[Music]

Let's now start part 2 of this first lecture on artificial neural networks, and I will do so by talking first about some similarities and differences between learning in one hand on symbolic systems which we spend some time on early on the course, and learning in sub symbolic systems which we will focus on this week. So as you may remember there are a variety of symbolic representations that we mentioned, some of them more some of them less logical representation, factual representation, object-oriented ones specific ones like Production rules, Decision trees, Bayesian Networks and Semantic Networks and so on. And the two sub symbolic systems that we touch on this course are artificial neural networks as for this week and earlier mentioned also genetic algorithms. So if we look at the differences here one can say that when you implement learning mechanism in a symbolic system you do it normally as an add-on to a static kernels so the static kernel is programmed and then you add on these other mechanism where you can adapt that static kernel to achieve the adaptive duality. While in some symbolic systems typically the learning mechanism are the core of the system as a whole, so therefore you get the impression is also true that problem-solving and learning are more tightly integrated or intertwined in a sub symbolic system. So that's a clear proof of sub symbolic system, there is all some con and one of them clearly is the following that what is learned and how it is learned is normally very concrete and explicit in symbolic systems. So you can actually look at the system before you learn and they can offer and you can see the difference and you can actually more or less kind of read it. While in a sub symbolic system which are made up of these tiny processing elements like in neuron or gene in the genetic algorithm it's not trivial to see actually what has happened, and it turns out that at this moment where sub symbolic systems start to be used a lot, this is easy the key problem and there are a lot of work in that area in order to interpret what has been learned, even if the system as such is very successful. Also one can say there is a depth following difference in a symbolic system you literally reshape and extend the current structures when you learn, which means that you in a symbolic system if you want extensive learning, changing the system not marginally but more drastically, you really have to make an extensive restoration of system, you have to add things, you have to add structures and that is not always trivial. The some symbolic approaches is in that very clever because actually what you do is that you pre-allocate empty space to say, because if you represent everything in uniform form you can say, ok let's play allocate a lot of space for our system, but initially we only use a subset. And then we take the rest introduced when is needed. So we can say the structure is already there but it's not used to its full extent. I mean another can be actually when people talking about the brain or the mind is that the brain are used to certain extent but maybe not fully utilized but because there are potential reserves in the brains neurons structure for more connections you can take it into use. So there is this distinction between parameter learning and structure learning in machine learning, and I think one can phase the approach in your networks that you convert structure learning into parameter learning and because parameter learning is normally considered simpler that's a positive thing. If you only want to learn parameters in the first place the differences between the symbolic system and the neural system is also not so great.

Learning in artificial neural networks is normally accomplished through an adaptive procedure known as a learning rule or algorithm whereby the weights and also the biases of the network are incremental adjusted. Potentially also other parameters the so called hyper parameters we talked out earlier can be learned and they are such I mean the function actually that controls the transfer activation is an example of a parameter that can be modified but there are also factors like learning rate, now the parameter is occurring in these models and they are not part of the system in the sense that the basic learning rules handle them but additional rules can be added for that purpose. For several reasons the learning process is best viewed as an optimization process. You probably get the same picture again here I mean we have talked about this earlier, it's very common in artificial intelligence to look at things as a search process, and with the purpose of optimization. So actually here what we do is to make a search in a multi-dimensional parameter weight space where a tentative solution is gradually optimizes a pre-specified objective function. So the term objective function means that we have a function that map's on a event situation on to a real number and intuitively many times it represents the cost for achieving what we have. So it could be called the cost function or loss function and of course always we want to minimize the cost. However one can design this in the opposite direction so we can have the cost function we can have a reward function or a profit function or a fitness function or something positive and of course that in that case if we design it like that then the optimization is maximizing that kind of function, but it's all up to us and we want to engineer it.

