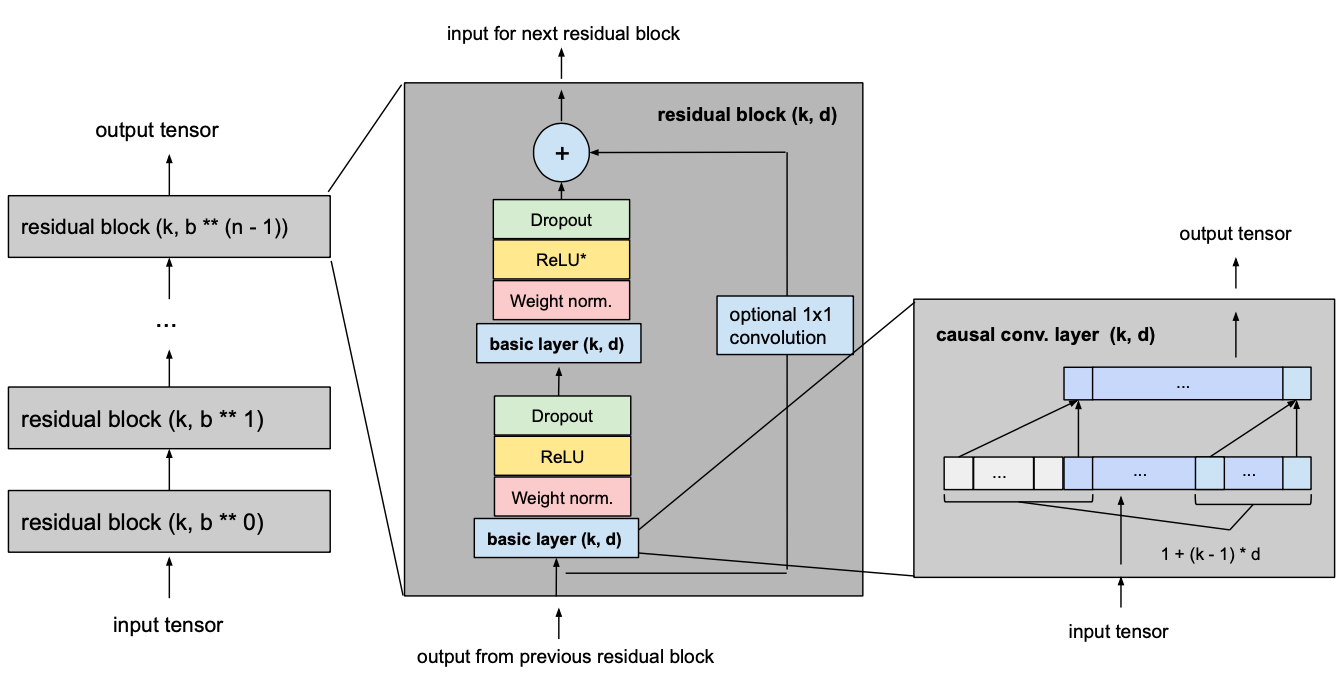
**Deep TCN**

For most deep learning practitioners, sequence modeling is synonymous with recurrent networks. Yet recent results indicate that convolutional architectures can outperform recurrent networks on tasks such as audio synthesis and machine translation. So Bai et al. developed an algorithm based on convolutional networks, Deep TCN (Temporal Convolutional Network), which outperforms canonical recurrent networks such as LSTMs across a diverse range of tasks and datasets.

