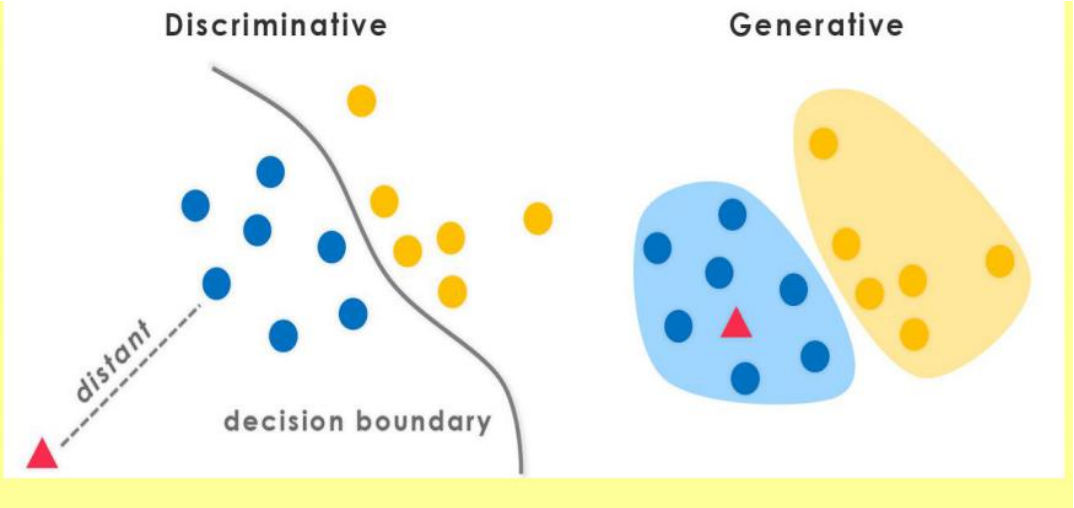
From discriminant model to understand neural network

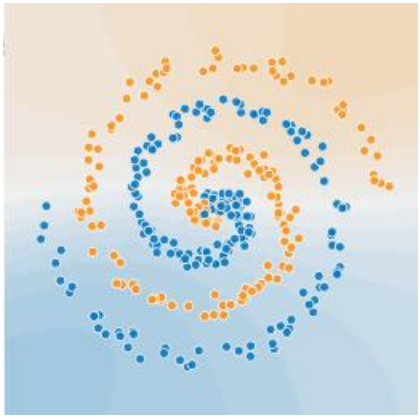
Neural network is one kind of discriminant model meaning we draw boundaries through the whole input dimensional space and put a label to each point. Although the training set become larger and larger and the final boundary can become extremely complex,due to the use of multi layers and nonlinear activation function,we can still approximate that boundary well by increasing the trainable parameters.Any two-layer(or greater) nonlinear neural network are powerful enough to approximate any function to any precision.Tons of work are studied on the representational power of neural net(https://arxiv.org/abs/1611.03530). This character leads neural network to outperform any other traditional machine learning models and keep shining with the growing amount of available data.But, as discriminant model, neural network directly map input data to labels will always suffer some inborn deficiencies.Concretely, the requirment of large amount of training set, generalize badly through transformation and easily cheated by adversarial samples. I will first discuss the discriminant characters of neural network and then elaborate my understandings of the reasons causing above mentioned deficiencies.

The discriminant characters

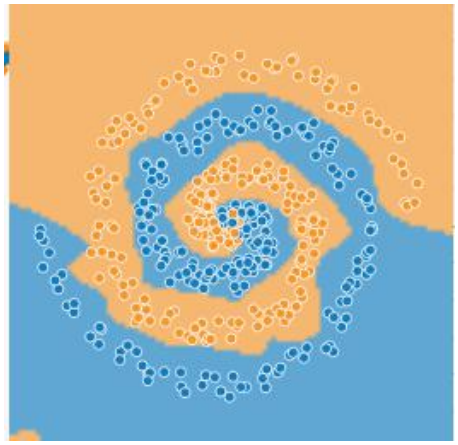
The differences between discriminant model and generative model can be elaborated in figure below:



If we want to train a neural network to approximate the function of y=f(x) and use the trained network to predict with new samples.For example, there are two classes colored by blue and orange. Because they are 2-d dots, the input are the location of each dot (x1,x2).



