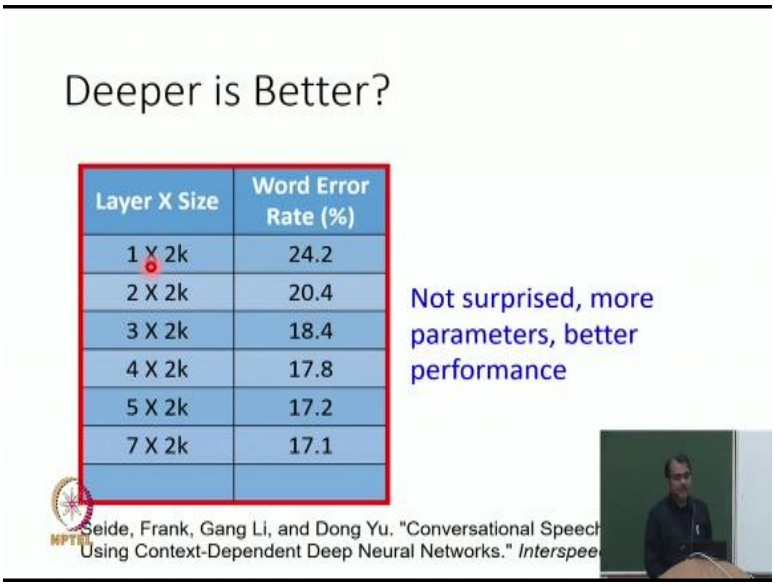
Artificial Intelligence
Prof. Mausam
Department of Computer Science and Engineering
Indian Institute of Technology-Delhi

Lecture - 89
Deep Learning: Thin Deep Vs Fat Shallow Networks

Now before we move to deep reinforcement learning, I want to talk about two quick things. The first question I thought we should discuss in this class is why should we train deep neural networks just at least what is the philosophy why do they work better and do they work better? We can also train a one layer neural network or a two layer neural network what is so beautiful or what is so important about the depth, right.

(Refer Slide Time: 00:40)



Why do we say that if you can train a large layer neural network then you will have large salary, right. Why do we say that? So here is a specific example, but this is illustrative and similar examples exist in other branches where I trained deeper neural networks. Each layer had 2000 nodes, 2000 neurons, okay. I believe it is a speech recognition task.

And as you can see, that the as you increase the number of layers, your accuracy increases, your error decreases. So starting from 24, you can get to 17% in 7 layers. This is not particularly surprising, because you can argue that of course, more layers

means more parameters, right. Suppose it is all fully connected, you can actually compute the number of parameters, right.

Suppose there are 2000 nodes, 2000 input and 2000 output problem, let us say. Then you can easily figure out that for each neuron I will have 2000 inputs or 2000 plus a bias term, so 2001 right, times 2000 neurons or 2000 square, right. You can easily do that kind of a computation if it is fully connected. So you can always figure out how many parameters are here.

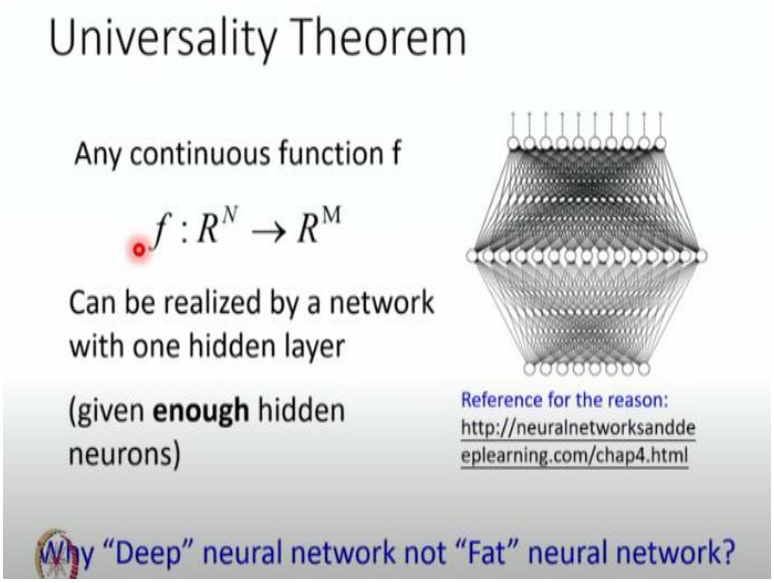
I believe this table is for the number of parameters and not the number of neurons, so I am not sure. Oh, this is 200 okay. So then it is number of parameters, number of neurons 200. Sorry, I was reading it wrong. So far so good. So now this should not surprise us because in fact, and this is very funny 2013 we got AlexNet, which had increased the performance from 25% error rate to say, let us say 16% error rate or something.

