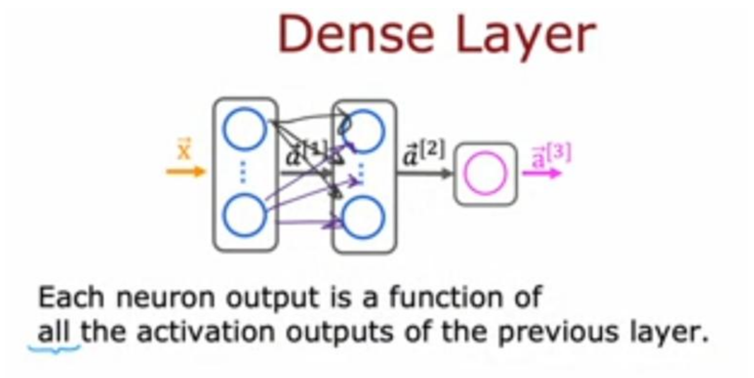
Types of additional layers

All the neural network layers with you so far have been the dense layer type in which every neuron in the layer gets its inputs all the activations from the previous layer. And it turns out that just using the dense layer type, you can actually build some pretty powerful learning algorithms. And to help you build further intuition about what neural networks can do.

It turns out that there's some other types of layers as well with other properties. In this video I'd like to briefly touch on this and give you an example of a different type of neural network layer. Let's take a look to recap in the dense layer that we've been using the activation of a neuron in say the second hidden layer Is a function of every single activation value from the previous layer of a one.

But it turns out that for some applications, someone designing a neural network may choose to use a different type of layer.



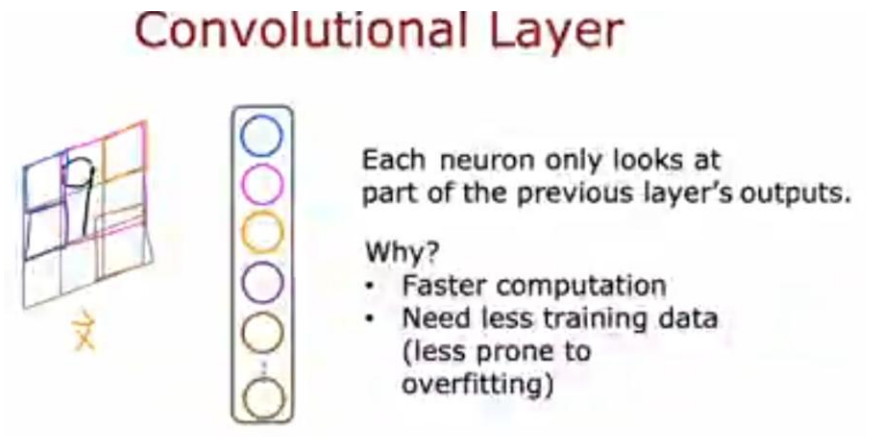
Dense Layer

Each neuron output is a function of
all the activation outputs of the previous layer.

$$\vec{a}_1^{[2]} = g\left(\vec{w}_1^{[2]} \cdot \vec{a}^{[1]} + b_1^{[2]}\right)$$

One other layer type that you may see in some work is called a convolutional layer. Let me illustrate this with an example. So what I'm showing on the left is the input X. Which is a handwritten digit nine. And what I'm going to do is construct a hidden layer which will compute different activations as functions of this input image I might say this neuron can only look at the pixels in this little rectangular region.

Second neuron, which I'm going to illustrate in magenta is also not going to look at the entire input image X instead, it's only going to look at the pixels in a limited region of the image. And so on for the third neuron and the 4th neuron and so on and so forth. Down to the last neuron which may be looking only at that region of the image.



Convolutional Layer

Each neuron only looks at
part of the previous layer's outputs.

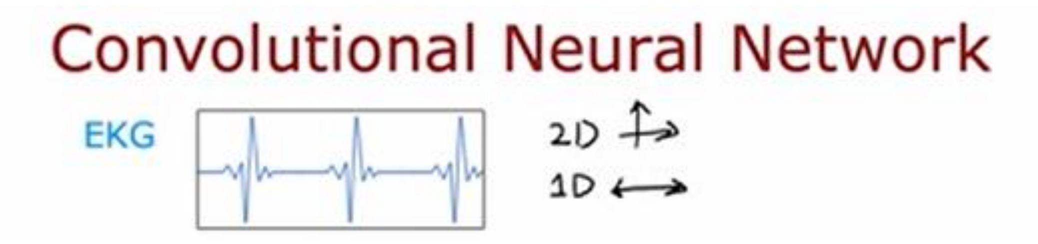
Why?
- Faster computation
- Need less training data
  (less prone to
  overfitting)

So why might you want to do this? Why won't you let every neuron look at all the pixels but instead look at only some of the pixels? Well, some of the benefits are first, it speeds up computation. And second advantage is that a neural network that uses this type of layer called a convolutional layer can need less training data or alternatively, it can also be less prone to overfitting.

You heard me talk a bit about overfitting in their previous course but this is something that will dive into greater detail on next week as well. When we talk about practical tips for using learning algorithms and this is the type of layer where each neuron only looks at a region of the input image is called a convolutional layer.

