Understand neural network as compositions of small decision boundaries

Currently, almost all the popular neural networks we use are discriminative models means we draw boundaries through the whole N dimension input space (N is the number of dimensions of input data. E.g for MNIST, N=28*28*1=784) and put a label to each point of the space(For MNIST, if we discrete the value bournday [0,1] to 10 sub-parts equally, 0, 0.1 , 0.2 ,... to 1.0, each dimension can take one of these eleven values,then the total points in the N space is 11^{784}). And each training sample is corresponding to one point in that space, the effect of whole training set to the space is to mark some points with different labels,always the training set can only label a very small part of the space. Neural networks act as classifiers to seperate these points with different labels. And the process of training neural network can be described as follows:

1. We have labeled training set. Each sample is corresponding to one point in the N space, so the whole training set is to mark some points with labels in that space.
2. Using back-propagation based on gradient descent, we approximate the 'nearest' boundaries which can seperate all points corresponding the labels.

This process is data driven and easily to implement, leading deep learning outperforms other machine learning methods even in some particular tasks beaten human. Altough training set become larger and the boundaries to classify all the classes become more complex, we can still use neural networks to fit all the data by increasing number of layers and neurons with more powerful optimizers. You shouldn't doubt the power of neural network to correctly classify all the training data as long as there exist such boundaries through the space. Such ability of neural nerwork has already been proven many years ago called 'Universal Approximation Theorem'.

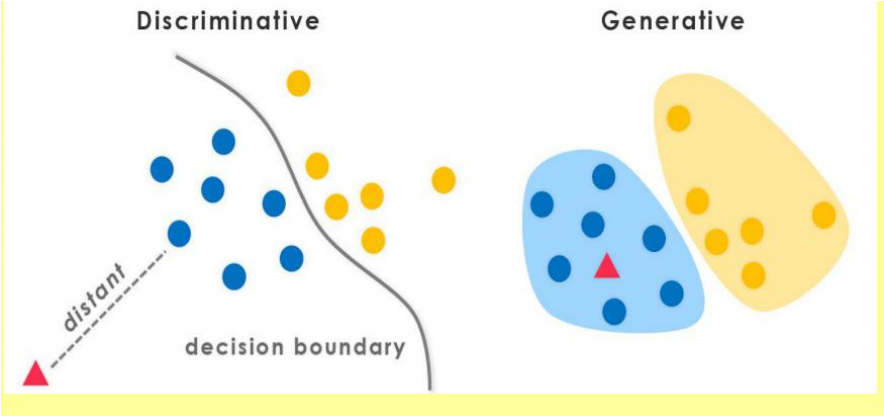
But due to the points in the whole space is so large and the discriminative characters, this process also cause some inborn limitations in neural networks.

1. Requirment of large training samples
2. Hard to explain the trained model and how well it generalize
3. Easily cheated by adversarial samples

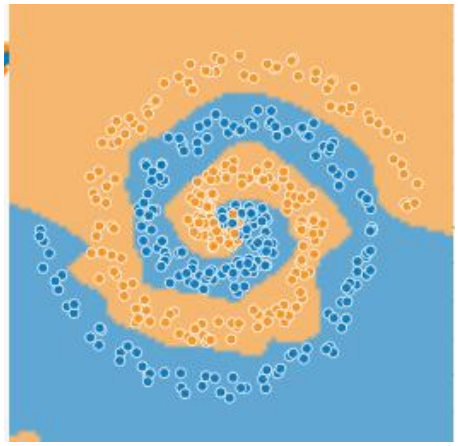
Even deep learning is very fruitful now, we should keep a clear head with it. Deep learning is still very far from the true AI, it may still need dacades to achieve it.