And then for the next three years, or four years, everybody kept coming up with a better and better neural network to do better and better performance. Do you know what was the main contribution of each next paper? Just increase the depth. It was just like almost I mean, I am oversimplifying, but they will beat me when I say if I say no contribution, of course they had contributions, they made it work.

There were engineering contributions and maybe some algorithmic contributions as well. But by and large, it was I can increase it to 50 layer, I can increase it to 75 layer, I can increase it to 100 layer, I can increase it to 150 layers game. And they kept getting better and better performance.

So an easy way if you know all the optimization tricks and you know all of that business, an easy way to get more performance and better performance is just increase the number of layers without thinking too much about the problem. So this does not surprise us the fact that if we have more layers, the performance is better.

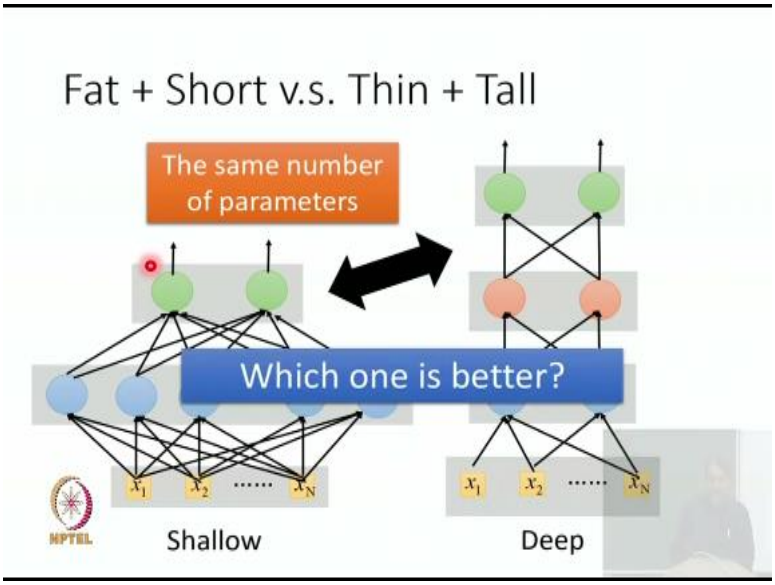
(Refer Slide Time: 03:49)



But moreover, there is a theoretical result called the universal universality theorem or the you know neural network is a universal approximator kind of a theorem. It says for anything continuous function f that takes that goes from, let us say n dimensional real space to say whatever m dimensional real space, any such function f can be realized by one network with one hidden layer only.

Just one hidden layer can realize any function whatsoever. And the fine print is given enough hidden neurons, right. So then that begs the question why should we train a deep neural networks why not a fat neural networks.

(Refer Slide Time: 04:44)



In fact, if we have a fixed number of parameters, why should we not train a fat and short, short as in not too deep, a fat and short neural network because technically it

can approximate any continuous function. Why should we train a thin and tall neural network which would be deep network. Let us be shallow. Let us be fat depending upon what we are playing sumo wrestling or basketball but.

"Professor - student conversation starts" Yes, what is your name? Ashray. Yes Ashray. In the slide which you just showed if could begin getting approximation of x with just only 10 layers with enough neurons. But in the previous case when we were talking about increasing layers, so there why do we not have any concept of like enough layers like what if number of layers are taken into account. I do not know. **"Professor - student conversation ends".**

Ashray asks you know this theorem is only for one hidden layer what happens when you have many layers? Do we have a bound, do we have theoretical results which tell us that okay these many layers are good enough for the problem etc. I will tell you the trick here. This theorem is a representability theorem. It is not a learnability theorem. Let us see if we can dissect these two words.

