Deep Learning: Feature Detection

ENEE 4584/5584 CV Apps in DL

Dr. Alsamman

Slide Credits:

Handcrafted Feature Engineering

- Features are unique descriptors used to identify objects
- Traditional data science relied on manual feature engineering
- Use domain and business knowledge
  - Very tedious process
  - Subject to human bias
- Requires complex math to formulate
  - Requires relaxed assumptions and simplifications to conceive
- Requires complex programming to realize
  - Accuracy vs speed

Deep Learning Features

- DL unifies domain knowledge, formulation, and programming
  - focuses on DL
- Let the architecture learn from data presented
  - No need for handcrafting, approximations, assumptions
- Emphasis on data
  - Data must be accurate otherwise “garbage in garbage out”.
  - Data must be inclusive otherwise it results in learning bias
  - Data collection, accurate labeling is expensive

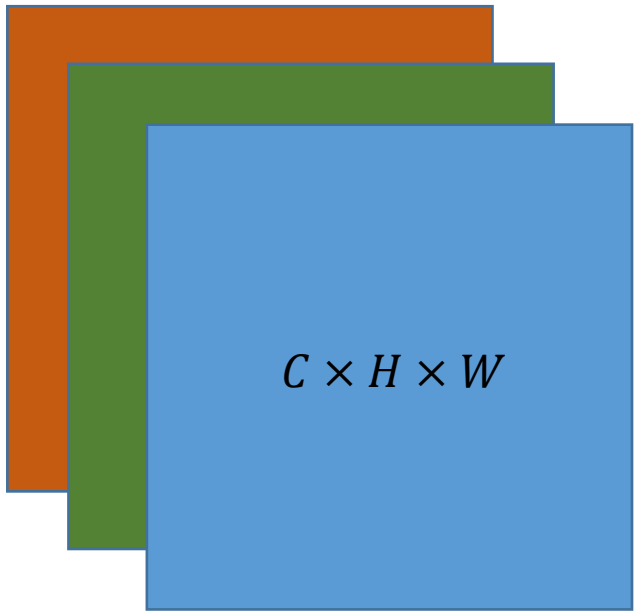
CNN Feature Detection

- Start with a database: desired feature points are labeled
- Convolution NN architecture:
  - Convolutional layer
  - Activation layer
  - Pooling layer
  - FC (dense layer)

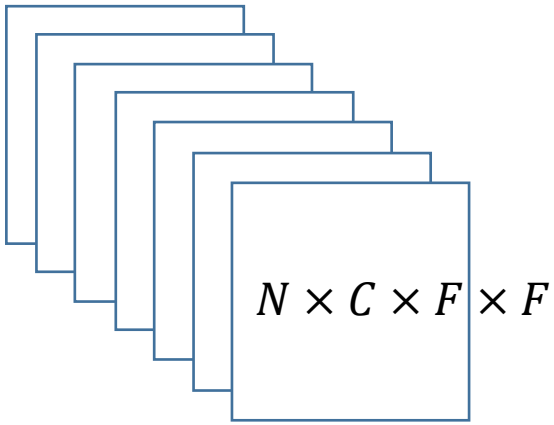
2D Convolutional Layers

- Odd filter sizes
  - Typically square
- Use many filters
  - Faster to use more filters as you go deeper
  - Traditional approach is to choose 2^n filters
- Channels (color)
  - Keep color channels if features are associated with them
  - Apply grayscale if features are structural not color related
- Stride (S) of convolution
  - Affects the convolution output size
  - Spatial size of output
  - Padding (P)