The choice of using artificial neural networks as a representation or something else, is orthogonal to the categorization of machine learning techniques in supervised reinforced or unsupervised. So actually ANN can be used for all these kinds of three purposes. So it's also for the supervised case the input received from the environment is associated with a specific desired target pattern. So then the weights are synthesized gradually as an updating when we treat data item per data item, so that when we went through the data set we have available data of the ways have stabilized in such a way that they may minimize the actual classification or regression error with respect to the predefined target reference point. But also the neural networks can be used for unsupervised learning, it can be used for clustering of course the architecture of that network and that is fit for that purpose would be differently engineered. So therefore as you if you remember from this slide where you saw all the kinds of networks that is possible or have been designed some of them are designed for the particular for the supervised case while others are more typically designed for the unsupervised case. Final reinforcement learning can also be handled and I would say that roughly the ANN architectures used for supervised learning can be rather easily adapted to the reinforcement learning case I mean the difference essentially in the reinforcement case is that there is actually not target preference for the value you want to achieve it's rather reward or grading from an external environmental or trainer on the quality of the result. But that can easily be remodeled in the same kind of framework as for the supervised learning case.

Every artificial neural network model have different learning rules. So what I want to exemplify on this slide and actually just two common types. So what you see if the bottom is this on this slide is variance of what is called the Delta learning rule and the reason for the name Delta is that at the center of this approach is that updating all weights because we when we talk about learning and we talk about normally updating of weights. So for the updating of weights, a key factor is the difference between the target output the reference value and the actual value produced by our network. So in the middle you can see T here which is the target and Y which is the outcome. So the difference because to which you could call the error of every problem-solving process are at the core of the calculation of the next value, so it's as I say it's a big difference it will be a big change so it also depends on all of the sign, so any in any kind of difference well a fact the way it's different. Okay so in contrast to that approach which is widely known and there are many versions of that, there is another style of updating which I'll call you the Hebbian learning rule which relates to this basic idea I mentioned from earlier idea by Donald Hebb, that what matters for the updating weight, is the simultaneous compatibility of all the firing of two connected Neuron, so it's also if they fire at the same time that should be the basis for in increasing the weight of that connection but if they don't fire what they find opposite directions that should be the basis for a diminishing or decreasing the value of the weight on that connection. So when you can see on top you have the factor Y and X which are actually the output from two neurons in that case and as you can understand if those values are compatible and positive values will increase away if they are have a opposite direction you will have a negative effect on the update. So these are two genres of updates but as you can see and that the main message is learning is updating awaits.

Looking at the Delta learning a little closer and as I mentioned the Delta learning rule is the kind of rule that controls the update of weights in a lot of cases, in general for most cases for what is called feed-forward ANN which includes the perceptron, a single layer neural network multiple layer which we put back propagation or so on, so it's a pretty wide range of different signs that were more or less the Delta learning rule is used and actually two main principles behind the Delta learning rule. One principle it is based on something called gradient descent and there is also some way of calculating the error, which is the basis for the update, that approach is called the mean square error. So I will now shortly say something about these two phenomena.

So the gradient descent approach or algorithm is an optimization algorithm for finding a minimum of a function. So you know sometimes it's called Steepest Descent. The gradient is actually a multivariable generalization of the variety I mean you have a space of any dimension I mean in a two dimensional space you have the diverter but in the highest place you have it's generalization this means that you can take the derivative in different directions. It's a vector valued function as opposed to derivative which is a scalar value and the gradient is represent the slope of the tangent of the graph of a function, more precisely when in points in the direction of the greatest rate of the trees of the function. To find a local minimum using gradient descent on takes steps proportional to the negative of the gradient at the current point. So as I earlier mentioned it's when we model our problem we can model it as a minimization problem, we can probably mark the same model the same problem in another way so that we can construct a maximization problem. So if you have constructive model or problem in the later way then we actually want to maximize function we want to climb you don't want to go into a valley, we want to climb a hill and in that sense if we do it like that and we can talk about gradient ascent rather than gradient descent naturally. But I mean the thinking is the same. I mean there is a technique we mentioned called Hill Climb and actually Hill climbing and diving into a valley in this case are similar but not equivalent because the pure gradient approach strictly prefers to go in the steepest direction, so if you are on the Hill you find the steepest point. While hill climbing selects the most promising next state, so I mean if you have an analogy with the skier probably the hill climbing or it's reverse is more natural for a normal ski you look down and you usually have feasible is to go somewhere, so you find a feasible state you're trying to find that you don't necessarily go on the steepest part actually or rather the opposite while steepest central wedges and always go for the steepest route to go down. Both of these approaches are Greedy in the sense they do a local observation you stand by your stand, and then you look around and either you go for the steepest or you go for the best place. But you always take a local decision which means that when you come further down you may have seen that actually enough if you want the wrong way because if you view it from a goggle angle there are deeper valleys or higher hills.

