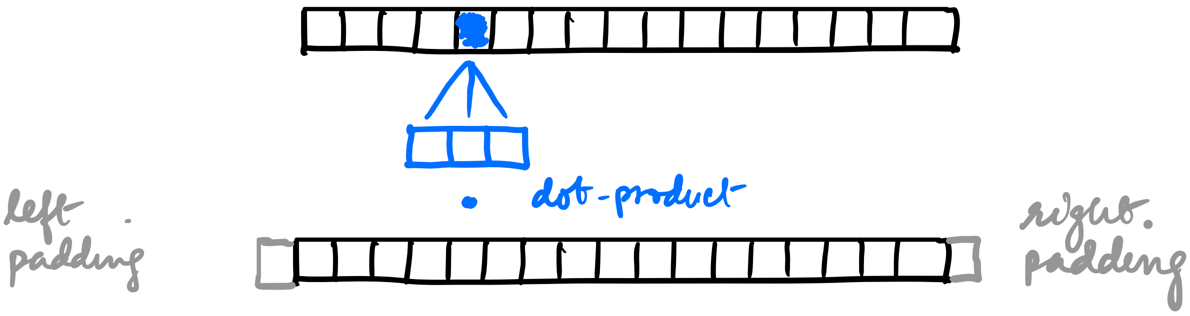
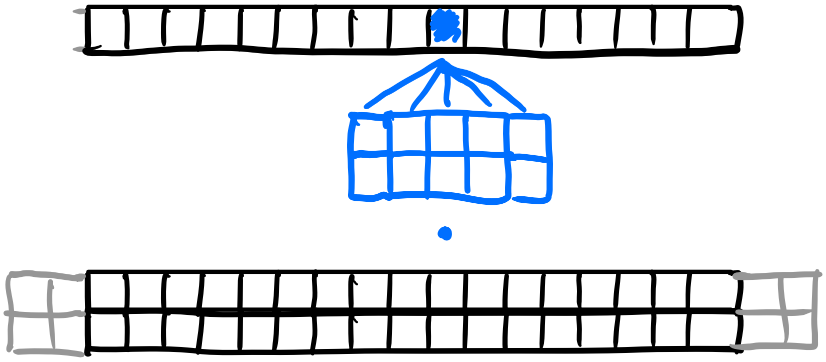
TEMPORAL CONVOLUTION NETWORKS (TCN)