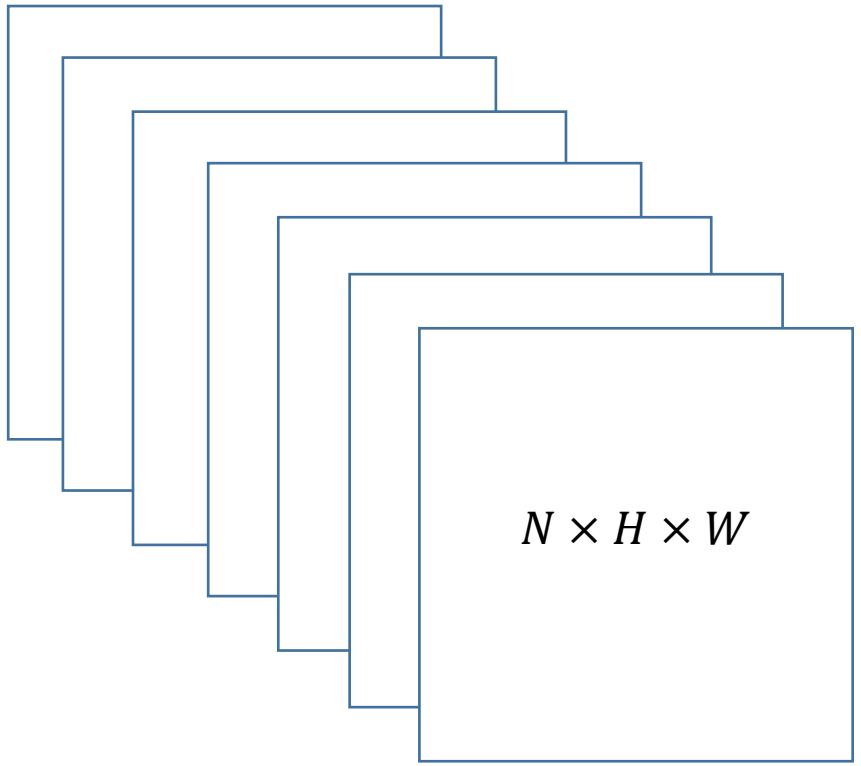
$$X \times Y = \left(\frac{H - F + 2P}{S} + 1\right) \times \left(\frac{W - F + 2P}{S} + 1\right)$$



input



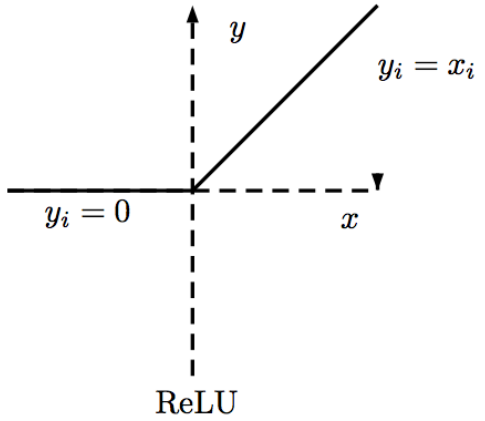
filters

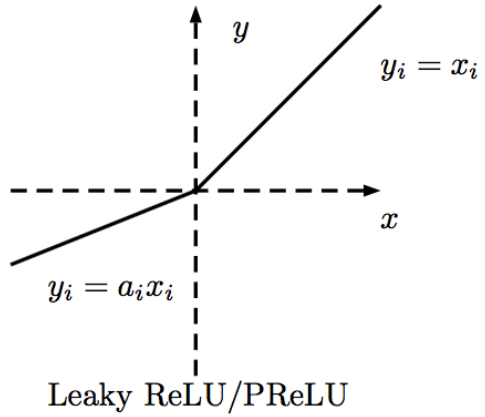


Convolution output
(padded, stride=1)

Activation

- Adds non-linearity
- Typically Rectified Linear Units (ReLU)
  - Fast gradient calculation
- Leaky ReLU
  - Not as fast
  - Non-flat negative domain
  - Must specify “leak” a_i





Pooling

- Reduce dimensionality
- Improves localization
- Max pooling is typical
- Pooling is even sized
- Stride (S) of pooling affects size of *receptive field* (output)

$$\left(\frac{X}{S}+1\right)\times\left(\frac{Y}{S}+1\right)$$

FCNN Layers

- A number of convolutional-activation-pooling layers are employed
  - Each reducing the receptive field
  - Experimentally determined
- A fully connected (dense) neural network is then added
  - All the receptive fields are flattened
    - converted into a single row vector
- Activation, layers, depth of layer is experimental
  - ReLU-like activations are preferred

FCNN Output

- Output size = number of features × size of feature
  - If features are locations in the image then size of feature = 2 (row and column)
- Output neurons must match feature type
  - If the feature location is desired then don't use softmax
- Error (learning criteria) must match output
  - Use MSE, MAE, when generating data (i.e not doing classification)

Additions to Boost Performance

- Dropout
  - Definitely apply to FCNN
  - Experiment with CNN layers
    - drop out as a percentage of weights in a filter, e.g. 1 weight in 3x3 = 1/9 dropout
- Data normalization
  - Problematic with RGB images
  - Idea: normalize grayscale and then apply to RGB
- Normalization as a function/layer
  - Batch normalization, layer normalization
  - “Whiten” output
  - Applied after convolution, or activation