An Introduction to Generative Adversarial Networks (GANs)

Yu Zheng

The list of sections is given below:

1. Introduction

Generative Adversarial Networks, or GANs for short, are an approach to generative modeling using deep learning methods, such as convolutional neural networks.

Generative modeling is an **unsupervised learning task** in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that **the model can be used to generate or output new examples that plausibly could have been drawn from the original dataset**.

GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with **two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to classify examples as either real (from the domain) or fake (generated).** The two models are trained together in a zero-sum game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples.

GANs are an exciting and rapidly changing field, delivering on the promise of generative models in their ability to generate realistic examples across a range of problem domains, most notably in image-to-image translation tasks such as translating photos of summer to winter or day to night, and in generating photorealistic photos of objects, scenes, and people that even humans cannot tell are fake.

Tero Karras, et al. in their 2017 paper titled "Progressive Growing of GANs for Improved Quality, Stability, and Variation" demonstrate the generation of plausible realistic photographs of human faces. They are so real looking, in fact, that it is fair to call the result remarkable. As such, the results received a lot of media attention. The face generations were trained on celebrity examples, meaning that there are elements of existing celebrities in the generated faces, making them seem familiar, but not quite.

Examples of Photorealistic GAN-Generated Faces.Taken from Progressive Growing of GANs for Improved Quality, Stability, and Variation, 2017.

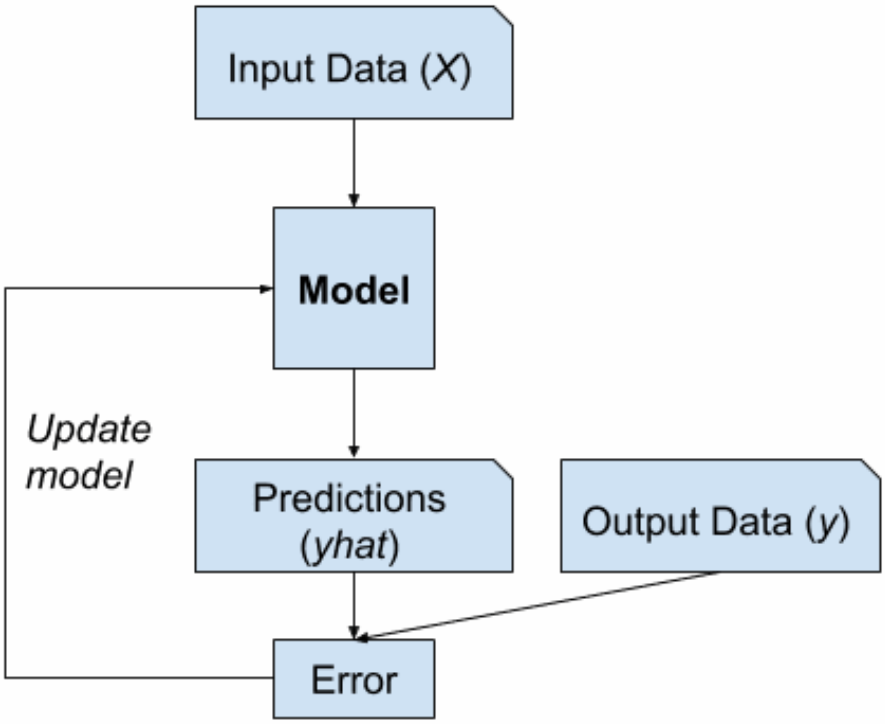
2. Generative Models

2.1 Supervised vs. Unsupervised Learning

A typical machine learning problem involves using a model to make a prediction, e.g. predictive modeling.

This requires a training dataset that is used to train a model, comprised of multiple examples, called samples, each with input variables (X) and output class labels (y). A model is trained by showing examples of inputs, having it predict outputs, and correcting the model to make the outputs more like the expected outputs.

This correction of the model is generally referred to as a supervised form of learning, or **supervised learning**.



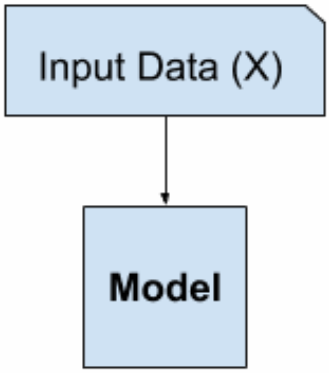
Example of Supervised Learning

Examples of supervised learning problems include classification and regression, and examples of supervised learning algorithms include logistic regression and random forest.

There is another paradigm of learning where the model is only given the input variables (X) and the problem does not have any output variables (y).

A model is constructed by extracting or summarizing the patterns in the input data. There is no correction of the model, as the model is not predicting anything.

This lack of correction is generally referred to as an unsupervised form of learning, or **unsupervised learning**.



Example of Unsupervised Learning

Examples of unsupervised learning problems include clustering and generative modeling, and examples of unsupervised learning algorithms are K-means and Generative Adversarial Networks.

2.2 Discriminative vs. Generative Modeling

In supervised learning, we may be interested in developing a model to predict a class label given an example of input variables.

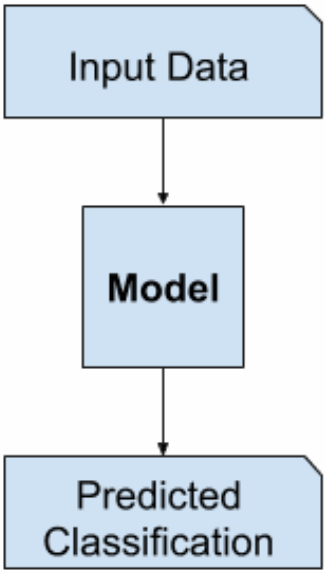
This predictive modeling task is called classification.

Classification is also traditionally referred to as **discriminative modeling**.

> … we use the training data to find a discriminant function f(x) that maps each x directly onto a class label, thereby combining the inference and decision stages into a single learning problem.

— Page 44, Pattern Recognition and Machine Learning, 2006.

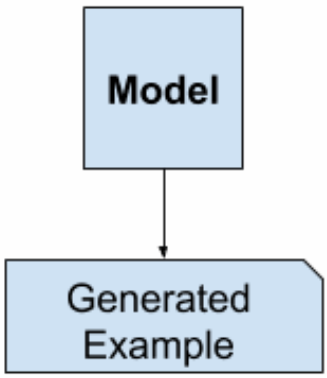
This is because a model must discriminate examples of input variables across classes; it must choose or make a decision as to what class a given example belongs.



Example of Discriminative Modeling

Alternately, unsupervised models that summarize the distribution of input variables may be able to be used to create or generate new examples in the input distribution.

As such, these types of models are referred to as generative models.



Example of Generative Modeling

For example, a single variable may have a known data distribution, such as a Gaussian distribution, or bell shape. A generative model may be able to sufficiently summarize this data distribution, and then be used to generate new variables that plausibly fit into the distribution of the input variable.

In fact, a really good generative model may be able to generate new examples that are not just plausible, but indistinguishable from real examples from the problem domain.

2.3 Examples of Generative Models

Naive Bayes is an example of a generative model that is more often used as a discriminative model.

For example, Naive Bayes works by summarizing the probability distribution of each input variable and the output class. When a prediction is made, the probability for each possible outcome is calculated for each variable, the independent probabilities are combined, and the most likely outcome is predicted. Used in reverse, the probability distributions for each variable can be sampled to generate new plausible (independent) feature values.

Other examples of generative models include Latent Dirichlet Allocation, or LDA, and the Gaussian Mixture Model, or GMM.

Deep learning methods can be used as generative models. Two popular examples include the Restricted Boltzmann Machine, or RBM, and the Deep Belief Network, or DBN.

Two modern examples of deep learning generative modeling algorithms include the Variational Autoencoder, or VAE, and the Generative Adversarial Network, or GAN.

3. Generative Adversarial Networks (GANs)

GANs, are a deep-learning-based generative model. More generally, GANs are a model architecture for training a generative model, and it is most common to use deep learning models in this architecture.

The GAN architecture was first described in the 2014 paper by Ian Goodfellow, et al. titled "Generative Adversarial Networks."

A standardized approach called Deep Convolutional Generative Adversarial Networks, or DCGAN, that led to more stable models was later formalized by Alec Radford, et al. in the 2015 paper titled "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks".

The GAN model architecture involves two sub-models: a *generator model* for generating new examples and a *discriminator model* for classifying whether generated examples are real, from the domain, or fake, generated by the generator model.

- **Generator**. Model that is used to generate new plausible examples from the problem domain.
- **Discriminator**. Model that is used to classify examples as real (*from the domain*) or fake (*generated*).

3.1 The Generator Model

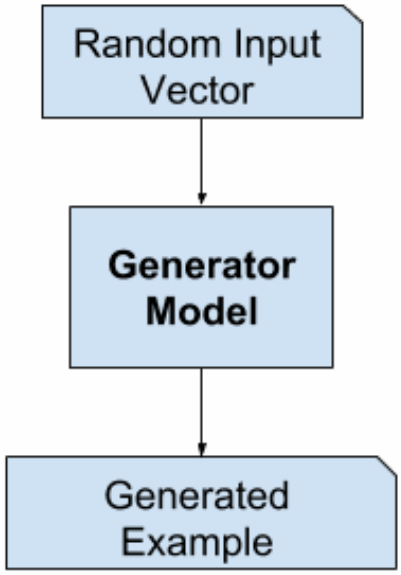
The generator model takes a fixed-length random vector as input and generates a sample in the domain.

The vector is drawn from randomly from a Gaussian distribution, and the vector is used to seed the generative process. After training, points in this multidimensional vector space will correspond to points in the problem domain, forming a compressed representation of the data distribution.

This vector space is referred to as a latent space, or a vector space comprised of latent variables. Latent variables are those variables that are important for a domain but are not directly observable.

