figure eight

The Essential Guide to Training Data

BY ROBERT MUNRO
CHIEF TECHNOLOGY OFFICER
FIGURE EIGHT

BY QAZALEH MIRSHARIF
MACHINE LEARNING SCIENTIST,
COMPUTER VISION, FIGURE EIGHT

THE ESSENTIAL GUIDE TO TRAINING DATA

BY ROBERT MUNRO & QAZALEH MIRSHARIF

INTRODUCTION

It's common knowledge that every machine learning solution needs a good algorithm powering it. Plenty of ink is spilled on tech sites about advances in deep learning and how the newest models are driving business success for everything from personalized shopping to national security.

What gets far less press is what actually powers these algorithm: **the data itself**.

There are plenty of analogies you can use to drive this point home. Data is the oil, the model is the car. Data is the ingredients, the algorithm is the recipe. The point is: neither works without the other.

In this guide, we'll cover everything you need to know about creating the training data necessary to drive successful machine learning projects. Let's start with a simple question.

After all, no matter how cutting edge a model happens to be, it's useless without enough high-quality training data. In fact, when you dig into the history of the major breakthroughs in AI, a vast majority were preceded by voluminous datasets:

Year	Breakthroughs in AI	Datasets (First Available)	Algorithms (First Proposed)
1994	Human-level spontaneous speech recognition	Spoken Wall Street Journal articles and other texts (1991)	Hidden Markov Model (1984)
1997	IBM Deep Blue defeated Garry Kasparov	700,000 Grandmaster chess games, aka "The Extended Book" (1991)	Negascout planning algorithm (1983)
2005	Google's Arabic- and Chinese-to-English translation	1.8 trillion tokens from Google Web and News pages (collected in 2005)	Statistical machine translation algorithm (1988)
2011	IBM Watson became the world Jeopardy! champion	8.6 million documents from Wikipedia, Wiktionary, Wikiquote, and Project Gutenberg (updated in 2010)	Mixture-of-Experts algorithm (1991)
2014	Google's GoogLeNet object classification at near-human performance	ImageNet corpus of 1.5 million labeled images and 1,000 object categories (2010)	Convolution neural network algorithm (1989)
2015	Google's Deepmind achieved human parity in playing 29 Atari games by learning general control from video	Arcade Learning Environment dataset of over 50 Atari games (2013)	Q-learning algorithm (1992)
Average No. of Years to Breakthrough:		**3 years**	**18 years**

HOW DO MACHINES UNDERSTAND THE WORLD?

The short answer: from labeled examples.

Fei Fei Li, the woman behind ImageNet, the most famous and widely cited dataset for computer vision projects, once gave a talk where she described teaching her daughter what a dog was. When a child is born, of course, it has no idea what a dog (or really anything) actually is. When a toddler sees a golden retriever for the first time and her parent says, "that's a dog," she now has a word for that four-legged furry thing that's running around, hoping to get pet. She can observe how the dog moves, how it behaves, all the while knowing: this is a thing that's called a dog.

But say she sees a cat. Well, it's got four legs and it's soft. It wants to be pet. She very well may assume that's a dog. Her parent can teach her differently though, telling her what this new animal is called and pointing out how it looks and behaves differently than the concept of "dog" she's already learned. The cat is smaller, for example. It purrs. It's not going to bring that tennis ball back to you.

Now, this young girl has, for all intents and purposes, new training data. A cat has been "labeled" by her mother and she can observe it, understand what distinguishes it from a dog, and so on.

Broadly speaking, for supervised learning, this is how machines understand things. They learn from labeled examples. ImageNet, the dataset we mentioned above, is a massive library with examples of everyday objects, from chairs to pizzas to, yes, dogs and cats. And it's widely cited as the basis for a vast amount of computer vision projects, both professional and academic.