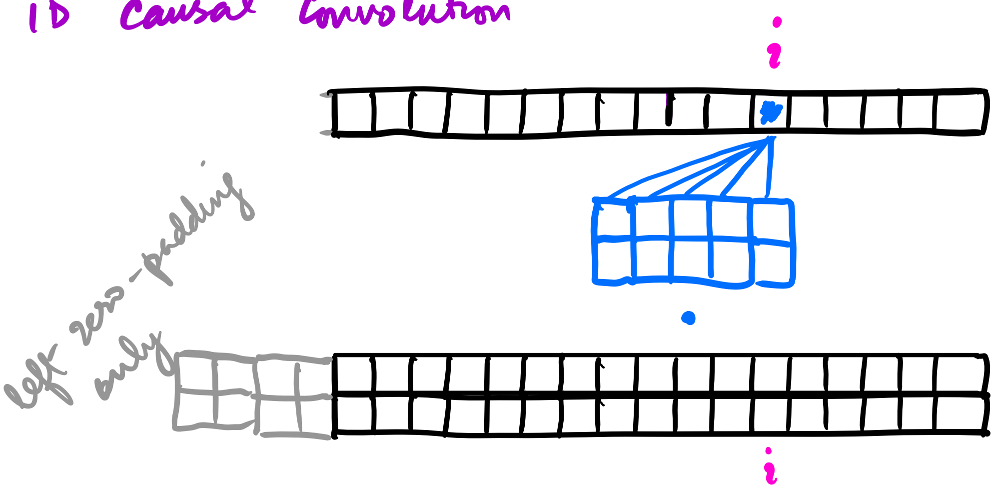
1D Convolution



1D Convolution with multi-dimensional input features



1D Causal Convolution



Output value at i depends upon input values

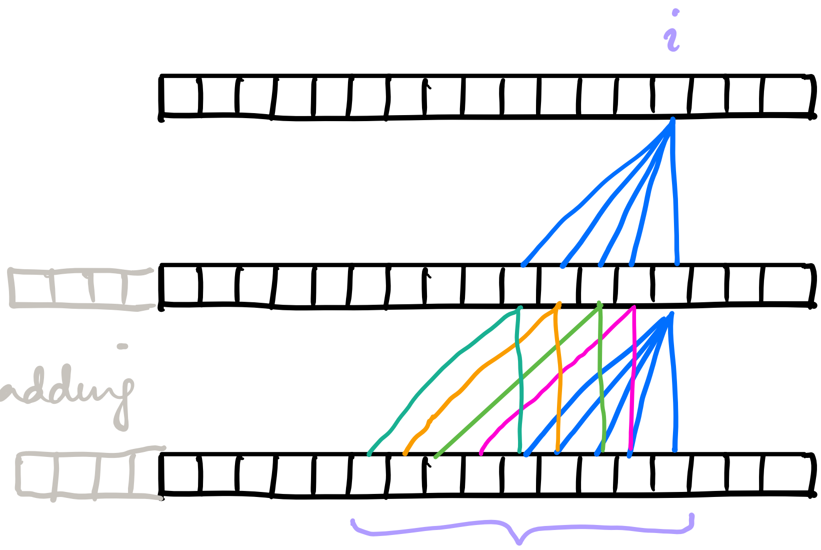
$$i,\ i-1,\ i-2,\ \dots,\ i-(2w+1)+1$$

half width of the kernel

$2w+1$ is the size of the kernel.

We saw a similar limitation in the case of CNNs. That output depends upon a local neighbourhood. There we dealt with it using dilated convolutions.
Also we can stack multiple layers on top of each other.

Stack multiple Layers



zero-padding

levels = 2
kernel = 5
receptive field = 9

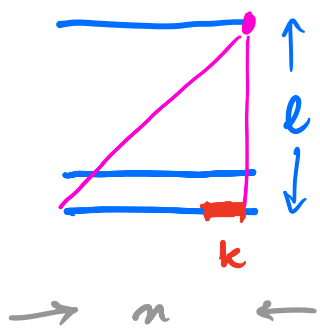
kernel width = 2w+1

Receptive field = $1 + l(k-1)$

levels

How many layers do we need to make receptive field equal to the input?

l = number of layers
k = kernel width
n = input size.



$$n = 1 + l(k-1)$$

$$\Rightarrow l(k-1) = n-1$$

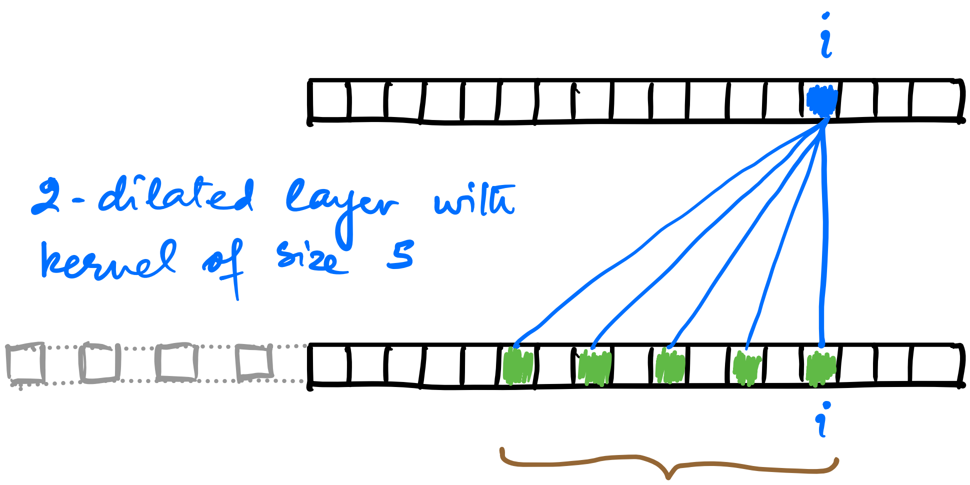
$$\Rightarrow \quad l = \left\lceil \frac{n-1}{k-1} \right\rceil$$

For a fixed kernel size, the number of layers required to cover the entire input is linear in the length of the input.

- → Network will become very deep and this will result in large models, resulting in slow training.
- → Large number of layers also lead to degradation problems related to the gradient of the loss function.