The discriminative characters and learning process

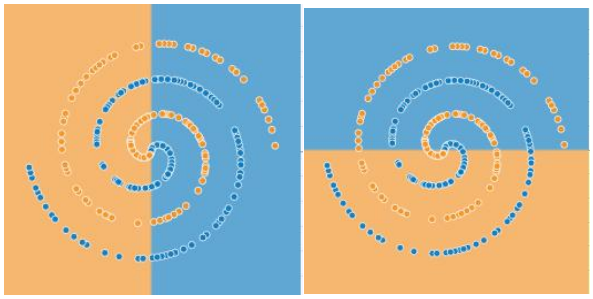
The differences between discriminative model and generative model can be elaborated in the figure below:

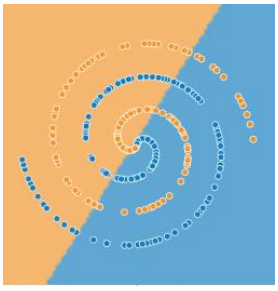


In deep learning disciminative models, we want to train a neural network to approximate the direct mapping function of y=f(x) and use the trained network to predict with new samples.For example, there are two classes colored by blue and orange. In this example, blue dots(area) are postive and orange dots(area) are negative. Because they are 2-d dots, the input are 2-D vectors [x1,x2] representing the location of each sample. We then use neural network to distinguish them by approximating the 'nearest' perfect boundary based on gradients from the initial point. After one random training (the structure used in this example is 2-8-8-8-1 FC with tanh) we can get the decision boundary and the label of each point shown below:



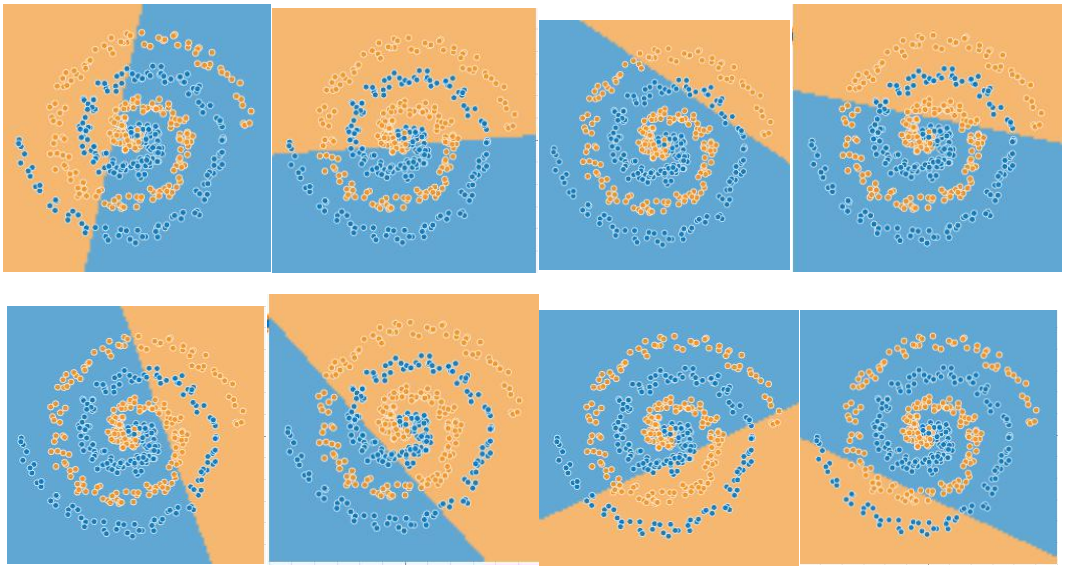
Units in the same layer are weight summed(we can treat the bias term as one additional input with weight -1 for easy elaboration) then followed by 'tanh' activation (I choose tanh here because it is symmetry in positive and negative axis for better visualization). The weights between units and units in the following layer represent 'how important' the previous units are. Outputs from the previous layer can also be viewed as small boundaries, the next layer takes these output boundaries to create new boundaries. We can also think the 2-D input data [x1,x2] as boundaries. And the first hidden layer uses these two boundaries to create new boundaries by multiply them with trainable matrix 'W' followed by nonlinear affine. For examples:



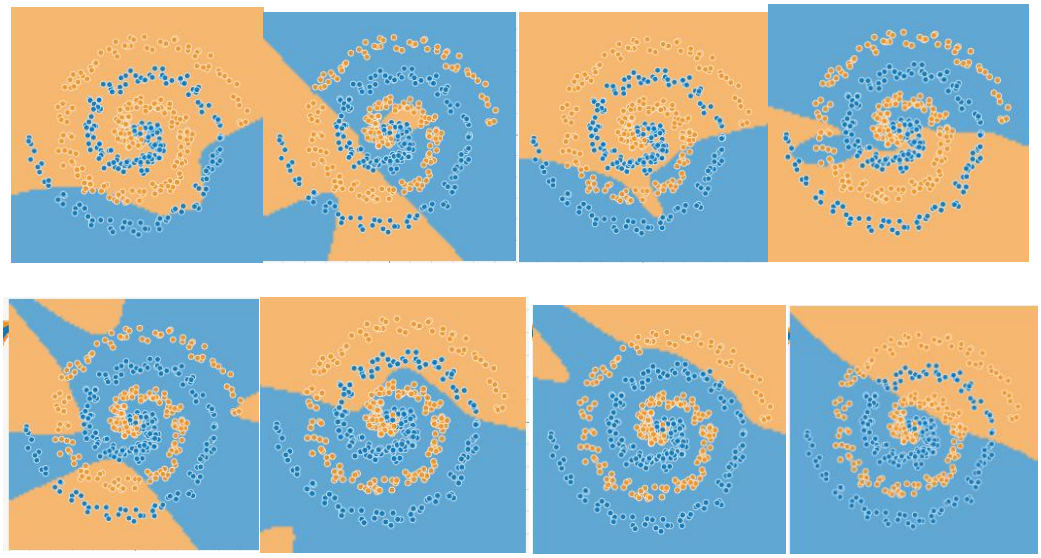


Input:	X1	X2	One unit of first hidden layer
Weights:	0.41	-0.26	

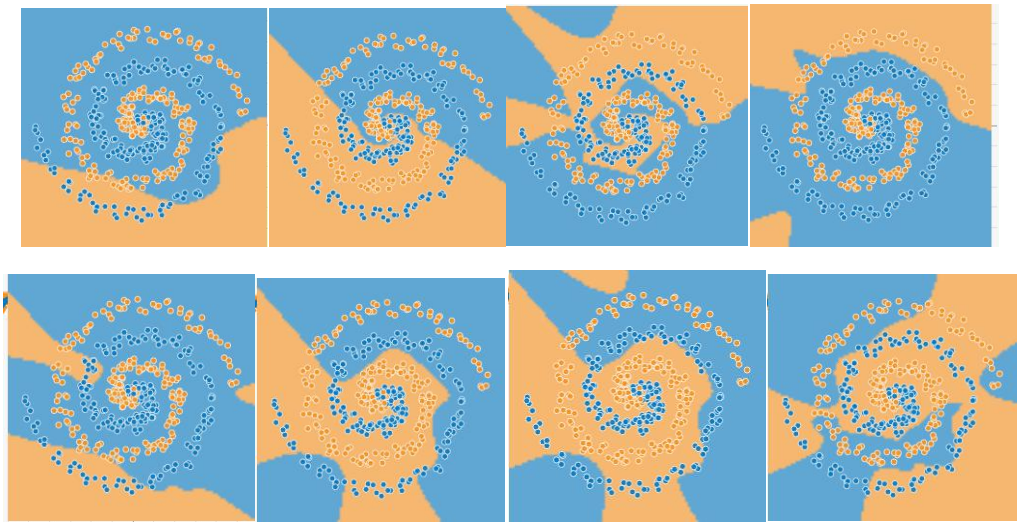
Each layer takes the boundaries from previous layer to create new boundaries. Although one hidden layer is enough for this problem, it's better to visualize several layers to see the effect of stacking affine mapping which I believe is the key in deep learning to model more complex problems.