It says that any continuous function can be realized, can be represented. There exists some set of parameters such that this function can be realized by those set of parameters. It does not say that those parameters can be learned by backpropagation or training data, which is always some fixed size training data even if large.

Now what happens is that when we start to learn a big fat network in the hope that, a bit shallow, in the hope that it is going to realize the function that we are looking for, we do not usually succeed. And there are limited theoretical results for why that happens. In fact, if you can prove a theorem in an interesting theorem about neural networks, you can be famous. Of course, theoreticians do not become rich.

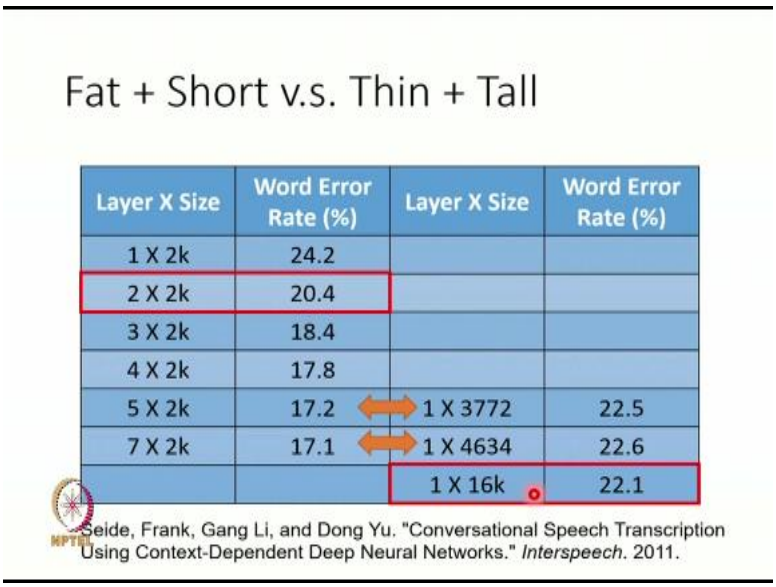
So you will probably not be rich and famous. But you can at least be famous. Because it is a very hard question today. People prove things about 3 neuron neural networks or you know 4 neuron neural networks. I mean, it is crazy, 2 layer neural networks. I mean, it is very early ages in with respect to our theoretical understanding of what they can and cannot do, okay.

So I do not think there are many good results for really how much depth do we need and so on so forth? Yes. right. And that is that is what we are going to get to, right. So Vishwajit says, this theorem of course, does not say how many neurons are needed. We can check. I am sure there are some bounds in the literature regarding that. But maybe if we make it deep and thin maybe we need less neurons, right?

And again, that is exactly what I said, this is our intuition. But can we prove it? I do not know, right? Probably not yet. So here is a quick comparison that you would want to do. Let us take a shallow fat network and a deep thin network with the same number of parameters. So we will just balance, we will control 4 parameters and say, okay apples to apples in terms of number of parameters, tell me which one is better.

That will tell us whether they are the same or big fat network is better or thin tall network is better, right.

(Refer Slide Time: 08:34)