We often refer to latent variables, or a latent space, as a projection or compression of a data distribution. That is, a latent space provides a compression or high-level concepts of the observed raw data such as the input data distribution. In the case of GANs, **the generator model applies meaning to points in a chosen latent space**, such that new points drawn from the latent space can be provided to the generator model as input and used to generate new and different output examples.

After training, the generator model is kept and used to generate new samples.



Example of the GAN Generator Model

3.2 The Discriminator Model

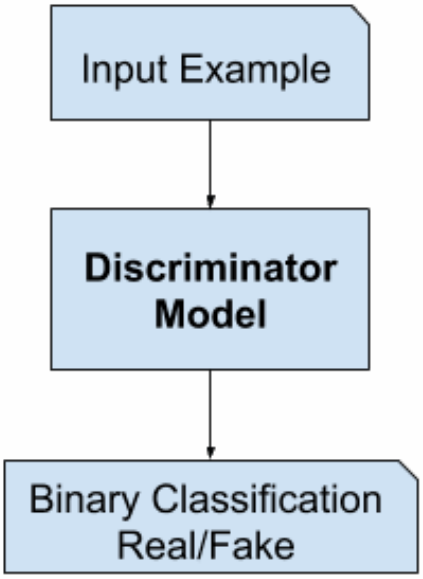
The discriminator model takes an example from the domain as input (real or generated) and predicts a binary class label of real or fake (generated).

The real example comes from the training dataset. The generated examples are output by the generator model.

The discriminator is a normal (and well understood) classification model.

After the training process, the discriminator model is discarded as we are interested in the generator.

Sometimes, the generator can be repurposed as it has learned to effectively extract features from examples in the problem domain. Some or all of the feature extraction layers can be used in transfer learning applications using the same or similar input data.



Example of the GAN Discriminator Model

3.3 GANs as a Two Player Game

Generative modeling is an unsupervised learning problem, as we discussed in the previous section, although a clever property of the GAN architecture is that the training of the generative model is framed as a supervised learning problem.

The two models, the generator and discriminator, are **trained together**:

- The generator generates a batch of samples, and these, along with real examples from the domain, are provided to the discriminator and classified as real or fake.
- The discriminator is then updated to get better at discriminating real and fake samples in the next round, and importantly, the generator is updated based on how well, or not, the generated samples fooled the discriminator.

> We can think of the generator as being like a counterfeiter, trying to make fake money, and the discriminator as being like police, trying to allow legitimate money and catch counterfeit money. To succeed in this game, the counterfeiter must learn to make money that is indistinguishable from genuine money, and the generator network must learn to create samples that are drawn from the same distribution as the training data.

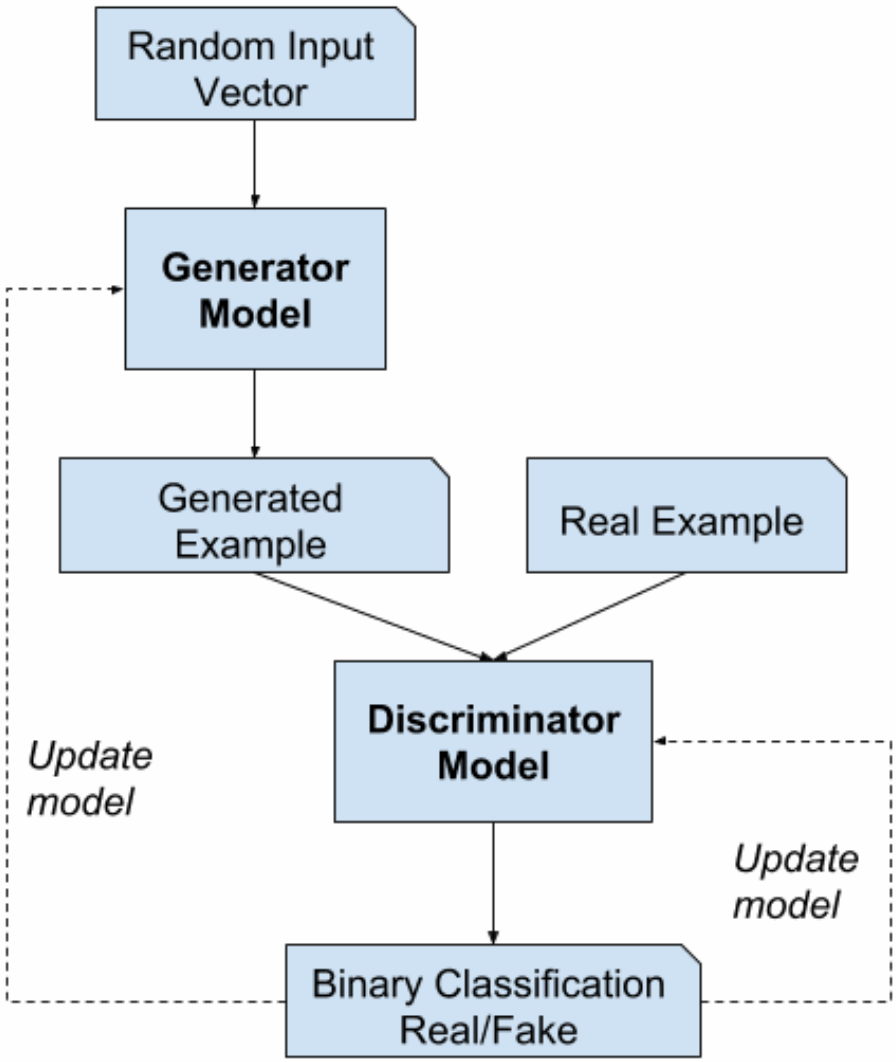
— NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.

In this way, the two models are competing against each other, they are adversarial in the game theory sense, and are playing a zero-sum game.

In this case, zero-sum means that when the discriminator successfully identifies real and fake samples, it is rewarded or no change is needed to the model parameters, whereas the generator is penalized with large updates to model parameters.

Alternately, when the generator fools the discriminator, it is rewarded, or no change is needed to the model parameters, but the discriminator is penalized and its model parameters are updated.

At a limit, the generator generates perfect replicas from the input domain every time, and the discriminator cannot tell the difference and predicts "unsure" (e.g. 50% for real and fake) in every case. This is just an example of an idealized case; we do not need to get to this point to arrive at a useful generator model.



Example of the Generative Adversarial Network Model Architecture

> [training] drives the discriminator to attempt to learn to correctly classify samples as real or fake. Simultaneously, the generator attempts to fool the classifier into believing its samples are real. At convergence, the generator's samples are indistinguishable from real data, and the discriminator outputs 1/2 everywhere. The discriminator may then be discarded.

— Page 700, Deep Learning, 2016.

3.4 Conditional GANs

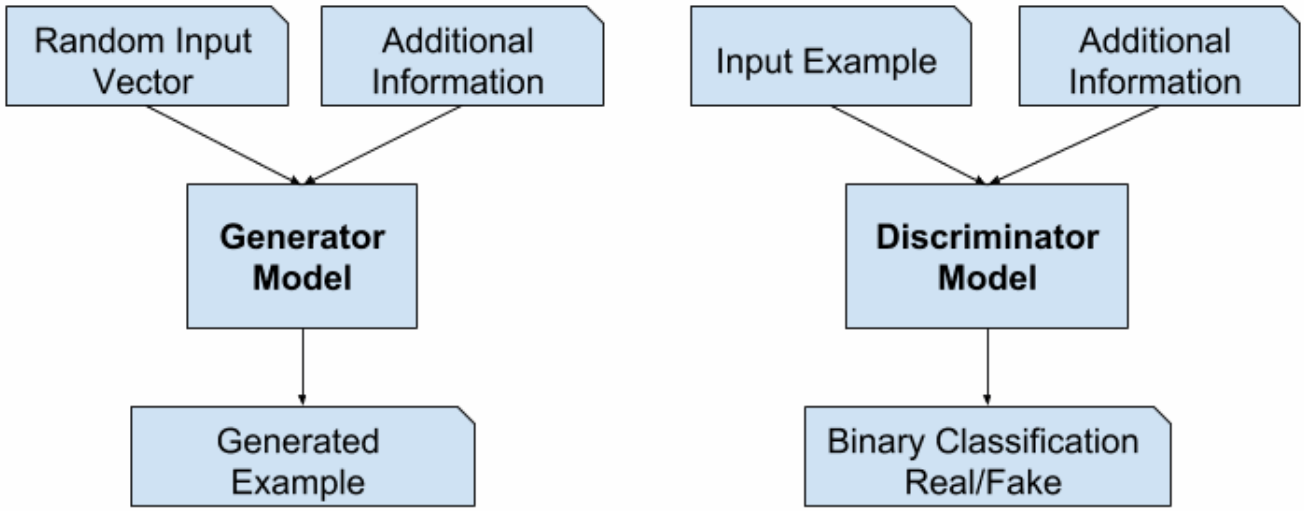
An important extension to the GAN is in their use for conditionally generating an output.

The generative model can be trained to generate new examples from the input domain, where the input, the random vector from the latent space, **is provided with (conditioned by) some additional input**.

The additional input could be a class value, such as male or female in the generation of photographs of people, or a digit, in the case of generating images of handwritten digits.

The discriminator is also conditioned, meaning that it is provided both with an input image that is either real or fake and the additional input. In the case of a classification label type conditional input, the discriminator would then expect that the input would be of that class, in turn teaching the generator to generate examples of that class in order to fool the discriminator.

