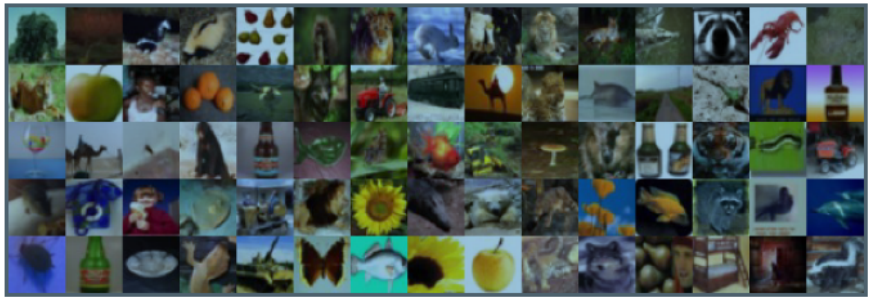
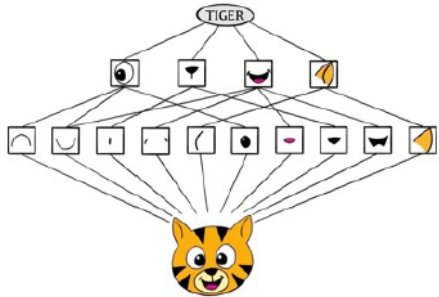
10.3 Convolutional Neural Networks

Convolutional Neural Networks (CNN)



- Major success story for image classification.
- Images above are from `CIFAR100` database
  - 32×32 color natural images, with 100 classes.
  - 50K training, 10k test
  - Each image is a 3D array or *feature* map, where the third dimension represents the RGB color channel.

CNN Concepts



- Network takes in an image and identifies local features.
- Features are then combined to create compound features, like eyes or ears.
- Compound features then used to output a label.

Convolution Layers

- A *convolution layer* is made up of a large number of *convolution filters*, each of which is a template for whether a particular feature is present in an image.
- A convolution filter relies on a simple operation, a *convolution*, which amounts to repeatedly multiplying matrix elements and then adding the results.

Convolution Filters

Consider a 4×3 image and a 2×2 filter.

$$\text{original image} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{bmatrix}$$

$$\text{convolution filter} = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}$$

When we *convolve* the image with the filter, we get

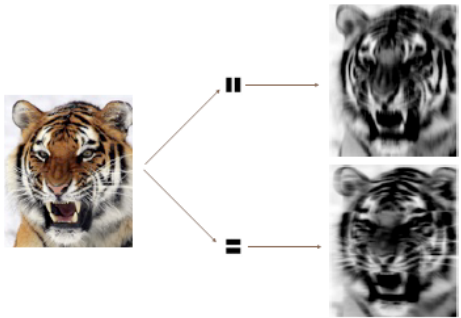
$$\text{convolved image} = \begin{bmatrix} a\alpha + b\beta + d\gamma + e\delta & b\alpha + c\beta + e\gamma + f\delta \\ d\alpha + e\beta + h\gamma + e\delta & e\alpha + f\beta + h\gamma + i\delta \\ g\alpha + h\beta + k\gamma + e\delta & h\alpha + i\beta + k\gamma + l\delta \end{bmatrix}$$

Convolution Filters

Idea:

- The filter is itself an image, represting some small feature.
- We slide it around the original image, scoring for matches.
  - Scoring done via dot-products.
- If the subimage of the input image is similar to the filter, the score is high, otherwise low.
  - Regions in the original image which resemble the convolution filter are highlighted.
- The actual filters are *learned* during training.

Convolution Example



- Each filter is a 15×15 image of mostly white with a black stripe.
- When the filter is convolved with the image, the stripes are given large values.
  - Horizontal stripe filter picks out horizontal stripes; vertical filter vertical stripes.
- The *weights* in the filters are the hidden layer weights.
  - Weights are reused for all possible patches in the image, so in the CNN case the weights are constrained.

Some Details

- Color images have three sets of 32×32 feature maps, one for each RGB color.
- A single convolution filter will also have three channels, one for each color, with potentially different filter weights.
- The results of the three convolutions are summed to form a 2D output feature map.
  - At this point, the color information has been used and is not passed to subsequent layers except through its role in the convolution.