Layer 1



Layer 2



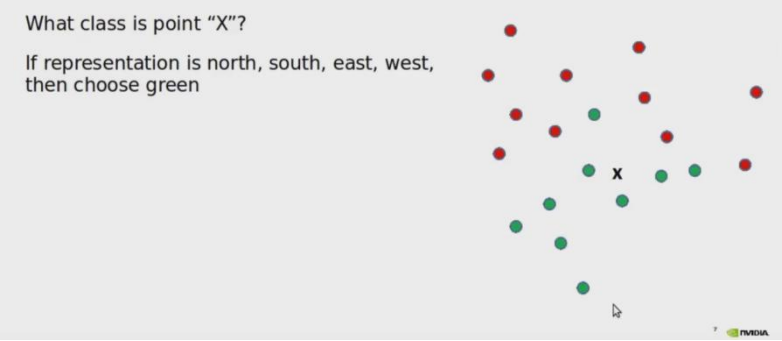
Layer 3

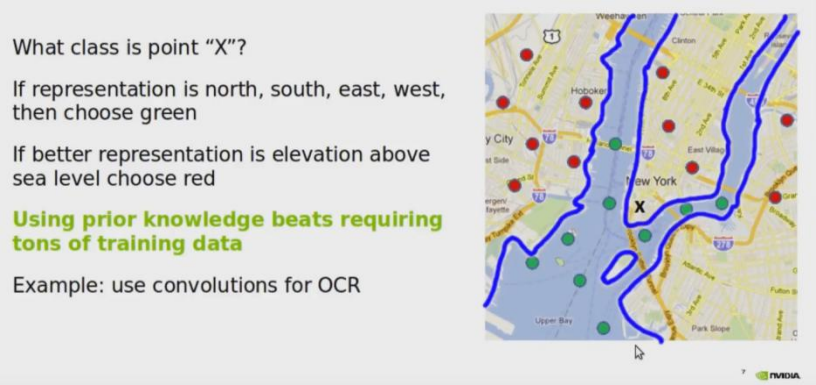
By treating input vector as boundaries, there is a straightforward way to speed up training neural network: we not only use raw [x1,x2] as input, we can also mannualy add more nonlinear affine to x1 and x2, eg sin(x1), x1^2, x1*x2 together forimng a large vector [x1,x2,sin(x1),x1*x2] to train neural network. My experiments show such phenomenon is general, but it needs rigor mathematical deduction.
This idea of fulfill complex tasks by using simple elements is very general, from genes to all kinds of living creatures, from 0-1 on-off switch to intergrated circuit. And units in deep learning to create the final dicision boundary.

Next I will elaborate my understandings of the limitations in neural network. It's irresponsible to accredit them only to the model. The gradient based back propagation training procedure (only follow one mostlikely path) and the way of using training samples (each only represents one labeled point in space) are also should be considered.

1. Requirment of large training samples

Each class of input data inherently have some 'connections' in the whole input space by following some patterns. Whether training data can represent problems well is crucial for generalization.





Sometimes the data may fail to represent the preblem and causing the model generalize badly to new test data. In neural network, we learn everything only from data without any prior assumption which will lead the model following wrong cues sometimes and can't transfer to real-life problems. For examples, current deep learning methods in computer vision is very amazing, but using 2-D images to represent 3-D objects is misleading. The use of static 2-D images will introduce so many variations compared with the way we learn in a consecutive 3-D environment. Convolution neural networks can handle problems well if test data are also 2-D static symbols like digits, but I believe only rely on current CNN won't lead us to the true AI vision as we dreamed.