This concept doesn't just apply to computer vision, though. It holds true across nearly every successful machine learning deployment:

For natural language processing (NLP), you need contextual examples to teach a machine what words mean. After all, the sentences "that burrito was so bad" and "I want a burrito so bad" share a lot of common words but mean vastly different things. Labeling the words in the sentence or the sentiment of the statement are the examples a machine needs to make sense of similar language.

Labeled examples teach a machine that "CNN" can mean the cable channel in one context and "convolutional neural net" in another. It's really about the context in which you find that abbreviation and the language that surrounds it that clue in human listeners. Machines need to be shown those rules as well.

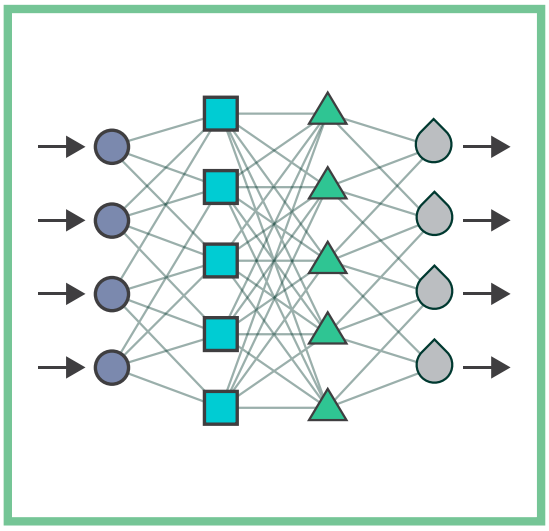
The same goes for audio. We're fairly good at understanding different vernaculars and accents, but that's because we've heard them and learned to decipher the nuances of different speech styles. An audio assistant trained by AI practitioners in California will almost certainly understand a mom in Silicon Valley when she's verbally ordering something. But until it's heard a woman in Alabama order the same, it might have trouble understanding just what's she's saying.

Simply put: identifying anything–a word, an object, a sound, a product, a moving image–requires examples. None of us knew what a dog looked like until we were told and given an example, then internalized, learned, and extrapolated from it.

Machines need the same things. The big thing to remember is: they need a lot more examples. And those examples need to be meticulously labeled.

CNN – NEWS

VS.



CNN – CONVOLUTIONAL NEURAL NET

WHY DOES DATA NEED TO BE LABELED ANYWAY?

The short answer: so that a machine can actually understand it.

There's an old Indian parable here that can actually get us into the discussion. It goes as follows:

> A group of blind men heard that a strange animal, called an elephant, had been brought to the town, but none of them were aware of its shape and form. Out of curiosity, they said: "We must inspect and know it by touch, of which we are capable". So, they sought it out, and when they found it they groped about it. In the case of the first person, whose hand landed on the trunk, said "This being is like a thick snake". For another one whose hand reached its ear, it seemed like a kind of fan. As for another person, whose hand was upon its leg, said, the elephant is a pillar like a tree-trunk. The blind man who placed his hand upon its side said, "elephant is a wall". Another who felt its tail, described it as a rope. The last felt its tusk, stating the elephant is that which is hard, smooth and like a spear.

In most versions of this story, the blind men start arguing. There's no way they all touched the same animal. After all, their experiences were markedly different. Eventually, they start listening to each other, collaborate, and start piecing together what an elephant actually is. It's not that any of them were wrong. It's that they all had an incomplete picture of what "elephant-ness" actually is.

Machines behave a lot like this. If you train on algorithm only on pictures of an elephant from the front, it will have a lot of trouble identifying one from a profile image. That's because it simply hasn't seen that angle.

Unlike the child whose parent tells her, "that's a dog," machines don't have the luxury of a mom describing the world to them. What they do have, however, is labeled data. And while a child can continue observing a dog from multiple angles, understanding how it moves, what it sounds like, how it feels, and on and on, essentially seeing hundreds, if not thousands of "labeled images" of the animal, a machine needs myriad, discrete examples of an object or word or concept to truly understand it. That's where labeling comes in.