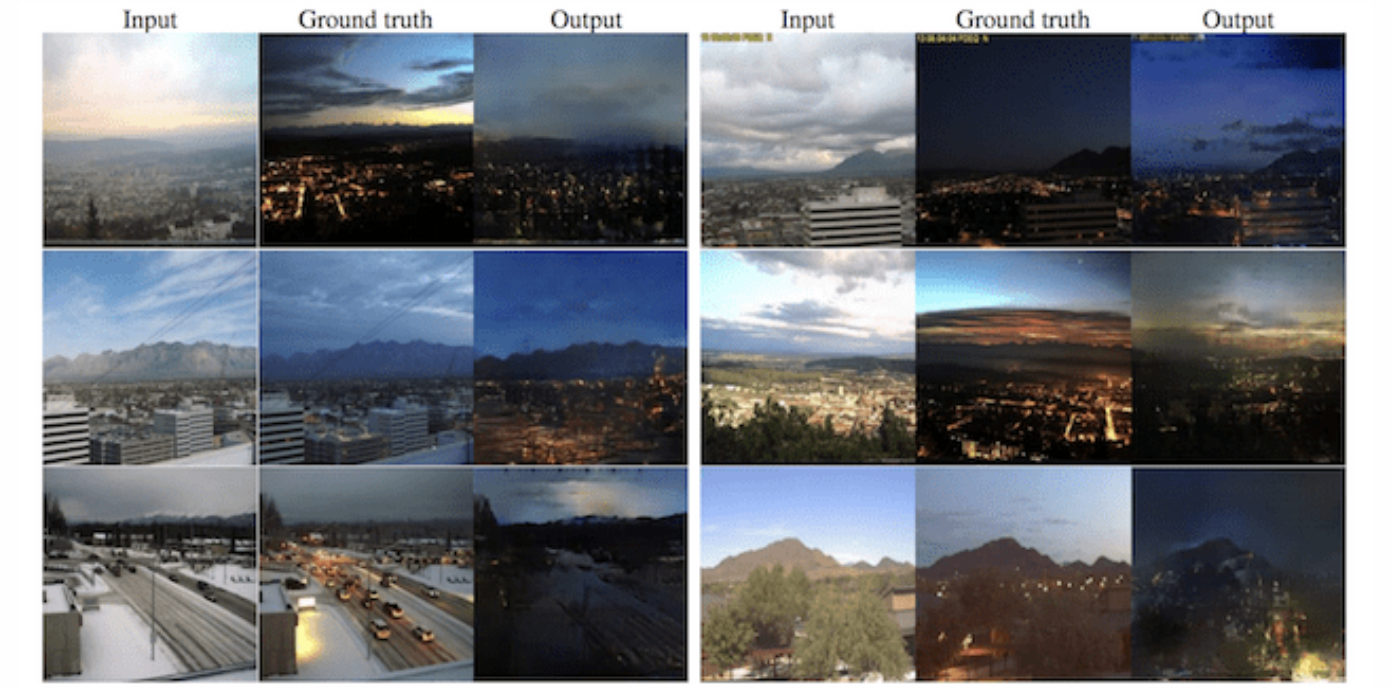
In this way, a conditional GAN can be used to generate examples from a domain of a given type.



Example of a Conditional Generative Adversarial Network Model Architecture

Taken one step further, **the GAN models can be conditioned on an example from the domain, such as an image**. This allows for applications of GANs such as text-to-image translation, or image-to-image translation. This allows for some of the more impressive applications of GANs, such as style transfer, photo colorization, transforming photos from summer to winter or day to night, and so on.

In the case of conditional GANs for image-to-image translation, such as transforming day to night, the discriminator is provided examples of real and generated nighttime photos as well as (conditioned on) real daytime photos as input. The generator is provided with a random vector from the latent space as well as (conditioned on) real daytime photos as input.



Example of Photographs of Daytime Cityscapes to Nighttime With pix2pix.Taken from Image-to-Image Translation with Conditional Adversarial Networks, 2016.

4. Why Generative Adversarial Networks?

4.1 Data Augmentation

One of the many major advancements in the use of deep learning methods in domains such as computer vision is a technique called data augmentation.

Data augmentation results in better performing models, both increasing model skill and providing a regularizing effect, reducing generalization error. It works by creating new, artificial but plausible examples from the input problem domain on which the model is trained.

The techniques are primitive in the case of image data, involving crops, flips, zooms, and other simple transforms of existing images in the training dataset.

Successful generative modeling provides an alternative and potentially more domain-specific approach for data augmentation. In fact, data augmentation is a simplified version of generative modeling, although it is rarely described this way.

4.2 GANs are Astonishing

There are many research reasons why GANs are interesting, important, and require further study. Ian Goodfellow outlines a number of these in his 2016 conference keynote and associated technical report titled "NIPS 2016 Tutorial: Generative Adversarial Networks."

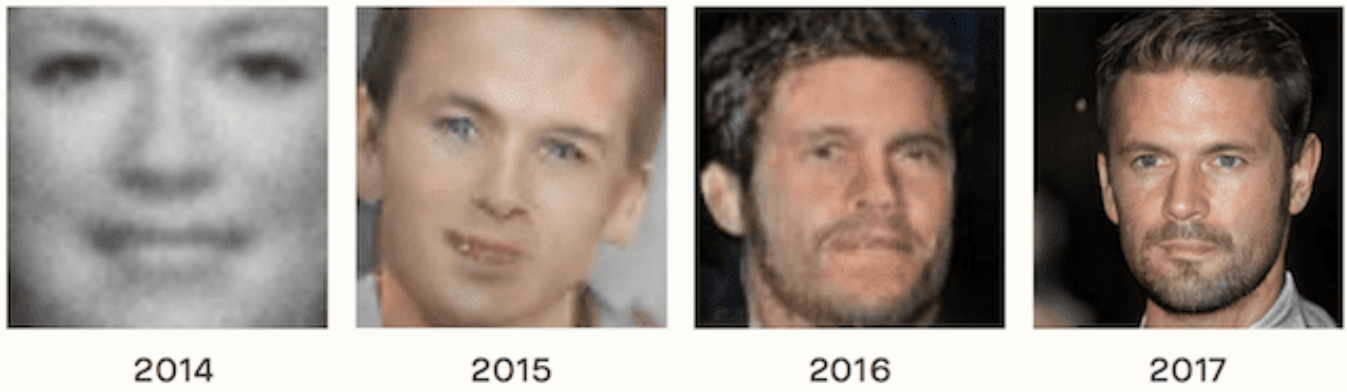
Among these reasons, he highlights GANs' successful ability to **model high-dimensional data, handle missing data, and the capacity of GANs to provide multi-modal outputs or multiple plausible answers**.

Perhaps the most compelling application of GANs is in conditional GANs for tasks that require the generation of new examples. Here, Goodfellow indicates three main examples:

- **Image Super-Resolution**. The ability to generate high-resolution versions of input images.
- **Creating Art**. The ability to great new and artistic images, sketches, painting, and more.
- **Image-to-Image Translation**. The ability to translate photographs across domains, such as day to night, summer to winter, and more.

Perhaps the most compelling reason that GANs are widely studied, developed, and used is because of their success. GANs have been able to generate photos so realistic that humans are unable to tell that they are of objects, scenes, and people that do not exist in real life.

Astonishing is not a sufficient adjective for their capability and success.



Example of the Progression in the Capabilities of GANs From 2014 to 2017. Taken from The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation, 2018.

Example of Photorealistic GAN-Generated Objects and ScenesTaken from Progressive Growing of GANs for Improved Quality, Stability, and Variation, 2017.

4.3 Some Other Applications of GANs

Generate Realistic Photographs

Andrew Brock, et al. in their 2018 paper titled "Large Scale GAN Training for High Fidelity Natural Image Synthesis" demonstrate the generation of synthetic photographs with their technique BigGAN that are practically indistinguishable from real photographs.

Image-to-Image Translation

This is a bit of a catch-all task, for those papers that present GANs that can do many image translation tasks.

Phillip Isola, et al. in their 2016 paper titled "Image-to-Image Translation with Conditional Adversarial Networks" demonstrate GANs, specifically their pix2pix approach for many image-to-image translation tasks.

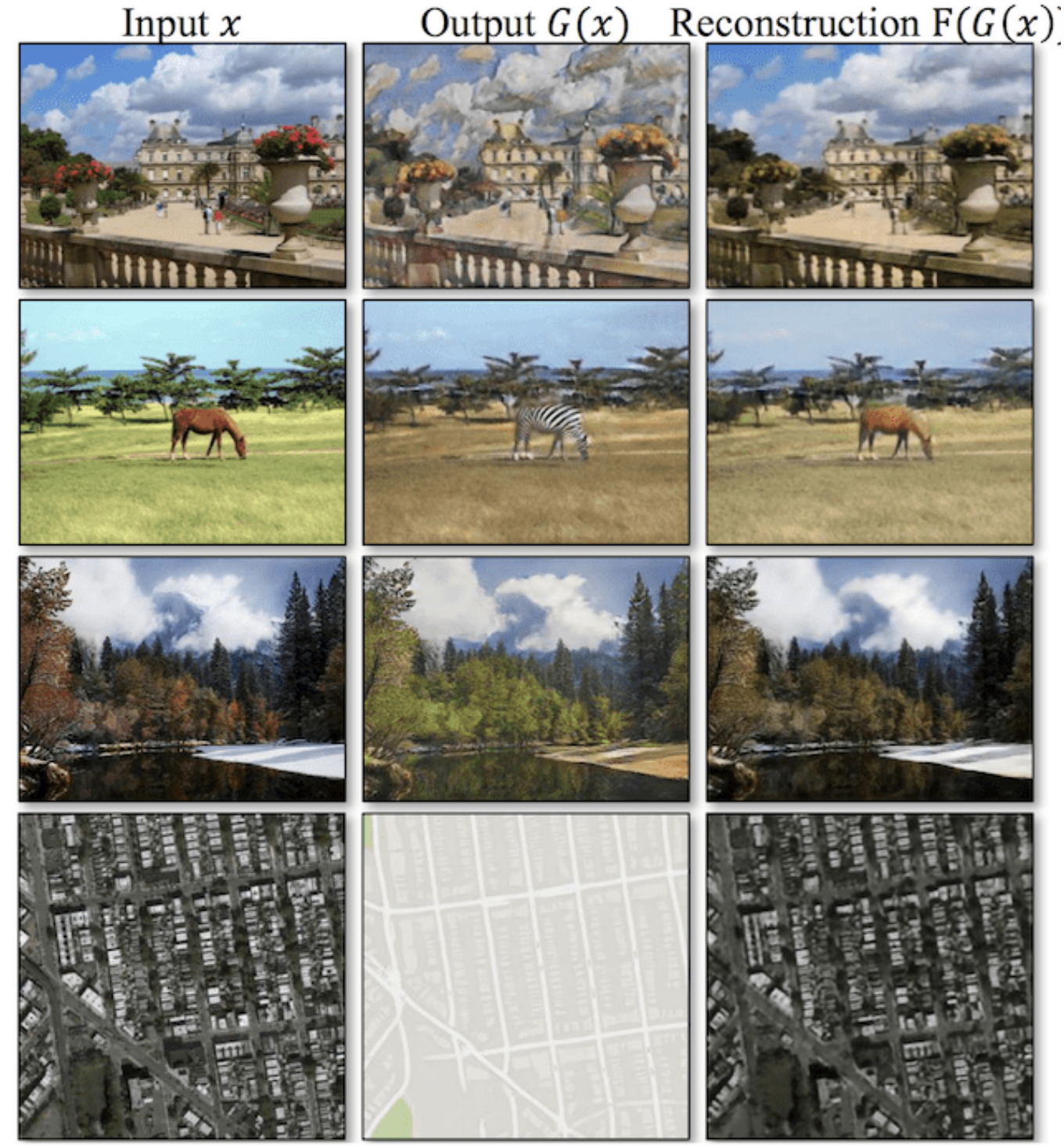
Examples include translation tasks such as:

