eBook

August 2022

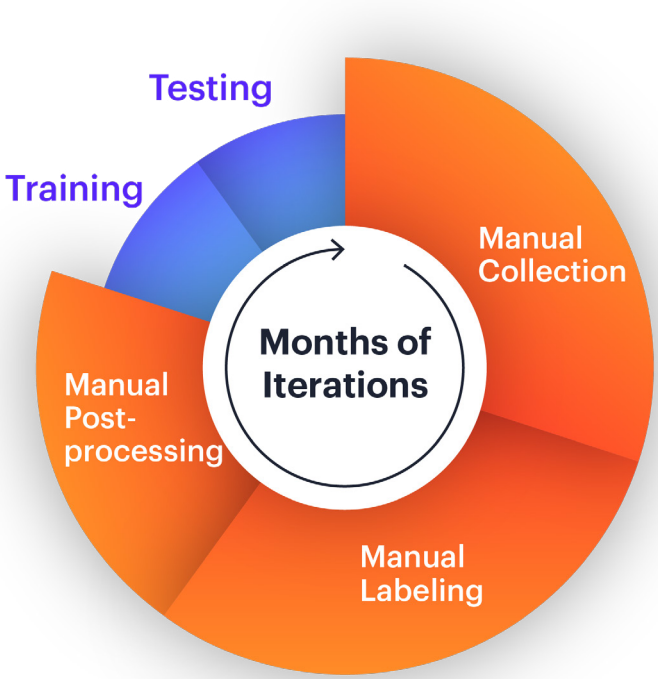
Designing a **Synthetic Data Solution**

datagen

What are the core principles of creating synthetic data?

In this eBook, we'll teach you the three core principles of synthetic data generation: variance, generalization over realism and reflecting the task, not the world. You'll learn about how to robustly represent your domain, testing and training.

Traditional data



With standard data acquisition processes, data is gathered through expensive, slow, and operationally intensive processes. It is then annotated manually with distributed human labor and cleaned before being used to train neural networks. This is a constraining process that doesn't allow for quick iteration or to improve the data that is used to train.

Synthetic data

- Automation
- Consistency
- Bias Control
- Privacy
- Compliance
- 3D Simulation

With synthetic data, this equation can be turned upside down. Instead of giving in to this "standard" process, data can be generated with full control. You can define the environment, the objects, the lights, the camera and get perfect fully annotated datasets - without dependence on manual, error prone processes. With all this control, you can robustly represent the target visual domain and spend less time on data acquisition with more efficient testing and training.

