Architecture

- 6 Configurations ranging from 8 conv layers to 16 conv layers and 3 fully connected layers. ReLU applied after each conv layer. There are 5 maxpool layers. The last layer of the network is a softmax.
    - Convolutional layers were 3x3 with stride 1, padding is such that output and input have same dimensions (except for number of channels)
        - Other networks reduce spatial resolution of feature maps more aggressively in first layers to decrease computation
    - *Channels of conv layer increase by factor of 2 after each maxpool layer*
    - Maxpool layers were 2x2 with stride 2 which reduced dimensions by half
    - First two FC have 4096 channels, third performs 1000-way ILSVRC classification (one channel per class)
- *One configuration (A-LRN) used local response normalization which didn't improve performance and only increased memory consumption and computation time.*
    - *LRN: ReLU neurons have unbounded activations and LRN allows for normalization of local neighborhoods of excited neurons. This dampens uniformly large activations in a given neighborhood and makes excited neurons more obvious.*

- Tricks: A stack of 2 3x3 convolutional layers without pooling is the same as a 5x5 receptive field, and a stack of 3 3x3 conv layers without pooling is the same as a 7x7 receptive field.
    - Using multiple non-linear rectification layers (ReLU) instead of a single one makes the decision function more discriminative. [regularization]
    - Reduces number of parameters
    - *Can be seen as imposing regularization on the 7x7 conv filters, forcing them to have a decomposition through the 3x3 layers (with non-linearity [ReLU activation function] injected in-between)*
- Use of 1x1 conv layers in config C is a way to increase non-linearity of the decision function without affecting receptive fields of the conv layers
- *Contrast: Research has shown that increased depth led to better performance, but GoogLeNet which uses small convolution filters has a more complex topology (22 weight layers, multiple filter sizes) and the spatial resolution of the feature maps was reduced more aggressively in the first layers to decrease computation at the expense of loss information loss/performance*

Preprocessing

- Subtracting the mean RGB value (computed on the training set) from each pixel

Training

- Mini-batch gradient descent with momentum based on back-propagation
- Using momentum allows gradient descent to move more quickly through areas where surface curves much more steeply in one dimension than in another/local minimas

    - Accelerates convergence
    - When momentum = 0 we recover gradient descent formula
- Batch size 256, momentum 0.9, learning rate initially 10^-2 and reduced by factor of 10 when validation accuracy stopped improving
    - Learning rate decreased 3 times and learning stopped after 370 K iterations (74 epochs)
- Training regularized by weight decay (L2 penalty multiplier set to 5E-4) and dropout regularization for first two fully connected layers (dropout ratio set to 0.5)
    - Reduces model training error and overfitting via smaller weights
    - Weight decay: to prevent overfitting, each time weight w is updated with gradient loss with respect to w, we subtract from w multiplier*w
    - Gives weights a tendency to decay towards 0

- Data augmentation: Input images were randomly cropped from rescaled training images
    - Crops underwent random horizontal flipping and random RGB color shift
- S is the smallest side of an isotropically-rescaled (square) training image from which the Conv Net input is cropped to size 224x224
- S can be any value not less than 224, S=224 → crop=entire image, S>>224 → crop=small part of image
- 1) Single scale training 2) multiscale training
    - 1) Fix S. Evaluated models trained at two fixed scales/values of S = 256 and S = 384.
    - 1 cont.) Given ConvNet config, trained network using S = 256. To speed up training of S = 384 network, initialized with weights pre-trained with S = 256 and given smaller initial learning rate of 10E-3
    - 2) Individually rescale training images using S randomly sampled from range [256, 512].
        - Relevant as objects can be different sizes, taken into account during training, seen as training set augmentation by scale jittering (model is trained to recognize objects over wide range of scales)
    - 2) For speed reasons, multi-scale models trained by fine-tuning all layers of a single scale model with same configuration, pre-trained with fixed S = 384

- Tricks: Configuration A (most shallow) was shallow enough to be trained with random initialization. To train the deeper architectures, the first four convolutional layers and last three fully connected layers were initialized with the values from the layers of net A. The intermediate layers were initialized randomly. [parallelization]
    - Learning rates of pre-initialized layers not decreased, allowing for them to change during learning
    - Random initialization used weights sampled from a normal distribution with 0 mean and 10^-2 variance. Biases were initialized to 0.

        - After paper was submitted, found it was possible to initialize the weights without pre-training by using the random initialization procedure of Glorot & Bengio
- Despite larger number of parameters/greater depth of these nets compared to others, these nets required less epochs to converge due to 1) implicit regularization imposed by greater depth/smaller conv filter sizes 2) pre-initialization of certain layers