See, to a machine, a picture is simply a series of pixels. Those pixels have values that correspond to colors but those pixels don't have values that represent the object, just a tiny dot on a massive canvas of other pixels. But labeled images show machines that certain collections of pixels are certain objects. And labeling images is something that's best done by humans-in-the-loop.

Let's go back to ImageNet. Every image in that dataset was labeled by a person. The end result: thousands of examples of different objects. From those labels, machines can make sense of the pixels of which they're made up.

Now, image labeling can be done in many different ways. You can run rudimentary labeling tasks like "is there a dog in this picture," but it's going to take a ton of images for a machine to start to understand that dataset. It's usually better practice to use bounding boxes, dots, or to actually label an image pixel by pixel.

Generally speaking, the more examples a machine sees, the better it understands. This usually holds true no matter the use case—images, text, audio, what have you. You can run into overfitting problems if you have too much of a certain label in your training data, but that's a problem we'll discuss (and help you solve) in the second half of this guide.

The point is that the data you have likely isn't the data you need to create effective machine learning algorithms. It's far more common that the data you have needs to be labeled or annotated in some way, shape, or form so that a machine can learn actually understand it. And the more labels a piece of data has, the more complicated an ontology it can create.

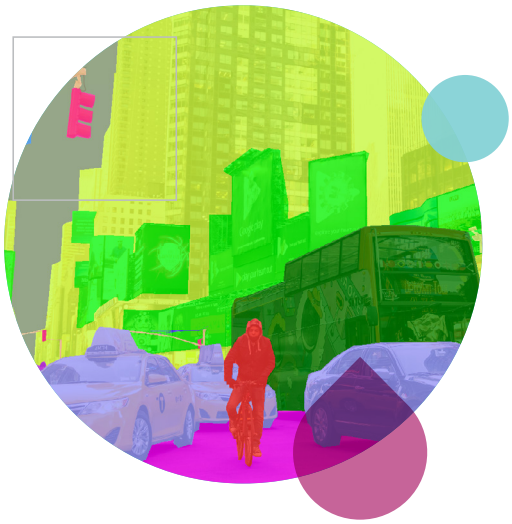
So far, we've covered two important concepts:

- Machines understand the world by learning from examples
- Humans need to labeling data so a machine understands what the examples mean.

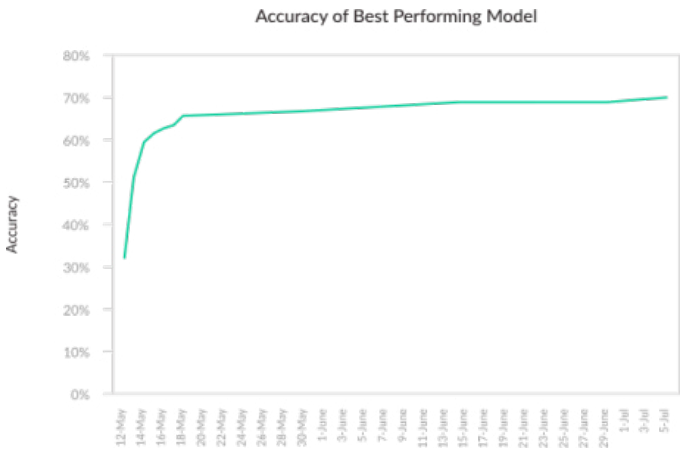
Now, let's look at a key concept you may remember Peter Norvig popularizing a while back.

THE UNREASONABLE EFFECTIVENESS OF DATA

A while back, we sponsored a competition on Kaggle. Kaggle, in case you aren't familiar, is a site where organizations can run data science competitions and practitioners compete to craft the most effective algorithms. These contests range from fun stuff like "Santa Gift Matching Challenges" to more professional-grade model building like speech recognition on TensorFlow or object detection. Teams constantly update their models, and, with contests running over weeks, some teams will submit hundreds of models just by themselves.