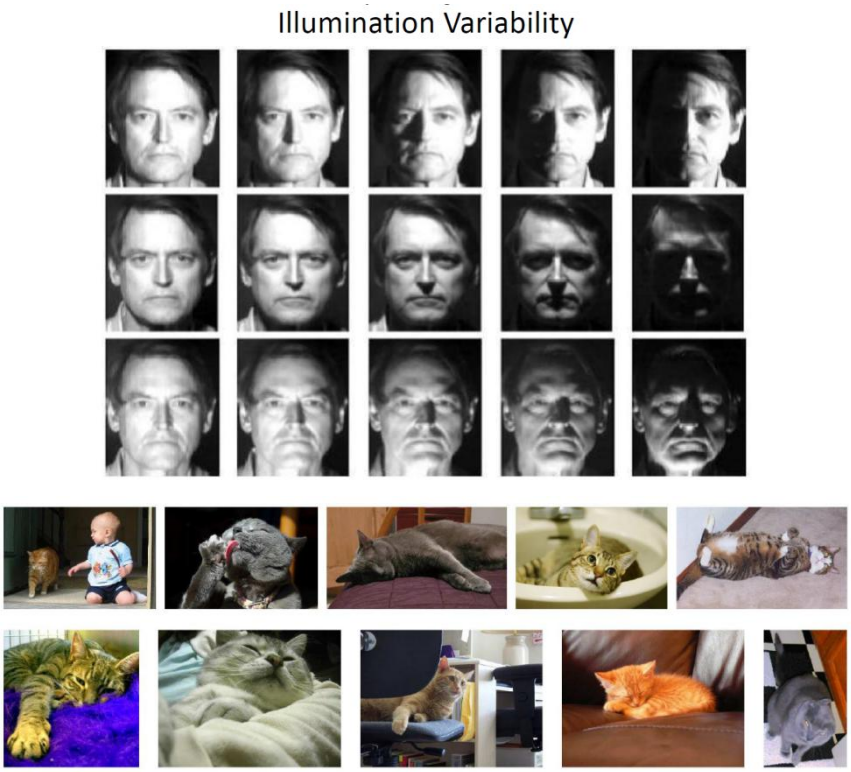
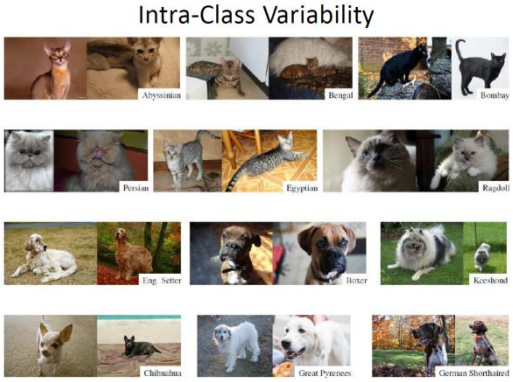
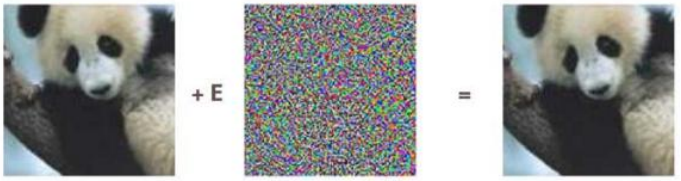


Figure 1. **The deformable and truncated cat.** Cats exhibit (al-



Parkhi et al. "Cats and dogs." 2012.

What's more, each image sample is just one labeled point in the space, the model can't see the connections between these points as we see the image. Some funny results may arise.The two images looks exactly same to us, but the model misclassify the right input image as gibbon with very little gradient based noise. Such samples are also called adversarial samples, the fault is not only in the model but also due to how we use data to represent such problems. For discriminative models, if the training size is too small to label enough points compared with the whole points, then adversarial samples will be pervasive no matter what kind algorithm it is. Even the famous Imagenet contains over one million samples, there are still plenty space (if we only use Integers, the number of points is $256^{3*224*224}$) for adversarial samples. Even we arbitary follow one direction in that space, the label of points may change dramatically. For discriminative models, if the input data are pixel-like images, then it will inherently cause adversarial samples.



