Toy Models of Superposition Replication and Findings

Zephaniah Roe zroe@uchicago.edu
Undergraduate Student at the University of Chicago

Abstract

Toy Models of Superposition[1] is a groundbreaking paper published by researchers affilated with Anthropic and Harvard University in 2022. By investigating small models with under 100 neurons, the paper demonstrates that neural networks can represent more features than they have dimensions. Additionally, they use these so called "toy models" to understand the relationship between how neural networks are trained and how they represent the data internally. Becasue the paper is quite extensive, this replication only focuses on reproducing the most important results from the introduction and sections 2 and 3 of the original paper. It also includes some comentary on section 1.

1 Introduction

Toy Models of Superposition motivates the idea of superposition with the following graphic:

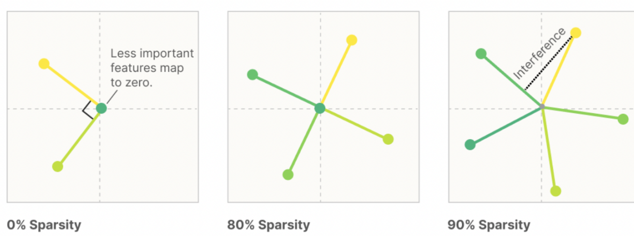

Figure 1: Graphic from *Toy Models of Superposition* showing superposition in 2D.

The basic idea is this: if you think of each feature as being represented inside of a neural network by a direction, you can graph these directions and observe them. By doing this, the authors of *Toy Models of Superposition* demonstate that the internal struture of a model depends on the sparsity of it's training data. A replication of this phenomenon can be found below and the code used to generate it can be found here.

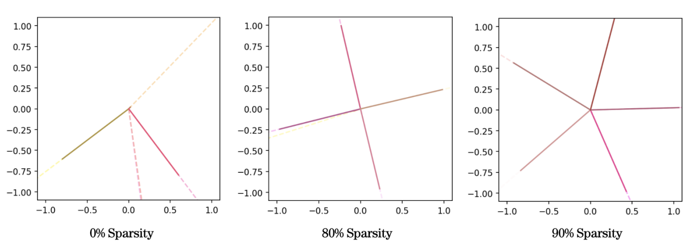

Figure 2: Replication of Figure 1

The model studied in Figure 2 is designed such that each column in the weight matrix corresponds to a given input. The weight matrix in this example contains only 2 neurons, so each column can be graphed as as 2D vector. Obersving these vectors while increasing the sparsity of the model's input reveals that the model can represent five features despite having only 2 neurons. This kind of representation is what the authors call "superposition." In future sections we will study superposition extensivly and reproduce this phenomenon in larger networks.

2 Background and Motivation

Before studying superposition in detail, the authors of *Toy Models of Superposition*, provide context by explaining concepts and defining terms. In this section, I provide some additional comentary on the definitions and explanations the authors use.

(1) Defining Features: *Toy Models of Superposition* defines features broadly as "properties of the input which a sufficiently large neural network will reliably dedicate a neuron to representing." The authors do however describe this definition as "slightly circular" and note that they are not "overly attached to it." I find the definition especially problematic because certain archietctures may be incentivized to ignore an aspect of the input that differently designed models may "want" to internally represent. It is unclear to me weather there is sufficent evidence to support the idea of a sort of ground truth for features. As a result, I propose an alternative definition: features are aspects of the input that a neural network represents accurately with a significantly higher probability than a randomly initialized network. In other words, features are parts of the input that a model determines to be important enough to represent internally.

(2) Role of Linear Representations in Neural Networks: The original authors of the paper study interpretability by trying to understand the linear representations within neural networks. It is worth noting that this isn't the only way to approach mechanistic interpretability research. Understanding the role of non-linearities at each level is likely also very important (and perhaps more neglected).

(3) Defining Superposition: The original paper has a compelling yet simple definition for Superposition: "Roughly, the idea of superposition is that neural networks 'want to represent more features than they have neurons', so they exploit a property of high-dimensional spaces to simulate a model with many more neurons." This is the definition I will use throughout this paper.