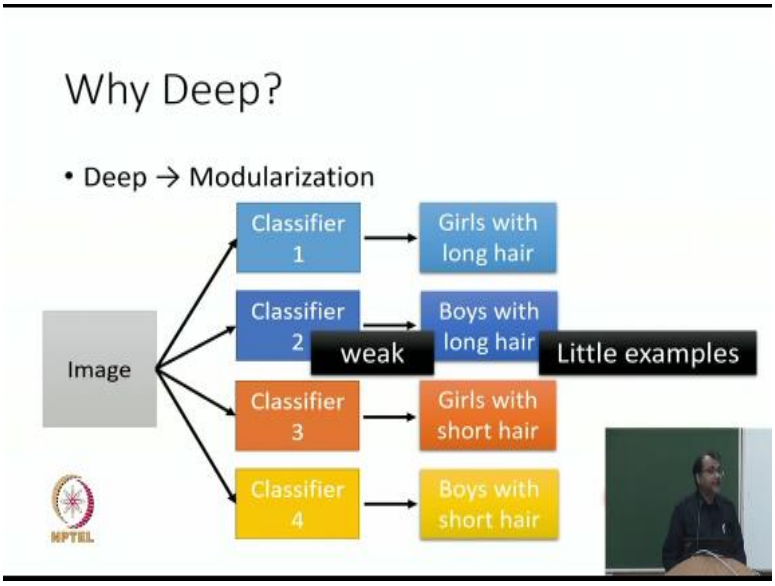
And so in the same example, they did this experiment, so they took one layer, one hidden layer neural network with these many neurons such that the total number of parameters are the same, approximately the same and notice the errors. 17% for the 5 layer one and 22% for the one level one or 17%, for the 7 layer one and 22 again 22% for the one layer.

In fact, even after increasing this number much higher, anyway we are not even in this on the left columns, we are not even comparable, right, it will be much deeper here if

we if we take this, it still went to 22.1. And notice just a 2 layer neural network. So it was better than a one layer 2000 network of course, but it was already worse than the two layer neural network.

So 2 layer neural networks were able to learn much better than a very large fat one layer network. So clearly there is something about the deep networks that makes this work.

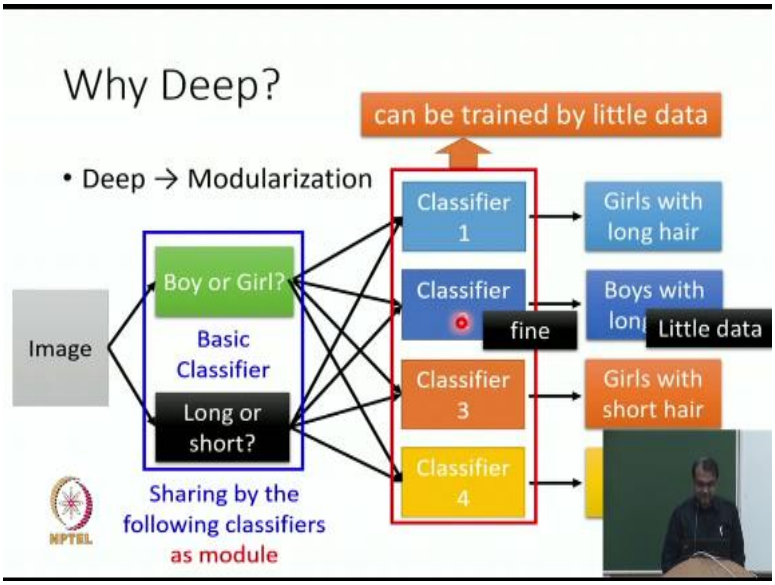
(Refer Slide Time: 09:49)



And here is a very nice intuition that this slide that gives. This is not my example. But I think that intuition is beautiful. That suppose I give you an image and my goal is to identify, you know girls with long hair, boys with long hair. Girls with short hair boys with short hair, right? And I send it to a neural network. And it learns, you know 4 representations internally and outputs these things automatically, right.

Now will we have enough examples for all 4 classes? Which class will we have very limited number of examples for? Why is it long hair? It is sad, but it is true. Someday we will have that parody too, right but not yet. So therefore, we have a little bit of parody in the class, thanks to our French friends, but not much, right. So we have to do better. Now what will happen is this classifier 2 is going to be incredibly weak because it does not have enough data.

(Refer Slide Time: 11:06)



On the other hand, if I did deep, and let us say for whatever reason I first learnt two basic classifiers is it boy or girl? Is it long or short hair, right? Only on the attributes of the data on the final classes, and then I still learnt my four classifiers. Do you think this particular way would do better? Because for the first class, boy or girl, will we have enough data? Yes, for the second class long or short hair, will we have enough data?

