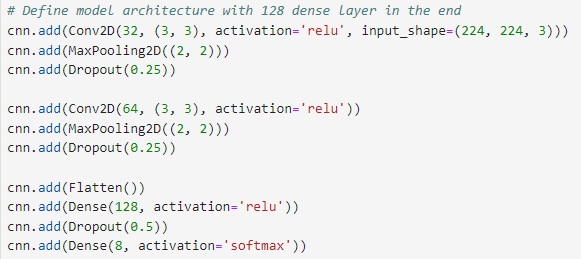
**Convolutional Neural Networks** is a widely used deep learning technique for accurate and advanced machine learning projects. Large companies use CNN’s for creating accurate and efficient deep learning models, some examples include advanced image classification, sign language translation, lip reading algorithms and way more. CNN’s are in many cases used with image data, either frames in a video or standalone images.

The way a Convolutional Neural Network works is by learning patterns within data. With images, the CNN would be able to learn certain patterns and recognize those patterns with the classes. The structure of the network is highly configurable and requires a lot of experience to understand what really happens behind the scenes. The machine learning engineer will define each layer of the network, some layers search for larger patterns within an image, while other layers might search for smaller patterns. The most common library for such deep learning is TensorFlow, which is developed by Google. An example of how the layers in a convolutional neural network are structured are shown below in (*code example 1*). There are Conv2D layers, which are smaller portions of the images that will try to learn patterns in that shape, the first Conv2D layer is also called the input layer in our case, where the input shape of the image is specified. The Dropout layers will randomly disable some neurons in the network, this can help with preventing overfitting. The last layer is a dense layer that is specified with 8, which is the number of classes. An activation function, in our case it is the SoftMax activation function which will return a value between zero and one for each class, the class with the highest value is the predicted class.



*code example 1*

When training a CNN, the model will generate training data loss and validation data loss. It then uses these results to adjust the weights in the network to gain a better result in the next epoch. The progress of the training can be plotted with matplotlib to get a visual representation on how the training process went, also to compare the training accuracy versus the validation data accuracy. If the training accuracy ends up being too high, it might be a sign of overfitting, which will result in the model poorly predicting unseen images.

When stepping into this project, we wanted to learn more about CNN’s even though it is not a part of this course. We learned a lot about how the Convoutional Neural Networks are built and how they operate. However, we had a hard time getting good results with the little knowledge we had at the time. We managed to properly load in the data using the TensorFlow data pipeline, define a model structure and run predictions. The results of the CNN training seemed very promising, however when predicting images, we never got good results. In most cases, the model overfitted and was very bias in one of the classes.

We are sure that with proper knowledge and experience with convolutional neural networks, it would be the best possible model in this projects case. It has been proven to give very good results, even on data with smaller details. The model structure must be engineered and by trial and error, to eventually get the wanted results.