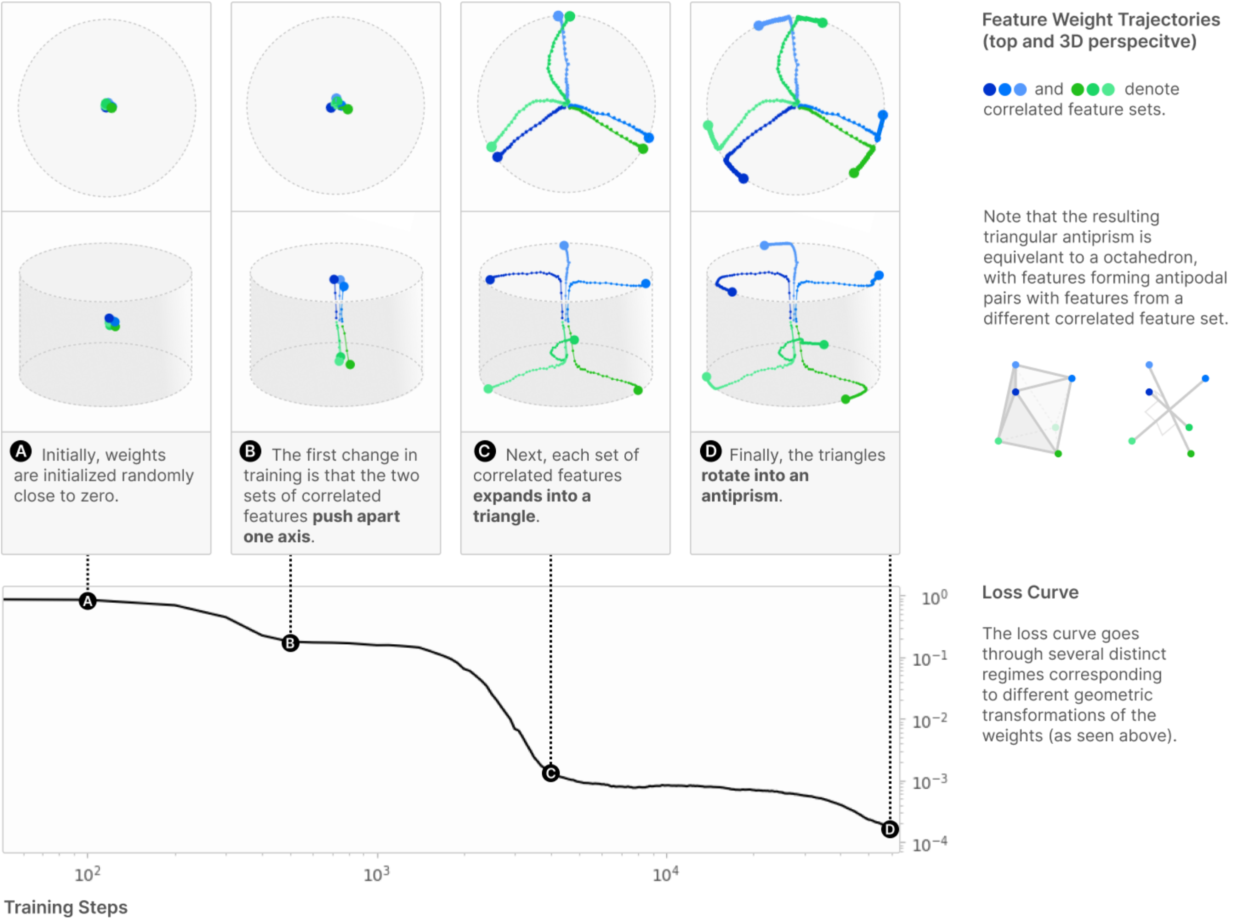


Ⓐ Initially, weights are initialized randomly close to zero.

Ⓑ The first change in training is that the two sets of correlated features **push apart one axis**.

Ⓒ Next, each set of correlated features **expands into a triangle**.

Ⓓ Finally, the triangles **rotate into an antiprism**.

Feature Weight Trajectories (top and 3D perspecitve)

●●● and ●●● denote correlated feature sets.

Note that the resulting triangular antiprism is equivelant to a octahedron, with features forming antipodal pairs with features from a different correlated feature set.

Loss Curve

The loss curve goes through several distinct regimes corresponding to different geometric transformations of the weights (as seen above).

(Although the last solution – an octahedron with features from different correlated sets arranged in antipodal pairs – seems to be a strong attractor, the learning trajectory visualized above appears to be one of a few different learning trajectories that attract the model. The different trajectories vary at step Ⓒ: sometimes the model gets pulled directly into the antiprism configuration from the start or organize features into antipodal pairs. Presumably this depends on which feature geometry the model is closest to when step Ⓑ ends.)

The learning dynamics we observe here seem directly related to previous findings on simple models. [30] found that two-layer neural networks, in early stages of training, tend to learn a linear approximation to a problem. Although the technicalities of our data generation process do not precisely match the hypotheses of their theorem, it seems likely that the same basic mechanism is at work. In our case, we see the toy network learns a linear PCA solution before moving to a better nonlinear solution. A second related finding comes from [31], who looked at hierarchical sets of features, with a data generation process similar to the one we consider. They find empirically that certain networks (nonlinear and deep linear) “split” embedding vectors in a manner very much like what we observed. They also provide a theoretical analysis in terms of the underlying dynamical system. A key difference is that they focus on the topology—the branching structure of the emerging feature representations—rather than the geometry. Despite this difference, it seems likely that their analysis could be generalized to our case.