To reduce the number of samples we need, I believe there are two directions: 1.Take in prior assumption 2. Build connections within data. The first approach already has many cases in neural network, but the second one is still missing. Current directions in designing new models in deep learning is to introduce extra trainable matrix in connections and loose restrictions to some hyperparameters. Foe example, we can view all the highway models like Resnet,Densenet as some specific types of boltzman machine with zero-weight connections where no connections within such models. And view deformable convolution and non-local net as the second part. So my idea is to build connections in data by extra trainable matrix, and feed the trained matrix together with raw data to neural networks.

If we can find some way to build connections within data, I believe the number of needed samples will reduce sharply.

2. Generalization

The final decision boundary and the label of the space where no directly mapping can only determined by the available training data. In neural networks, we rely on the final loss function to do the training. The only one target for models is to approximate the perfect boundaries which can classify all the different points. And the laterer the layer is, the influence will be greater. Typically, we call the influence as back progapation gradient. At least, this is the popular reason for why deeper models are harder to train for vanilla neural networks. With skip connections between layers and other tricks, deep models can be trained. But I personally perfer to call that reason as model degradation rather then gradient vanishing. Because even we manlly scale the gradients in lower layers, deep models are still hard to get a small training loss when it should be. We degrade the model as we simply stocking more layers when still train the whole model with BP. Make the power of modeling worse than shallow models for deep models.

Other people believe the flatness around the global minimum is the key to generalization well and avoid adversarial samples. So regularization methods like weight decay and dropout always perform better with test samples. By leveraging weights to reduce the noises of dominant units. In the example of the spiral dots of orange and blue, we can see some boundaries in layer 3 already have reasonable shape, if happened to there are no training samples in the noise part, then the noise will have a greater influence and drop the test accuracy.

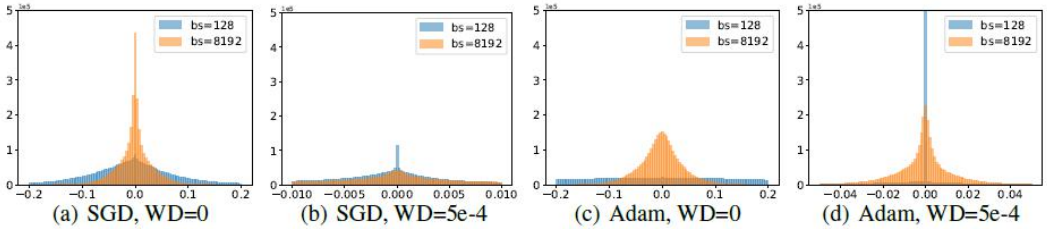
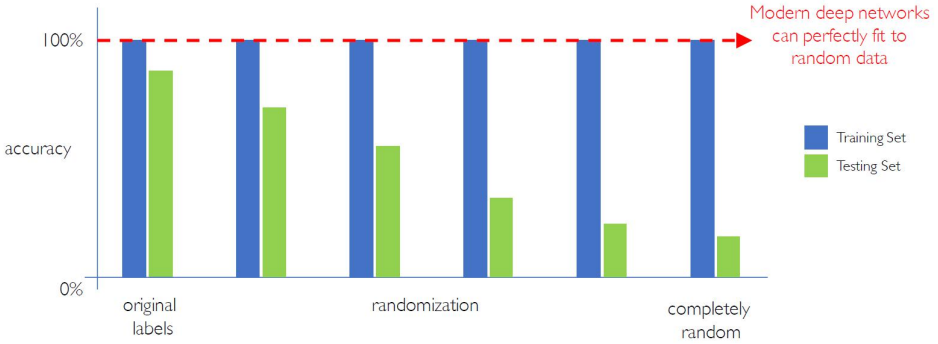
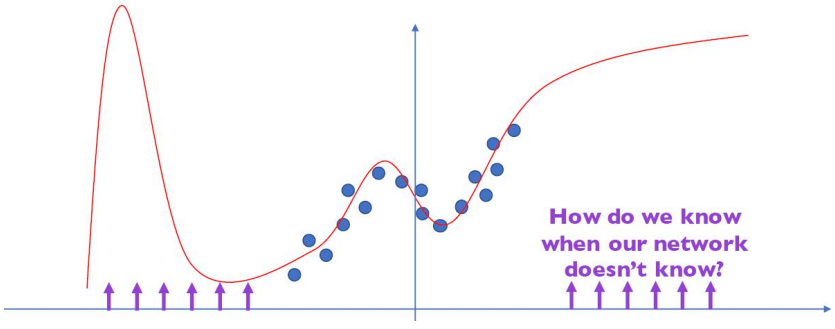


