

14. Classification



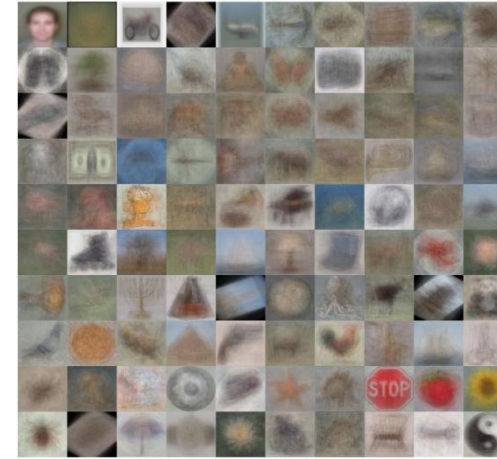
Classification

- Classification allows us understand, predict, and interact with the surrounding environment
 - Is it dangerous? How fast does it run? Can I poke with it?

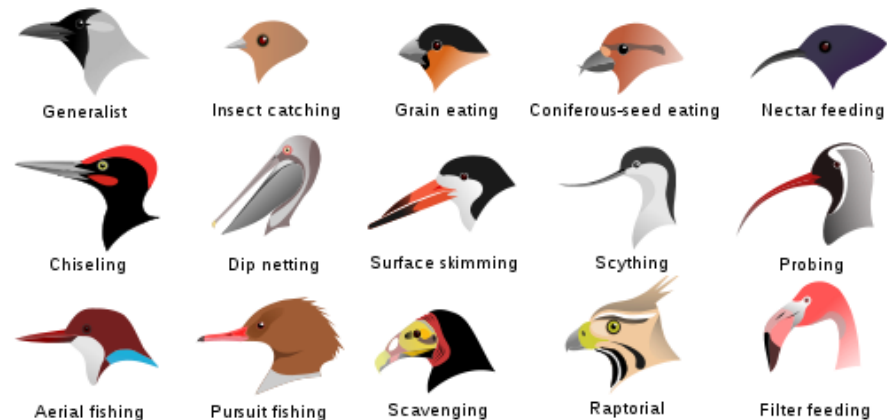


Sample Classification Problems

- Object recognition
- Place recognition
- Fine-grained recognition



Caltech 101 Average Object Images

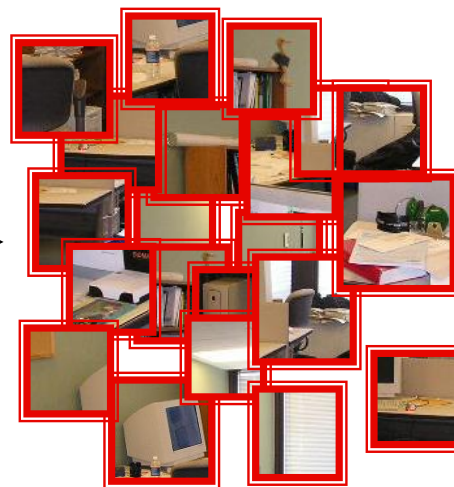


Places Database [[Zhou et al. NIPS 2014](#)]

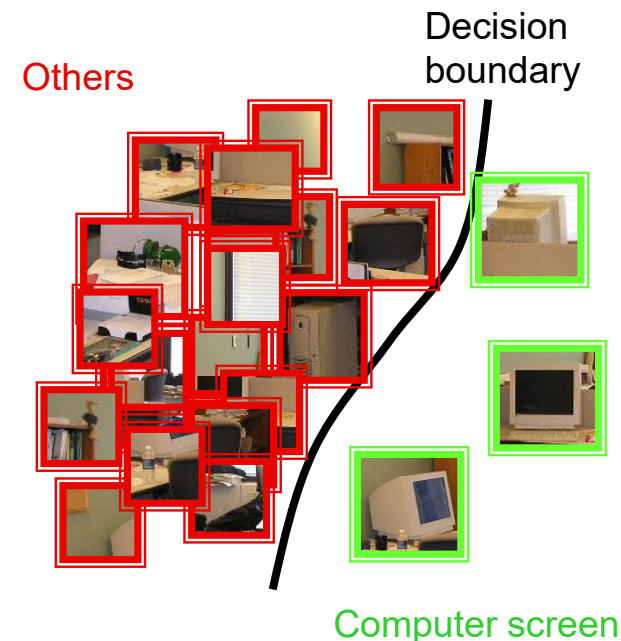
Object Detection by Classification

Object detection can be formulated as a classification problem.
The image is partitioned into a set of overlapping windows
... and a decision is made at each window about if it contains a target object or not.

Where are the screens?