Domain Analysis

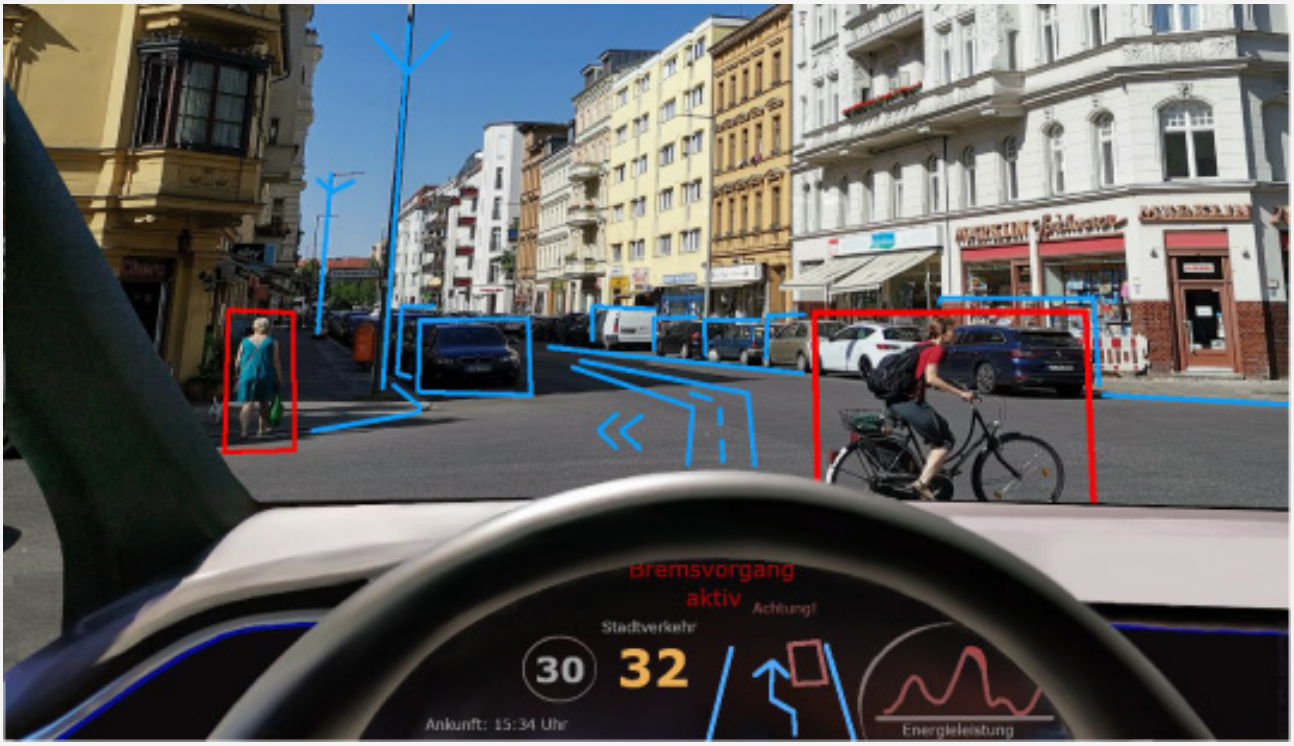
How do you know what data to create? What are the ways to look at and analyze the target visual domain to design your synthetic data?

To begin, you need to understand when to combine real and synthetic data, how to break down large visual domains into more practical sub-domains that can be solved, and what components of the domain you need to be explicit about when designing your synthetic data.

The first question to ask yourselves when analyzing your target visual domain is: **Are you dealing with a narrow or wide visual domain?**

A narrow visual domain is one with few dimensions of visual variance and small variance in each dimension.

For instance, let's take a scenario in which we have a robot, in a sterile environment, and a specific set of objects that are being picked up and placed on a conveyor belt. The only semantic variance is which objects are currently on the table and the orientation of those objects. This would be considered a narrow domain.