Our competition was about creating a search relevance algorithm from a modestly-sized dataset of semi-random products. But the particulars of that dataset aren't really what's relevant. Instead, look at what happened to the top algorithm accuracy over time:



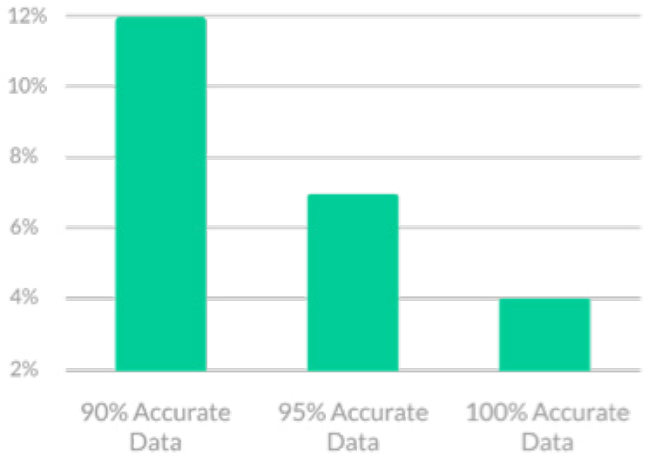
See that? In the first throes of the competition, models improved immensely as smart data scientists found features and iterated on promising avenues. But as time wound on, their models barely got better, improving by fractions of fractions of a percent.

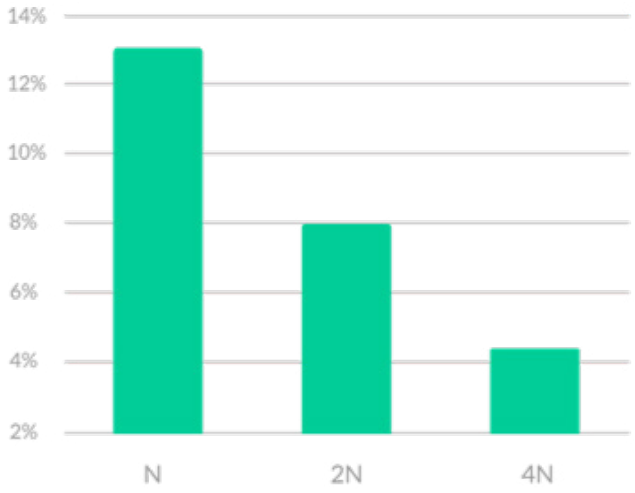
So why exactly did that happen? Simple. Because that's as good as the algorithm could get with the data it had.

At a certain point, without the inclusion of more and better training data, models will simply plateau. Yes, you can find marginal improvements, always, but those improvements are of vanishing importance.

Of course, that doesn't mean your models can't get more accurate and confident. Because even when you've squeezed what you can out of the data you have, marked improvement comes from, you guessed it, more training data.

Now take a look at this:





Both graphs show the benefit of data quantity and quality on the same algorithm. A classifier with an error rate of 13% gets nearly twice as accurate with twice as much data. If you give it four times as much data? Its error rate drops to less than 5%. And the same general principle holds for cleaner, more accurate data.

This is exactly what Norvig was referring to when he wrote about the unreasonable effectiveness of data.

The best data scientists working with the same data will arrive at very similar solutions. That's what Kaggle contests–and not just ours, mind you–show time and time again. But more and cleaner data are simply more effective than better algorithms. **Nothing helps machine learning projects more than improving the data they run on.**

But not all data is created equal.

HOW TO MAKE THE DATA YOU HAVE THE DATA YOU NEED

The short answer: by labeling it.

The reality is, most data is messy or incomplete. At least as it applies to machine learning, that is.

Take a picture for example. To a machine, an image is just a series of pixels. Some might be green, some might be brown, but a machine doesn't know this is a tree until it has a label associated with it that says, in essence, this collection of pixels right here is a tree:

If a machine sees enough labeled images of a tree, it can start to understand that similar groupings of pixels in an unlabeled image also constitute a tree.

Not every piece of data is like this. Take an example from earlier in this guide, the corpus of grand master chess games. That data is plenary. It's complete. Nothing needs to be added because the data itself is a series of chess moves and a series of chess moves is the entirety of the game.