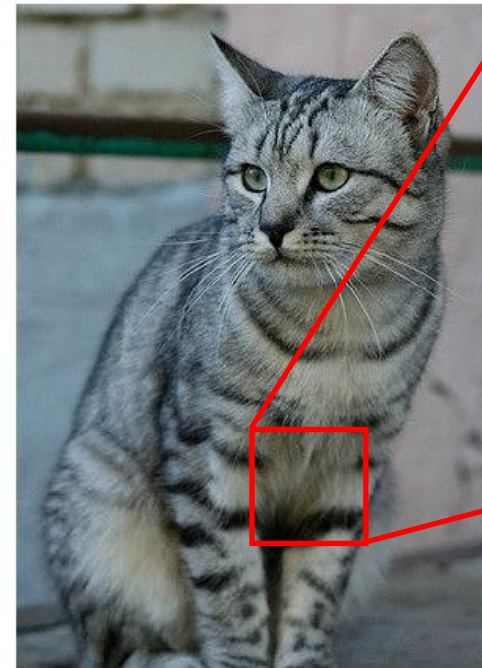
overlapping image patches



In some feature space

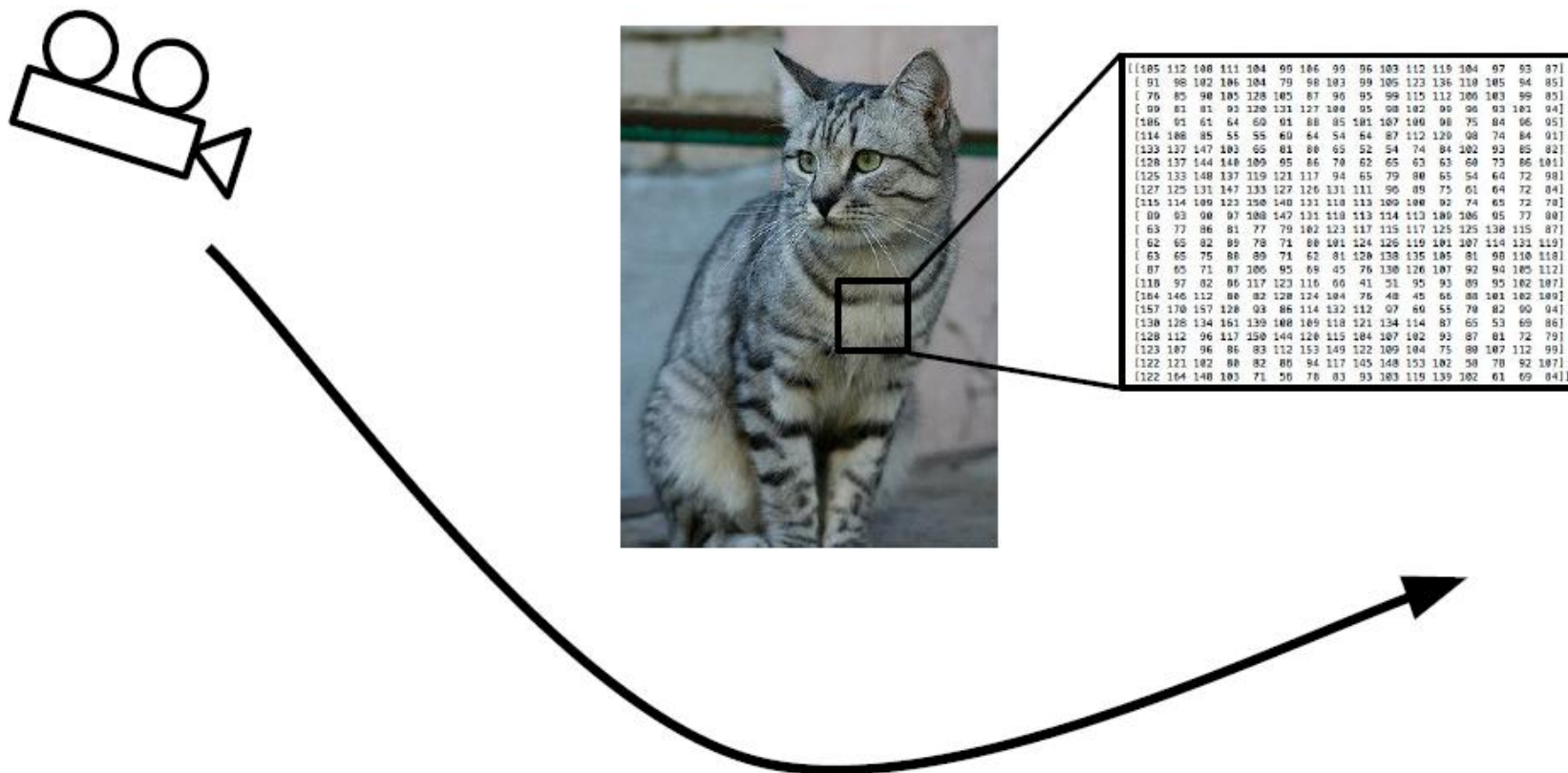
The Problem: Semantic Gap

- Images are represented by RGB values
- The computer does not 'see' the semantics of the image!
 - No easy way to write a program to recognize a cat
 - Unlike sorting a set of numbers