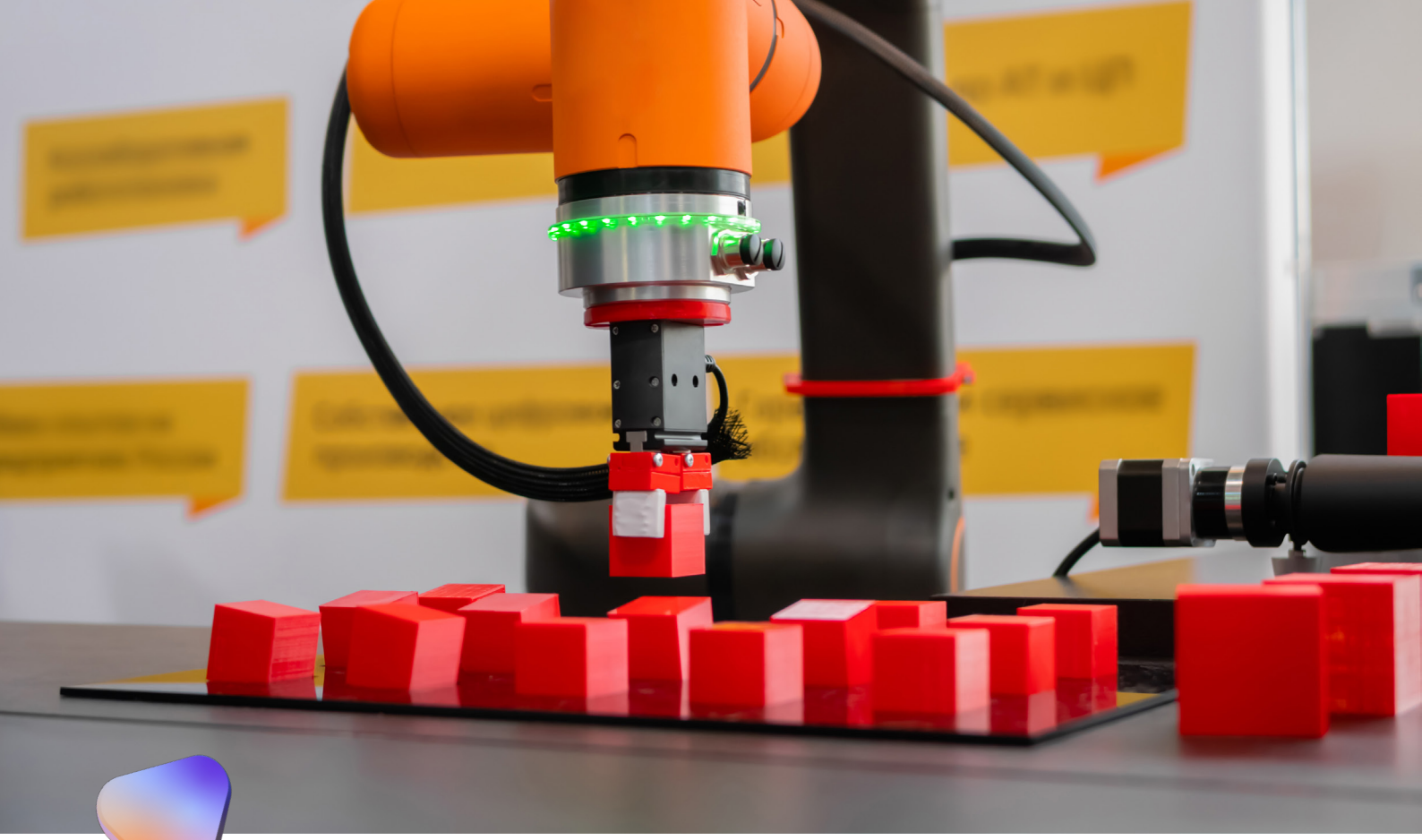


A wide visual domain, on the other hand, is a visual domain with many semantic dimensions of variance and large variance in each dimension.

Autonomous vehicles are the perfect example, as they drive in an extremely wide visual domain. There are an enormous amount of dimensions of visual variance, for instance, an infinite number of unique object classes which could become obstacles for the vehicle; various weather conditions, sensor conditions and a wide range of lighting scenarios, to name a few. And if that isn't enough, in each dimension there are substantial visual variations; the same vehicle driving in the same location on a sunny summer day or a snowy winter day looks extremely different.

From our experience at Datagen, narrow visual domains can be fully simulated. The entire challenge can be solved end-to-end with synthetic data. On the other hand, in the case of very wide visual domains, synthetic data must supplement gathered data from the target visual domain. You solve this by breaking down the wide domain to a set of more narrow sub-domains and solve each one with a mix of real and synthetic data.

To understand the target visual domain at a deeper level, let's analyze three parts of the visual domain: Domain core, domain edge and domain dynamics.



Domain Core

The domain core defines the semantic and visual variance that exists within the domain.

Semantic variance is changes that can be made to the image that will affect the output. For example, for the robot in a manufacturing setting, the semantic variance can be changed in the placement of the objects.

Visual variance is changes that can be made to the image that should not affect the output. For the robot, lighting changes shouldn't affect what the robot understands.

You need to map out both types of variations. This mapping will be critical when designing your synthetic data solution.

Domain Edge

With the domain edge, ask yourselves “what can go wrong?”. Whatever can go wrong, will go wrong when millions of users use the algorithms created within the designed products.

There are two types of things that can go wrong. Things that can be foreseen - for instance, objects that are heavily occluded. And then there are unknown unknowns, which are challenges that you don’t know exist within the target visual domain until you deploy the device or product to the masses and start to run into them.

Synthetic data can deal with edge-cases in a substantial way. It can simulate them with the same ease that the domain core can be simulated.

Regarding unknown unknowns, think outside the box, and in practice, deal with them in a product-focused manner and not with purely technological solutions.

