1. **Filter**: A filter in a convolutional network is a matrix that has a smaller shape than the input matrix. It looks for features by essentially sliding across the input matrix, multiplying its values (random weights or zeros and ones) by the corresponding input values, and summing them so that each new cell is a summation of the multiplied input. The locations of the features will have higher values. This approach is beneficial because the model only has to figure out the weights for the values in the smaller region of the filter vs. weights for every value in the image, which is a better use of computational resources.
2. **Feature**: A feature is some property of the input that the network uses to make predictions. Rather than trying to learn the weights for features at every possible location in the image like fully connected networks, convolutional networks look for features in smaller regions that aren’t necessarily localized. Early layers look for lower-level features that are common to almost everything (e.g. edges, orientations, colors, etc.), and then subsequent higher-level layers look for more complex patterns by combining these lower-level features in meaningful ways.
3. **Feature map**: A feature map is the output of a convolutional layer. Each feature map has nodes that are looking for the same feature. The map shows which parts of the image activated that feature the strongest, i.e. the locations where the nodes found the feature and how strongly they found it. As the subsequent convolutional layers look for more complex combinations of these features, the maps will reflect those locations and values.
4. **Pooling**: A pooling operation takes a convolutional layer and reduces it to a smaller size, typically by either taking the maximum or average value across the area. Pooling helps the model’s classification accuracy on images where the position or orientation of the objects change. This is important because it’s looking for features that aren’t necessarily localized.