But that's simply not the kind of data most machine learning projects are built upon. Sentiment algorithms need in-domain labels to understand the vernacular and conventions of a platform. Audio data needs to be translated to words that a machine can more easily understand so that it can make sense of the myriad ways people say the same words and phrases.

So how do you prepare training data so that it has the features and labels your model needs to succeed? **The best way is with a human-in-the-loop. Or, more accurately, humans-in-the-loop.**

Take Facebook, for example. A few years back, they published a report that boasted the accuracy of their facial recognition algorithm known as DeepFace. In fact, DeepFace's accuracy rivaled that of people, hovering around 97% correct. And without taking anything away from the machine learning experts behind the project, there's simply no way DeepFace succeeds without human labels.

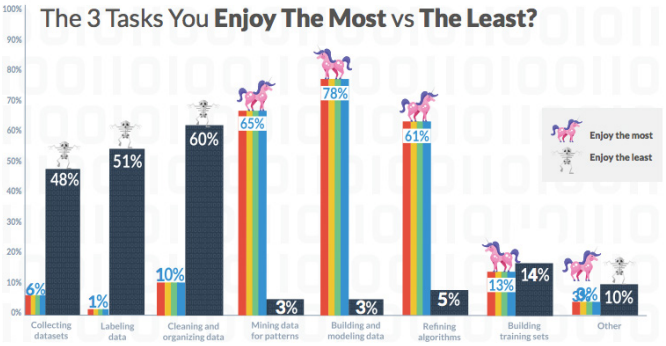
The catch here is that those labels are provided by all of us.

Every time you tag your significant other, your best friend, your mother, child, or coworker, **that's a human label**. You're pointing to a collection of pixels and saying, "this is Ian" or "this is Ashley." Those labels, in conjunction with our relationships and "friend" circles, make up the ingredients of that model's success. Without them, DeepFace doesn't work. Or, at the very least, it doesn't work very well.

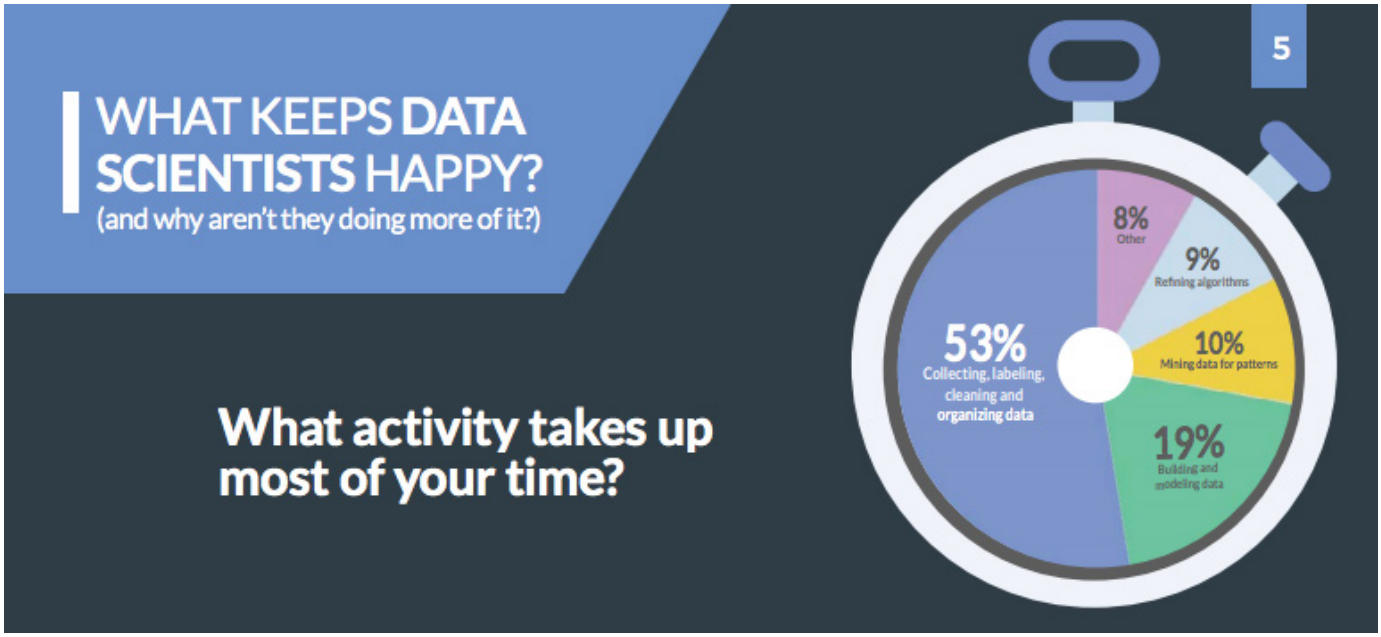
Of course, not every company can count on a user base of human labelers. Very few can, in fact. That audio data we just mentioned above? That sentiment data? It needs people to add discrete labels, place item into their proper categories, or otherwise annotate it so your models can make accurate predictions.