Domain Dynamics

Target visual domains are constantly changing and expanding. Various visual cues are changing at different frequencies. Some cycle, some slowly drift, and some quickly shift. For example, in a smart store scenario, there are new items constantly being added to the shelves, existing items with updated packaging, and other items that are being removed from shelves. These are shifts to the domain that are happening on a daily basis. An example of cyclical shifts are outdoor scenes, where the seasons cycle annually, as well as weather patterns shifting constantly.

These temporal shifts occur in every target visual domain. This is something that needs to be well understood and mapped out, when analyzing the target visual domain.

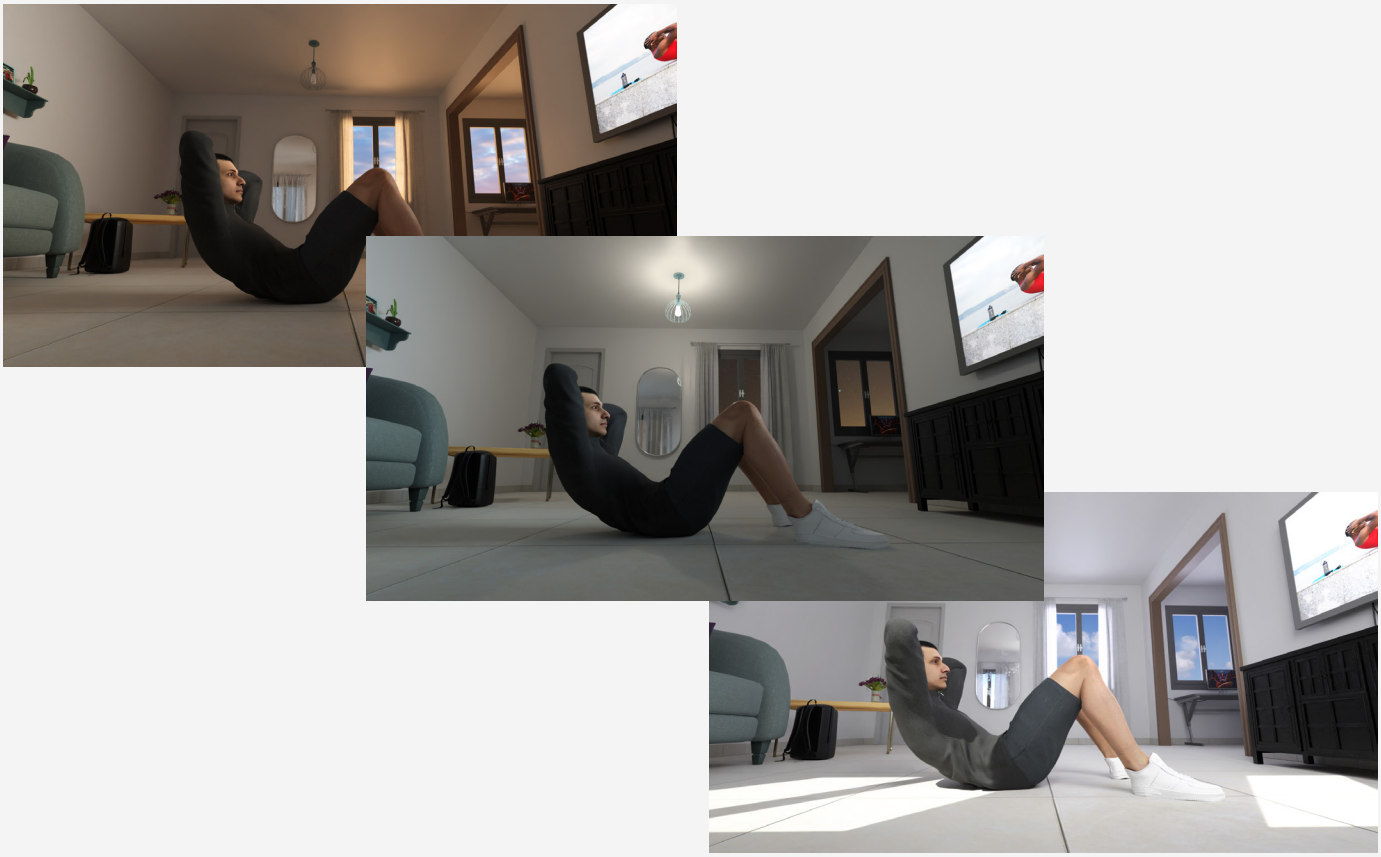
When trying to bring together an analysis, in order to clearly define the synthetic data that will be generated, the minimal task-specific target domain needs to be defined, meaning the minimal target domain, that if represented robustly within the data, would produce solid results for your specific task.

