- **Decomposing Activation Space.** The most fundamental challenge of any interpretability agenda is to defeat the curse of dimensionality. For mechanistic interpretability, this ultimately reduces to whether we can decompose activation space into independently understandable components, analogous to how computer program memory can be decomposed into variables. Identifying features is what allows us to decompose the model in terms of them.
- **Describing Activations in Terms of Pure Features.** One of the most obvious casualties of superposition is that we can't describe activations in terms of pure features. When features are relatively basis aligned, we can take an activation – say the activations for a dog head in a vision model – and decompose them into individual underlying features, like a floppy ear, short golden fur, and a snout. (See the "semantic dictionary" interface in Building Blocks [37] .) Solving superposition would allow us to do this for every model.
- **Understanding Weights (ie. Circuit Analysis).** Neural network weights can typically only be understood when they're connecting together understandable features. All the circuit analysis seen in the original circuit thread (see especially [38]), see specially was fundamentally only possible because the weights connected non-polysemantic neurons. We need to solve superposition for this to work in general.
- **Even very basic approaches become perilous with superposition.** It isn't just sophisticated approaches to interpretability which are harmed by superposition. Even very basic methods one might consider become unreliable. For example, if one is concerned about language models exhibiting manipulative behavior, one might ask if an input has a significant cosine similarity to the representations of other examples of deceptive behavior. Unfortunately, superposition means that cosine similarity has the potential to be misleading, since unrelated features start to be embedded with positive dot products to each other. However, if we solve superposition, this won't be an issue – either we'll have a model where features align with neurons, or a way to use compressed sensing to lift features to a space where they no longer have positive dot products.

Three Ways Out

At a very high level, there seem to be three potential approaches to resolving superposition:

- **Create models without superposition.**
- **Find an overcomplete basis** that describes how features are represented in models with superposition.
- **Hybrid approaches** in which one changes models, not resolving superposition, but making it easier for a second stage of analysis to find an overcomplete basis that describes it.

Our sense is that all of these approaches are possible if one doesn't care about having a competitive model. For example, we believe it's possible to accomplish any of these for the toy models described in this paper. However, as one starts to consider serious neural networks, let alone modern large language models, all of these approaches begin to look very difficult. We'll outline the challenges we see for each approach in the following sections.

With that said, it's worth highlighting one bright spot before we focus on the challenges. You might have believed that superposition was something you could never fully get rid of, but that doesn't seem to be the case. All our results seem to suggest that superposition and polysemanticity are phases with sharp transitions. That is, there may exist a regime for every model where it has no superposition or polysemanticity. The question is largely whether the cost of getting rid of or otherwise resolving superposition is too high.