Take-home messages for building a CNN architecture

- Filter (Kernel) size:
  - Start by using small filters to collect local information then increase filters size to represent more global information.
  - If you think that a big amount of pixels are necessary for the network to recognize the object you will use large filters (as 11x11 or 9x9). If you think what differentiates objects are some small and local features you should use small filters (3x3 or 5x5). **In general we use filters with odd sizes.**
- Filter number:
  - The number of filters should be low in the beginning such that it detects low-level features.
  - The number of filters is increased to increase the depth of the feature space thus helping in learning more levels of global abstract structures.
- Keep adding layers until you over-fit.
  - Once you achieved a considerable accuracy in our validation dataset we can use regularization components like l1/l2 regularization, dropout, batch norm, data augmentation etc. to reduce over-fitting
  - Or stop at the layer before over-fitting.
- Use classic networks as inspiration:
  - Trends in the layers Conv-Pool-Conv-Pool or Conv-Conv-Pool-Conv-Conv-Pool
  - Trend in the number of filters 32–64–128 or 32–32-64–64 or trend in filter sizes, Max-pooling parameters etc.
- Use padding.
- Pooling-layer:
  - Max-Pooling is generally used. The objective is to down-sample an input representation, reducing its dimensionality by keeping the max value.
- Dropout layer:
  - Prevents over-fitting. A fully connected layer occupies most of the parameters, neurons develop co-dependency amongst each other during training which leads to over-fitting of training data.
- Batch normalization layer:
  - The normalized input after passing through all the initial layers becomes too big or too small while it reaches the last layers which causes a problem that impacts learning. To solve this we add a batch normalization layer to standardize the input given for the following layers