TCN

The simplest way to examine the contribution of each architectural parameter is to vary it in isolation

RNN’s suffer from vanishing gradient

We cannot consider values from the future -> casual convolutions

Casual vs standard convolutions

Casual just adds padding so limits the kernel to only use values up to timestep p , to predict n

Receptive field the outputs dependencies from the original input

We want it very large to capture long range dependencies

Might use large filters , but this adds weights , one matrix for each filter , and we have lots of filters per convolution

Instead we stretch by dilation factor

If we only have one layer == extreme sparsity , but we stack layers so uses all inputs as as receptive field

For learning if the entwor is deep , use residual connections , help improve the propergation of gradience , dropout puts away some hidden representation values so can be generalized better.