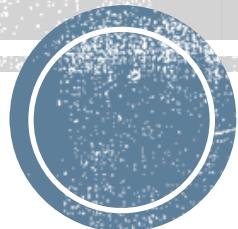


Applied Data science

Spring 2017_Project 3_Group 14

- Han, Ke (kh2793)
- Li, Mengchen (ml3890)
- Mison, Virgile (vcm2114)
- Pan, Yijia (yp2424)
- Xiang, Yi (yx2365)



Models Comparison

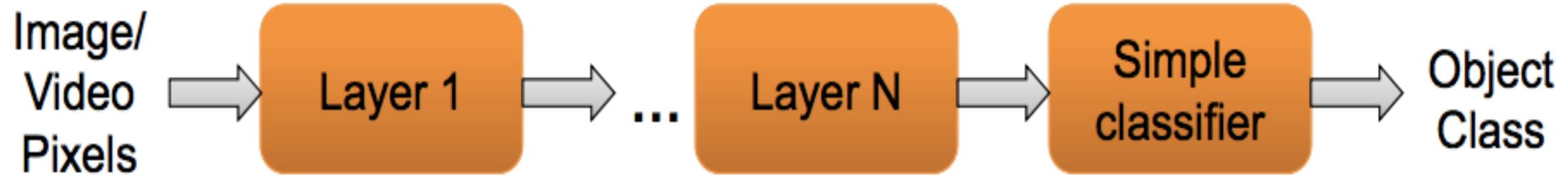
feature	model	parameters	accuracy
sift	GBM	ntree = 100, depth =1, shrinkage = 0.1 (CV)	0.73
sift (LASSO)	GBM	ntree = 100, depth =1, shrinkage = 0.1 (CV)	0.64
sift (LASSO)	KNN	k = 3	0.65
sift (LASSO)	XG boost	objective = 'logistic', max_depth = 7, eta = 0.11, gamma = 0.01	0.68
sift (LASSO)	random forest	ntree = 600 (CV)	0.67
sift (LASSO) + texture	majority vote (XG boost)	...	0.69
DNN	DNN	num.round = 750, learning.rate = 0.3, dropout = 0.7	0.792

Training: 80% vs. Test: 79.24% improved nearly 7%



Deep Learning (DNN)

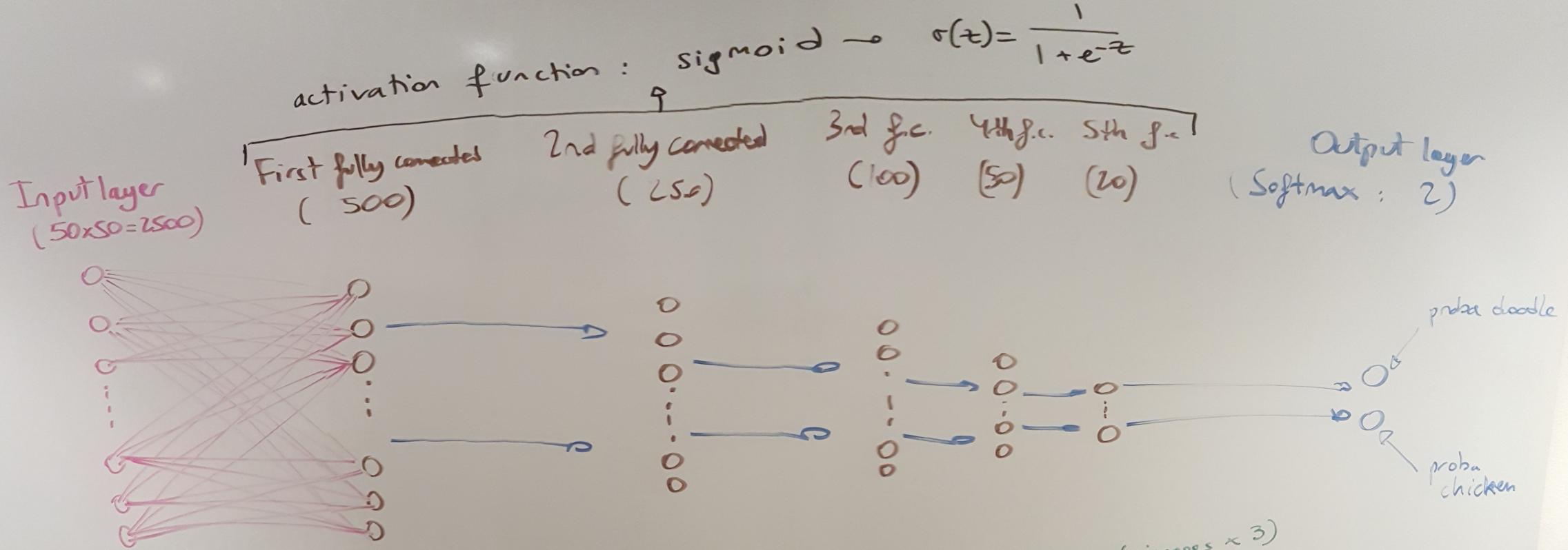
Deep learning: “Deep” architecture



Problems + Strategies

- 2000 images -> 1600 train + 400 test
- Deep Learning (mxnet)
- Results: train acc=95% vs. test acc=75%
- Problem: Overfitting !!!
- Reason: Deep networks need to be trained on a huge number of training images to achieve satisfactory performance
- How to solve? Ans: data augmentation (add to the general training dataset images that have been flipped horizontally and also with one rotation of small angles)
- Strategies: 2000 images -> 4000 + 2000 images, improved!





Most important points ::

- Total numbers of parameters::
matrices { $2500 \times 500 + 500 \times 250 + 250 \times 100$
+ $100 \times 50 + 50 \times 20 + 20 \times 2$
bias { $1 + 500 + 50 + 100 + 50 + 20$

$\approx 1.4 \text{ M}$ parameters,

- Training accuracy $\approx 80\%$.
- Test accuracy $\approx 79.24\%$.

\rightarrow time to train ≈ 20 minutes
on CPU of
MacBook Pro 2013

① Data augmentation to avoid overfitting (images $\times 3$)

② Dropout to avoid massively overfitting (set to $p=0.9$)

③ Larger learning rate to avoid getting stuck in a "hole" ∇J

④ Larger weights initialization matrix W : $W_{ij} \sim N(0, 0.6)$

↳ original initialization diverges for $W_{ij} \approx 0$

⑤ Use of sigmoid activation function

↳ tanh and relu all diverge