When it's analyzed, look to define three aspects explicitly. The domain actors, domain scene, (see below) and the ideal information you want to be labeled.

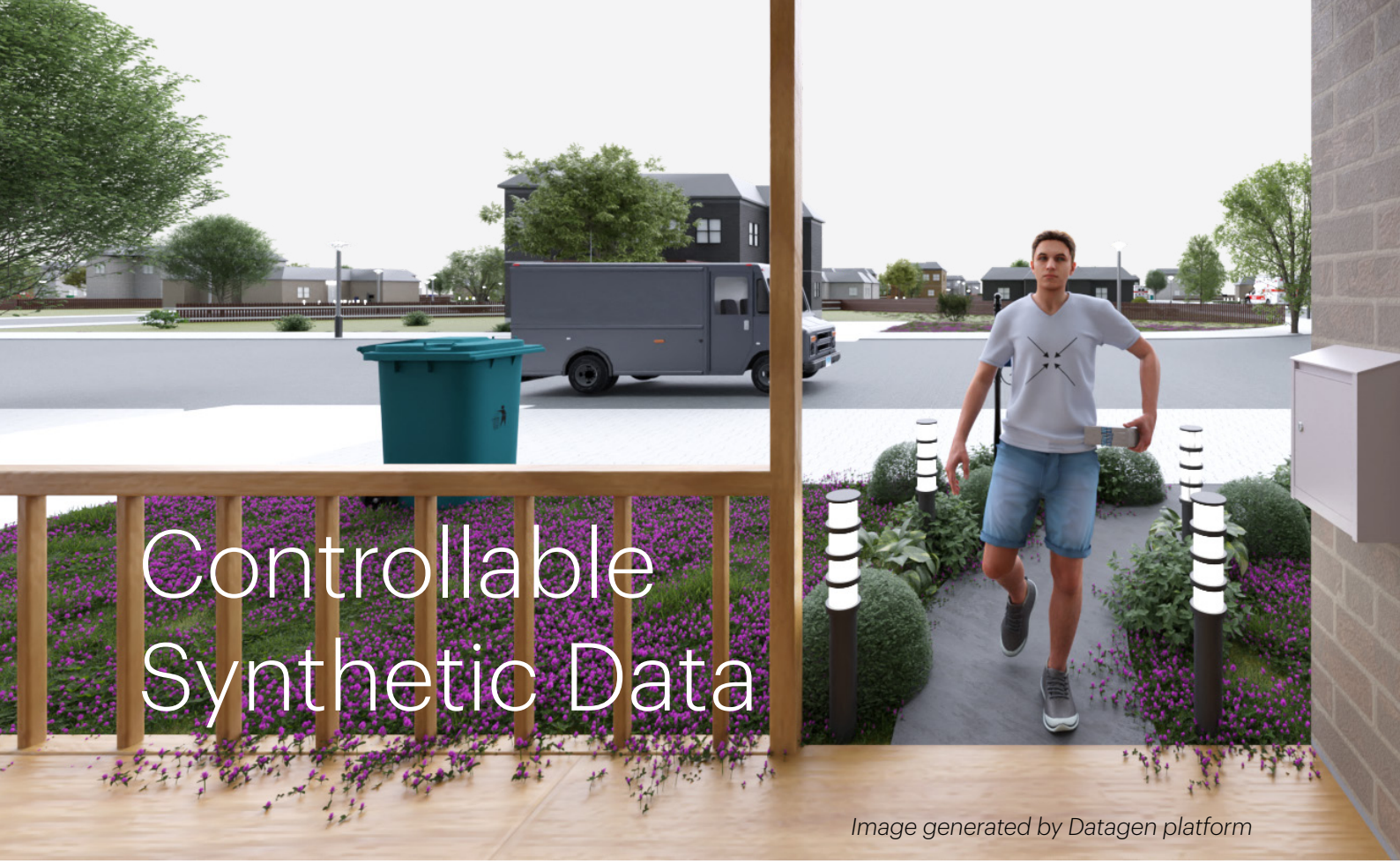
Actors can be people, objects of interest or distractor items that may make our lives difficult. Most use-cases are focused around a finite set of actors.

