For CNN, the math is difficult to grasp, but I understand that we are comparing regions of the input image with the kernel to compute an output score that shows how much the input region matches the kernel. (Higher number = better match)

For padding, I guess padding can be useful in case you want to apply the kernel onto the boundaries of the image, although I'm not sure how useful the boundary information is. Maybe it is useful in self-driving to detect parts of a car coming near your own car.

For stride, it seems useful in case the input resolution is so large that it would take too long to compute the output. I wonder if it is significantly faster than resizing the original image or if it's a similar principle.

For pooling, I understand that it can be used for downsampling (compression) of images, but I'm not sure how it would work for color. Does it take the max of each Red, Green, Blue and form a new color based on that?

For the CNN example, it seems like the original Lenet has a high training loss near the end because it did not learn some features, while the modernized vesion quickly attained a low training loss. However, the modernized version may have to be careful of overfitting.