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.

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 *Patch*GAN – 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.

Such a discriminator effectively models the image as a Markov random field, assuming independence between pixels separated by more than a patch diameter. This connection was previously explored in [38], and is also the common assumption in models of texture [17, 21] and style [16, 25, 22, 37]. Therefore, our PatchGAN can be understood as a form of texture/style loss.

3.3. Optimization and inference

To optimize our networks, we follow the standard approach from [24]: 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 $\log D(x, G(x, z))$ [24]. 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 [32], with a learning rate of 0.0002, and momentum parameters $\beta_1 = 0.5$, $\beta_2 = 0.999$.

At inference time, we run the generator net in exactly the same manner as during the training phase. This differs from the usual protocol in that we apply dropout at test time, and we apply batch normalization [29] using the statistics of the test batch, rather than aggregated statistics of the training batch. This approach to batch normalization, when the batch size is set to 1, has been termed "instance normalization" and has been demonstrated to be effective at image generation tasks [54]. In our experiments, we use batch sizes between 1 and 10 depending on the experiment.

4. Experiments

Results Starts with a detailed description of all the experiments ran, and the datasets used. Note the naming is "Experiments", not "Results".

To explore the generality of conditional GANs, we test the method on a variety of tasks and datasets, including both graphics tasks, like photo generation, and vision tasks, like semantic segmentation:

- *Semantic labels↔photo*, trained on the Cityscapes dataset [12].
- *Architectural labels→photo*, trained on CMP Facades [45].
- *Map↔aerial photo*, trained on data scraped from Google Maps.
- *BW→color photos*, trained on [51].
- *Edges→photo*, trained on data from [65] and [60]; binary edges generated using the HED edge detector [58] plus postprocessing.
- *Sketch→photo*: tests edges→photo models on human-drawn sketches from [19].
- *Day→night*, trained on [33].
- *Thermal→color photos*, trained on data from [27].
- *Photo with missing pixels→inpainted photo*, trained on Paris StreetView from [14].

Experiment and dataset used

Details of training on each of these datasets are provided in the supplemental materials online. In all cases, the input and output are simply 1-3 channel images. Qualitative results are shown in Figures 8, 9, 11, 10, 13, 14, 15, 16, 17, 18, 19, 20. Several failure cases are highlighted in Figure 21. More comprehensive results are available at `https://phillipi.github.io/pix2pix/`.

Data requirements and speed We note that decent results can often be obtained even on small datasets. Our facade training set consists of just 400 images (see results in Figure 14), and the day to night training set consists of only 91 unique webcams (see results in Figure 15). On datasets of this size, training can be very fast: for example, the results shown in Figure 14 took less than two hours of training on a single Pascal Titan X GPU. At test time, all models run in well under a second on this GPU.

4.1. Evaluation metrics

Evaluating the quality of synthesized images is an open and difficult problem [52]. Traditional metrics such as per-pixel mean-squared error do not assess joint statistics of the result, and therefore do not measure the very structure that structured losses aim to capture.

To more holistically evaluate the visual quality of our results, we employ two tactics. First, we run "real vs. fake" perceptual studies on Amazon Mechanical Turk (AMT). For graphics problems like colorization and photo generation, plausibility to a human observer is often the ultimate goal. Therefore, we test our map generation, aerial photo generation, and image colorization using this approach.

Since this work is algorithmic, they outlined the evaluation methods before presenting results, giving readers an idea what to expect.

Figure 4: Different losses induce different quality of results. Each column shows results trained under a different loss. Please see `https://phillipi.github.io/pix2pix/` for additional examples.

Second, we measure whether or not our synthesized cityscapes are realistic enough that off-the-shelf recognition system can recognize the objects in them. This metric is similar to the "inception score" from [52], the object detection evaluation in [55], and the "semantic interpretability" measures in [62] and [42].

AMT perceptual studies For our AMT experiments, we followed the protocol from [62]: Turkers were presented with a series of trials that pitted a "real" image against a "fake" image generated by our algorithm. On each trial, each image appeared for 1 second, after which the images disappeared and Turkers were given unlimited time to respond as to which was fake. The first 10 images of each session were practice and Turkers were given feedback. No feedback was provided on the 40 trials of the main experiment. Each session tested just one algorithm at a time, and Turkers were not allowed to complete more than one session. $\sim$ 50 Turkers evaluated each algorithm. Unlike [62], we did not include vigilance trials. For our colorization experiments, the real and fake images were generated from the same grayscale input. For map$\leftrightarrow$aerial photo, the real and fake images were not generated from the same input, in order to make the task more difficult and avoid floor-level results. For map$\leftrightarrow$aerial photo, we trained on 256×256 resolution images, but exploited fully-convolutional translation (described above) to test on 512×512 images, which were then downsampled and presented to Turkers at 256×256 resolution. For colorization, we trained and tested on 256×256 resolution images and presented the results to Turkers at this same resolution.