Figure 3: Histogram of weights. With zero weight decay, small-batch methods produce large weights. With non-zero weight decay, small-batch methods produce smaller weights.

But there are some premises to hold it true. One crucial premise is the distributions of training and testing samples over the space. Neural networks can remember training samples well with random labels, but of course the test accuracy is no better than random guess.



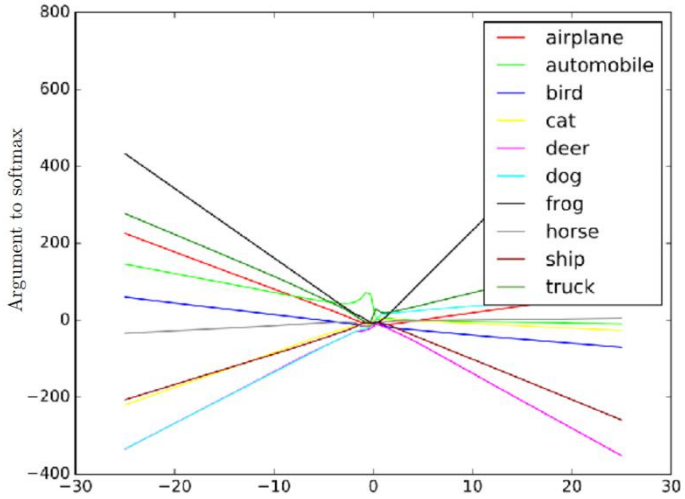
Another one is the spreads of the distribution. If we only collect training samples from part of the real distribution, no matter how well the model is trained, the generalization will be inferior. We can only learn the mapping of Y=F(X) and use the mapping to test with new samples, if the training data fails, the model fails too. Current deep learning models are just preliminaries to the true AI.



3. Adversarial samples

Adversarial samples are prevalent in all kinds of discriminative models. An adversarial example is a sample of input data which has been modified very slightly in a way that is intended to cause a machine learning classifier to misclassify it. I am personally not very supportive to the definition. Because training data only maps small part of points in the whole space, it is very partial observable for the models. And neural networks are discriminative models which can remember the labels of trained samples. It can not form new reasoning beyond training data. Gradient

based back propagation will also push the model to follow the gradient without exploration. All the reasons cause current popular deep learning models are vulnerable by adversarial samples. And adversarial samples are easily to transfer and attack models even we have no preknowledge of it. For example, neural networks trained with CIFAR-10 can only successfully classify points within a small area covered by training data.



Conclusion

Neural networks are easier to train and employ if the number of training data is large and the representing quaility is good for real problems. But we can't satisfy with it, current neural network models are functions of nonlinear mapping Y=f(X). It only covers part of AI, we still need to work hard to pursuit the path to the real Artificial General Intelligence which can see, learn and think like us and even with emotions in it.