That's where Figure Eight can help. We've been preparing training data for machine learning for nearly a decade and understand how to leverage human intelligence at scale to get you the training data you need for your AI projects. We've worked with video, images, audio, and text and chances are, if you need more and better training data, we can provide it for you with human-in-the-loop annotations.

Because there's another dirty secret that most practitioners know but many executives simply don't: data and machine learning professionals spend way too much time preparing their own data.



What's more? They don't particularly enjoy it.



After all, labeling and collecting data isn't what they went to school for. It's not what their expertise should be squandered on. Because, really, most of these labeling projects are fairly rote. But as we've learned, you need a lot of quality data to make smart models. And even if your model is already performing, more data will make it perform better. After all, data is unreasonably effective.

These human-in-the-loop machine learning workflows are sometimes called active learning. That said, active learning, often refers to algorithm tuning as opposed to the preparation of training data, but this can be an important part of model success as well. Typically, humans will look at an output–say, a model's prediction about whether an image is in fact a dog–and verify or correct that output (i.e. "yes, this is a dog" or "no, this is a cat"). That's active learning–where people are actively teaching a model and helping shed clarity on some of its blind spots.

THIS IS NOT AN OUTPUT FOR "DOG"

Remember that graph a few sections ago? The one that showed how more accurate data improves models? This active learning process is the cousin to that approach. Gauging your model's accuracy with humans-in-the-loop is a simple but highly effective way of drastically increasing its performance. The constant feedback loop has long been the best way to create models that work in the real world.

But this is a guide about training data. So far, we've established:

- Machines learn from examples (a.k.a. training data), just like people
- Training data is as important–if not more important–than the algorithm itself
- High quality and high quantities of training data are the surest way to improve models
- Training data needs labeling to be truly useful
- Training data labeled by humans is the most accurate way to do this; often it's the only way.

Let's dig in a little deeper though.

A FEW IMPORTANT THINGS YOU SHOULD KNOW ABOUT TRAINING DATA

We've been in the training data business long enough to not only hear the most common questions and concerns but to solve them. We'll start with some basics and move into some more technical bits later on.

HOW MUCH TRAINING DATA DO YOU NEED?

Anyone who's dealt with a lawyer will recognize the classic response here: it depends.

There are a lot of factors in play for deciding how much training data you need. First and foremost is how important accuracy is.

Say you're creating a sentiment analysis algorithm. Your problem is complex, yes, but it's not a life or death issue. A sentiment algorithm that gets to 85 or 90% is more than enough for most people's needs and a false positive or negative here or there isn't going to substantively change much of anything.

Now, a cancer detection model or a self-driving car algorithm? That's a whole different story. A car that's 85 or 90% safe is actually remarkably unsafe and should never see the road. A cancer detection model that could miss important indicators is literally a matter of life or death.

