This appears to be a relatively standard *sparse coding* problem, where we want to take the activations of neural network layers and find out which directions correspond to features.[20]

The advantage of this is that we don't need to worry about whether we're damaging model performance. On the other hand, many other things are harder:

- **It’s no longer easy to know how many features you have to enumerate.** A monosemantic model represents a feature per neuron, but when finding an overcomplete basis there’s an additional challenge of identifying how many features to use for it.

- **Solutions are no longer integrated into the surface computational structure.** Neural networks can be understood in terms of their surface structure – neurons, attention heads, etc – and virtual structure that implicitly emerge (*e.g.* virtual attention heads [40]). A model described by an overcomplete basis has "virtual neurons": there's a further gap between the surface and virtual structure.

- **It's a different, major engineering challenge.** Seriously attempting to solve superposition by applying sparse coding to real neural nets suggests a *massive* sparse coding problem. For truly large language models, one would be starting with something like a millions (neurons) by billions (tokens) matrix and then trying to do an extremely overcomplete factorization, perhaps trying to factor it to be a thousand or more times *larger*. This is a major engineering challenge which is different from the standard distributed training challenges ML labs are set up for.

- **Interference is no longer pushing in your favor.** If you try to train models without superposition, interference between features is pushing the training process to have less superposition. If you instead try to decode superposition after the fact, whatever amount of superposition is "baked in" by the training process and you don't have part of the objective pushing in your favor.

APPROACH 3: HYBRID APPROACHES

In addition to approaches which address superposition purely at training time, or purely after the fact, it may be possible to take "hybrid approaches" which do a mixture. For example, even if one can't change models without superposition, it may be possible to produce models with *less* superposition, which are then easier to decode.[21] Alternatively, it may be possible for architecture changes to make finding an overcomplete basis easier or more computationally tractable in large models, separately from trying to reduce superposition.