Yes. And moreover, even if we have limited data for boys with long hair, would we need that much data for classifier 2. In this case, at least it is a simple right and function of a certain kind that we have to learn. So we can actually train it with very limited amount of data. So we are fine there. So this is the modularization of a deep network that the first layer is learning something which is used by the second layer which is learning something which is then used by the third layer.

So even if you use words to explain somebody what is the face. What would you say. A face has two eyes a nose and a mouth and the person will say what? What is two eyes? What is two, what is eyes? So now you say okay an eye is one which has this eyelid above the two eyelids and there is a you know whatever eyeball in the middle and blah blah blah. And then they will say okay what is a ball?

And it will say okay, it is a round object. It will say what is round and you say it has this edge and that edge. If you keep deconstructing this thing about this deconstruction. If you keep deconstructing this, at the lowest layer you will be

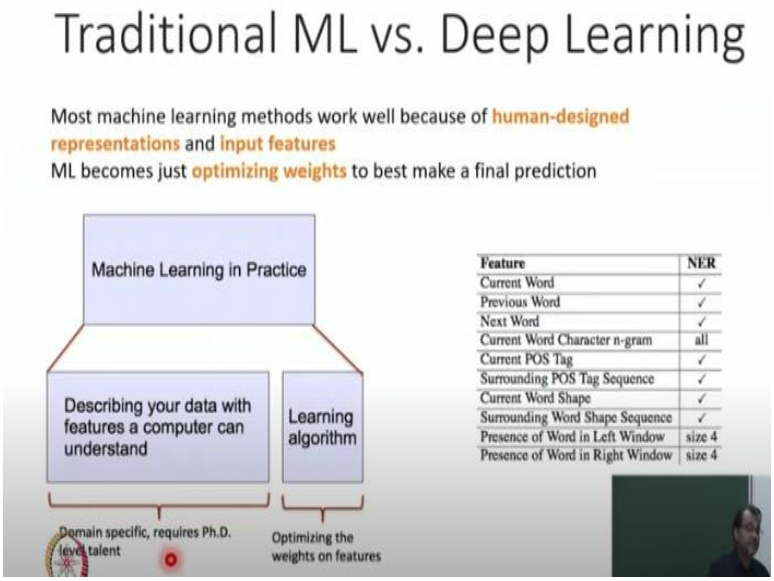
looking at only the physical attributes the very fine like this is an edge, circular edge looking upward.

This is a you know edge slanting downwards, then they will combine to create a v shape or a semi-circular or circular shape. Then they will combine to create a ball to create a triangle to create blah. Then they will combine to create an eyelid or whatever it is. Then they will combine to create an eye. Then they will combine to create a face. This depth naturally brings out the compositionality of our world.

This is called compositionality. That something is composed of parts, the parts are composed of further parts, the parts are composed of further parts. If you want to do it at one step, it is going to be very hard for the classifier. If you do it in multiple steps, it allows the model to automatically train the lowest level features, use them to train the higher level features, use them to train the final class prediction.

So these are all our intuitions where deep networks have better properties in practice, have better learnability properties, have better performance. And of course, there are issues like how to train them well, and that requires some skills.

(Refer Slide Time: 14:26)



Now this completely changed the way machine learning used to be done.

Because in the more in the pre neural era of AI of machine learning, the style of machine learning was the following. You give me the Bayesian network, I will let us say I am a machine, you give me the Bayesian network, I will train the parameters.

Who gives the Bayesian network? The human. Okay, I can do some structure learning so I can take your Bayesian network and tweak it a little bit.