3 Demonstrating Superposition

In the introduction, the authors of the original paper proved that models with two neurons could exhibit superposition (this result was reproduced in Figure 2). Later in the paper, however, the authors demonstrate that superposition is also observed in models with more than two neurons. Specifically they begin by exploring two models with a weight matrix $W_{5\times20}$: a linear model defined by W^TWx+b and a ReLU model defined by $\text{ReLU}(W^TWx+b)$. The objective of both models was to reconstruct the input x. The authors used a weighted mean squared error loss function making accurately representing some features more important than others. For more information about the loss see section 2 of the original paper.

While the model without an activation function appears to only represent features orthogonally, the authors claim that the ReLU model can exibit superposition if it is trained on sparse enough data. Becuase we defined each weight matrix with 5 neurons ($W_{5\times20}$) this claim cannot be validated by graphing features in 2D like in Figure 2. But by graphing W^TW, superposition in the ReLU model can be shown visually in Figure 3. In this figure, positive numbers in the matrix W^TW are labeled red while negative ones are colored blue. They also graph the length of each feature by treating each column in W as a vector. Features that are orthogonal to others in W are labeled black while features that aren't are labeled yellow (the exact details for how this is calculated is discussed in 3.1).

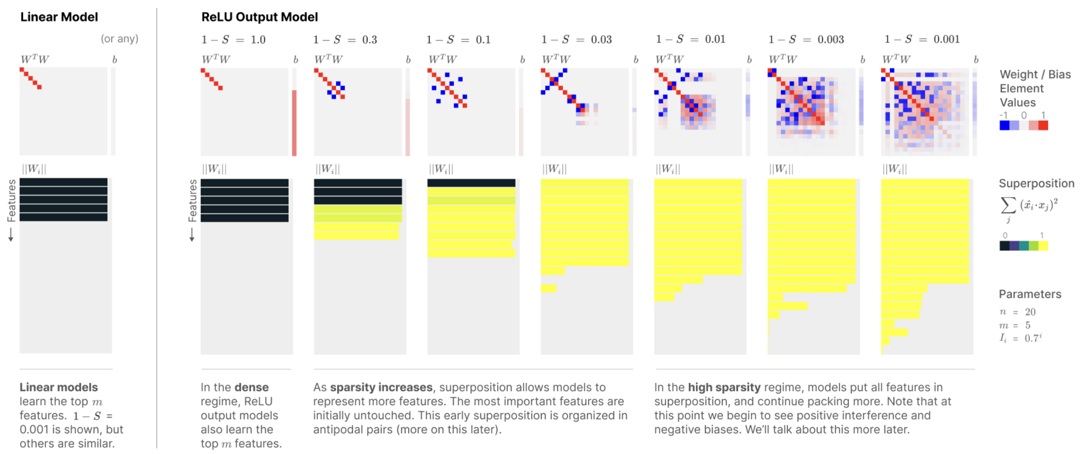

Figure 3: Superposition in linear and ReLU models from *Toy Models of Superposition*[1]

The visualisation of W^TW (top of Figure 3) and the chart of feature representations (bottom of the figure) both show that by increasing sparsity, the ReLU model ceases to represent features orthogonally. The yellow bars shown in the models on the right side of the figure illustrate that the model maps all it's features in superposition. This is also shown in the graphs of the weight matrices W^TW for sparse models in Figure 3. These representations appear to show the model embedding more than 5 features, but the representation is noisy.

The first step in replicating these findings was to train the linear and ReLU models that don't perform computation in superposition. The objective of each model was to reconstruct the input x. Both models were trained with the Adam optimizer (learning rate $= 1 * 10^{-3}$) on 20,000 batches of 256 examples.