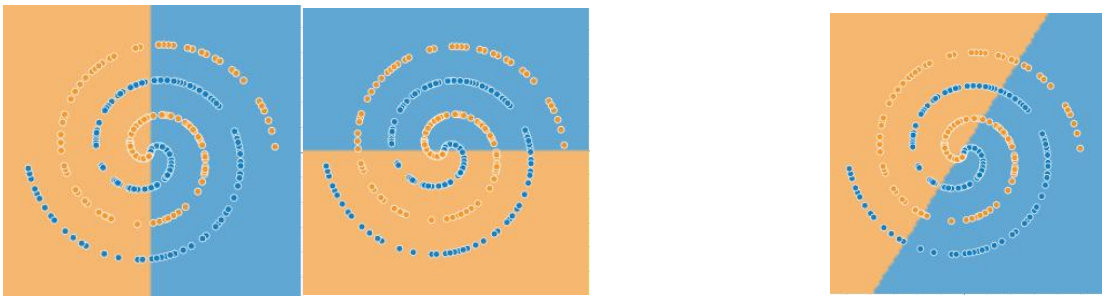
We then use neural network to distinguish them by drawing boundary(here I use 3 layers with 8 units each layer and tanh activation function), we can get one decision boundary shown below:



Units in the same layer are weighted summed(we can treat the bias term as one additional input with weight -1) then followed by 'tanh' nonlinear activation. The weights between units of this layer and units in the following layer represent 'how much ' we use the output results from previous units to create new outputs. We can take each output from previous unit is also a boundary like 'wall' to serperate the two classes, the absolute value of the weight is the height of the wall, the sign of the weight is the direction of the wall(up or down),we add all the different walls(layer 3) followed by tanh nonlinear mapping to build our final decision wall(the one output unit). So the boundary generated by unit in the output layer is the weightd sum of boundaries from previous layer(layer 3).

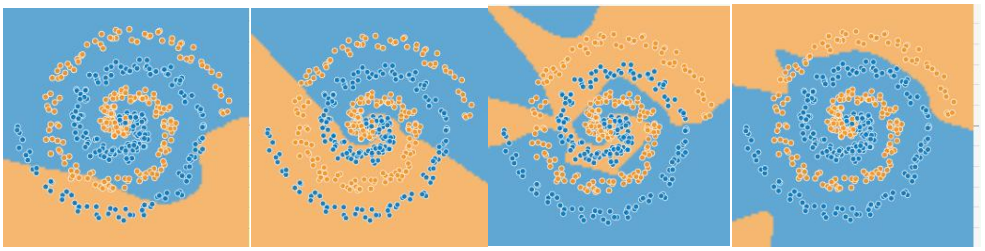
Some elaborations are shown below(blue area is positive), if we use two boundaries (x1=0,x2=0) with weights below the figure, after tanh we can get the output boundary shown in right part.

Input boundaries Output boundary

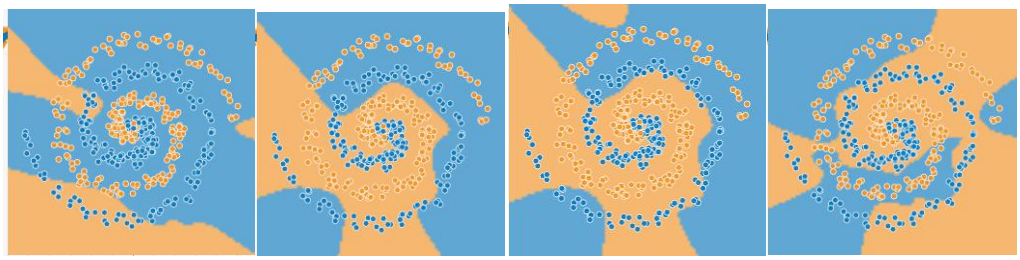


Weights: 0.41 -0.26

We now back to the boundary generated by the three-layer neural network. The final well-shaped boundary generated by the output unit is weighted sum of the third layers output boundaries. I list all the 8 output boundaries below and their weights to verify.



