order to model high-frequencies, it is sufficient to restrict our attention to the structure in local image patches.

Therefore, we design a discriminator architecture – which we term a PatchGAN – that only penalizes structure at the scale of patches. This discriminator tries to classify if each $N \times N$ patch in an image is real or fake. We run this discriminator convolutionally across the image, averaging all responses to provide the ultimate output of D.


In Section 4.4, we demonstrate that N can be much smaller than the full size of the image and still produce high quality results. This is advantageous because a smaller PatchGAN has fewer parameters, runs faster, and can be applied to arbitrarily large images.





Optimization

To optimize our networks, we follow the standard approach from : we alternate between one gradient descent step on D , then one step on G . As suggested in the original GAN paper, rather than training G to minimize $\log(1 - D(x, G(x, z)))$, we instead train to **maximize $\log D(x, G(x, z))$** . In addition, **we divide the objective by 2 while optimizing D** , which **slows down the rate at which D learns relative to G** . We use minibatch SGD and apply the Adam solver , with a learning rate of 0.0002, and momentum parameters $\beta_1 = 0.5$, $\beta_2 = 0.999$.





n4

EXPERIMENTAL RESULT
& CONCLUSION

Experiments



Figure 5: Adding skip connections to an encoder-decoder to create a “U-Net” results in much higher quality results.

Experiments

Loss	Per-pixel acc.	Per-class acc.	Class IOU
L1	0.42	0.15	0.11
GAN	0.22	0.05	0.01
cGAN	0.57	0.22	0.16
L1+GAN	0.64	0.20	0.15
L1+cGAN	0.66	0.23	0.17
Ground truth	0.80	0.26	0.21

Table 1: FCN-scores for different losses, evaluated on Cityscapes labels↔photos.

Loss	Per-pixel acc.	Per-class acc.	Class IOU
Encoder-decoder (L1)	0.35	0.12	0.08
Encoder-decoder (L1+cGAN)	0.29	0.09	0.05
U-net (L1)	0.48	0.18	0.13
U-net (L1+cGAN)	0.55	0.20	0.14

Table 2: FCN-scores for different generator architectures (and objectives), evaluated on Cityscapes labels↔photos. (U-net (L1- cGAN) scores differ from those reported in other tables since batch size was 10 for this experiment and 1 for other tables, and random variation between training runs.)

Experiments



Figure 6: Patch size variations. Uncertainty in the output manifests itself differently for different loss functions. Uncertain regions become blurry and desaturated under L1.

The 1x1 PixelGAN encourages greater color diversity but has no effect on spatial statistics.

The 16x16 PatchGAN creates locally sharp results, but also leads to tiling artifacts beyond the scale it can observe.

The 70x70 PatchGAN forces outputs that are sharp, even if incorrect, in both the spatial and spectral (colorfulness) dimensions.

The full 286x286 ImageGAN produces results that are visually similar to the 70x70 PatchGAN, but somewhat lower quality according to our FCN-score metric (Table 3).

Experiments

Discriminator receptive field	Per-pixel acc.	Per-class acc.	Class IOU
1×1	0.39	0.15	0.10
16×16	0.65	0.21	0.17
70×70	0.66	0.23	0.17
286×286	0.42	0.16	0.11

Table 3: FCN-scores for different receptive field sizes of the discriminator, evaluated on Cityscapes labels→photos. Note that input images are 256×256 pixels and larger receptive fields are padded with zeros.

This may be because the ImageGAN has many more parameters and greater depth than the 70×70 PatchGAN, and may be harder to train.

