Uncertainty Estimation

2026-05-06 · cheerful mango Haubentaucher

I know that I know nothing.

The Why

In the previous chapter we learned to compress high-dimensional data into structured latent spaces and embeddings and even to draw new samples from those spaces. Closing we even explored and stepped through the latent feature space itself. Neither capability told us how much we should trust those embeddings or outputs for any particular input. A reconstruction looks the same whether the embedding lay in the heart of the training distribution or far from any encoded neighbour in feature space; the model has no built-in way to communicate its own uncertainty.

Neural networks are trained to predict, and they do so with impressive accuracy on familiar data, which makes them particularly misleading on unfamiliar data.

The Confidence Value and Prediction Probabilities are ambiguous. Neural networks calibrate poorly on out-of-distribution inputs by design: the softmax normalises to sum to one. Confidence collapses to a measure of similarity within the known domain itself and fails out-of-domain distributions. As a consequence we need to establish ways to determine the actual confidence we have in a model.

Adversarial inputs lift the problem to a sharper edge, where the model is high-confidence wrong on imperceptible perturbations.

[1]

Calibration[1]. In a supervised setting the accuracy of the model is known, and using that information the confidence can be calibrated by the actual accuracy in a specific confidence bin.

Confidence Head. A complementary route is to train a small head whose only job is to predict whether the main model is actually confident to be right on the current input, turning the model's own likelihood of being correct into a prediction target.

CONFIDENCE BY RECONSTRUCTION. In the spirit of the Variational Autoencoder Architecture, the confidence in an embedding gets varified when its reconstruction gets compared with its decoding. Even when out of domain samples fall within the latent feature space of our training data, the reconstruction collapses onto the training domain, highlighting out of domain samples.

EPISTEMIC UNCERTAINTY. While uncertainty within the data, which is called aleatoric, is a concern the models own uncertainty is the focus when brought to application. Introducting noise into our learned parameters during inference, the variance in prediction result is measured and used as an indicator for uncertainty. This approach is the basis for Monte Carlo Dropout.

APPLICATION AND ACTING UNDER UNCERTAINTY. Uncertainty estimations turn confidence into a signal that informs our downstream modules.

IN ORDER TO MOVE FROM OVERCONFIDENT PREDICTORS TO UNCERTAINTY ESTIMATIONS we have good reason to find ways to,

- separate aleatoric from epistemic uncertainty, so that each source informs the predictive confidence of our model.
- measure or predict model uncertainty in out of domain distributions.
- act on uncertainty estimates in practice, routing unreliable predictions to human reviewers, falling back to fail-safe defaults, and detecting inputs the model was never trained to handle.

THE HEAVY TAIL OF ALEATORIC UNCERTAINTY. Real-world data is heavy-tailed: A small number of common situations cover most of the volume, but a long tail of rare configurations refuses to vanish no matter how much data is collected. Autonomous vehicles thus have no means to have learned those rare events, so at least they need to realize that they move out of domain.

ODD AND DRIFT. The operational design domain names the conditions under which a system was designed to work, namely the slice of the world the training data covered. ODD detection asks at runtime whether the current input falls inside that slice. Data drift is the slow change in input statistics over time; model drift is the resulting decoupling of stated confidence from empirical accuracy.

ANOMALY DETECTION AND ON-DOMAIN LEARNING. Making a virtue of necessity. We can use uncertainty estimates for anomaly detection on one-class embeddings: Deliberately fit only normal data, then flag any input the model finds unusual without ever showing it a labelled anomaly.

SELF-REFLECTION questions to guide your thinking:

- Why does softmax turn confidence into a similarity score within the training distribution rather than an absolute correctness estimate?
- Why is high softmax confidence on an adversarial perturbation the same failure mode as overconfidence out of domain, not a separate one?
- In a reliability diagram, why does the gap at the heaviest bin determine the user-visible failure mode of a deployed system?
- Why does histogram binning close the in-domain ECE almost completely while leaving the out-of-domain ECE almost untouched?
- How does histogram binning differ in assumption and applicability from a separately trained confidence head?
- How does MC Dropout turn a single trained network into an estimator of predictive variance, and what assumption links that variance to epistemic uncertainty?
- Which kind of out-of-domain input is structurally invisible to MC Dropout, and why?
- In a DeVries-Taylor confidence head, every hint towards the label costs $-\log c$. Why does that loss alone tend to saturate at $c \to 1$, and what does outlier exposure add that closes the loophole?
- Why can the reconstruction error of a one-class VAE flag an out-of-domain input even when its latent embedding falls inside the training cloud, and what kind of input slips through that signal?
- Why do VAE reconstruction error and VAE latent geometry tend to confuse the same out-of-domain inputs, and what does that say about geometric vs semantic OOD detection?
- What separates aleatoric from epistemic uncertainty, and which of the two is reduced by collecting more training data?
- Why does the heavy tail of real-world data imply that the epistemic component cannot be driven to zero, and what does that mean for safety-critical deployment?
- How do data drift and model drift, defined relative to the operational design domain, decouple stated confidence from empirical accuracy at runtime?

- Why can a one-class generator serve as an anomaly detector without ever observing a labelled anomaly, and what does that imply for the choice of objective?
- On what axes would you compare softmax, calibration, MC Dropout, a confidence head, and a VAE OOD score when picking one for a concrete deployment?

Recap of Key Concepts:

- Softmax encodes similarity within the training distribution, not correctness, and so fails out of domain and on adversarial inputs.
- ECE measures the gap between stated confidence and empirical accuracy; histogram binning closes it on the distribution it was fit on but does not transfer.
- Uncertainty splits into aleatoric (irreducible noise) and epistemic (model ignorance); MC Dropout and a confidence head estimate the latter, and only the latter shrinks under more training data.
- A one-class VAE turns reconstruction error and latent geometry into OOD signals, and inherits a blind spot whenever the unknown looks geometrically like the known.
- The operational design domain delimits the input slice the model was trained for; data drift and model drift decouple stated confidence from empirical accuracy at runtime.

A notion of domain. Every method in this chapter inherits its notion of domain from a narrow training distribution, and the long tail of novel inputs is structurally invisible to a narrowly-trained detector. The data covers only an increment of the input space the deployed model will encounter.

Transfer learning. In order to build ever more capable models and expand beyond known domains, our target is to widen the feature space before a new domain or a new task even arrives.

Teaser. How do our embeddings need to be formalized and trained in order to generalize and transfer between tasks and domains?

feedback