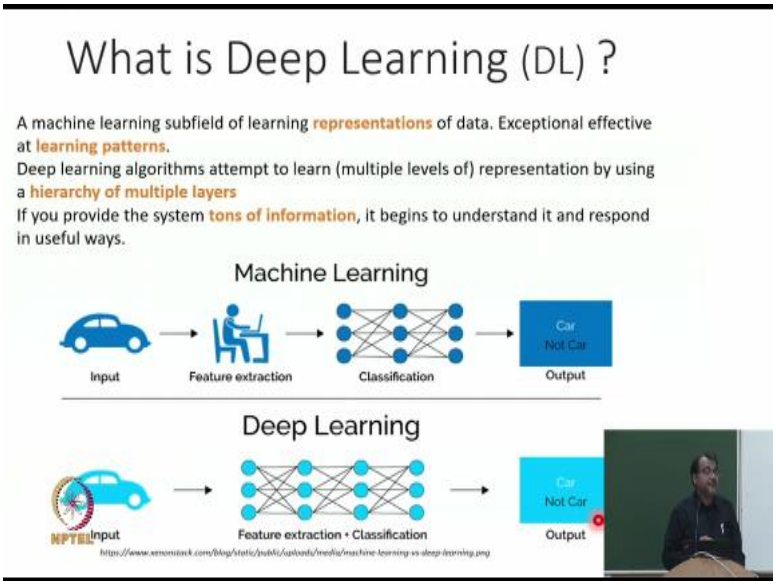
Say this edge is also important, this edge is unnecessary, very small updates. In practice fine. You rarely do structure learning from scratch which does not work. So features of the problem are given by the human. The structure of the problem is given by the human. And the machine basically trains the weights. For more complicated examples, like language processing, image classification, speech recognition, lots of features were being given by the human.

The features were okay, what is the current word? What is the previous word? What is the next word? What are the various engrams present in the current word, what is the part of speech tag of the current word, etc., etc., etc. All these are features. And then the features had weights. The weights were trained by the machine, given the training data. So that was what traditional machine learning was.

Optimizing the weights was a small part of the big picture. In fact, if you had taken my course five years ago, I would have said that you should look at the data and you should come up with good features because if you do not have good features, you cannot do good machine learning. That was what we were trained as in machine learning. It was all about human insight, not machine insight.

It was all about our ability to come up with the right features so that the machine can learn good weights for it. If you do not give it a good feature, the machine will basically not do very well. And as it says in the slide also this is very domain specific, I can do it for language, I cannot do it for image. People who can do it for image can do it for image, usually not do it for language and so on so forth, required PhD level talent.

(Refer Slide Time: 16:31)



However, deep learning completely changes. Deep learning says, forget the human, let us bring the human out. Let us send them out. That is not completely true. And I will explain why. But for now imagine that there is no human. The input comes in, the output has to go out. Not only do I have to learn the weights, I also have to learn the features. In other words, and I am going to do it together.

I am going to learn the features and learn the weights and make the prediction in one big optimization. So therefore, I will hopefully end up learning the features that are important for my current task. There may be other features of the input, which are relevant for some other task, I may not learn them, I do not need to learn them, my machine may not learn them. So feature extraction and classification happens together.

They all happen in this one universal representation, the neural network. And now you can think about that the middle layers are learning features at different layers of abstraction, different levels of abstraction. And at the last layer, I am making the prediction. So all of neural network learning is learning representation of the data, learning the features of the data, learning patterns about the data.

And using those patterns, using those features, using those representations to create a my final class, predict my final class. And not only does it learn features, it learns hierarchy of features. And then also fits the way we think because the world is a

compositional model. Now where does the human come in, in the whole paradigm other than for creating the data, input data and defining what the output should be?

What else? Yes, so the human does the architecture design. What does that mean? It figures out which neuron should be connected to what? What should be the nonlinearities? How many weights should be there also how many weights you can even do search, right? In fact, people are saying let us remove the human even from there. Let us have a suite of architectures. Let give me the lot of data.

I will just learn lots of architectures on it. And using some performance measure I will keep doing local search on it and finally come up with the best architecture and maybe human may not will not be able to come up with the best architecture. This field of machine learning is called somebody knows, no okay this is called meta learning, auto ML, automatic machine learning. No human in the loop whatsoever.

On the other hand there is a lot of work on human in the loop AI. It says you the machine learning algorithm always have to work with me the human, so do not mess me up. Do not completely change your predictions. Do not you know give a new data, completely change your classifier because I will go completely crazy. How can we the two of us work together?

That is called human in the loop machine learning or human in the loop AI. That is a different story. That is long term working together. So everybody is going towards the vision that AI and humans will be working together in the long run.