Experiments

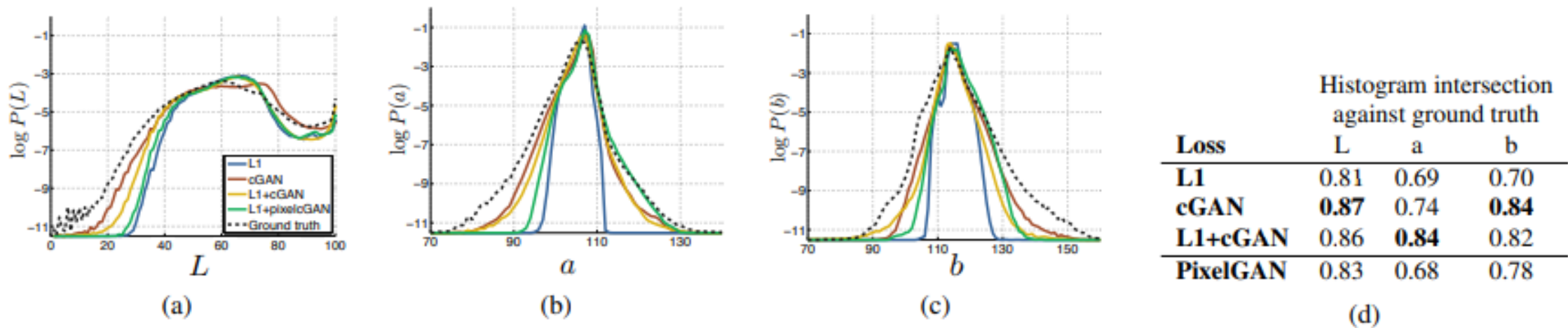


Figure 7: Color distribution matching property of the cGAN, tested on Cityscapes. (c.f. Figure 1 of the original GAN paper). Note that the histogram intersection scores are dominated by differences in the high probability region, which are imperceptible in the plots, which show log probability and therefore emphasize differences in the low probability regions