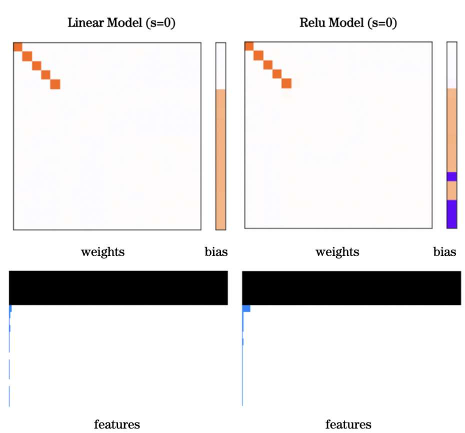

Figure 4: The generated graphics show that the linear and ReLU models trained on dense inputs represent only five features (one for each dimension in the model).

Note that in Figure 4, I use orange to indicate positive numbers and purple to indicate negative ones. This is different from the red and blue in Figure 3 to distinguish my work from that of the original authors. Similarly, while the original authors use yellow to indicate features in superposition, I use blue (This is hard to see in Figure 4 but it will be more obvious going forward).

3.1 Calculating Superposition

The models in Figure 4 do not exhibit superposition. They encode the five most important features orthogonally (one feature for each neuron in the model). In this section, we will be investigating models that do not behave in this way, instead encoding features as vectors that interfere with eachother. The model explored in this section is defined by the same architecture and objective as the models in Figure 4. The difference is that the models in this section are trained on sparse input data and, as a result, map features interally in superposition.

In order to explain this phenomenon and demonstate how models with sparse input are able to represent features in superpostion, it will be useful to dive into the math behind the concept of feature interference. The extent to which features interfere with eachother is defined by the following equation:

$$\text{Interference} = \sum_{j \neq i} (\hat{W}_i \cdot \hat{W}_j)^2 \tag{1}$$

For a given column i in weight matrix W, interference is calculated by taking the dot product with every other column in W. Non-zero dot products indicate that the columns in W are not orthogonal. As a result, summing these dot products gives a general idea of how much the network is representing a given feature in superposition. Note that $\hat{W}_i$ is the unit vector for W_i. This is necessary because when calculating interference, we are interested in the direction of a given feature, not its length.

In Figure 5, the length of a feature (calculated by taking the length of the vector W_i) determines the width of the bars in the feature graph (shown in the bottom half of the figure). The interference equation (Equation 1) determines the color of the columns: black indicates a low value for interference while blue indicates a higher value. This means that blue bars show that a given feature is represented in superposition while black bars indicate that the feature is mapped orthogonally.

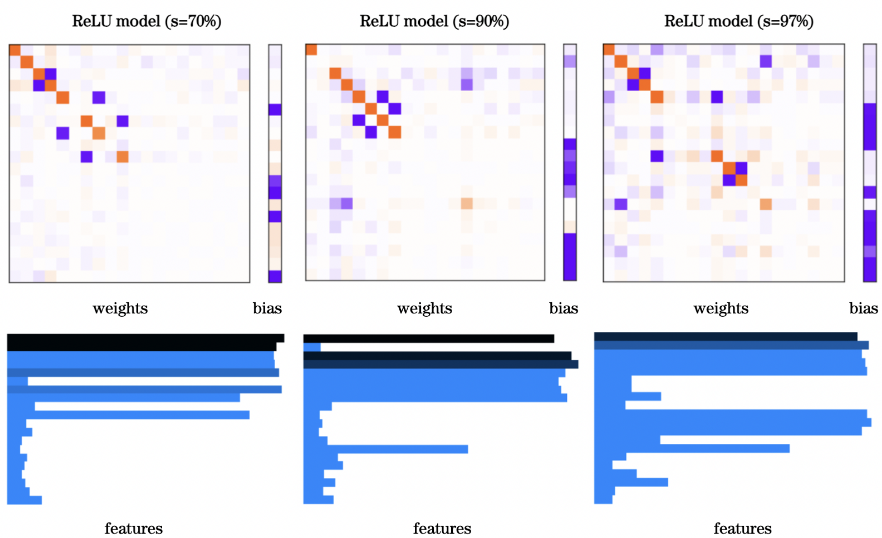

Figure 5: Superposition is observed in models trained on 70%, 90% and 97% sparse inputs (code to generate these figures can be found here). The 70% and 90% sparse models were trained on 50,000 batches of 256 examples with the "RMSProp" optimizer (learning rate = 10^{-2}). The 97% sparsity model was trained on 100,000 batches of 256 examples using the Adam opimizer (learning rate = 10^{-2}).

