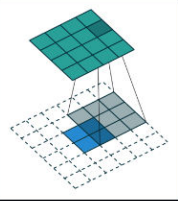
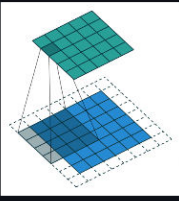
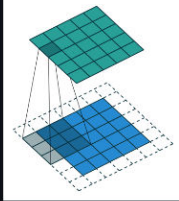
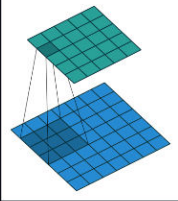
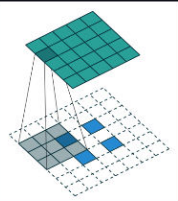
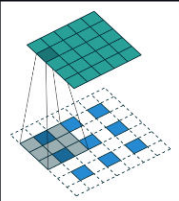
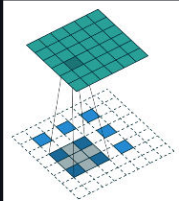


- Architecture consists of contracting patch to capture context
- Symmetric expanding path that enables precise localisation
- The input and the output are not the same
 - This is partially because of padding used on the input
- The main feature of the U-Net is that it has a **downsampling path followed by a matching upsampling path**
- It also contains skipped connections, which means that some downsampling layers have connections to their matching upsampling layers
- 2 layers followed by a max pool layer
- Upsampling is done using a **transposed convolution**

- https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

			
No padding, no strides, transposed	Arbitrary padding, no strides, transposed	Half padding, no strides, transposed	Full padding, no strides, transposed
			
No padding, strides, transposed	Padding, strides, transposed	Padding, strides, transposed (odd)	

- The skipped connections are done such that half of the connections come from the previous layer and the other half from the layer below, then the nodes are simply concatenated together
- **Downsample:** use two 3x3 valid (unpadded) convolutions, RELU, followed by 2x2 (2 stride) max pooling
 - Each downsampling step doubles the number of feature channels
- **Upsample:** upsample feature map followed by a 2x2 up-convolution that halves the number of feature channels
 - Followed by a **concatenation with correspondingly cropped feature map**
 - **Two 3x3 convolutions, RELU**
 - **Cropping necessary due to loss of border pixels in every convolution**
- **Final layer** uses a 1x1 convolution to map feature vectors to desired number of classes
- **Output** uses pixel-wise softmax with cross entropy loss
 - **Instead of that, you can also have a sigmoid with one single channel and binary cross entropy instead**
- If we always used a 3x3 kernel with same padding, our dimensions would stay the same, **however:**
 - This would be quite expensive as our image would stay very large
 - Our **receptive field** wouldn't be very large because it **won't grow fast enough** despite it always growing
- On the **contraction** path, we are trying to learn **what** is in the image but at the same time we are **losing spatial information** - losing where that information is
- On the **expansion** path, we try to **figure out where** the information is supposed to be again
 - Pooling operators are replaced by **upsampling operators**
- The skipped values are supposed to help us get back on the right path
 - **To make the sizes match, the skipped connections are cropped to match in size**
- The network does not have any fully connected layers, which is not usual of convolutional networks
- The network only uses valid part of each convolution, meaning that the segmentation map only contains pixels for which full context is available, meaning that the output is of a different resolution than the input

- The way we make this network work for large images is that we put in a small section of the image at a time.
- We add mirrored padding around the **entire** image such that the subimages at the borders contain the mirrored padding.
 - We do this so that the border pixels have some context to learn from
 - You then stitch everything up so that you get the final output
- They used **weighted loss** so that they could prioritise having the borders be very accurate
 - This was to help with separation of touching objects of the same class
- For seamless tiling of segmentation map, input tile size is chosen such that **all 2x2 map-pooling operations are applied to a layer with an even x- and y- size.**