Experiments

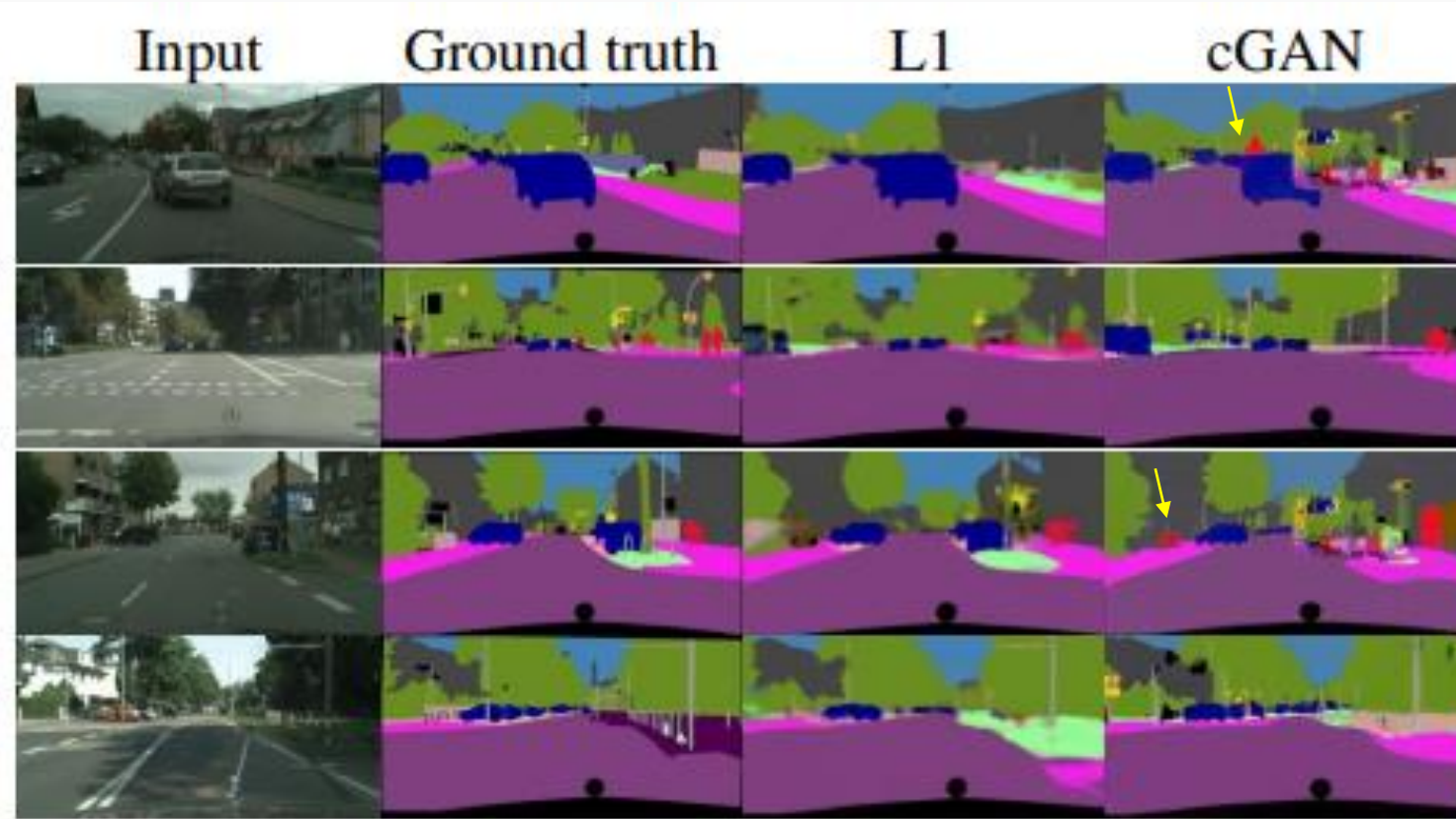



Figure 10: Applying a conditional GAN to semantic segmentation. The cGAN produces sharp images that look at glance like the ground truth, but in fact include **many small, hallucinated** objects.



Conclusion

The results in this paper suggest that conditional adversarial networks are a promising approach for many image-to-image translation tasks, especially those involving **highly structured** graphical outputs.

These networks learn a loss adapted to the task and data at hand, which makes them applicable in a wide variety of settings.



Other Examples

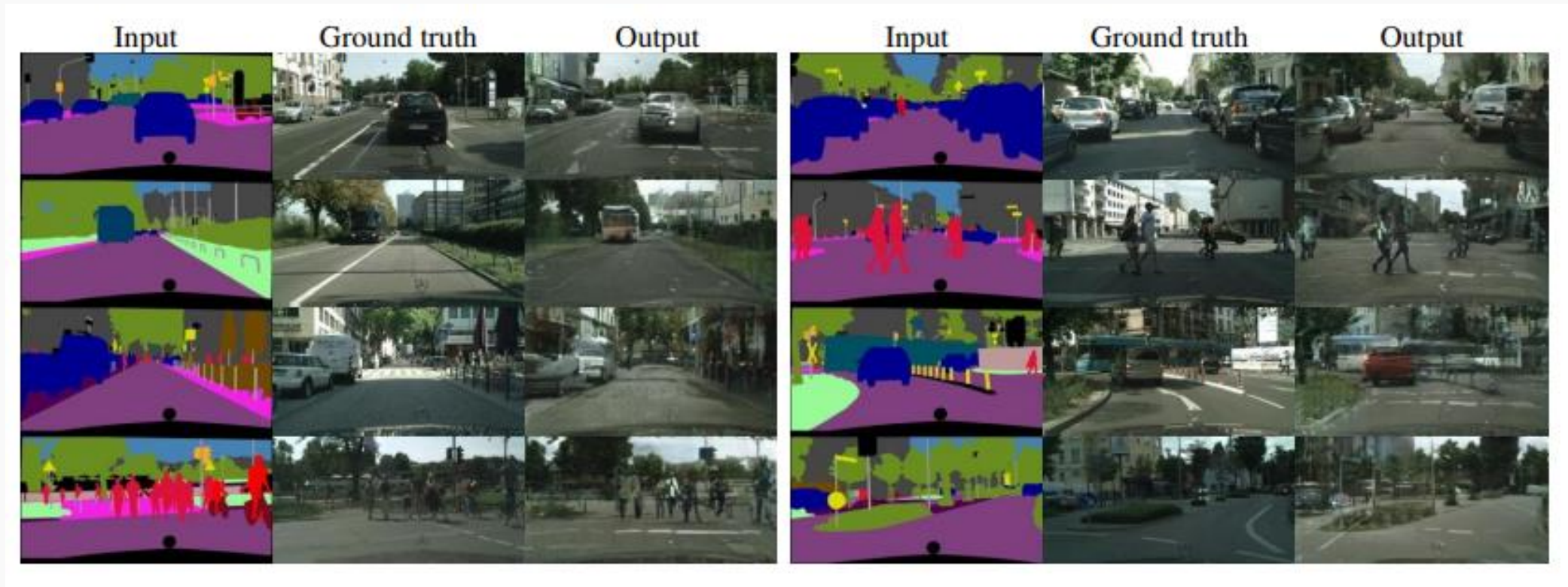


Figure 13: Example results of our method on Cityscapes labels→photo, compared to ground truth