The scene includes semantic relationships between the various actors, lighting scenarios, camera parameters, and background scenery.

The task labels are the set of ideal labels that you want to extract from the target visual domain, in order to bring the best capabilities to your product. This requires some imagination, to step beyond the standard detection bounding boxes. With synthetic data, you have all of the flexibility in the world, so the task labels are up to you.



Images generated by Datagen platform



Controllable Synthetic Data

Image generated by Datagen platform

In the past, the target domain was understood intuitively and helped decide what data to gather. In practice, due to the limited control of the exact data gathered, there wasn't a strong emphasis on this.

Now, with synthetic data in mind, you can actually control all these aspects of the data. You can generate from the domain core, domain edge and deal with the domain dynamics.

The actors and scene define the synthetic data itself. And the annotations to be extracted are no longer limited by the cost and time of manual labor.

We don't know what the perfect dataset is ahead of time, but by breaking it down and analyzing the domain, you can create solutions for each part of the domain, until it is completely covered. And by constantly iterating and improving the data, you can reach better capabilities in a fraction of the time it took before.

The Core Principles

What is important when generating data?

Now that you understand target visual domain, you understand the mix of synthetic and real data that makes sense for your use-case, and you have designed a set of clearly defined datasets that cover your target domain. You must understand the three main principles of synthetic data generation.

There are three main principles that are important when generating synthetic data - high-variance, that attempts to create a robust representation of the target domain, generalization, between the synthetic domain and the target domain and controlled biases, which represent the world equally without inserting unwanted correlations into our data.

Images generated by Datagen platform

Principle

1 Variance

Variance is the most important aspect of data.

A million images of the same visual scene within our target domain won't add the same value as a smaller but highly variant set of images from the same domain.

To create strong algorithms, you need robust representations of the target visual domain. Now that gathering and annotating isn't an issue due to synthetic data, you need to define the domain and attempt to generate data that is uniformly distributed across it.

You want to start by representing the Domain Core robustly. This is usually done with multiple designed datasets; each covering a specific set of actors and parts of the scene.

Later, you can make your way to the Domain Edge and add variance to encompass the Domain Dynamics.

At Datagen, we highly recommend using augmentations to remove the network's dependence on specific visual cues.

For example: If you want to remove the network's dependence on the brightness of the images, you can use random brightness augmentation to make sure that the visual signal isn't correlated with the output of your network.

If the signal isn't critical for the task, it should improve results and speed up convergence. The bottom line is that random augmentation is still relevant - but should be used in the right context, on specific visual cues.

Principle

2 Generalization Over Realism

Using synthetic data to train neural networks is like using a simulation to train pilots. A pilot simulation should train a pilot to deal with real-life scenarios. The goal of the pilot training simulation is to make sure that pilots can use the skills learned in the real world, even if it doesn't look photorealistic. A certain level of realism makes it easier for the pilot to learn, but this isn't the most critical component. The goal is for the data generated to generalize well to the real world.

What generalization means is the ability to train with data from domain A, and reach good results when tested on domain B.

Domain A
Simulation

Domain B
Real Cockpit

In the context of synthetic data, you want to train with the synthetic data from the synthetic data domain and receive good quality test results in the target visual domain.

Benchmarking is the key. Benchmarking is one of the most important aspects of good synthetic data. And due to the iterative nature of synthetic data, it is more actionable than ever before.

Images generated by Datagen platform

Principle 3

Reflect the Task, Not the World

This is a bit counterintuitive because even though the world may be biased, your models shouldn't be. There are biases that can be caused by the gathering methods, things that are naturally less frequent or are harder to gather, appear less in the data. There are biases that can be caused during the annotation process, so things that are harder to annotate have more annotation mistakes.