- Translation of semantic images to photographs of cityscapes and buildings.
- Translation of satellite photographs to Google Maps.
- Translation of photos from day to night.
- Translation of black and white photographs to color.
- Translation of sketches to color photographs.

Jun-Yan Zhu in their 2017 paper titled "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks" introduce their famous CycleGAN and a suite of very impressive image-to-image translation examples.

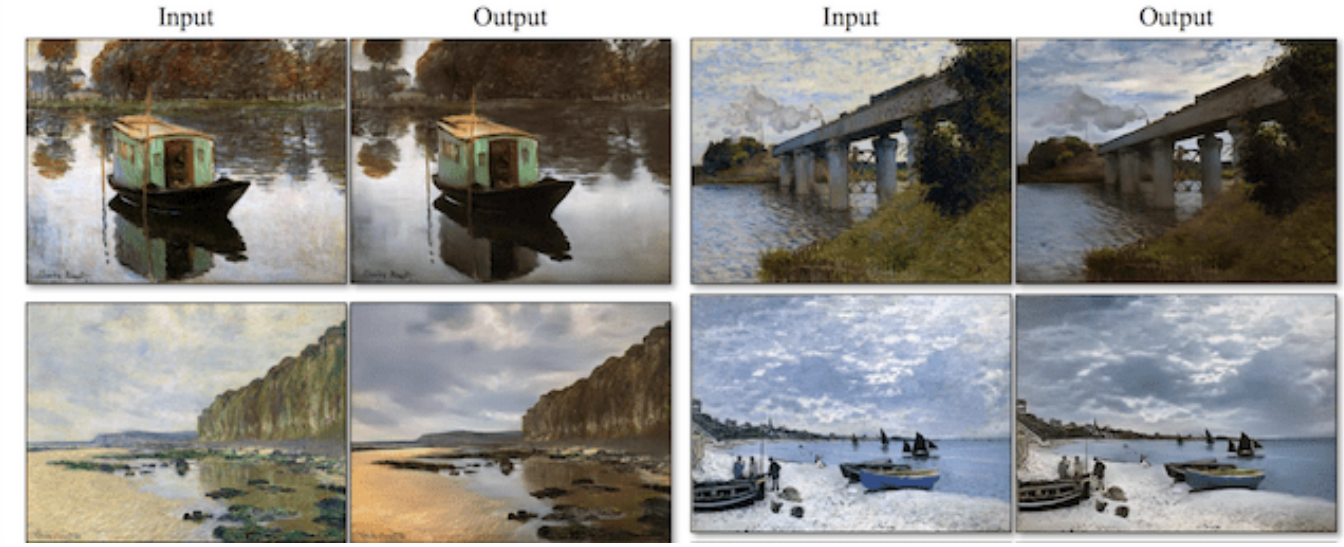
The example below demonstrates four image translation cases:

- Translation from photograph to artistic painting style.
- Translation of horse to zebra.
- Translation of photograph from summer to winter.
- Translation of satellite photograph to Google Maps view.



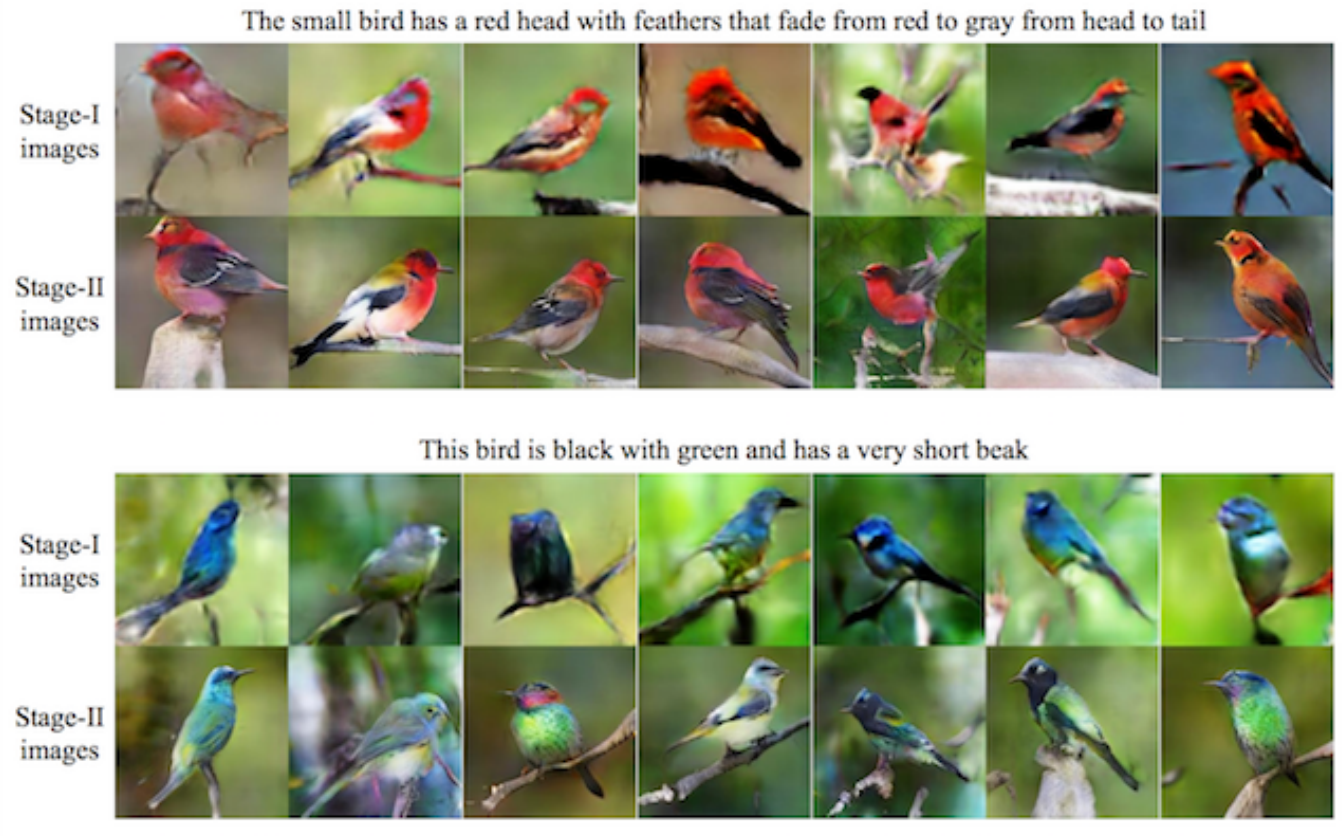
The paper also provides many other examples, such as:

- Translation of painting to photograph.
- Translation of sketch to photograph.
- Translation of apples to oranges.
- Translation of photograph to artistic painting.