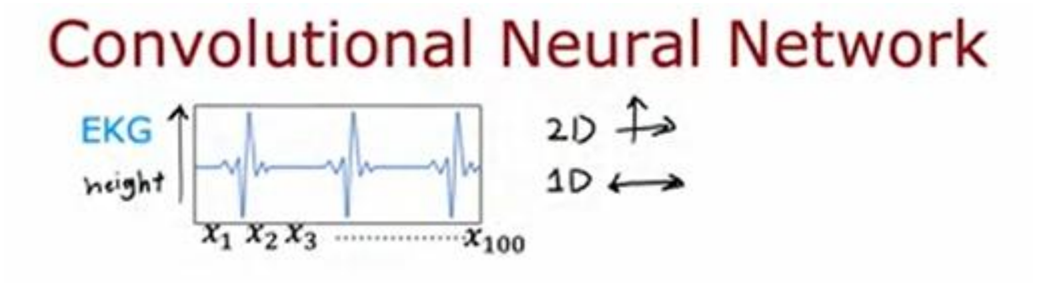
It was a researcher John Macoun who had figured out a lot of the details of how to get convolutional layers to work and popularized their use. Let me illustrate in more detail a convolutional layer. And if you have multiple convolutional layers in a neural network sometimes that's called a convolutional neural network.

To illustrate the convolutional layer of convolutional neural network on this slide I'm going to use instead of a two D image input. I'm going to use a one dimensional input and the motivating example I'm going to use is classification of EKG signals or electrocardiograms.



So if you put two electrodes on your chest you will record the voltages that look like this that correspond to your heartbeat. This is actually something that my Stanford research group did research on. We were actually reading EKG signals that actually look like this to try to diagnose if patients may have a heart issue.

So an EKG signal and electoral cardia graham ECG in some places EK G. In some places there's just a list of numbers corresponding to the height of the surface at different points in time. So you may have say 100 numbers corresponding to the height of this curve at 100 different points of time. And the learning tosses given this time series, given this EKG signal to classify say whether this patient has a heart disease or some diagnosable heart conditions.

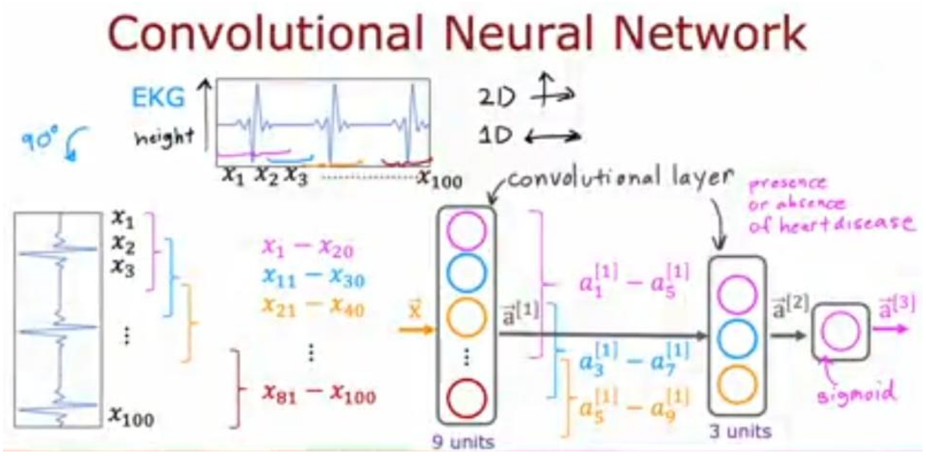


Here's what the convolutional neural network might do. So I'm going to take the EKG signal and rotated 90 degrees to lay it on the side. And so we have here 100 inputs X one X two all the way through X 100. Like. So and when I construct the first hidden layer Instead of having the first hidden unit take us input all 100 numbers. Let me have the first hidden unit. Look at only X one through X 20.

So that corresponds to looking at just a small window of this EKG signal. The second hidden unit shown in a different color here. Well look at X 11 through X 30, so looks at a different window in this EKG signal. And the third hidden there looks at another window X21 through X 40 and so on. And the final hidden units in this example. Well Look at X 81 through X100. So it looks like a small window towards the end of this EKG time series.

So this is a convolutional layer because these units in this layer looks at only a limited window of the input. Now this layer of the neural network has nine units. The next layer can also be a convolutional layer. So in the second hidden layer let me architect my first unit not to look at all nine activations from the previous layer, but to look at say just the first 5 activations from the previous layer.

And then my second unit In this second hidden there can look at just another five numbers, say A3-A7. And the third and final hidden unit in this layer will only look at A5 through A9. And then maybe finally these activations.



A2 gets inputs to a sigmoid unit that does look at all three of these values of A2 in order to make a binary classification regarding the presence or absence of heart disease. So this is the example of a neural network with the first hidden layer being a convolutional layer.

The second hidden layer also being a convolutional layer and then the output layer being a sigmoid layer. And it turns out that with convolutional layers you have many architecture choices such as how big is the window of inputs that a single neuron should look at and how many neurons should layer have. And by choosing those architectural parameters effectively, you can build new versions of neural networks that can be even more effective than the dense layer for some applications.

To recap, that's it for the convolutional layer and convolutional neural networks. I'm not going to go deeper into convolutional networks in this class and you don't need to know anything about them to do the homework and finish this class successfully.

But I hope that you find this additional intuition that neural networks can have other types of layers as well to be useful. And in fact, if you sometimes hear about the latest cutting edge architectures like a transformer model or an LS TM or an attention model. A lot of this research in neural networks even today pertains to researchers trying to invent new types of layers for neural networks. And plugging these different types of layers together as building blocks to form even more complex and hopefully more powerful neural networks. So that's it for the required videos for this week.