Other Examples

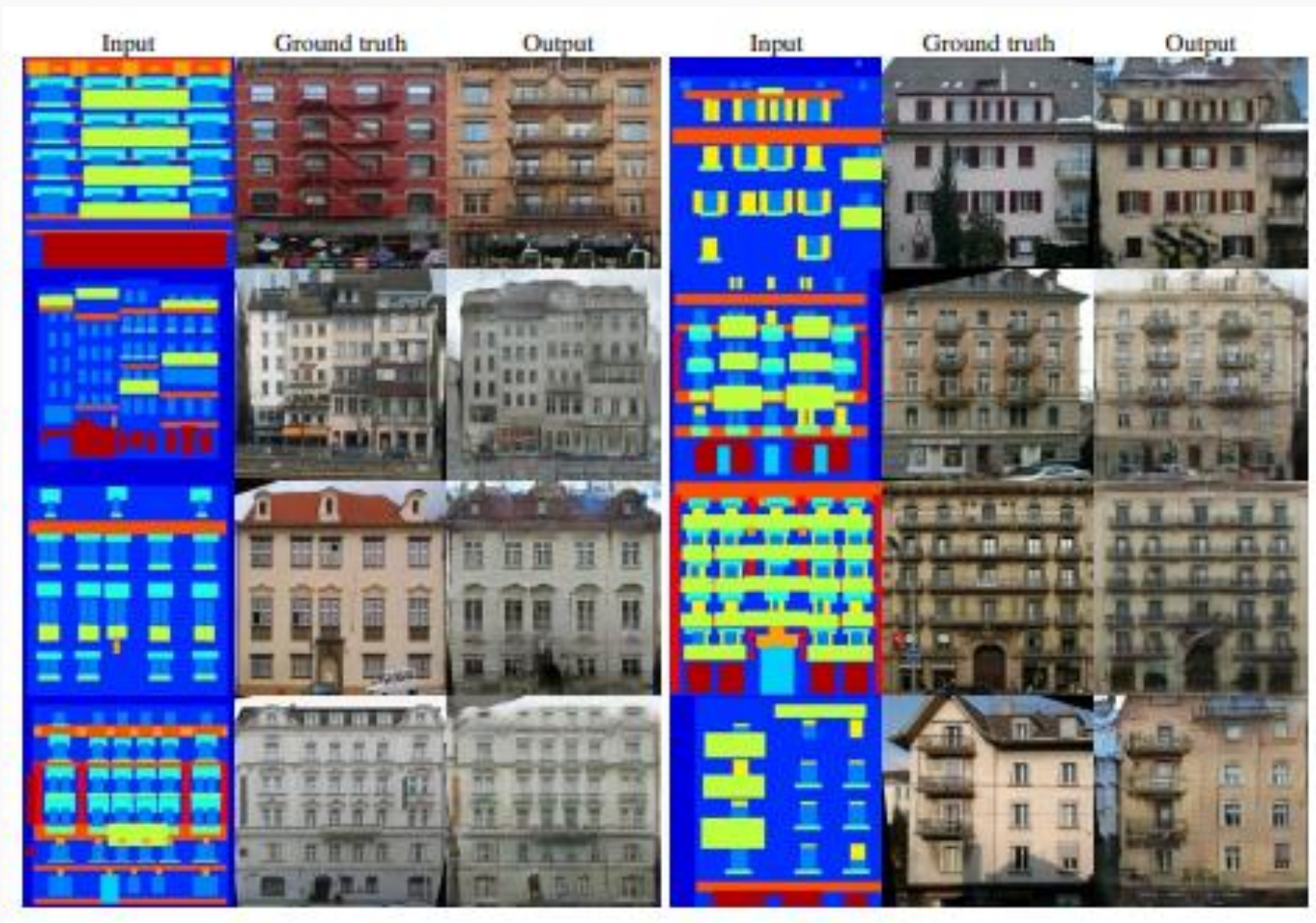


Figure 14: Example results of our method on facades labels→photo, compared to ground truth.

Other Examples



Figure 15: Example results of our method on day→night, compared to ground truth.

Other Examples



Figure 16: Example results of our method on automatically detected edges→handbags, compared to ground truth.