<https://realpython.com/python-keras-text-classification/>

When you are working with sequential data, like text, you work with one dimensional convolutions, but the idea and the application stays the same. You still want to pick up on patterns in the sequence which become more complex with each added convolutional layer.

You can see that 80% accuracy seems to be tough hurdle to overcome with this data set and a CNN might not be well equipped. The reason for such a plateau might be that:

* There are not enough training samples
* The data you have does not generalize well
* Missing focus on tweaking the hyperparameters

CNNs work best with large training sets where they are able to find generalizations where a simple model like logistic regression won’t be able.

<https://machinelearningmastery.com/develop-n-gram-multichannel-convolutional-neural-network-sentiment-analysis/>

A standard model for document classification is to use an Embedding layer as input, followed by a one-dimensional convolutional neural network, pooling layer, and then a prediction output layer.

The kernel size in the [convolutional layer](https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/) defines the number of words to consider as the convolution is passed across the input text document, providing a grouping parameter.

A multi-channel convolutional neural network for document classification involves using multiple versions of the standard model with different sized kernels. This allows the document to be processed at different resolutions or different n-grams (groups of words) at a time, whilst the model learns how to best integrate these interpretations.

This approach was first described by Yoon Kim in his 2014 paper titled “[Convolutional Neural Networks for Sentence Classification](https://arxiv.org/abs/1408.5882).”

In the paper, Kim experimented with static and dynamic (updated) embedding layers, we can simplify the approach and instead focus only on the use of different kernel sizes.

This approach is best understood with a diagram taken from Kim’s paper:

Each channel is comprised of the following elements:

* Input layer that defines the length of input sequences.
* Embedding layer set to the size of the vocabulary and 100-dimensional real-valued representations.
* One-dimensional convolutional layer with 32 filters and a kernel size set to the number of words to read at once.
* Max Pooling layer to consolidate the output from the convolutional layer.
* Flatten layer to reduce the three-dimensional output to two dimensional for concatenation.

The output from the three channels are concatenated into a single vector and process by a Dense layer and an output layer.

The function below defines and returns the model. As part of defining the model, a summary of the defined model is printed and a plot of the model graph is created and saved to file.