The Heavy Tail Safety Ceiling

Philip Koopman
Carnegie Mellon University

Abstract

Creating safe autonomous vehicles will require not only extensive training and testing against realistic operational scenarios, but also dealing with uncertainty. The real world can present many rare but dangerous events, suggesting that these systems will need to be robust when encountering novel, unforeseen situations. Generalizing from observed road data to hypothesize various classes of unusual situations will help. However, a heavy tail distribution of surprises from the real world could make it impossible to use a simplistic drive/fail/fix development process to achieve acceptable safety. Autonomous vehicles will need to be robust in handling novelty, and will additionally need a way to detect that they are encountering a surprise so that they can remain safe in the face of uncertainty.

Reasonable Behavior

Beyond the need to follow accepted safety engineering practices, a significant challenge in validating the safety of autonomous vehicles is ensuring that they will behave in a reasonable way when something unusual happens (i.e., a "surprise"). The scope of potential surprises is broad, and includes exceptional situations that are unforeseeable for practical purposes. Deploying self-driving cars on public roads requires addressing a number of topics in this area, including:

- Defining what "reasonable behavior" actually is. This includes not only detailed traffic rules, but also appropriate handling of exceptional situations.
- Creating robust descriptions of operational scenarios, obstacles, and environments relevant to expected vehicle operations. On-road surprises should be infrequent.
- Finding a way to ensure that the system's behavior is robust to novelty and surprises. In particular, the system should not be brittle when encountering inputs that are only slightly different than training data.
- Ensuring that the system behavior is appropriately humble. False confidence in interpreting the environment can lead to mishaps. The system should be good at knowing when it doesn't know what's going on.

While human drivers are certainly not perfect, any human driver has many years of experience in perceiving objects and events in the real world, and building predictive mental models as to what is likely to happen next. Humans build upon those skills that aren't specific to the driving task when they learn to drive. A particularly important safety skill is having enough self-awareness for a driver to realize that it's unclear what's happening and take steps to reduce risk until the uncertainty is resolved. In other words, it's important to recognize when a surprise driving situation is occurring.

Heavy Tail Distribution

The reason that detecting novel situations is so important is that there is an essentially infinite supply of surprises awaiting in the real world, even for simple deployment concepts. While human drivers are imperfect, they are incredibly adaptable. Machines, on the other hand, can be brittle in unexpected ways.

Consider a system which is pretty good, but not quite as safe as it needs to be, and how that might be fixed. As a hypothetical example, assume that potentially fatal "surprises" are showing up about once every 1 million miles in on-road testing. The question is, will finding and fixing these surprises as they appear make the system safer? The answer is that it depends upon the statistical arrival rates of the surprises.

Hypothesize that there are 100 total surprises awaiting the system, with each surprise arriving on average every 1 million miles. That means each individual surprise happens every 100 million miles. A test program of perhaps one or two billion miles could potentially identify, correct, and validate mitigation of all the surprises with adequately high probability.

But, on the other hand, consider the possibility of 100,000 total surprises awaiting the system, with each type of surprise arriving once every 100 billion miles. The average surprise arrival rate is still once every million miles. But you'd need to repeat a test/fail/fix cycle many times longer – perhaps a *trillion* miles – to be reasonably sure of mitigating all the surprises.

While real systems will have varied average arrival rates, the important point here is that many things in life have a heavy tail distribution, in which a significant fraction of the population arrives very infrequently.

The Heavy Tail Safety Ceiling

A heavy tail safety ceiling will exist for autonomous vehicles to the degree that there is a population of surprises with average arrival rates that are long compared to development and validation exposure, but short enough that an operational fleet will encounter them on a regular basis. Keeping things simple, if a particular type of surprise happens less often on average than the total development and validation test exposure, then probably it won't be seen until deployment. That in turn means that when a larger scale deployment encounters that surprise, it can potentially result in a mishap if not handled robustly.