Of course, more complicated use cases generally require more data than less complex ones. A computer vision that's looking to only identify foods versus one that's trying to identify objects generally will need less training data as a rule of thumb. The more classes you're hoping your model can identify, the more examples it will need.

Which is all to say: the amount of training data you'll need is contingent on the complexity of your ontology and how necessary high levels of accuracy are.

Interestingly, that cancer model we teased above? That needs far less data than an autonomous vehicle model will need. That's because a cancer classifier is looking at cells whereas a self-driving car model will need to identify everything from other cars and pedestrians to street signs and medians, among countless other classes. Though the cancer model's accuracy is incredibly important, it should need far less data to get to an acceptable accuracy threshold than one looking to drive on city streets.

It's worth resurfacing something here: there's really no such thing as too much data. Better training data, and more of it, will improve your models. You need to set the threshold for success, but know that with careful iterations, you can exceed that with more and better data.

WHY YOU NEED SEPARATE TRAINING AND VALIDATION DATASETS

Typically, when you're building a model, you split your labeled dataset into training and validation sets (though, sometimes, your validation set may be unlabeled). And, of course, you train your algorithm on the former and validate its performance on the latter.

What happens when your validation set doesn't give you the results you're looking for? If you're like most folks, you'll update your weights, drop or add labels, try different approaches, and retrain your model. But when you do this, it's incredibly important to do it with your datasets split in the exact same way.

Why is that? It's the best way to evaluate success. You'll be able to see the labels and decisions it has improved on and where it's falling flat. Different training sets can lead to markedly different outcomes on the same algorithm, so when you're testing different models, you need to use the same training data to truly know if you're improving or not.

That said, you need to be careful here:

BE AWARE OF WHAT LABELS YOU USE TO EVALUATE SUCCESS

Your training data won't have equal amounts of every category you're hoping to identify. A sentiment classifier won't be split 33% positive, 33% neutral, and 33% negative. It's just not how randomly selected training datasets work. And the most of a particular label you have, the higher the chances it will perform well on that particular label. To use a simple example: if your computer vision algorithm sees 10,000 instances of a dog and only 5 of a cat, chances are, it's going to have trouble identifying cats. As we've covered, machines need training data as examples to learn from. Without them, it simply has trouble learning.

The important thing to keep in mind here is what success means for your model in the real world. If your classifier is really just trying to identify dogs, then it's less-than-stellar work on cats or fish identification is probably not a deal-breaker. But you're going to want to evaluate model success on the labels you'll need in production.

Now, what happens if you simply don't have enough information to reach your desired accuracy level? Chances are, you'll need more training data. Models built on a few thousand rows are generally not robust enough to be successful for large-scale business practices.

HOW TRAINING DATA CAN SOLVE FOR OVERFITTING

If you have too much of a certain label in your training set, you could run into problems with overfitting. To go back to our toy problem above, where you have 10,000 dog instances and a handful of cat labels out of, say, 12,000 data rows total. If you push your model into production, chances are, it's going to assume a lot of animals are dogs. That's because, for your model, somewhere close to 80% of its training data is dogs. It's going to see dogs everywhere.

A more even distribution of training data helps. So does increasing your training data, adding other labeled rows that aren't part of the overfit class. In other words, show your model more cats and it will start to learn the distinctions between a class it already understands and one it doesn't. With retraining and additional (or better) data, you can often solve your overfitting problem.

It's worth pointing out that active learning practices can help here. When your model is a bit overconfident about a certain class, using human judgements to correct it can be a big help. Remember: human-in-the-loop machine learning should never mean "just label some training data." You can also test and tune your algorithms with human judgments.

WHY CANONICAL TRAINING DATASETS MATTER AND HOW TO USE THEM

A computer can learn to "see" from a large, canonical dataset and then be tuned to see a specific set of classes that weren't in the original dataset at all. This is usually referred to as transfer learning and it can be a great way to create smart models when your training dataset is a bit smaller than you'd like.