Dilated Convolution

Another scheme to increase receptive field size



kernel width

Receptive field = $1 + d(k-1)$

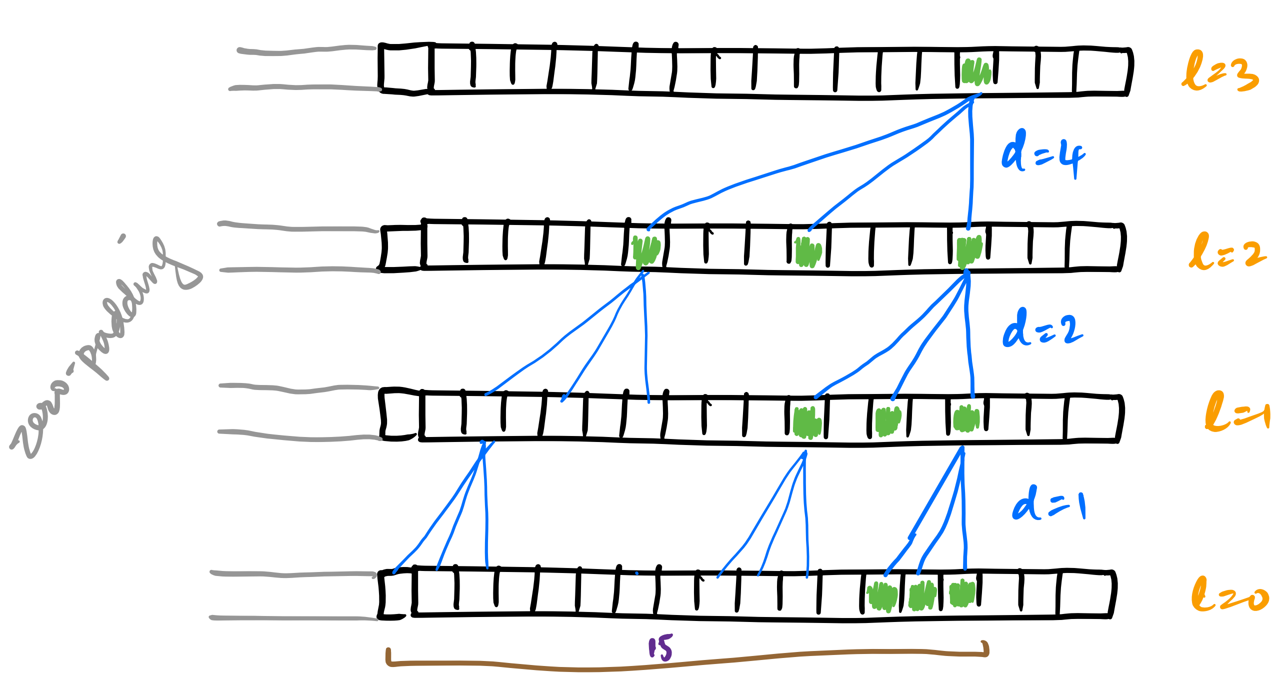
dilation

We can combine dilated kernels with levels to further increase the receptive field size.

- → However for fixed dilation parameters, the number of levels required to achieve full coverage (of the input) is still linear in the length of the input.

A solution is to increase dilation size as we move up the levels.

→ This is achieved by fixing a base dilation for the first layer and exponentially increasing the dilation at every subsequent layer.



Here $d = b^l$ where b is base. For this example b is 2.

$$\text{Receptive-field } r = 1 + \sum_{l=0}^{L-1} (k-1) \cdot b^l$$

Combining dilated kernels with levels can lead to holes in the receptive field coverage. To avoid this ensure that kernel size k is atleast as large as the base.

Receptive Field computation $\quad r = 1 + \sum_{l=0}^{L-1} (k-1)\, b^l$

$$r = 1 + \underbrace{(3-1)(1)}_{l=0} + \underbrace{(3-1)(2)}_{l=1} + \underbrace{(3-1)(4)}_{l=2}$$

$$= 1 + 2 + 4 + 8$$

$$= 15$$

$$r = 1 + (k-1) \sum_{l=0}^{L-1} b^l$$

$$= 1 + (k-1) \frac{b^L - 1}{b - 1}$$

We can use this expression to find the number of layers that we need to make the receptive field equal to the size of the input.

Say we want to find L:

$$r = 1 + (k-1) \frac{b^L - 1}{b-1}$$

$$\Rightarrow \frac{r-1}{k-1} = \frac{b^L - 1}{b-1}$$

$$\Rightarrow b^L - 1 = \frac{(r-1)(b-1)}{(k-1)}$$

$$\Rightarrow b^L = \frac{(r-1)(b-1)}{(k-1)} + 1$$

$$\Rightarrow L \log b = \log\left[\frac{(r-1)(b-1)}{(k-1)} + 1\right]$$

$$\Rightarrow L = \frac{\log\left[\frac{(r-1)(b-1)}{(k-1)} + 1\right]}{\log b}$$

This suggests that the number of layers needed now are logarithmic in the size of the input.

We can use the following expression to compute the padding that we need at each layer.

$$p = b^i (k-1)$$

zero-padding needed at layer i.

Back to TCN

All that we have done thus far is to construct a fancy linear autoregressive model.
To deal with non-linearities we need to include activation fn.s, residual connections, etc.