Consider the population of latent surprises, which is the set of surprises not seen during development and validation. Each type of latent surprise has an arrival rate. The total population of latent surprises has an aggregate arrival rate faster than the arrival rate of any individual surprise (i.e., *some* surprise can happen relatively often, even if individual types of surprises each happen only rarely).

The heavy tail safety ceiling problem occurs when (a) the total population of latent surprises is relatively large, and (b) the aggregate arrival rate of unacceptably risky latent surprises is more frequent than the system safety target. In this situation the system is not acceptably safe. Worse, fixing surprises as they arrive won't resolve the problem, because mitigating one type of surprise will not substantively change the size of the large latent surprise population.

Mitigating heavy tail surprises likely requires more than just a billion miles of data collection and testing. If you can drive a billion miles with very few mishaps, then you might be able to infer you are good enough to deploy. (There are many caveats to that approach.) However, if the billion miles of driving reveals too many surprises, it might be impracticable use a repeated drive/fail/fix cycle to attain safety. The pool of latent surprises can simply be too big to be discovered and mitigated via a brute force approach.

Mitigating Heavy Tail Problems

Achieving a high level of safety with autonomous vehicles is likely to require a three prong approach: explore as deeply into the heavy tail as is practicable; encourage robustness in system behavior so that fixing one surprise has a chance of also fixing other similar surprises; and ensure that the system is good at knowing when it doesn't know what's going on.

The first step in dealing with a heavy tail ceiling situation is to validate with enough realistic data that you in fact are able to realize you've hit the heavy tail ceiling. In other words, first get all the low hanging fruit. At that point it is possible you'll be safe enough, but let's say for the sake of argument that the arrival rate of surprises is still too high after a brute force drive/fail/fix campaign.

The next step is to try to make your system more robust to surprises. For some surprises this can be done by hypothesizing a generic version of a surprise to improve system robustness. Consider, as an example, a sensor failure caused by a plastic bag blowing onto a vision sensor. A narrow fix would involve detecting or removing a single-sensor occlusion. A more robust fix would encompass the possibility of multiple sensor occlusions by either a batch of debris or a much larger single piece of debris. (For example consider a tarp blowing off a gravel truck that covers your own vehicle entirely. True story.)

Genericizing surprises to improve robustness should encompass novel operational environments, novel obstacles, and other aspects of the system's operation. This sort of approach might be able to depopulate the heavy tail surprise space more quickly by addressing a group of related surprises after only the first surprise of a particular type has been seen. But it is unlikely to be enough, because such an approach is limited to surprises that have been detected in some way during development and validation. There will always be unexpected types of surprises lurking out in the real world that haven't been seen yet.

Dealing with unknowns that are unknowable until after deployment requires dealing with the system's ability to understand its own limits. At some point a system needs to be able to know that it doesn't know what's going on, and do something reasonable to safe the system. This is an aspect of system robustness, which deals with the ability of a system to gracefully handle exceptional conditions.

One way to improve robustness is to inject noise into sensor values to test the system's operational brittleness. For example, injecting moderate amounts of noise in images or other sensor data should not cause catastrophic system failure. Similarly, small amounts of noise should not cause wildly varying classification results or vehicle behaviors. Rather, the result of noise that is small compared to signals from the environment should be a transition from reasonable certainty about object and scenario classification to an expression of uncertainty. In many cases it will also be appropriate to transition vehicle operation to less aggressive operational modes as uncertainty increases.

Injecting noise into a system to validate that it has a robust response is a variation on fault injection and robustness testing approaches. Application of creative and effective fault injection techniques has the potential to improve autonomous vehicle robustness and break through the heavy tail safety ceiling.

Contact Information

Dr. Philip Koopman is an Associate Professor of Electrical and Computer Engineering at Carnegie Mellon University, where he specializes in software safety and dependable system design. He also has affiliations with the Carnegie Mellon University Robotics Institute, National Robotics Engineering Center (NREC) and the Institute for Software Research. He is CTO and co-founder of Edge Case Research, LLC.
E-mail: koopman@cmu.edu
Web: http://users.ece.cmu.edu/~koopman