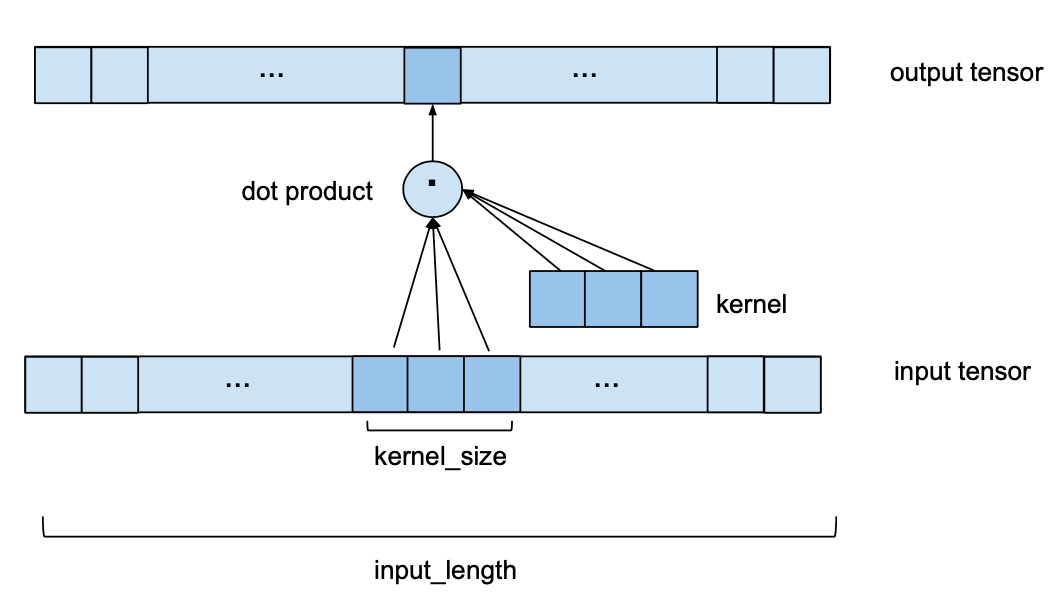
The structure of this model is s follow.



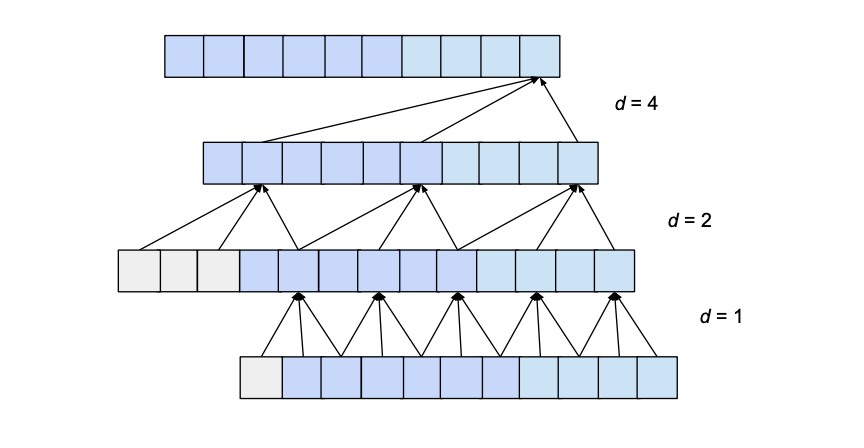
Generally speaking, the model has some residual blocks and the output of each block is the input of the next block. Each block has a basic layer which is the most important part of that and to make the model more than just an overly complex linear regression model, activation functions need to be added on top of the convolutional layers to introduce non-linearities. ReLU activations are added to the residual blocks after both convolutional layers. To normalize the input of hidden layers (which counteracts the exploding gradient problem among other things), weight normalization is applied to every convolutional layer. In order to prevent overfitting, regularization is introduced via dropout after every convolutional layer in every residual block. The following figure shows the final residual block.

Each basic layer has three general hyperparameters that should be determined before starting fitting process.

* **Kernel size**: kernel size explains how many number of elements contribute in computing each cell of output. For example in this picture the kernel size is 3.



* **Dilation**: Dilation refers to the distance between the elements of the input sequence that are used to compute one entry of the output sequence. For example the below picture demonstrates 1,2, and 4 dilation.



* **Number of layers**: It’s about the number of layer in each basic layer.

Based on the input length and output length some relation between these hyperparameters should be established. For example sometimes for matching the input and output size, we need zero-padding in layers or based on the input length, kernel size, and dilation; the length of next layer should be set. In addition to this, the performance of this model relies on the value of its hyperparameters. So it’s necessary to find the best value of the hyperparameters. PAI as an Automated AI solution in addition to find the best value of model’s parameters, investigates the best hyperparameters with hyperparameter optimization algorithm and you don’t need to set them manually and just enough to import your data and enjoy forecasting.