One of the most widely discussed biases, which poses a serious problem for real-world applications, are biases in demographics that are widely dependent on the geography of the gathering process.

This is counterintuitive because you don't want to represent the distributions of the real-world. You want to reflect high-level biases of the domain uniformly, in our training data. Ethnicities, ages, genders, lighting scenarios and smartphone camera type are a few examples.

In addition, task specific biases are important to represent equally and uncorrelated to the high-level biases.

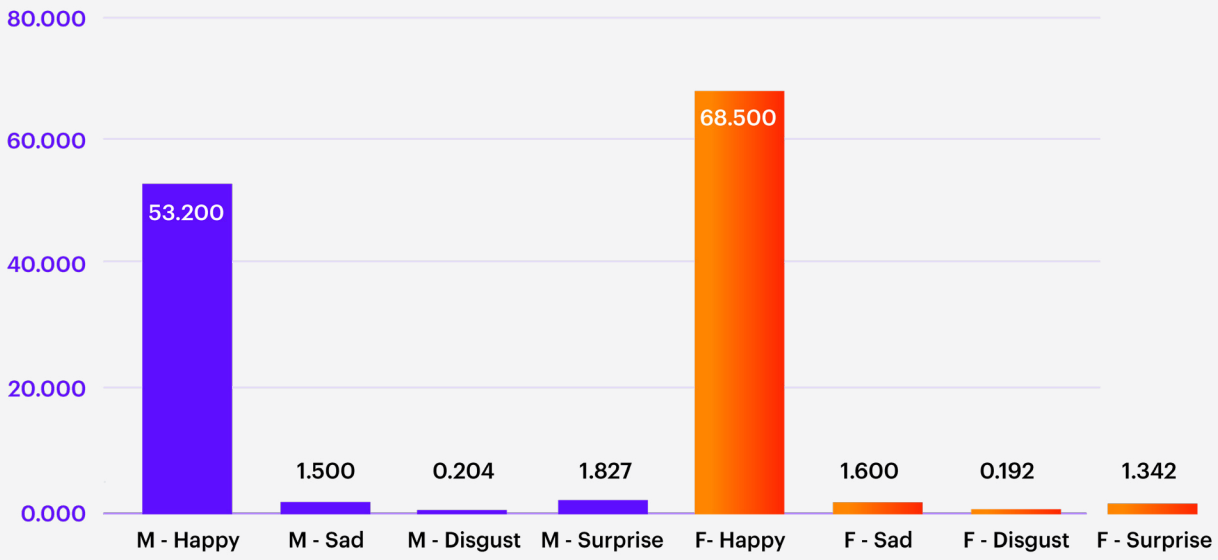
FFHQ Dataset

Here you see the FFHQ face dataset. A large dataset of 70,000 faces collected from Flicker, which is now widely used to train generative machine learning models.

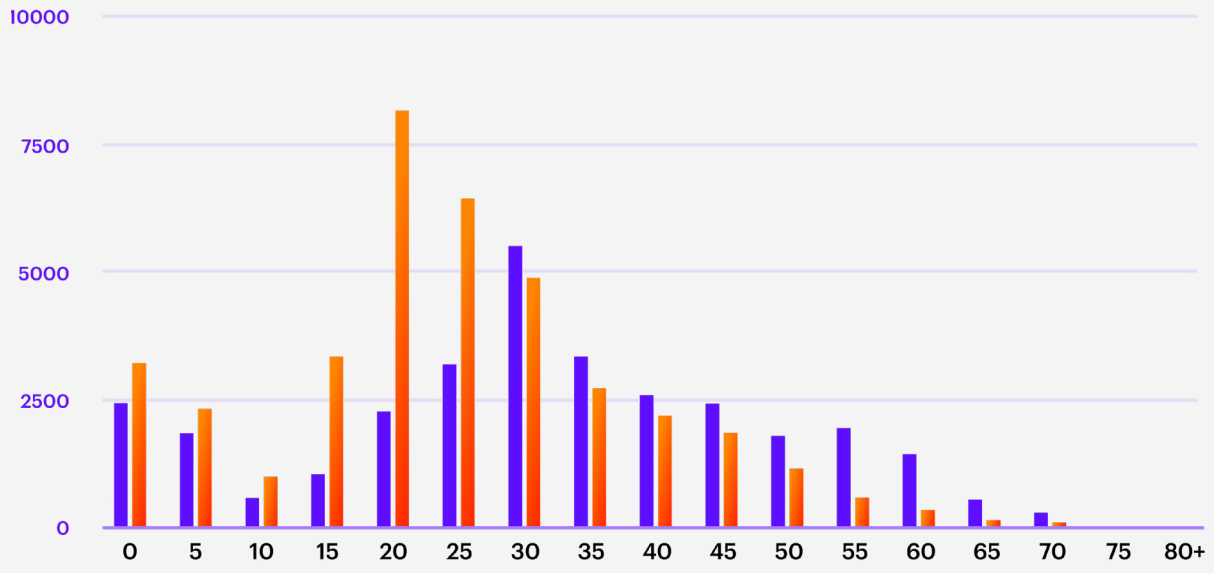
Let's use this data to train a network for various tasks.

