Consolidated CV notes:

Classification

CNN (plankton)

**class imbalance? Prob. Because then its biased, has poor perf on smaller classes -> transfer learning** (pre-train with class normalized data [reduce big classes via random sampling, using a threshhold. Too big -> no bias reduction, too small -> lose specificity], then fine-tune with orig. i.e. train again on whole dataset) -> better acc than w/o or using other data augmentation techs. **Future? Sequential transfer learning w/ multi thresholds (learn to classify big datasets first, then smaller later)**

CNN (deepmole)

**expensive, time-consuming to feature extract/use expert features, also limited health data. Rather, use CNN to feature extract then classify by pre-training on normal data.** Use activations from earlier layers to make complete classifier because health data so different. Build custom classifiers for the last 3 (FC) layers. Then data augment w/ distortions. Custom classifiers use KNN w/ cosine dist because “more robust to outliers, used in other works”. **Future? Combine CNN w/ handcrafted features (??)**

CNN (Maxmin)

**using ReLu gets rid of negatives (i.e. negative info ignored, whether strong/weak) - > preserve this using 2 reLus to keep high + and - vals**

using CNNs for low-res and in weak supervision (used these days)

ReLu = standard activation func for classification, then max/avg pooling (which seem best). Relu zeroes all negative vals , done to “facilitate the exploitation of discriminative information by de-noising filter detections”. Can do PreLu (**check out**).

PCA (PCAnet)

**Emulate CNN’s processing layers thru PCA => easy to use, train and adapt. Also gives theoretical basis for deep stuff**

using CNNs for classification – depend on fine-tuning params. Not easy to generalize.

PCAnet = simple unsupervised CNN. 1. Cascading PCA on input. 2. Binary Hashing. 3. Block histograms. **Future? More sophisticated filters/stages so can handle extreme variability data sets (e.g. PASCAL VOC)**

CNNs

LBCNNs (Local Binary CNNs)

**Efficient approximation of a convolutional layer -> big computational savings, less prone to over-fitting**

training convolutional kernels end to end = expensive, large and over-fits limited data (cos so many params). Use binary weights instead of floats to save space (because can do binary convolutions (?)). but if complete binarization -> worse perf . Rather than learning to optimize conv filters, learn to optimize linear weights.

RCNN (Text Class)

**recurrent CNN for text classification without human-designed features**

n-gram (= no. words. 5 words = 5gram)

Bag of Words (BoW) models ignore sequence/context. Recursive Nns use tree (0n\*2) which is untractable. RNNs 0(n) but biased in favor of later words. -> bi-direction (Left and Right) recurrent CNN (get ‘left context’ and ‘right context’ in conv layer then max pool)

Recognition/Detection

Emotion Recognition using RCNN (Emote)

**Use CNN as input to an RNN. can’t use cnn alone cos then no temporal info**

use cnn’s regression layer for each frame as input to RNN

Deep Residual Learning

**more accurate = stack more layers? No. -> accuracy stagnation and degradation. NOT overfitting but higher training error. Raher – use identity mapping to overcome this and have super deep networks**

rather than just stacking layers to hope they fit a desired underlying mapping, we explicitly let these layers fit a residual mapping. Do this using shortcut connections (ones that skip >=1 layers) -> no extra params and can have kaaaaaaak deep networks. **Prob: summing identity func and output of layer might cause info loss**

Densely Connected (DCNN)s

**Cnns better if connections close to input and close to output shorter. Instead of L layers having L connections, make all prev layers have connections to all subsequent layers and vice versa -> fewer params (don’t need to learn redundant feature-maps). Dense ones also deep supervised and less prone to overfitting cos self-regulate. Don’t go deeper/wider, do feature re-use.**

instead of creating a short path from early layers to late ones (a la resnet), connect *all* layers w/ matching feature map sizes directly [concatenate inputs]. Also prob with resnet is klomp params (each layer got own weights) but many of the layers do fokol.

Residual Attention Network (ran for img classification)

**CNNs + attention-aware features (thru *stacked* attention modules) -> more discriminative feature reps & higher obj classification accuracy and able to be super deep (also bc many attention modules -> can captture diff types of attention**