This article breaks down what a neural network does and why those things happen. Initially, it was predicted that the network would begin to recognize loop and line patterns that makeup numbers and than connect these patterns to build a number prediction. In actuality, while a few patterns are recognized, they are seemingly random. In fact if the network is given garbage input, these random patterns will still confidently give a prediction. There are a few reasons that the network reached this result. The first reason is due to the way networks are trained, meaning e found a local minimum of the function. This local minimum is good enough to solve the problem but is not the best solution. So if this isn't the best solution how could a better one be found? The answer to this is changing how the network is trained. This network was trained with numbers of consistent sizing perfectly centered in the box and every training example had a number, so there's no reason for it ever to give a non-number result, nor did we program a way for it to do so. If the data was changed to have numbers of varying sizes in different locations of the box, then the network would have had to learn better patterns to recognize the numbers. This network should also have one more element of NAN added to the output layer that the network would hopefully select if no pattern or many patterns were present. While this isn't directly discussed in the article, I think it's worth pointing out how the live drawing example has a preprocessing button that can change the incorrect prediction to a correct prediction. This shows the importance of sanitizing data and fitting it to our model, and the preprocessor seems to shrink the drawing and smooth out the edges, which should help the network recognize the numbers. However, even preprocessing will not solve the size error as a large eight commonly predicts a two before processing and a zero after processing just from playing with the tool.