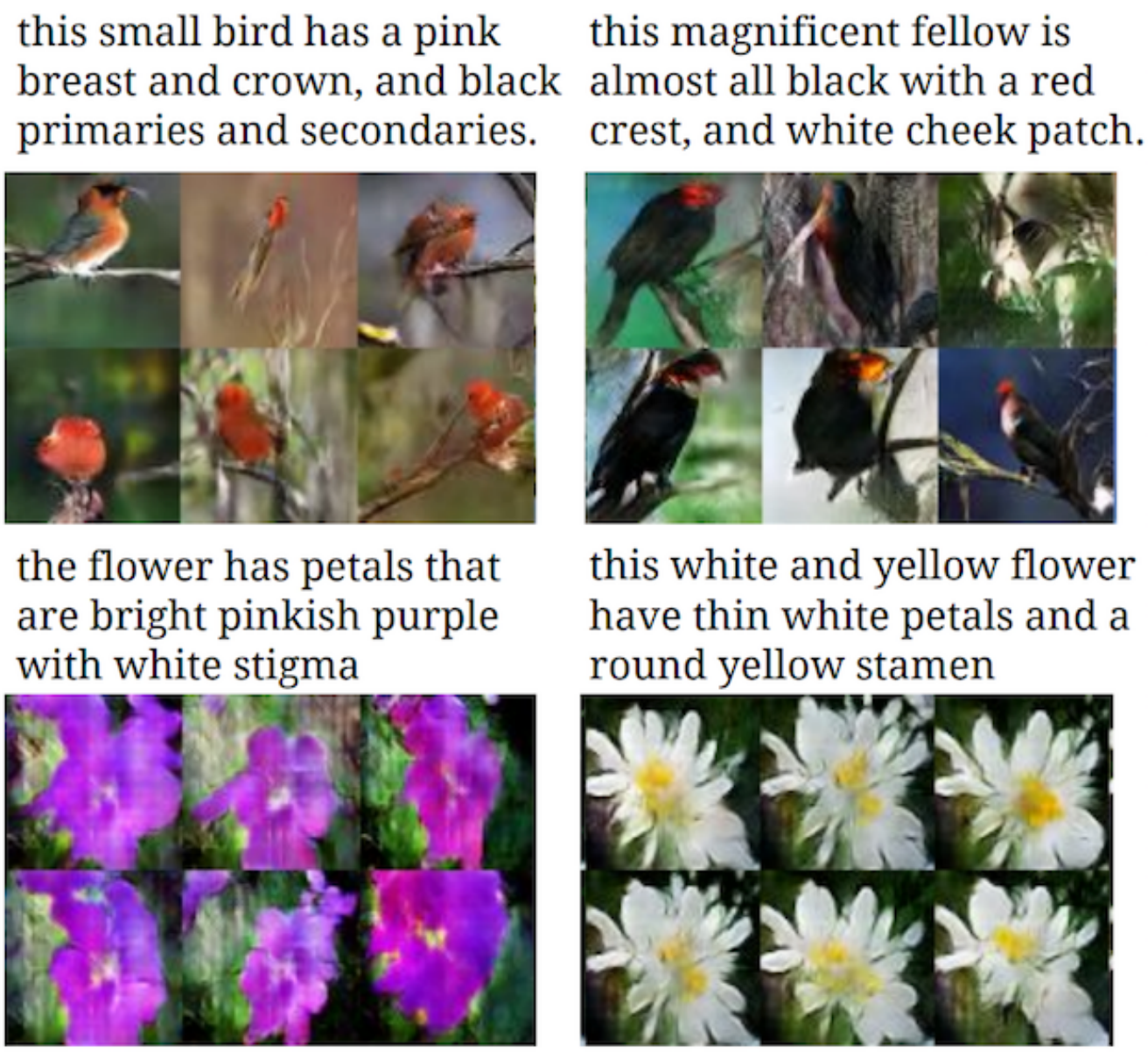


Text-to-Image Translation (text2image)

Han Zhang, et al. in their 2016 paper titled "StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks" demonstrate the use of GANs, specifically their StackGAN to generate realistic looking photographs from textual descriptions of simple objects like birds and flowers.

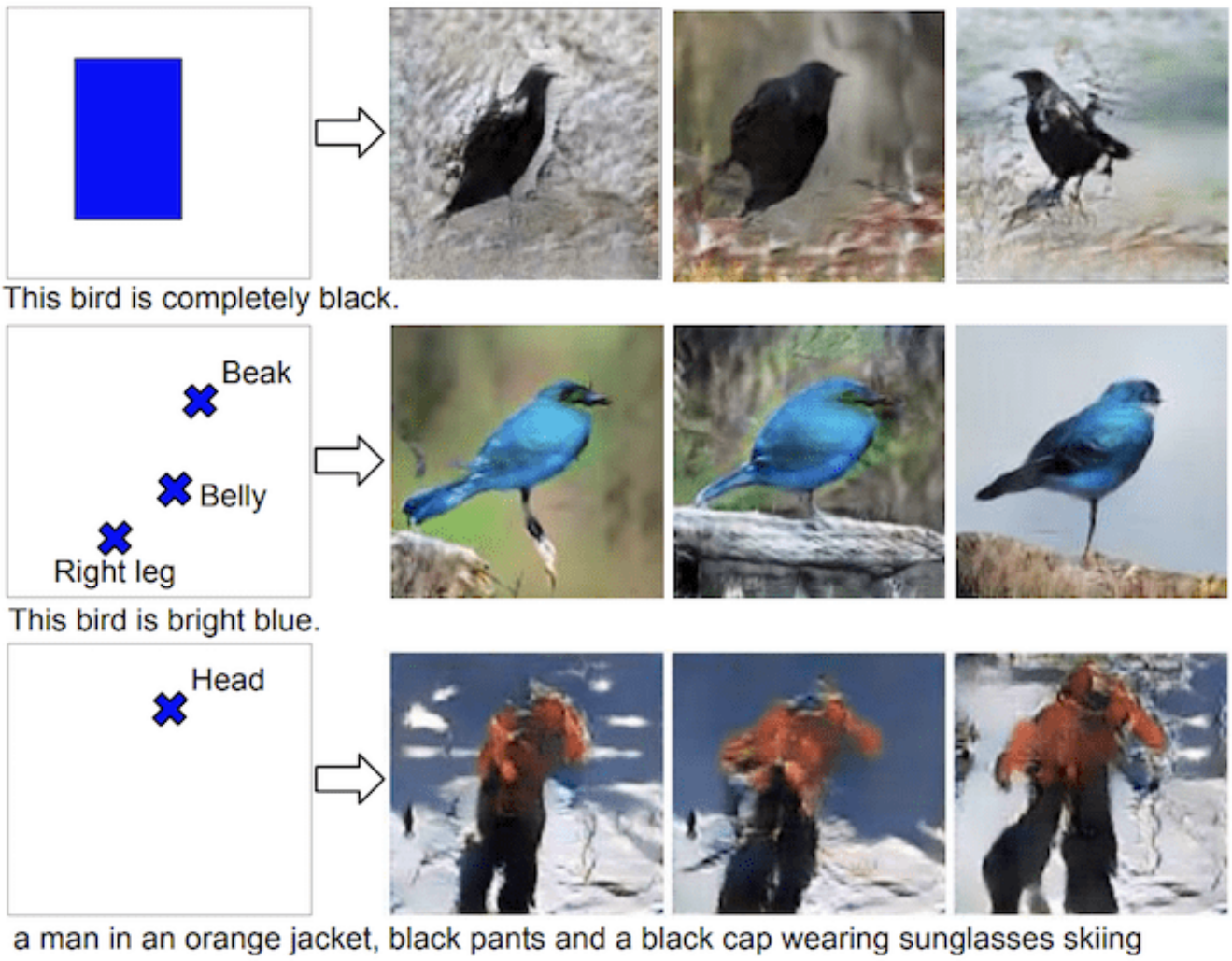


Scott Reed, et al. in their 2016 paper titled "Generative Adversarial Text to Image Synthesis" also provide an early example of text to image generation of small objects and scenes including birds, flowers, and more.



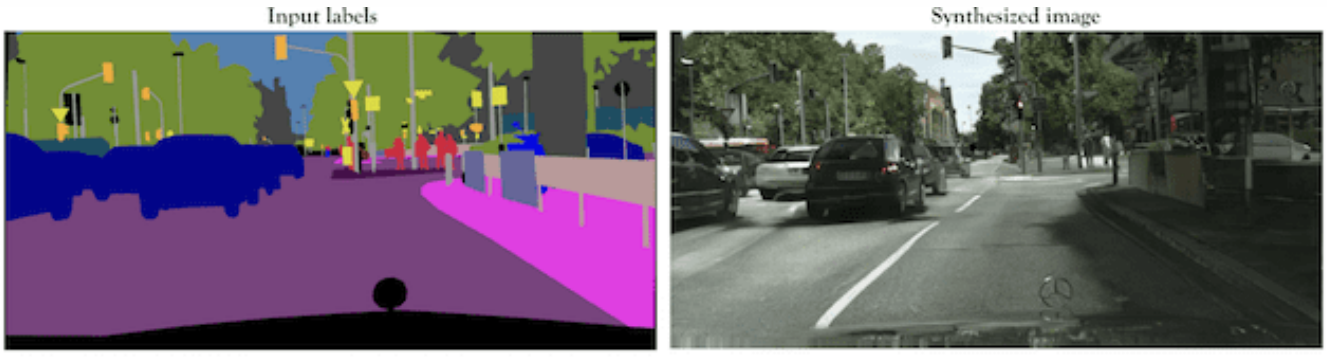
Ayushman Dash, et al. provide more examples on seemingly the same dataset in their 2017 paper titled "TAC-GAN – Text Conditioned Auxiliary Classifier Generative Adversarial Network".

Scott Reed, et al. in their 2016 paper titled "Learning What and Where to Draw" expand upon this capability and use GANs to both generate images from text and use bounding boxes and key points as hints as to where to draw a described object, like a bird.



Semantic-Image-to-Photo Translation

Ting-Chun Wang, et al. in their 2017 paper titled "High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs" demonstrate the use of conditional GANs to generate photorealistic images given a semantic image or sketch as input.