Unlike the models with 0% sparsity in Figure 4, the models in Figure 5 have higher levels of sparsity and, as a result, leverage superposition. The bottom half of Figure 5 shows that these models represent far many more features than the models in Figure 4, but by doing so, they are forced to represent many of their features in superposition. The models only have 5 neurons so if they "want" to represent more than 5 features, they can't represent each feature orthogonally. This tradeoff is intuatively more attractive when the model is trained on sparse inputs becase it is less likely that the model will be fed a combination of inputs that cause feature representations to conflict (because a significant percentage of the input is 0).

3.2 Models Trained on Very Sparse Data

As sparsity is increased to almost 100% the models stop representing any features orthogonally. This is displayed in Figure 6 where models are trained on 99%, 99.7% and 99.9% sparse inputs.

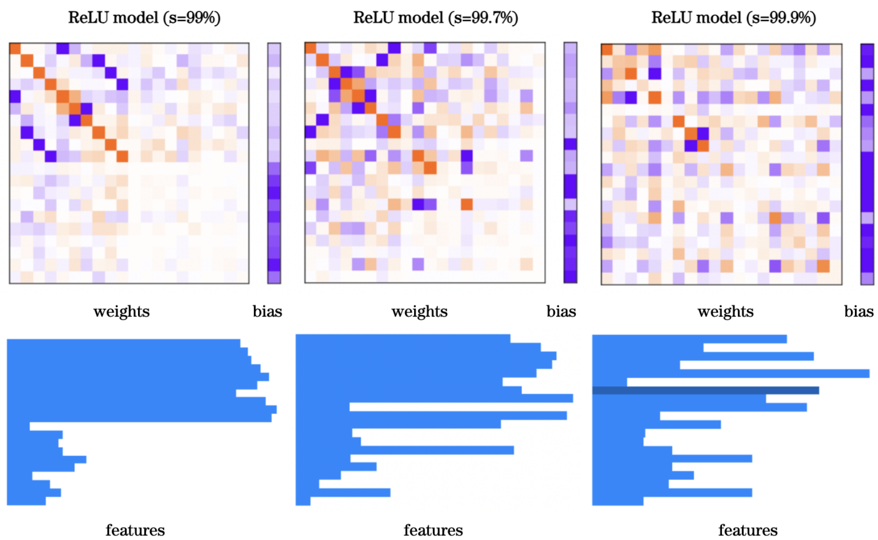

Figure 6: When models are trained on sufficiently sparse data, all feature representations are in superposition. The models in the figure were trained on 100,000 batches of 256 examples using the Adam opimizer (learning rate = 10^{-2}).

The representations of W^TW of these very sparse models (shown in the top half of Figure 6) is far less clean than previous representations we have seen. This is the same trend the original authors found when increasing sparsity of these models (Figure 3 illustrates how the original authors displayed this visually).

The feature representations, shown in the bottom have of Figure 6, are also consistent with the the findings of *Toy Models of Superposition*. Like the investigation from the original paper, these feature representations show no features mapped orthogonally (recall that features that interfere with eachother are shown in blue).

3.3 Scaling Results to Larger Models

In the previous subsections, we have explored superposition in models with 5 neurons and 20 inputs. This proves that the phenomenon we observed in Figure 6 applies to models with more than 2 neurons. The original paper expanded this finding by also studying models with 20 neurons and 80 inputs (Figure 7).

