Applied Data science

Spring 2017_Project 3_Group 14

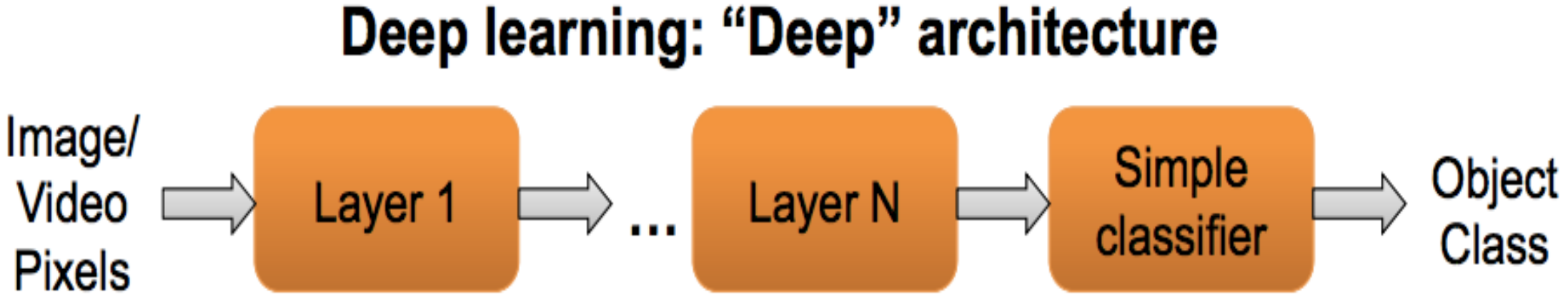
- Han, Ke (kh2793)
- Li, Mengchen (ml3890)
- Mison, Virgile (vcm2114)
- Pan, Yijia (yp2424)
- Xiang, Yi (yx2365)

Models Comparison

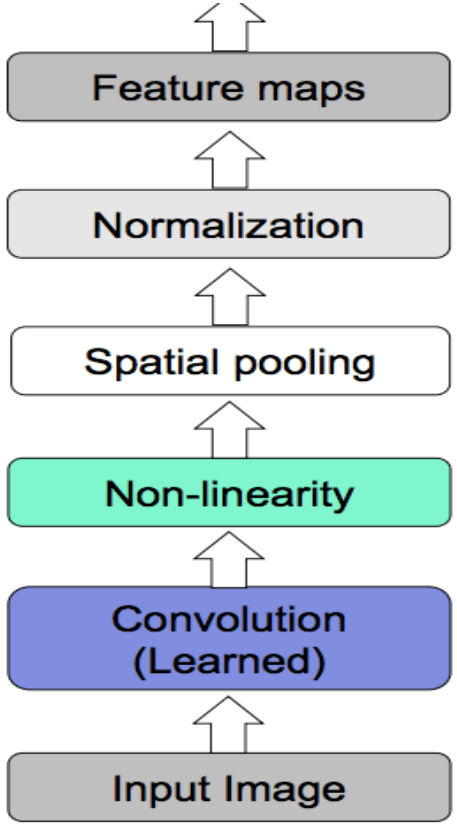
feature	model	parameters	accuracy	Time
sift	**GBM**	**ntree = 100, depth =1, shrinkage = 0.1 (CV)**	**0.73**	**21.5s**
sift (LASSO)	GBM	ntree = 100, depth =1, shrinkage = 0.1 (CV)	0.64	20s
sift (LASSO)	KNN	k = 3	0.65	10min
sift (LASSO)	XG boost	objective = 'logistic', max_depth = 7, eta = 0.11, gamma = 0.01	0.68	40s
sift (LASSO)	random forest	ntree = 600 (CV)	0.67	5min
sift (LASSO) + texture	majority vote (XG boost)	...	0.69	18min
DNN	**DNN**	**num.round = 750, learning.rate = 0.3, dropout = 0.7**	**0.81**	**15min (features) + 5min(model)**

Training: 80% vs. Test: 81.24% improved nearly 9%

Deep Learning (DNN)



Feature Extractions



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!

activation function : sigmoid → $\sigma(t) = \frac{1}{1+e^{-t}}$

Input layer (50×50 = 2500)

First fully connected (500)

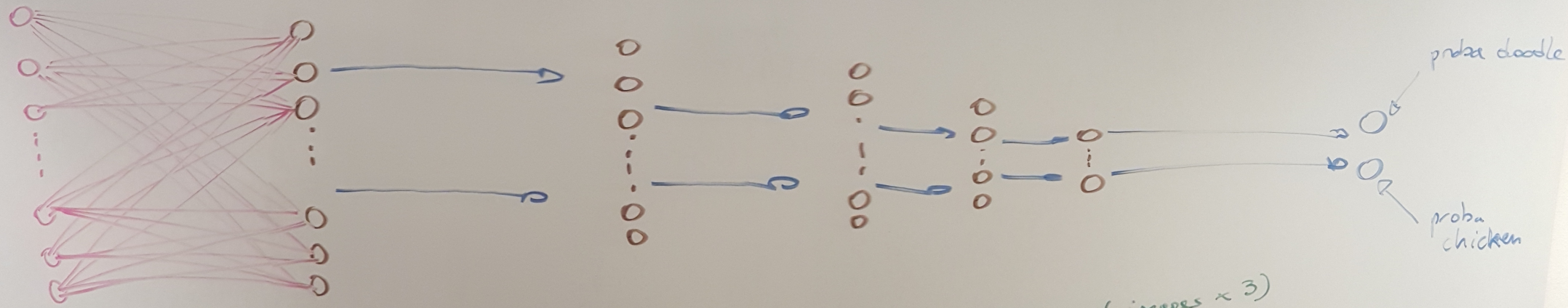
2nd fully connected (250)

3rd f.c. (100)

4th f.c. (50)

5th f.c. (20)

Output layer (Softmax : 2)



Most important points::

- Total number of parameters::

matrices { 2500×500 + 500×250 + 250×100
+ 100×50 + 50×20 + 20×2
bias { + 500 + 250 + 100 + 50 + 20

≃ 1.4 M parameters

- Training accuracy ≃ 80 %.
- Test accuracy ≃ 79.24 %.

→ time to train ≃ 20 minutes on CPU of MacBook Pro 2013

① Data augmentation to avoid overfitting (images × 3)

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

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

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

Final DNN model

```
mx.set.seed(2)
model <- mx.mlp(train.x, train.y, hidden_node=c(500,250,100,50,20), out_node=2, activation =
"sigmoid", out_activation="softmax",
                num.round=200, array.batch.size=50, learning.rate=0.08,
                eval.metric=mx.metric.accuracy, dropout = 0.9, momentum = 0.7,
initializer=mx.init.normal(0.6))

accuracy_model(model, test.x,test.y) # test accuracy 81.24%
```

```
[200] Train-accuracy=0.834583333333334
Auto detect layout of input matrix, use rowmajor..
          test.y
pred.label   0   1
         0 445 109
         1 128 518
[1] "Accuracy of model: 80.25%"
```