"FCN-score" While quantitative evaluation of generative models is known to be challenging, recent works [52, 55, 62, 42] have tried using pre-trained semantic classifiers to measure the discriminability of the generated stimuli as a pseudo-metric. The intuition is that if the generated images are realistic, classifiers trained on real images will be able to classify the synthesized image correctly as well. To this end, we adopt the popular FCN-8s [39] architecture for semantic segmentation, and train it on the cityscapes dataset. We then score synthesized photos by the classification accuracy against the labels these photos were synthesized from.

4.2. Analysis of the objective function

Which components of the objective in Eqn. 4 are important? We run ablation studies to isolate the effect of the L1 term, the GAN term, and to compare using a discriminator conditioned on the input (cGAN, Eqn. 1) against using an unconditional discriminator (GAN, Eqn. 2).

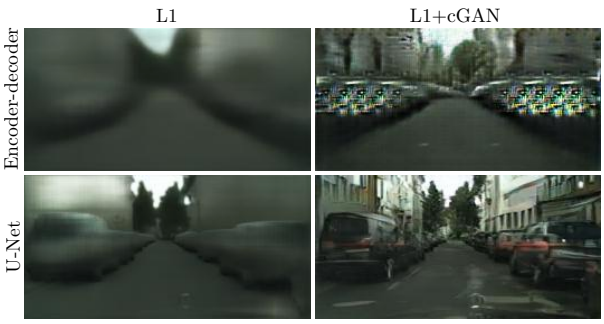

Figure caption stands alone and describes the experiment and the major finding

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

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 presents quantitative results over prior work.

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.)

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.

Figure 4 shows the qualitative effects of these variations on two labels→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.

We quantify these observations using the FCN-score on the cityscapes labels→photo task (Table 1): the GAN-based objectives achieve higher scores, indicating that the synthesized images include more recognizable structure. We also test the effect of removing conditioning from the discriminator (labeled as GAN). In this case, the loss does not penalize mismatch between the input and output; it only cares that the output look realistic. This variant results in poor performance; examining the results reveals that the generator collapsed into producing nearly the exact same output regardless of input photograph. Clearly, it is important, in this case, that the loss measure the quality of the match between input and output, and indeed cGAN performs much better than GAN. Note, however, that adding an L1 term also encourages that the output respect the input, since the L1 loss penalizes the distance between ground truth outputs, which correctly match the input, and synthesized outputs, which may not. Correspondingly, L1+GAN is also effective at creating realistic renderings that respect the input label maps. Combining all terms, L1+cGAN, performs similarly well.

Colorfulness A striking effect of conditional GANs is that they produce sharp images, hallucinating spatial structure even where it does not exist in the input label map. One might imagine cGANs have a similar effect on "sharpening" in the spectral dimension – i.e. making images more colorful. Just as L1 will incentivize a blur when it is uncertain where exactly to locate an edge, it will also incentivize an average, grayish color when it is uncertain which of several plausible color values a pixel should take on. Specially, L1 will be minimized by choosing the median of the conditional probability density function over possible colors. An adversarial loss, on the other hand, can in principle become aware that grayish outputs are unrealistic, and encourage matching the true color distribution [24]. In Figure 7, we investigate whether our cGANs actually achieve this effect on the Cityscapes dataset. The plots show the marginal distributions over output color values in Lab color space. The ground truth distributions are shown with a dotted line. It is apparent that L1 leads to a narrower distribution than the ground truth, confirming the hypothesis that L1 encourages average, grayish colors. Using a cGAN, on the other hand, pushes the output distribution closer to the ground truth.

4.3. Analysis of the generator architecture

A U-Net architecture allows low-level information to shortcut across the network. Does this lead to better results? Figure 5 and Table 2 compare the U-Net against an encoder-decoder on cityscape generation. The encoder-decoder is created simply by severing the skip connections in the U-Net. The encoder-decoder is unable to learn to generate realistic images in our experiments. The advantages of the U-Net appear not to be specific to conditional GANs: when both U-Net and encoder-decoder are trained with an L1 loss, the U-Net again achieves the superior results.

Each subsection outlines one major finding, and points to a Table and/or Figure with quantitative results.

Concise summary of the conclusions of the experiment. Explanation also does not repeat Figure caption. Notice there is no speculation of the conclusions.

4.4. From PixelGANs to PatchGANs to ImageGANs

We test the effect of varying the patch size N of our discriminator receptive fields, from a 1×1 "PixelGAN" to a