-0.43 -2.3 -1.3 2.6

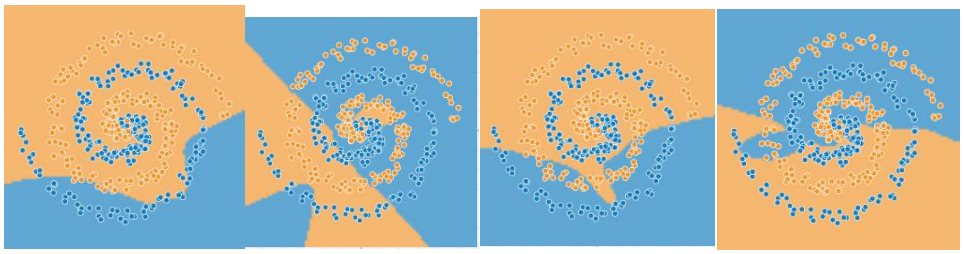


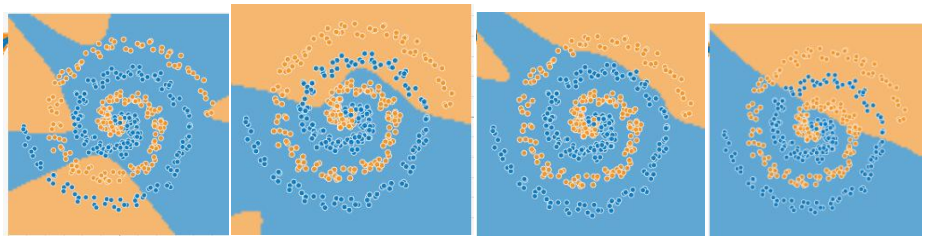
-0.29 2.6 1.2 -3.1

Of course the weird boundaries of each unit in layer 3 is generated by the weighted sum of second layer units' outputs.

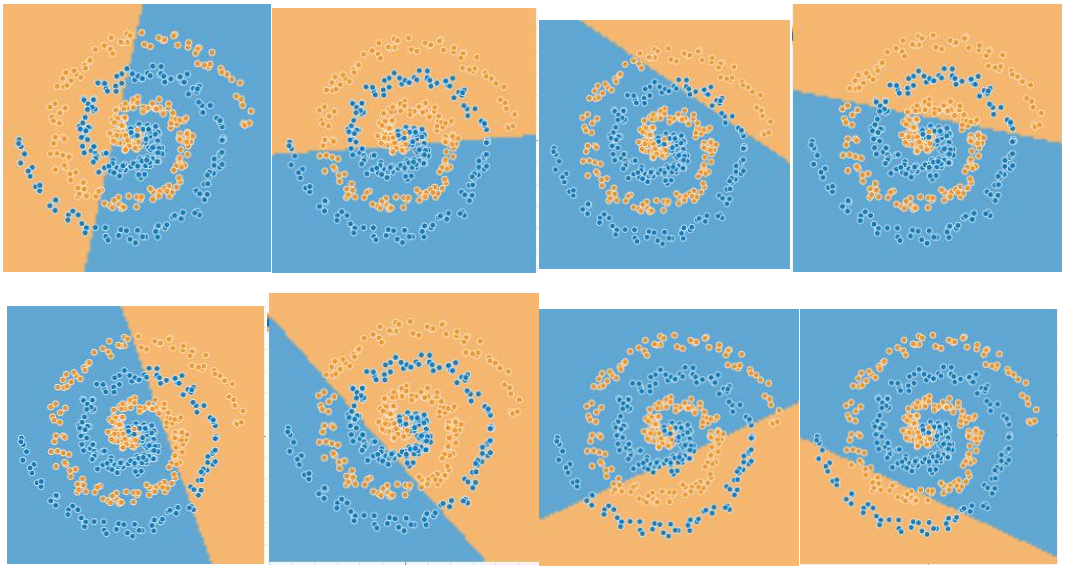
And from that, we can then discuss the explainations of why neural network works and their deficiencies. Each unit output boundary in intermediate layers looks 'nonsense' to the input data and can't seperate the data while every of them is somehow 'similar' to the finally decision boundary. I am not sure whether such phenomena have connections with the visiualization of CNN units.I will not discuss that part here. During the whole training process, we can only approximate the final decision boundaries (may exist many if the input dimensions are rudundant) given the labels of training data by gradient descent. The weights of all layers are learnt to minimize the differences of outputs and true labels. In discriminant models, we approximate the decision boundary that can perfectly seperate all the data. But the trainable parameters are huge and the input dimensions are always much higher than 2, it's impossible to visualize the training process loss and the boundaries perfectly(https://arxiv.org/abs/1712.09913). The training process may fall into different local minimum with different paths due to random initializations. But,from the view of discriminant models, the whole process is still draw boundaries, weighted sum them, through nonlinear mapping and create new bourndary to approximate the true boundary can seperate all the data.

I also list all the outputs of the other layers during the same training process to help elaborate this idea.