If the task is expression analysis, you can see that most of this dataset is smiling and you can expect strong bias in the network towards this expression.

Mean Emotion Scores by Gender



Age by Gender



If the task is age estimation, you can see that the men are older than the women in the data - so there is a correlation between the task and the high-level biases.

There are methods to mitigate the effects of correlations that you can anticipate, but the best thing to do is to make sure that the data is unbiased.

Computer vision neural networks can be seen as task-specific selective compression algorithms that extract specific patterns from within the data.

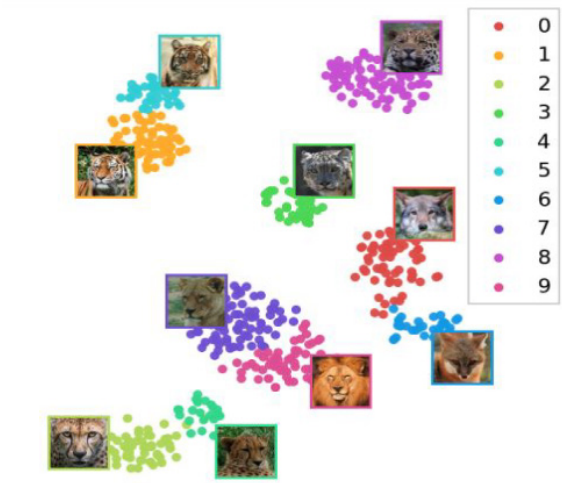
CV neural networks will use any correlations found in the data to help compress the data and infer these patterns from within the data. And this, of course, includes unwanted correlations such as biases.

Within your data, you need to uniformly represent the domain, in this case the various segments of the population, especially in relation to the task.

Training and Testing

Given a specific task, you want to generate a uniform representation of the domain and you need to specifically make sure it is uniform with relation to your task. This uniform representation needs to be in the training data and in the test data.

Training Data



Unit-Testing Framework

What we recommend, especially in wide domains, is to break down the domain into Narrow Visual Sub-Domains. In each of these subdomains, you need to make unit-tests that validate the quality of your neural networks on specific segments of the domain. With every update - adding new data, optimizing the network architecture or changing the collection device - you need to validate that you're maintaining or improving performance.

Like with code, whenever it's updated and you attempt to push it to production, it needs to first pass all the tests. This is what you should aim for with neural networks.

The Datagen Solution

Datagen is the Data-as-code company, a category-defining solution for the Computer Vision data market. Building on the principles of Infrastructure-as-code, Datagen's approach turns the heavy operational process of visual data collection and annotation into an easy-to-control programmable user interface, enabling CV teams to generate data and train and evaluate models, across the development lifecycle. Fortune 500 companies rely on Datagen's self-service human-centric synthetic data platform to develop their future products in the worlds of AR/VR/Metaverse, In-cabin Vehicle Safety, IoT Security and more. Founded in 2018, Datagen is led and backed by world-renowned AI experts.

Let's Talk