They original authors report that this experiment produced "qualitatively similar" results compared to the models with 5 neurons and 20 inputs. Because this experiment is almost identical to the previous

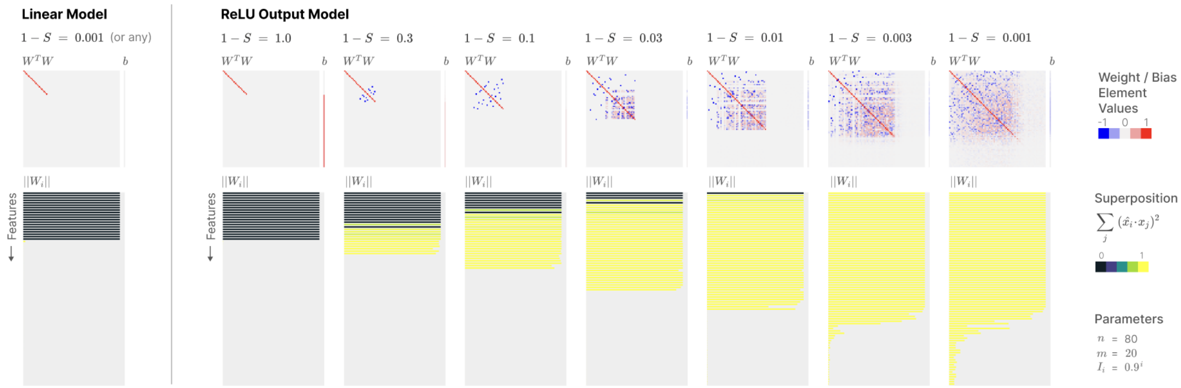

Figure 7: Superposition in models with 20 neurons and 80 inputs from *Toy Models of Superposition*[1]

ones, and becasue I would not expect to find results that conflict with the original paper, I have excluded this experiment from this replication.

4 Superposition as a Phase Change

The author's of *Toy Models of Superposition* claim that superposition can be thought of as a a kind of "phase change." Figure 8 shows a phase diagram for a single neuron model with 2 inputs. The models studied were defined by $\mathrm{ReLU}(W^TWx + b)$ and trained to simply reconstruct their inputs. The loss function used was a weighted mean squared error where the importance of the first output is always one and the importance of the second output is varied between 0.1 and 10. The relative importance of this second output is the x-axis for the graphs in Figure 8. The y-axis represents the model's feature density - in other words, the probability that a given input is not zero.

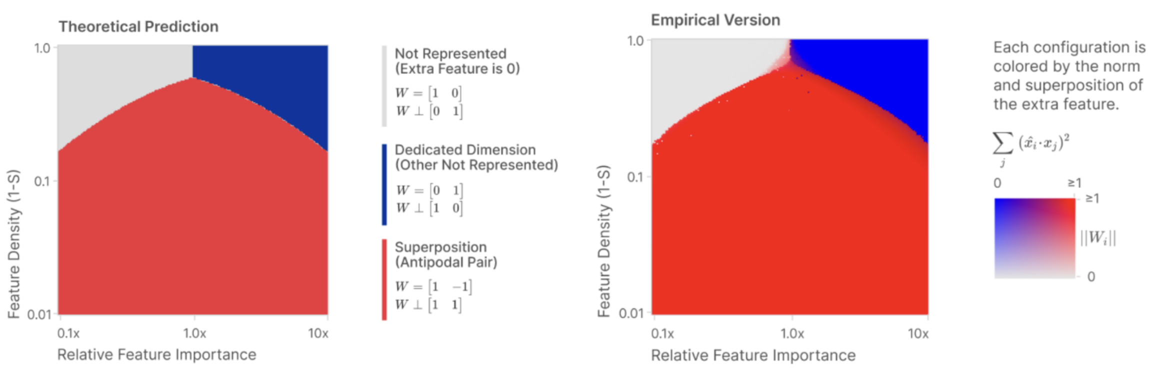

Figure 8: Theoretical and empirical superposition in one neuron models in *Toy Models of Superposition*[1]

These phase diagrams present three discreet posibilities for the way this single neuron model represents it's input internally. The first posibility is the model *only* represents the first input. In this case the weight matrix $W = [1, 0]$ (shown in grey in Figure 8). The oppositve option is also a possibility: the model could only embed the second feature making $W = [0, 1]$ (shown in blue). The third possibility is the model could represent *both* features by making $W = [1, -1]$ (shown in red).