Relationship to Adversarial Robustness

Although we're most interested in the implications of superposition for interpretability, there appears to be a connection to adversarial examples. If one gives it a little thought, this connection can actually be quite intuitive.

In a model without superposition, the end-to-end weights for the first feature are:

$$(W^TW)_0 \quad = \quad (1,\ 0,\ 0,\ 0,\ ...)$$

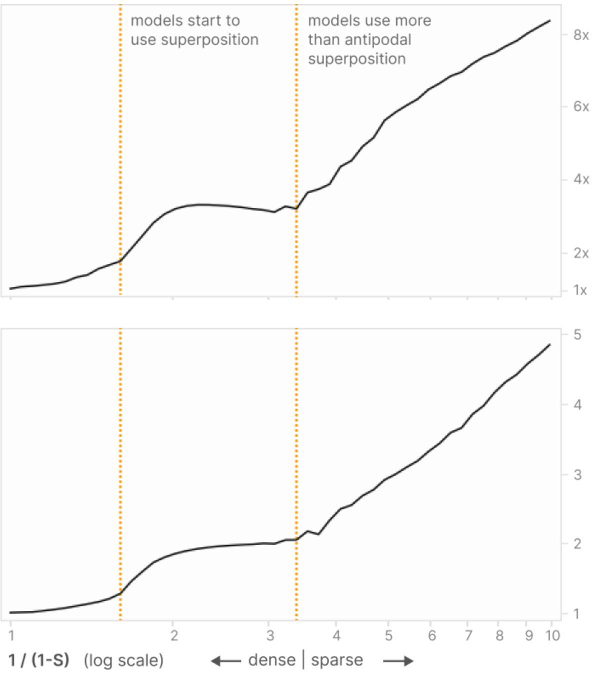
But in a model with superposition, it's something like:

$$(W^TW)_0 \quad = \quad (1,\ \epsilon,\ -\epsilon,\ \epsilon,\ ...)$$

The ϵ entries (which are solely an artifact of superposition "interference") create an obvious way for an adversary to attack the most important feature. Note that this may remain true even in the infinite data limit: the optimal behavior of the model fit to sparse infinite data is to use superposition to represent more features, leaving it vulnerable to attack.

To test this, we generated L2 adversarial examples (allowing a max L2 attack norm of 0.1 of the average input norm). We originally generated attacks with gradient descent, but found that for extremely sparse examples where ReLU neurons are in the zero regime 99% of the time, attacks were difficult, effectively due to gradient masking [32]. Instead, we found it worked better to analytically derive adversarial attacks by considering the optimal L2 attacks for each feature ($\lambda(W^TW)_i/||(W^TW)_i||_2$) and taking the one of these attacks which most harms model performance.

We find that vulnerability to adversarial examples sharply increases as superposition forms (increasing by >3x), and that the level of vulnerability closely tracks the number of features per dimension (the reciprocal of feature dimensionality).



Vulnerability to Adversarial Examples (Relative to Non-Superposition Model)

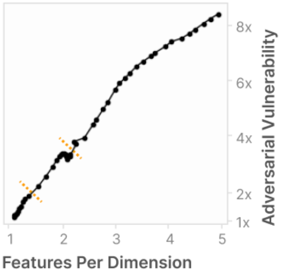
$(L' - L)/(L'_0 - L_0)$

The model's vulnerability to adversarial examples increases with feature sparsity, tracking the "features per dimension" measure of superposition plotted below.

Features Per Dimension

$1/D^* = ||W||_F^2/m$

Here we plot the number of features per dimension, the reciprocal of the "dimensionality" we defined in the pervious section. We see that the curve closely parallels vulnerability to adversarial examples.



We can also directly plot adversarial vulnerability agains the number of features per dimension. This reveals that adversarial vulnerability is highly correlated with the number of features stored in superposition per dimension.

We're hesitant to speculate about the extent to which superposition is responsible for adversarial examples in practice. There are compelling theories for why adversarial examples occur without reference to superposition (*e.g.* [33]). But it is interesting to note that if one wanted to try to argue for a "superposition maximalist stance", it does seem like many interesting phenomena related to adversarial examples can be predicted from superposition. As seen above, superposition can be used to explain why adversarial examples exist. It also predicts that adversarially robust models would have worse performance, since making models robust would require giving up superposition and representing less features. It predicts that more adversarially robust models might be more interpretable (*see e.g.* [34]). Finally, it could arguably predict that adversarial examples transfer (*see e.g.* [35]) if the arrangement of features in superposition is heavily influenced by which features are correlated or anti-correlated (see earlier results on this). It might be interesting for future work to see how far the hypothesis that superposition is a significant contributor to adversarial examples can be driven.

In addition to observing that superposition can cause models to be vulnerable to adversarial examples, we briefly experimented with adversarial training to see if the relationship could be used in the other direction to reduce superposition. To keep training reasonably efficient, we used the analytic optimal attack against a random feature. We found that this did reduce superposition, but attacks had to be made unreasonably large (80% input L2 norm) to fully eliminate it, which didn't seem satisfying. Perhaps stronger adversarial attacks would work better. We didn't explore this further since the increased cost and complexity of adversarial training made us want to prioritize other lines of attack on superposition first.