More Details

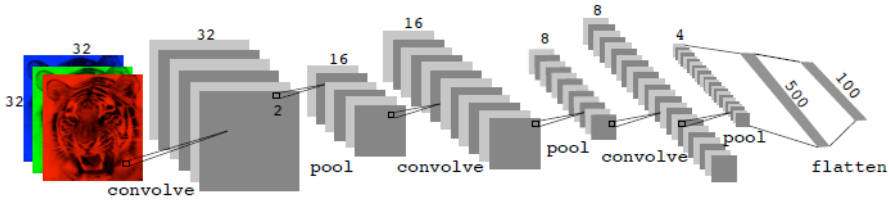
- If we use K different convolution filters at the first hidden layer, we get K 2D output feature maps.
- We view each of the K output feature maps as a separate channel of information, so we get K channels from the three original color channels.
- The 3D feature map is like the activations in a hidden layer of a simple neural network, just organized and produced in a spatially structured manner.
- We typically apply the ReLU activation function to the convolved image.
  - This is sometimes viewed as a separate layer in the CNN, referred to as a *detector layer*.

Pooling Layers

$$\text{Max pool} \begin{bmatrix} 1 & 2 & 5 & 3 \\ 3 & 0 & 1 & 2 \\ 2 & 1 & 3 & 4 \\ 1 & 1 & 2 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 3 & 5 \\ 2 & 4 \end{bmatrix}$$

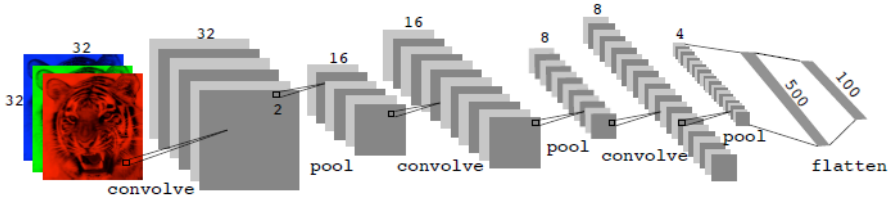
- A *pooling* layer provides a way to condense a large image into a smaller summary image.
- This is often done using a *max pooling* operation.
  - Summarizes each non-overlapping 2×2 block of pixels using the maximum value in the block.
  - This reduces the size of the image by a factor of 4 (2 in each direction)
  - Provides some *location invariance* $\rightarrow$ as long as there is a large value in one of the four pixels, the whole block registers as a large value in the reduced image.
- This basically just makes the image blurry while maintaining key features.

Architecture of a CNN



- Many convolution and pooling layers.
- Filters typically small, e.g., 3×3
- Each new filter creates a new channel in the convolution layer.
- As pooling reduces size, the numbers of filters/channels is typically increased.
- Number of layers may be very large (~50)

Architecture of a CNN



- Each subsequent convolve layer is similar to the first.
  - Takes as input the 3D feature map from the prior layer and treats it like a single multi-channel image.
- Since the channel feature maps are reduced in size after each pool layer, we increase the number of filters in the next layer to compensate.
- Sometimes we repeat several convolve layers before a pool later.
  - This effectively increases the dimension of the filter.

Architecture of a CNN

- The process is repeated until each feature map is just a few pixels in each dimension.
- At this point the 3D maps are *flattened* (the pixels are treated as separate units) before eventually being fed into the output later.
  - Typically a *softmax activation* for the image classes.

Architecture of a CNN

These can be super complex!

- Need to select the number, nature, and size of each layer.
- There are a lot of other tuning parameters to select, too.
- Can use dropout learning, lasso or ridge regularization, etc...
- Lot of software options for constructing these.
- Lots of ongoing work in this field, so things should continue to improve.
  - As of 2023, the `CIFAR100` official test set's best accuracy was 75%.

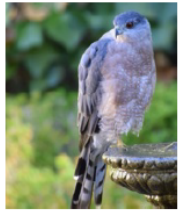
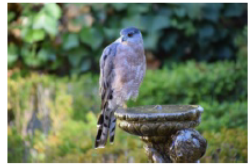
Data Augmentation

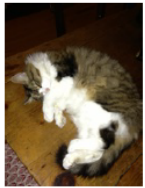
- Each training image is replicated many times, with each replicate randomly distorted.
  - This is done in such a way that human recognition is unaffected.
  - Typical distortions are zoom, horizontal and vertical shift, shear, small rotations, horizontal flips, etc.
- This increases the training set with somewhat different images, which helps protect against overfitting.
- This is also a form of regularization: building a cloud of images around each original image, all with the same label.

Results Using Pretrained Classifier

- Here we use an industry-level pretrained classifier to predict the class of some new images.
- You will use this `resnet50` classifier to predict the class of your own images in Project 3.

Results Using Pretrained Classifier




flamingo		Cooper's hawk		Cooper's hawk	
flamingo	0.83	kite	0.60	fountain	0.35
spoonbill	0.17	great grey owl	0.09	nail	0.12
white stork	0.00	robin	0.06	hook	0.07
Lhasa Apso		cat		Cape weaver	
Tibetan terrier	0.56	Old English sheepdog	0.82	jacamar	0.28
Lhasa	0.32	Shih-Tzu	0.04	macaw	0.12
cocker spaniel	0.03	Persian cat	0.04	robin	0.12