The authors claim it is possible to calculate theorically losses based on sparisty, realative feature importance, and the weight matrix W for each of these models. *Toy Models of Superposition* links notebook which specifies the equations for calculating these theorical losses in 3 dimesions, but they should work exactly the same in this 2 dimensional example:

$$\text{Loss for when } W \text{ is } [1,0] = \frac{s}{3} - \frac{s^2}{4} \tag{2}$$

$$\text{Loss for when } W \text{ is } [0,1] = r\left(\frac{s}{3} - \frac{s^2}{4}\right) \tag{3}$$

$$\text{Loss for when } W \text{ is } [1,-1] = \frac{(1+r)s^2}{6} \tag{4}$$

The variable r is the relative importance of the second feature. The variable s is the feature density—in other words, 1 - sparsity. For information about how these equations were derived, visit this notebook provided by the authors. Based on the sparsity of the input and the relitive feature importance of the second input, the authors were able to plot which weight configuration would theoretially lead to the lowest loss in Figure 8. They were then able to train models and average the results to show that the same pattern emerges imperically when training models with gradient descent.

4.1 Replicating Superposition as a Phase Change

Replicating the findings from *Toy Models of Superposition* in the previous subsection was a complex process that it described in detail in the succeeding sections. Figure 9 is a visual summary of my findings. It will be refered to in the subsections below as a way to clarify my claims.

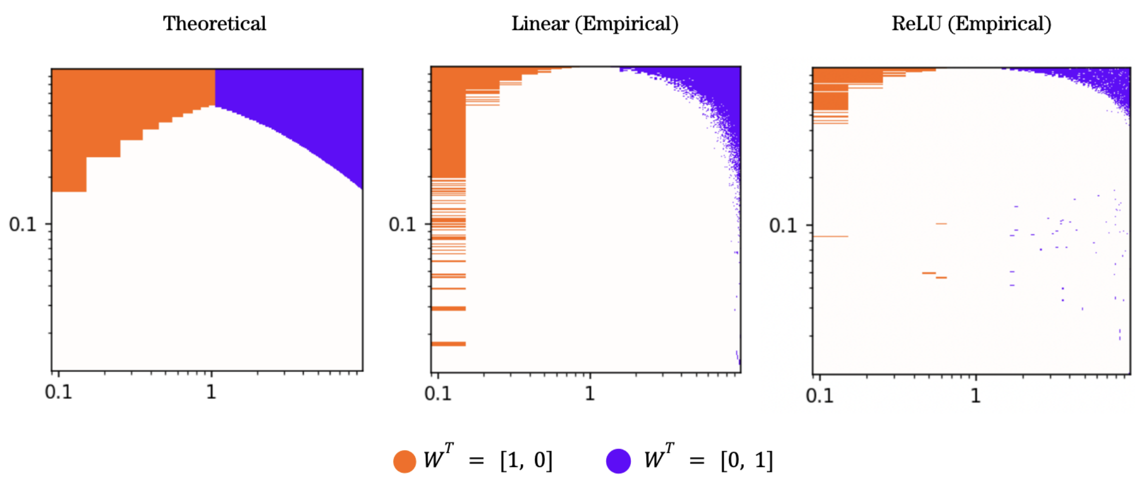

Figure 9: Theoretical and empirical superposition in one neuron models.

4.2 Replicating Theoretical Predictions

By using Equation 2, Equation 2 and Equation 2 I was able to replicate the theorical phase diagram in *Toy Models of Superposition*. The careful reader however, will notice that the theoretical prediction in Figure 8 (graphic from *Toy Models of Superposition*) looks more choppy than the one in Figure 9 (graphic generated for this replication). This is because in the theoretical prediction I generated, steps between each value on the x-axis is much larger. This however is intentional and is benificial for two reasons. First, by creating larger steps between numbers on the x-axis, I was able to intentionally tweak the step size to make the numbers easiest to work with when generating the graph. It is also benificaial because larger step sizes means that the graph includes far fewer theoretical models. This meant that making a emperical version of the theoretical graph required training fewer real models. Because my current setup is very compute-contrained, this made performing the tasks in the following subsections much more managable.

References

[1] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022.