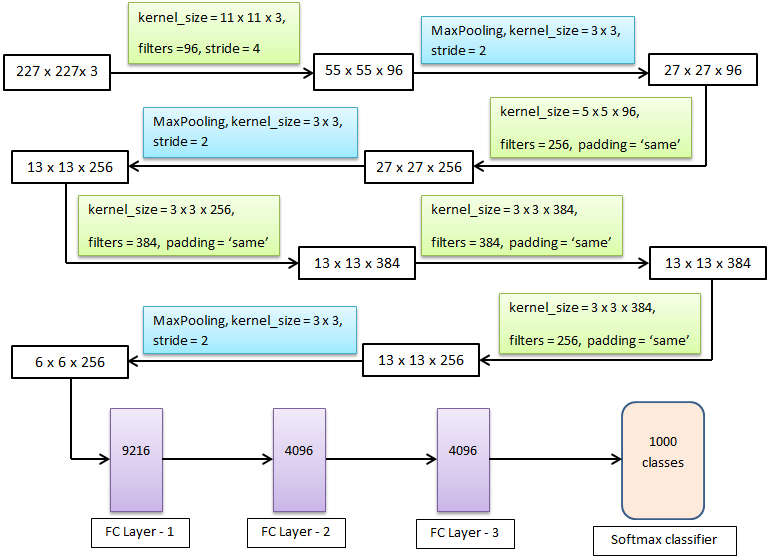
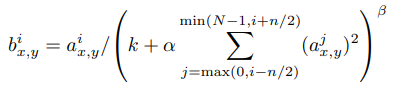
AlexNet Summary

* **Dataset used:** A subset of Imagenet
* **Size of dataset:** 1.2M training images, 50K validation images, 150K test images
* **Parameters of model:** ~60M
* **Neurons:** 650,000
* **Classes:** 1000
* **Dataset preprocessing:**
  + Rescaled each image to 256 x 256 and cropped out centrally.
  + Subtracted the mean activity from each pixel.
* **Parameter initialization:**
  + Each layer’s weights are initialized as N(0,0.01).
  + Biases of 2nd, 4th, 5th conv layers, and in all FC layers initialized as 1; in the remaining layers initialized as 0.
* **Activation function:** ReLU (max{0,x}) (non-saturating non-linear)
  + To overcome the problem of vanishing gradient with sigmoid or tanh.
  + For faster & cheaper computation.
* **GPU Utilization:** 
  + Parallelization scheme trained half neurons of each layer in each GPU.
  + The GPUs communicated in only certain layers.
* **Layers:** 5 convolutional layers + 3 fully-connected layers followed by a softmax-classifier with 1000 classes.

A brief outline of the network architecture (ignoring GPU utilization) :



* **Normalization:** Local response normalization (or brightness normalization) is used to implement lateral inhibition (capacity of an excited neuron to subdue its neighbors). If we normalize around the local neighborhood of an excited neuron, it becomes even more sensitive as compared to its neighbors. Its activation reaches a peak (local maximum), creating a high contrast in the area and enhancing the sensory perception. In case of inter-channel LRN (normalization across separate channels but same spatial location),



Where

**i, j** = index of channel (for a layer having N channels, i varies from 0 to N - 1),

**aix,y** = value of activation in a neuron in i-th channel with position (x,y) before normalization

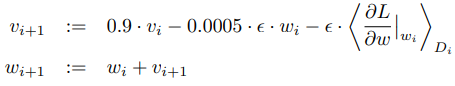
**bix,y** = value of activation in a neuron in i-th channel with position (x,y) after normalization

**k** = a hyperparameter used to avoid any singularity, i.e. division by zero (here, k = 2)

**∝** = normalization constant (here, ∝ = 10-4)

**ꞵ** = contrasting constant (here, ꞵ = 0.75)

**n** = neighborhood length (number of consecutive pixel values considered for normalization) (here, n = 5)

* **Overlapping Pooling:** Stride = 2, Kernel size = 3 x 3
* **Loss Function:** Cross-entropy loss function (multinomial logistic regression objective)
* **Regularization:** Dropout (dropout probability = 0.5)
  + To introduce model combination
  + To eliminate co-adaptation of neurons, thereby creating more robust features
  + Applied in first two fully-connected layers
  + Requires doubling the number of iterations for the model to converge
* **Stochastic Gradient descent details:** 
  + **Mini-batch size:** 128
  + **Momentum:** 0.9
  + **Initial learning rate:** 0.01 (divided by 10 if validation set accuracy stops improving)
  + **Decaying learning rate:** 0.0005
  + **Number of epochs:** 90
  + **Weight updation rule:** 

Where i = the iteration index,

v = the momentum variable,

ϵ = the learning rate,

 = the average over the i-th batch Di of the derivative of the objective with respect to w, evaluated at wi

* **Data Augmentation (to reduce overfitting):**
  + **Image translation & horizontal reflection:**
    - **During training** - cropped out random 227 x 227 sizes from each training image and used their horizontal reflection.
    - **During testing** - cropped out (4 corner patches + 1 central patch) = 5 patches of 227 x 227 size from each test image → took horizontal reflections of each of them, so 10 patches in total → made predictions on each of them, and finally took the average from those 10 predictions.
  + **Changing the pixel intensity by PCA:** For each training image,
    - The 3 x 3 covariance matrix of 3-dimensional RGB channels is calculated.
    - The eigenvectors & eigenvalues of the covariance matrix are calculated.
    - Let the eigenvectors be p1, p2, p3 & eigenvalues be λ1, λ2, λ3. Also, let ∝1, ∝2 & ∝3 be three variables randomly drawn from N(0,0.1). And, IR, IG, IB are pixel intensities of a point across RGB channels.
    - Then we use → [IR IG IB]T += [p1 p2 p3] [∝1λ1 ∝2λ2 ∝3λ3]T.