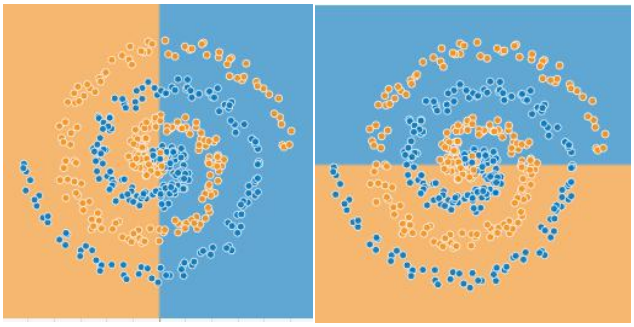




Layer 2



Layer 1



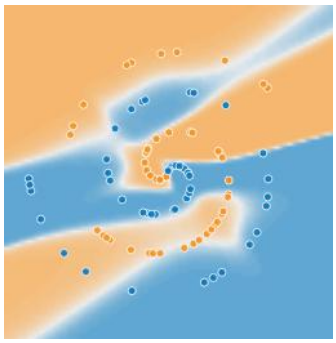
Input layer

Next I will elaborate my explainations of the deficiencies existed in neural networks.

1. Requirment of large training samples

The different classes of input data inherently have some connections and form different ‘clusters’ in the whole input dimensional space.But due to the complex structures of each class(we can take MNIST for example,it’s very hard to describe what structure form the different dights) the clusters maybe not like ‘sphere’ surrounding to exact one point in all dimensions,they may only ‘close’ in some dimensions(like belt or spiral or more weird) and far away(or have no relations) in other dimensions(image pixels are very rudundant, using dimension reduction methods like T-SNE/PCA we can already seperate digits well only rely on several dimensions). In neural networks, we can approximate any mathematical functions.But we can’t choose which boundaries to approximate as long as such boundaries can seperate all the data. So if the whole

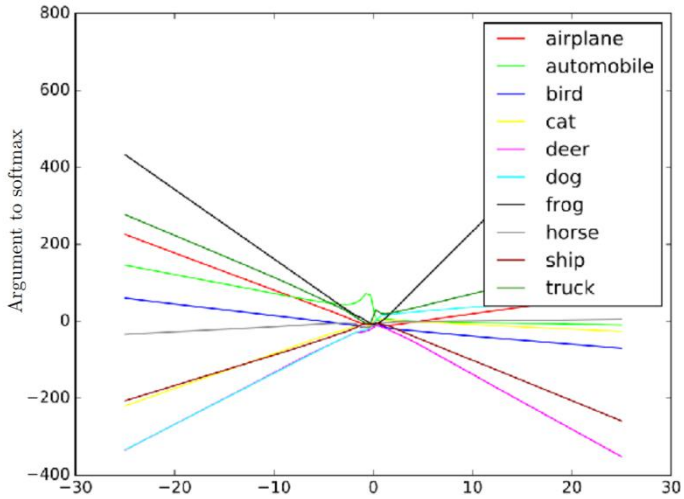
input dimensional space is very redundant(for image data), and the amount of data can't describe the structures and relations of different classes well in the high dimensions, the 'perfect' boundaries may be many and some of them can't really capture the structures to describe the data (refer to the story of detecting tanks https://www.jefftk.com/p/detecting-tanks). If we follow the wrong target, the space will be teared apart to fragments and causing the space labeled mistakenly. Elaboration is shown below.



However, we can still use neural network to approximate that 'broken' boundary well which is different from overfitting or underfitting but due to the lack of training samples. And when we use test samples even from the same distribution of input data, if it happened lie in mislabeled regions, the output label will be totally different.Because all the space is labeled by the decision boundary and we can't decide the process to approximate which boundary. The generality will also be ugly if training samples are too few and neural networks will be inferior compared to other traditional machine learning models if the amount of samples can't match the high dimensions and can't describe the relations in the 'needed' dimensions.