The mean square error is a statistical measure that is pretty common and actually the mean square error is to look at another variable and a target value for that variable and then compare that for the outcome of some measurements. And if you make a series of measurements of the same variable you take the difference between the target value in each measurement and you take the square of that and then you get a series of squared errors and you take the sum of that and you divide by the number as with how many measurements yeah so it's pretty straightforward. So this you can somehow apply in the context of an artificial neural network where we look at the error between or the difference between the target value for foreign analysis process and really the output from the network. So if we have a single day time then it will look like simply the difference between the two scalars of the target and the outcome, and the square of that divided by two. And if you take a series of value for Epoch I mean a number you do this for a number items of course you can then do as above to take the difference between all the outcomes from different data items you square them sum them divide by N and you divide by 2. So then you can ask the question why do we divide by 2 and actually that's an arbitrary choice, I mean you can define very much as you like, oh you're inspired here in this case by the mean square error method that just so about why divide by 2, it's because if you take the derivative which we typically do here for example in the Delta method, then as you know the derivative of our square is two times the variable so therefore by dividing by two we get rid of the other two. So this simplifies our calculation so it just put there for convenience and it's arbitrary social but for convenience.

As I hope you understood I try to keep this course a little mathematical as I can, however it's quite obvious that for the feel of machine learning artificial neural networks in particular, it's very difficult to keep entirely out of mathematics so these things creep up, so I collected on this slide some areas from mathematics where is very wise to have some prior knowledge if you've further studies on this subject these of interest. So matrices is an important area vectors coupled to that linearity issues, inner and outer products and show similarity are with difference in metrics and where you we call the chain rule of derivation are important and also some geometrical science are important because many times you want to look at your research space for example with some geometrical or spatial views also graphs of course you can see then are important the basic ideas and technologies perhaps simplifies life. But I'm not going into this I also point at it and I want you to understand that you have a problem in some area you it will be biased to go back to some lectures illiterate or literature all these particular mathematical errors and make a little study there, then you can continue with machine learning. It doesn't have to hinder you to go forward in this area.

Artificial Neural Network is a complex area so for that purpose I really tried now for the purpose of this week to make a little map, so we now started we may talk about the fundamentals, so we are here you can see at the top. So below you can see what we have before us this week as you can see in the middle we will talk about what I will see the main stream of work in ANN which is the kind of feed forward networks where you strictly go from an input layer and hidden layer to an output layer. To the right you can see a kind little different tradition where we which is really inspired very much of what was already mentioned about Hebbian learning. So I would say that this corner of neural network we call as associative memory, and I mean there is no short formula here but that roughly one can say that the right part here the search in memory is more related to handling unsupervised learning, while the feed-forward cases are more related to the genre of supervised learning. Reinforcement learning is more in like in the middle and but I would say it's more natural to adapt to supervised learning methods to handle reinforcement learning then the unsupervised case. However artificial neural networks can be used in all cases depending on how your engineer it. So the new next two parts of this week is this part which is called recurrent neural networks which essentially is about how to handle sequences in temporal data. And the third part or the fourth part is what is called Convolution Neural network which have to do with special mechanisms to handle perception issues particularly imaging. So then in the end we will try to tie things together in lecture. So this is a structure we will follow and yeah finally just a few words about the final deep learning as I already said it was imagined in 1986 actually it wasn't used in 1986, it was only in 2000 that anybody used the Deep learning for artificial neural network and it actually wasn't widely used until 2012 when there were some real success stories by application of this kind of systems, then the word deep learning was started to be used of this kind of systems. Now it's a dominating term and one can discuss whether is just a term for the state of the art artificial neural network or is something more specific and this we will discuss shortly in the final lecture. So that was the end of this lecture thank you very much the next lecture 6.2 will be on the topic of perceptrons.