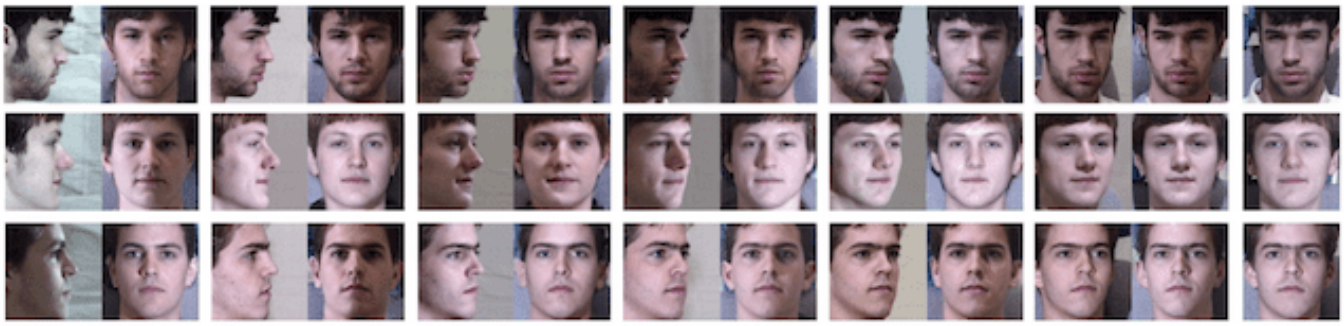
Specific examples included:

- Cityscape photograph, given semantic image.
- Bedroom photograph, given semantic image.
- Human face photograph, given semantic image.
- Human face photograph, given sketch.

They also demonstrate an interactive editor for manipulating the generated image.

Face Frontal View Generation

Rui Huang, et al. in their 2017 paper titled "Beyond Face Rotation: Global and Local Perception GAN for Photorealistic and Identity Preserving Frontal View Synthesis" demonstrate the use of GANs for generating frontal-view (i.e. face on) photographs of human faces given photographs taken at an angle. The idea is that the generated front-on photos can then be used as input to a face verification or face identification system.

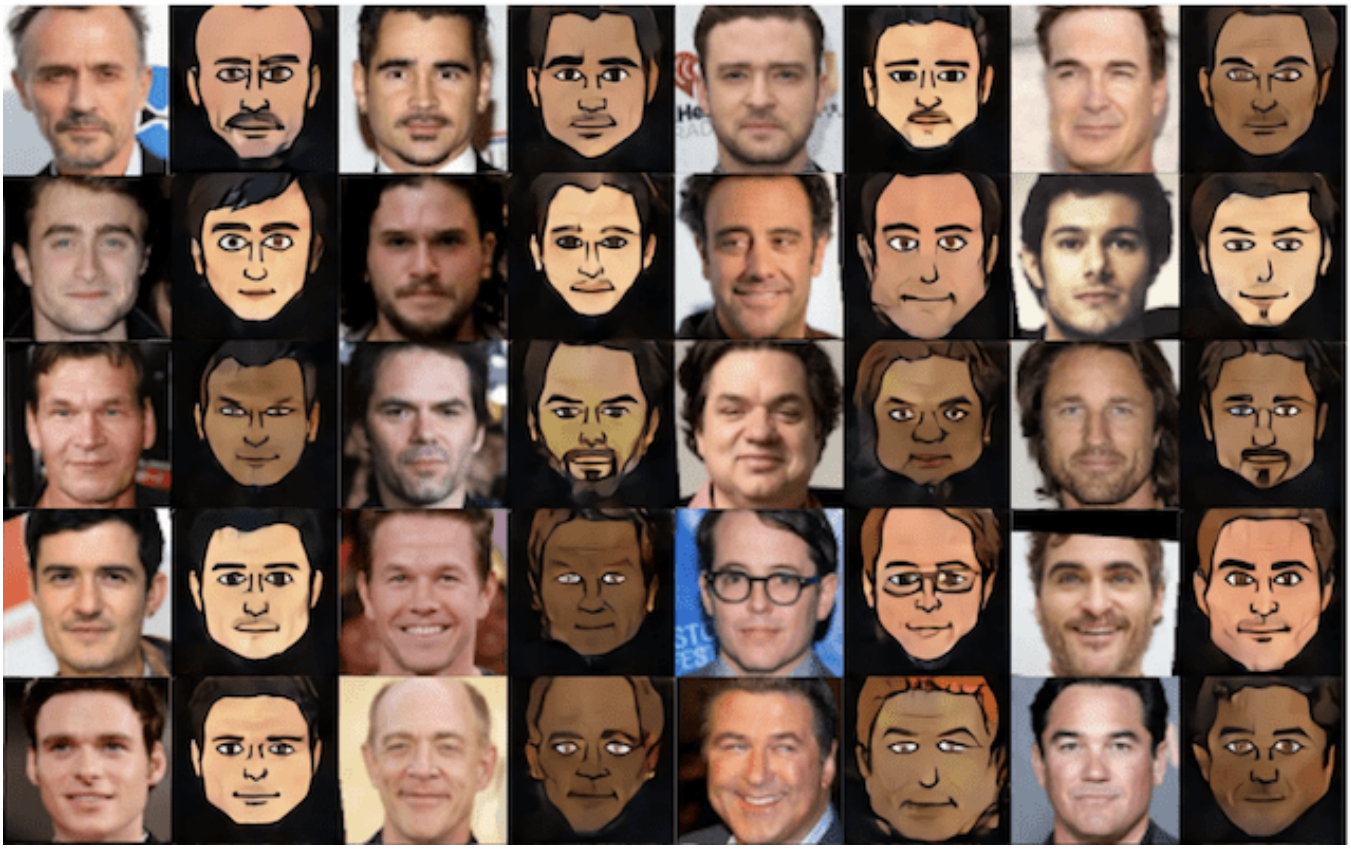


Generate New Human Poses

Liqian Ma, et al. in their 2017 paper titled "Pose Guided Person Image Generation" provide an example of generating new photographs of human models with new poses.

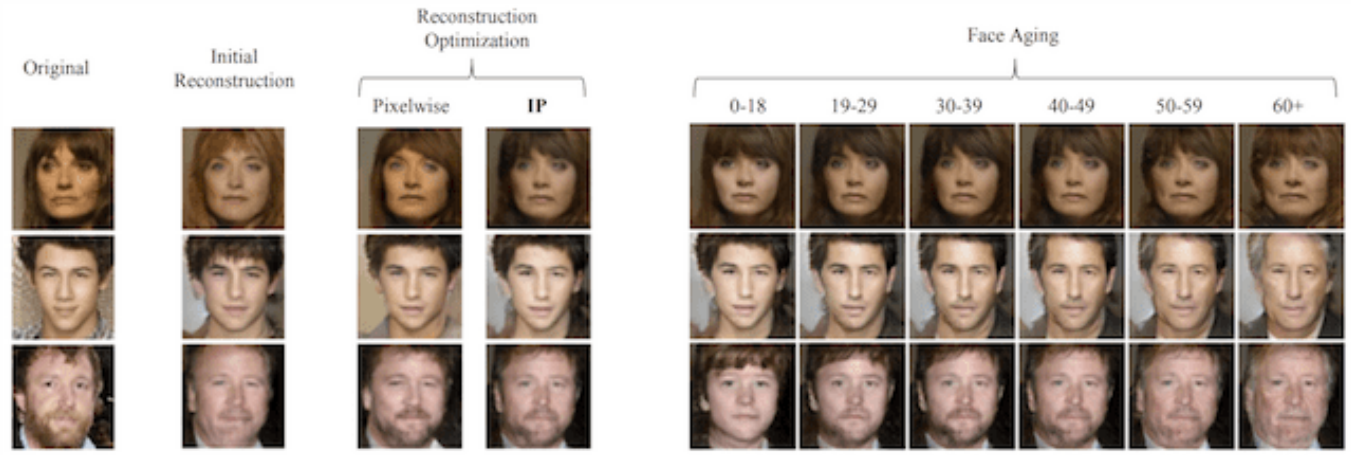
Photos to Emojis

Yaniv Taigman, et al. in their 2016 paper titled "Unsupervised Cross-Domain Image Generation" used a GAN to translate images from one domain to another, including from street numbers to MNIST handwritten digits, and from photographs of celebrities to what they call emojis or small cartoon faces.



Face Aging

Grigory Antipov, et al. in their 2017 paper titled "Face Aging With Conditional Generative Adversarial Networks" use GANs to generate photographs of faces with different apparent ages, from younger to older.



Zhifei Zhang, in their 2017 paper titled "Age Progression/Regression by Conditional Adversarial Autoencoder" use a GAN based method for de-aging photographs of faces.

