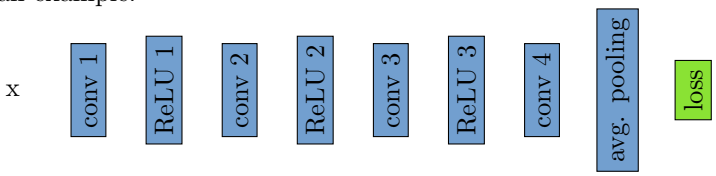
Exercise 6: Convolutional models - Solution

Name: ..... UTID: .....

Last week we looked at several popular architectures in class. We compared their model size, inference speed and accuracy. A recap of popular models can be seen on the right. While AlexNet and VGG have fully connected layers, all other models listed do not. They are all-convolutional. See below for an example.



In a all-convolutional network all computation, including the output classification, is performed using convolutions. Moreover, all-convolutional networks often produce more than a single prediction for an image. These predictions are then averaged in a pooling layer.

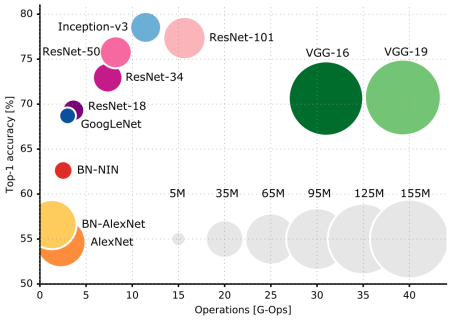


Figure 1: Comparison of popular architectures w.r.t. accuracy (on imagenet classification), operations, and model size.

a) What are some of the advantages of all-convolutional networks?

- ☒ Fewer parameters
- ☐ Faster inference
- ☐ Faster training
- ☒ Run on images of any resolution and aspect ratio

b) All-convolutional network also overfit less. In fact, they mimic some of the strategies we learned and implemented in class inside the network. Which strategies to all-convolutional networks implicitly employ to reduce overfitting?

- ☐ Batch Normalization
- ☐ Early Stopping
- ☐ Weight regularization
- ☒ Parameter Sharing
- ☐ Dropout
- ☐ Color augmentation
- ☒ (Random) cropping
- ☒ Ensembles

c) Should you always make your network all-convolutional (with multiple spatial outputs you average)? If yes, briefly explain why, if no, give a counter-example.

Counter-example: Any domain where images are strongly aligned (e.g. faces). Here we do not gain much from multiple spatial outputs that are averaged. In fact, it might be better to reason about the global image in a common reference frame.

d) You trained a fully-connected network (e.g. AlexNet). It took one month to train. Now that you know about all-convolutional networks, you'd like to convert your architecture to an all-convolutional one. Do you have to retrain your model, or is there a way to skip training the all-convolutional model?

Any fully connected layer can be converted into a 1×1 *convolution (on a* 1×1 *feature map). The first fully connected layer is special: Let's assume we have a* $w \times h$ *feature map. A convolution of this feature map with a* $w \times h$ *kernel is equivalent to a fully connected layer.*