2. Generalization

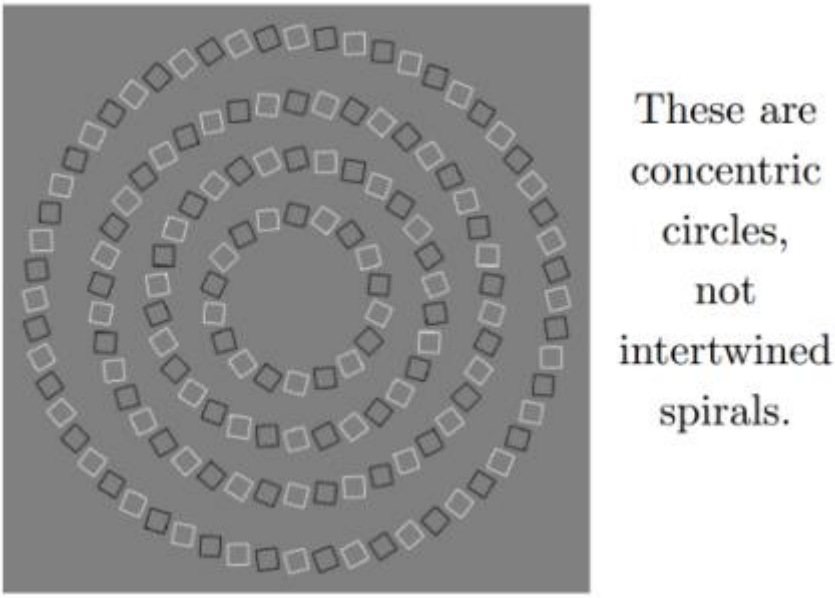
The final decision boundary and the label of the space can only determined by the available training data in neural network.The only job for the model is to successfully draw one boundary to classify the different classes.But the way to draw the boundary may be tricky and totally different from human.For examples, in CIFAR-10, we classify different classes by the shape of the object shown in the image, depend on the global information of the image. But the input data to neural network are just pixels and neural network has no preformed concepts of each object and how to classify them like us. And the size of CIFAR-10 is 32*32,the total input dimensions are very high (each pixel correspond to one dimension, and we can treat each dimension is a feature we rely to classify) and redundant. Even can't be perfect,neural network can still find some boundaries with very small error loss using extra dimensions exclude from the object(the background or the some specific color pixels shown in fixed positions in one class). In short, neural network may following the wrong cues to classify in some datasets. And this problem can't be eliminated by regularizations like weight decay or dropout. The figure below shows the likelihood of each class by disturbing the trained network following one specific direction(1024d-vector) in the whole input space.



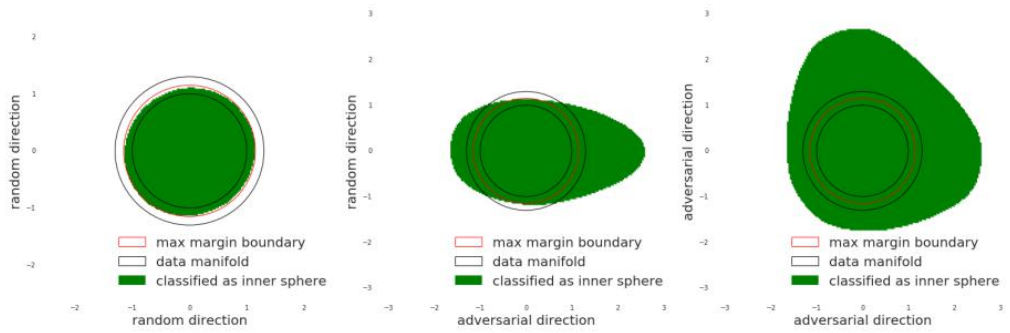
As we see, the correct label(automobile) is only within a small area of the whole space(no disturb in middle).if we follow one spefic direction of the whole space, the output label becomes incorrect.More data from different distributions will alleviate this problem, but it's extremly difficult to make sure the exact amount of per class to describle them perfectly using pixels. Particularly using 2-D images to describe all the characters of 3-D real-life objects.

3. Adversarial samples

Adversarial samples exist in every kind of discriminant models and even our human brains also suffer from it. Onel sample is shown below.



In discriminant models, we can accredit it due to the mistakenly label the space by carelessly drawing the decision boundaries.Like the elaboration image is in Sec.1, the whole space is randomly labeled if we can't using enough proper amount of data to cover the whole space.Because the only target we try to approximate is one of the 'perfect' decision boundary. The raliant dimensions to make the correct conclusion are too small compared to the number ofo pixels in images which causing the vulnerability exist prevalently in all popular computer vision models.T his paper(adversarial sphere): https://arxiv.org/pdf/1801.02774.pdf shows that even a perfect approximated boundary with tons of data(500 dimensions with 20 milion data), we still hard to make sure every dimension is correct just relying on the cost function.



So, these adversarial dimensions cause the vulnerability of the discriminant model. Especially in spaces where no data coverd, the label is less likely to be correct. Still, discriminant models will always suffer adversarial samples because brutally draw boundaries through the whole space and label them.

In conclusion, discriminant models are easier to train and employ, but the inborn deficiencies can't be overcomed easily. Maybe by changing the cost function to control the training process and more training data will alleviate the problems, but we can't eliminate them.The problems are caused by discriminant models themselves. I don't know whether similar deficiencies also exist in generative models, but I truly hope we can one day scale them up to promote the development of AI.

P.S During training, I find one interesting phenomenon. In above 2-D dots, if we not only use (x1,x2) but with some nonlinear function directly to the input (sin(x) or X^2) together with input (x1,x2) , the training process will be much faster and easier.