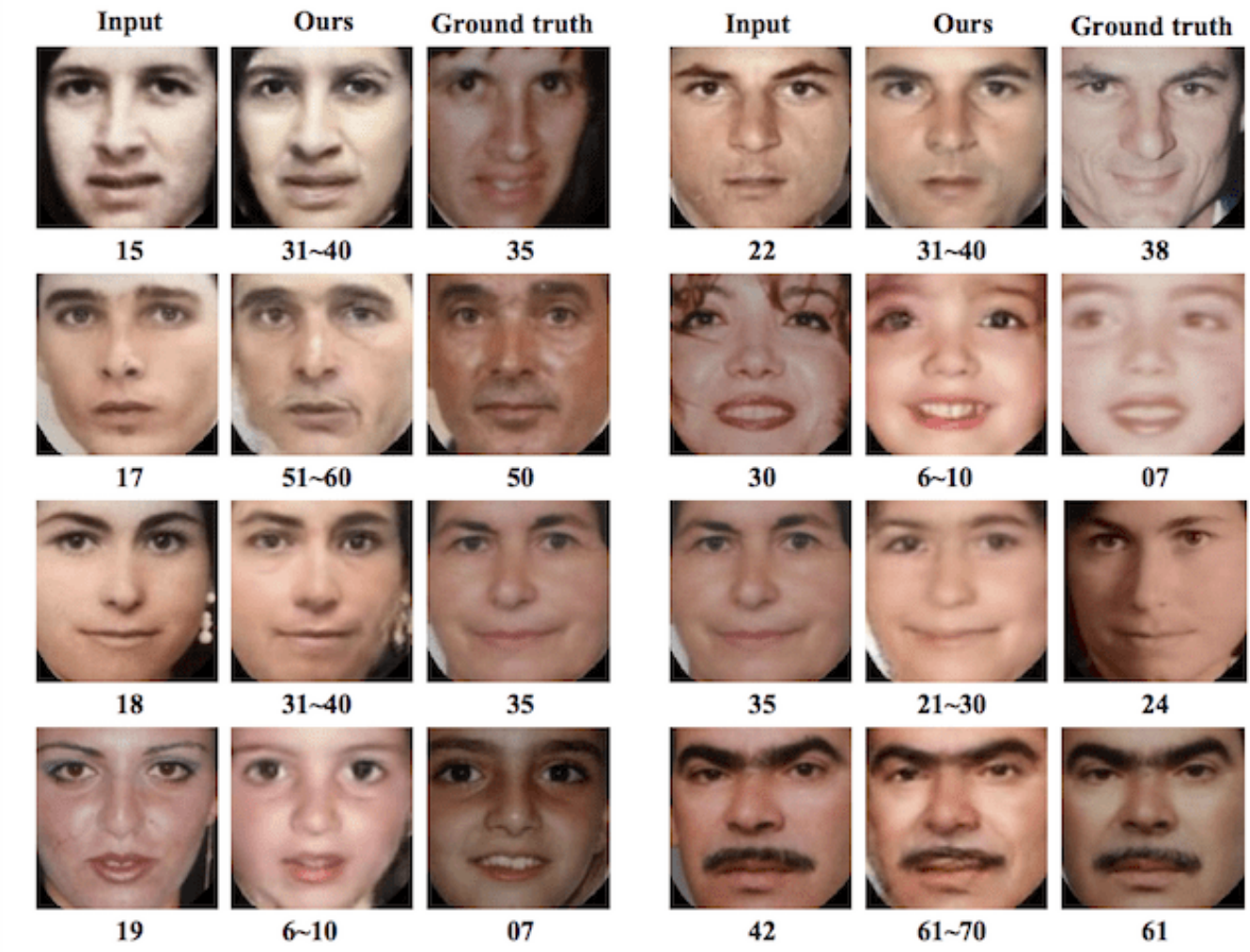


Photo Blending

Huikai Wu, et al. in their 2017 paper titled "GP-GAN: Towards Realistic High-Resolution Image Blending" demonstrate the use of GANs in blending photographs, specifically elements from different photographs such as fields, mountains, and other large structures.

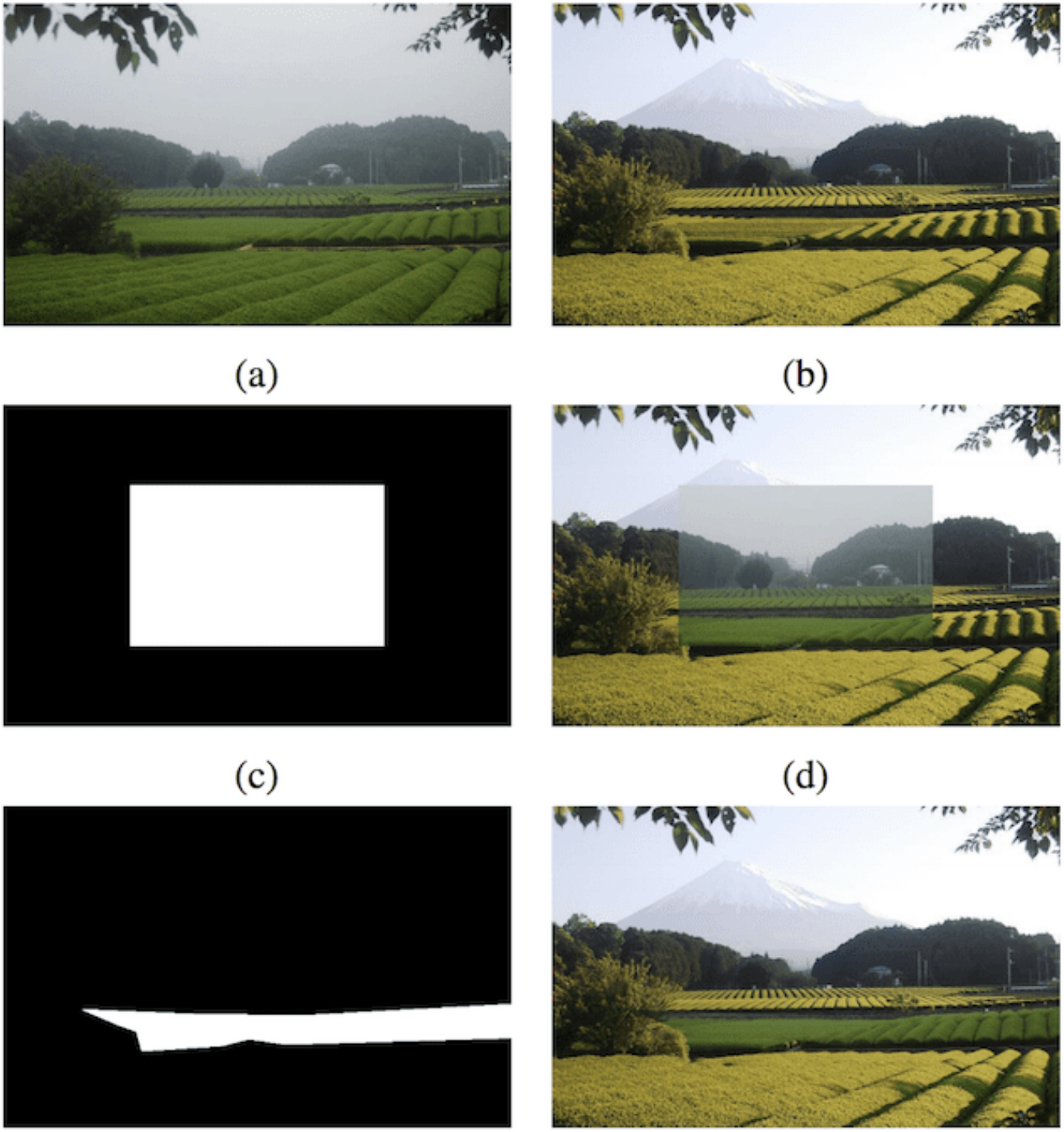
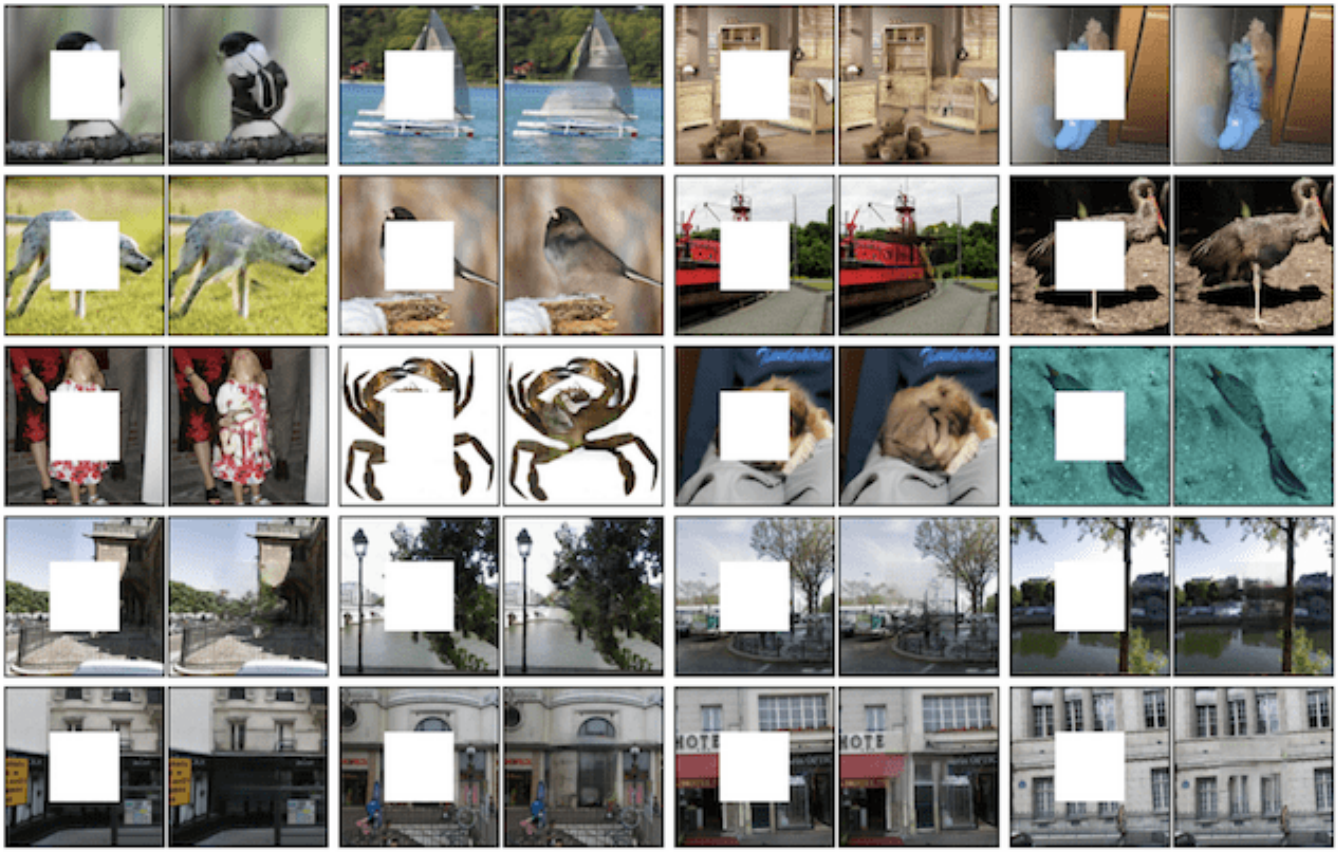
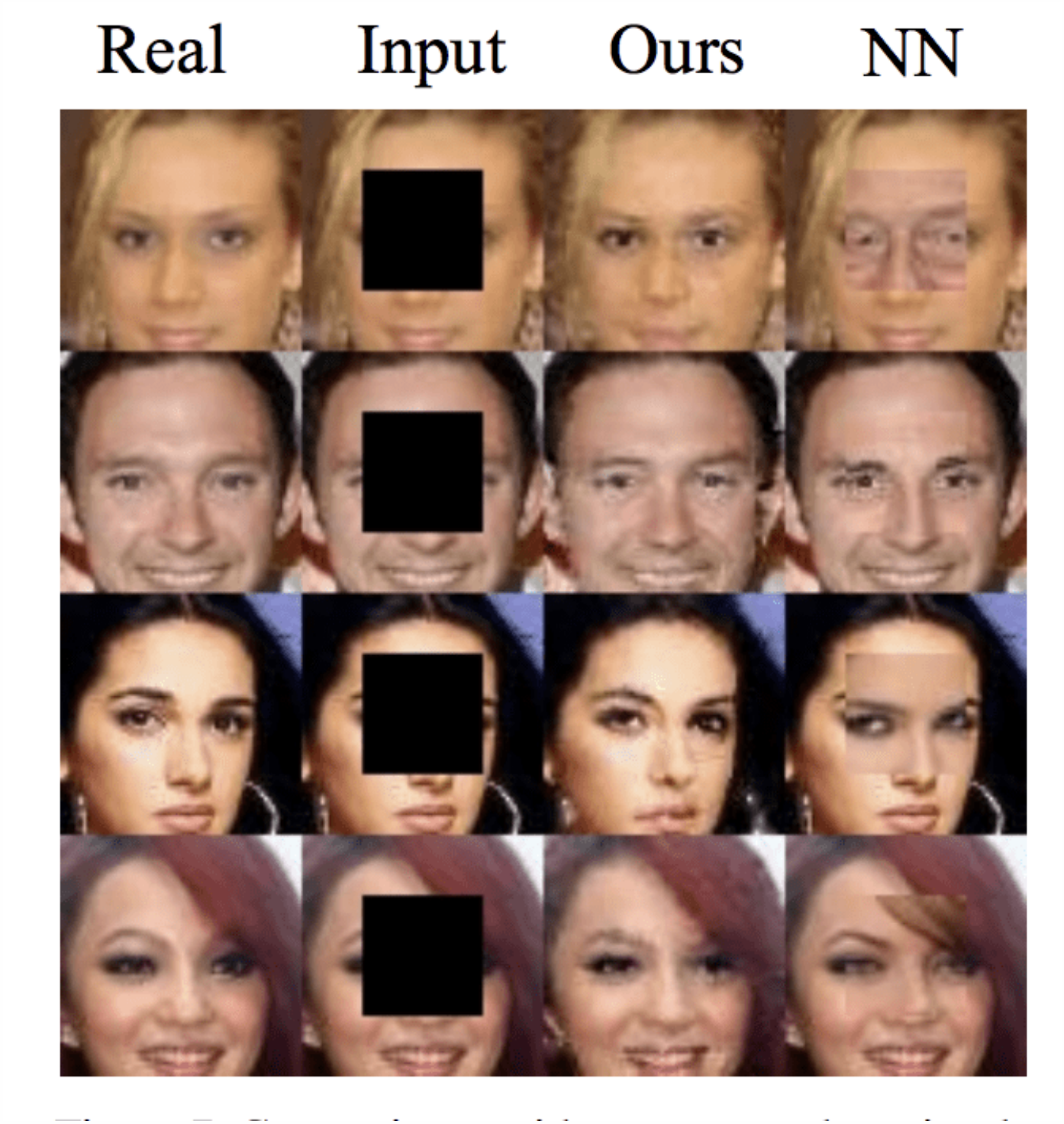