Here, consider something like the ImageNet dataset we mentioned previously. You can train your model on ImageNet and it will learn the images labeled there, but more importantly, it will understand how to "see." It will about edges, for example, figuring out where one object starts and another ends. Once it's learned to see, you can retrain the final layers in your neural net with the labels you care about.

In this way, a model that was trained on something trivial (say, pizzas and chairs) can be retrained to understand microscopy images and help diagnose cancer.

FIVE PUBLIC DATASETS YOU CAN USE TO BOOTSTRAP YOUR MODELS

Open Images (v4)

Served in collaboration with Google, Open Images v4 is a dataset of nearly 2 million annotated images. 600 object classes are represented here and this dataset can provide a great starting place for object recognition or computer vision algorithms.

Handwriting Recognition

Transcriptions of over 400,000 handwritten names from various cultures to inform Optical Character Recognition (OCR) models. Dataset includes an image of the name and the transcription of that image.

Medical Speech, Transcription, and Intent (English)

Useful audio datasets can be particularly difficult to find. This dataset contains over eight hours of audio utterances paired with text involving common medical symptoms and scenarios.

Parking Sign Detection

This dataset contains images of parking signs in different shapes, colors, orientations and sizes collected from different neighborhoods in San Francisco and annotated using Figure Eight platform, enabling model training for detecting parking signs in the city. These annotated parking signs can help train OCR models to understand relevant signage for parking and self-driving cars, teaching models to ignore store signage, billboards, and other potentially confusing outdoor text.

Medical Information Extraction

This dataset contains 3,984 medical sentences extracted from PubMed abstracts and relationships between discrete medical terms were annotated. This dataset focuses primarily on "treat" and "cause" relationships, with 1,043 sentences containing treatment relations and 1,787 containing causal ones.

HOW FIGURE EIGHT CAN HELP

At Figure Eight, we understand training data. We've labeled billions–with a "B"–rows of data. We've annotated millions of images. We've supported hundreds of real-world machine learning projects. We can help prepare your data for whatever sort of initiative you want. What's more, we can evaluate your algorithms to help tune them with active learning.

If you'd like to learn a bit about how we can help on your particular initiative, just shoot us an email. We'd be happy to help!

HOW DOES IT WORK? IT'S ACTUALLY PRETTY SIMPLE.

1. You upload a dataset you need annotated, enriched, categorized, or otherwise organized. This dataset can be as large as several hundreds of thousands of rows.

2. You give human labelers instructions for how you need your data annotated. For an image, you can ask for bounding boxes or pixel labels. You can have people look up URLs or classify businesses. You can labelers them judge or summarize text. It's all about what you need out of your training data and what your model will be doing in the real world.

3. Figure Eight recommends creating some gold rows (we call them test questions) where you can help train and test contributors on your job. This is a key quality control measure, making sure that human labelers who understand your job can actually work on it. We employ multiple quality measures, including contributor tracking and leveling, redundancy, and more.

4. Then, just launch your job. Humans-in-the-loop will add the labels you need to your data.

5. Once they're done, you download enriched, improved data to use on your machine learning project.

Figure Eight is the essential Human-in-the-Loop AI platform for data science and machine learning teams. The Figure Eight software platform trains, tests, and tunes machine learning models to make AI work in the real world. Figure Eight's technology and expertise supports a wide range of data types – text, image, audio, video – and use cases including autonomous vehicles, intelligent chat bots, facial recognition, medical image labeling, aerial and satellite imagery, consumer product identification, content categorization, customer support ticket classification, social data insight, CRM data enrichment, product categorization, and search relevance. The Figure Eight platform operates at an unprecedented scale having generated over 10 billion data labels to power AI applications.

Headquartered in San Francisco with a presence in Tel Aviv and backed by Canvas Ventures, Trinity Ventures, Industry Ventures, Microsoft Ventures, and Salesforce Ventures, Figure Eight serves Fortune 500 and fast-growing data-driven organizations across a wide variety of industries. For more information on the company, visit figure-eight.com