Photo Inpainting

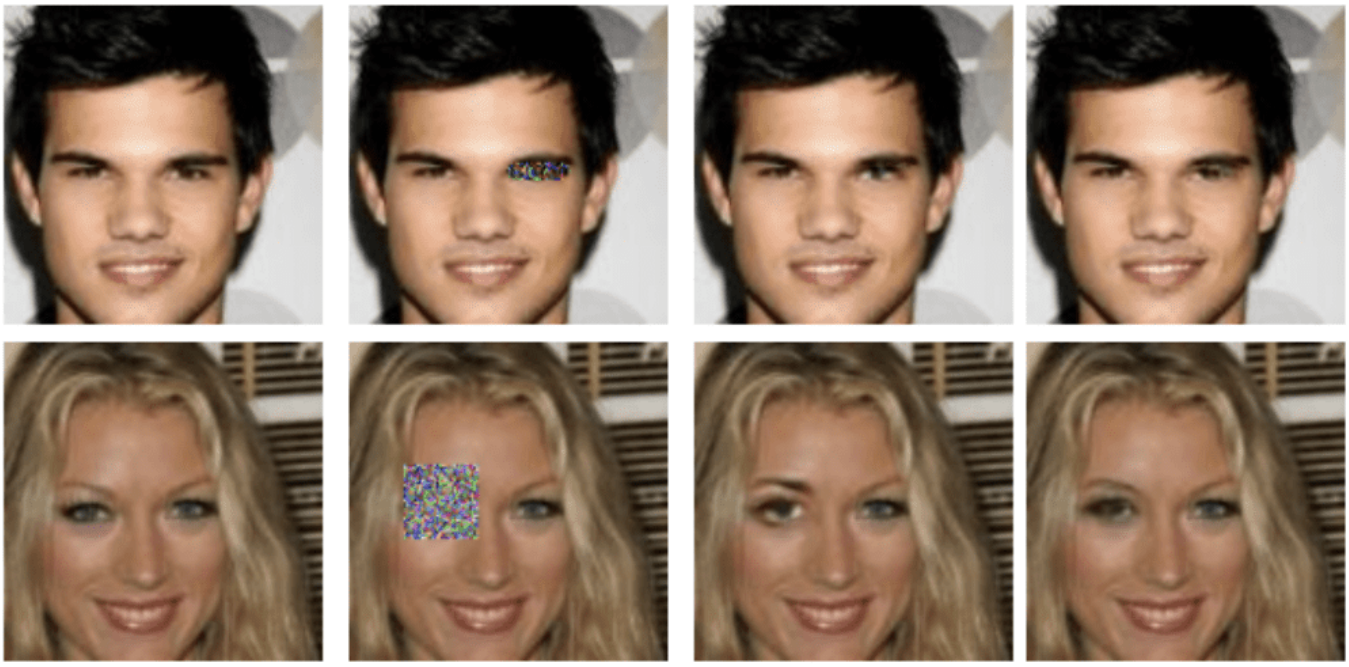
Deepak Pathak, et al. in their 2016 paper titled "Context Encoders: Feature Learning by Inpainting" describe the use of GANs, specifically Context Encoders, to perform photograph inpainting or hole filling, that is filling in an area of a photograph that was removed for some reason.



Raymond A. Yeh, et al. in their 2016 paper titled "Semantic Image Inpainting with Deep Generative Models" use GANs to fill in and repair intentionally damaged photographs of human faces.



Yijun Li, et al. in their 2017 paper titled "Generative Face Completion" also use GANs for inpainting and reconstructing damaged photographs of human faces.



5. Challenge of Training GANs

GANs are difficult to train.

The reason they are difficult to train is that both the generator model and the discriminator model are trained simultaneously in a game. This means that improvements to one model come at the expense of the other model.

The goal of training two models involves finding a point of equilibrium between the two competing concerns.

> Training GANs consists in finding a Nash equilibrium to a two-player non-cooperative game. […] Unfortunately, finding Nash equilibria is a very difficult problem. Algorithms exist for specialized cases, but we are not aware of any that are feasible to apply to the GAN game, where the cost functions are non-convex, the parameters are continuous, and the parameter space is extremely high-dimensional

— Improved Techniques for Training GANs, 2016.

It also means that every time the parameters of one of the models are updated, the nature of the optimization problem that is being solved is changed.

This has the effect of creating a dynamic system.

> But with a GAN, every step taken down the hill changes the entire landscape a little. It's a dynamic system where the optimization process is seeking not a minimum, but an equilibrium between two forces.

— Page 306, Deep Learning with Python, 2017.

In neural network terms, the technical challenge of training two competing neural networks at the same time is that **they can fail to converge.**

> The largest problem facing GANs that researchers should try to resolve is the issue of non-convergence.

— NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.

Instead of converging, GANs may suffer from one of a small number of failure modes.

A common failure mode is that instead of finding a point of equilibrium, **the generator oscillates between generating specific examples in the domain**.

> In practice, GANs often seem to oscillate, […] meaning that they progress from generating one kind of sample to generating another kind of sample without eventually reaching an equilibrium.

— NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.

Perhaps the most challenging model failure is the case where **multiple inputs to the generator result in the generation of the same output**.

This is referred to as "*mode collapse*," and may represent one of the most challenging issues when training GANs.

> Mode collapse, also known as the scenario, is a problem that occurs when the generator learns to map several different input z values to the same output point.

— NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.

Finally, there are **no good objective metrics for evaluating whether a GAN is performing well** during training. E.g. reviewing loss is not sufficient.

Instead, the best approach is to visually inspect generated examples and use subjective evaluation.

> Generative adversarial networks lack an objective function, which makes it difficult to compare performance of different models. One intuitive metric of performance can be obtained by having human annotators judge the visual quality of samples.

— Improved Techniques for Training GANs, 2016.

References:

A Gentle Introduction to Generative Adversarial Networks (GANs)

18 Impressive Applications of Generative Adversarial Networks (GANs)

Tips for Training Stable Generative Adversarial Networks

Further Reading:

How to Develop a 1D Generative Adversarial Network From Scratch in Keras

Lots of articles about GANs