[105	112	108	111	104	99	106	99	96	103	112	119	104	97	93	87]
[91	98	102	106	104	79	98	103	99	105	123	136	110	105	94	85]
[76	85	90	105	120	105	87	96	95	99	115	112	106	103	99	85]
[99	81	81	93	120	131	127	100	95	98	102	99	96	93	101	94]
[106	91	61	64	69	91	88	85	101	107	109	98	75	84	96	95]
[114	108	85	55	55	69	64	54	64	87	112	129	98	74	84	91]
[133	137	147	103	65	81	88	65	52	54	74	84	102	93	85	82]
[128	137	144	140	109	95	86	70	62	65	63	63	60	73	86	101]
[125	133	148	137	119	121	117	94	65	79	80	65	54	64	72	90]
[127	125	131	147	133	127	126	131	111	96	89	75	61	64	72	84]
[115	114	109	123	150	148	131	118	113	109	100	92	74	65	72	78]
[89	93	90	97	108	147	131	118	113	114	113	109	106	95	77	80]
[63	77	86	81	77	79	102	123	117	115	117	125	125	130	115	87]
[62	65	82	89	78	71	80	101	124	126	119	101	107	114	131	119]
[63	65	75	88	89	71	62	81	120	130	135	105	81	98	110	110]
[87	65	71	87	106	95	69	45	76	130	126	107	92	94	105	112]
[118	97	82	86	117	123	116	66	41	51	95	93	89	95	102	107]
[164	146	112	80	82	120	124	104	76	48	45	66	88	101	102	100]
[157	170	157	120	93	86	114	132	112	97	69	55	70	82	99	94]
[130	128	134	161	139	100	109	118	121	134	114	87	65	53	69	86]
[128	112	96	117	150	144	120	115	104	107	102	93	87	81	72	79]
[123	107	96	86	83	112	153	149	122	109	104	75	80	107	112	99]
[122	121	102	80	82	86	94	117	145	148	153	102	58	78	92	107]
[122	164	148	103	71	56	78	83	93	103	119	139	102	61	69	84]

Challenges: Viewpoint variation

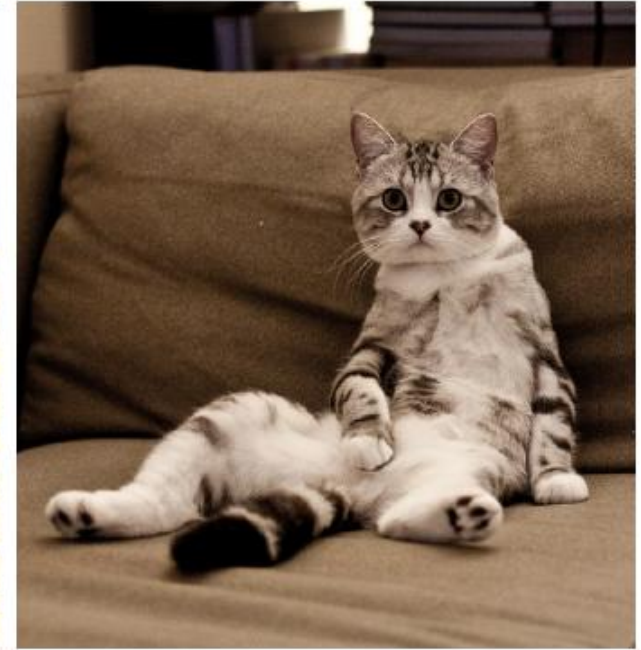


All pixels change when the camera moves!

Challenges: Illumination



Challenges: Deformation

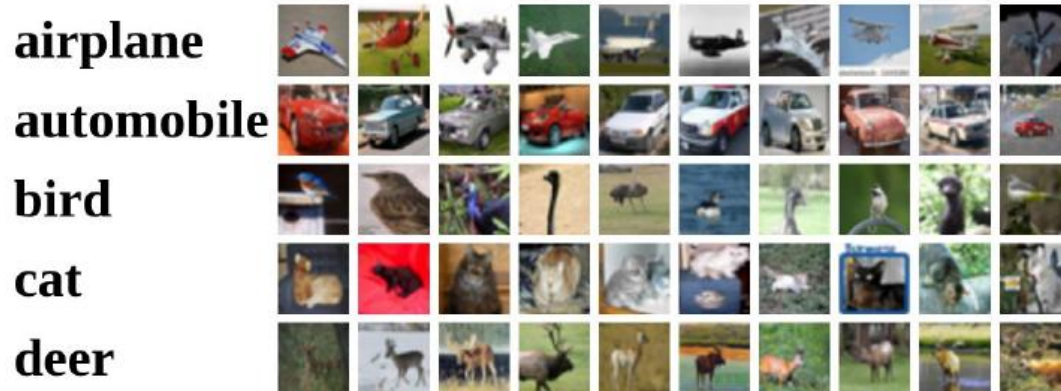


Challenges: Occlusion



Learned Classifier

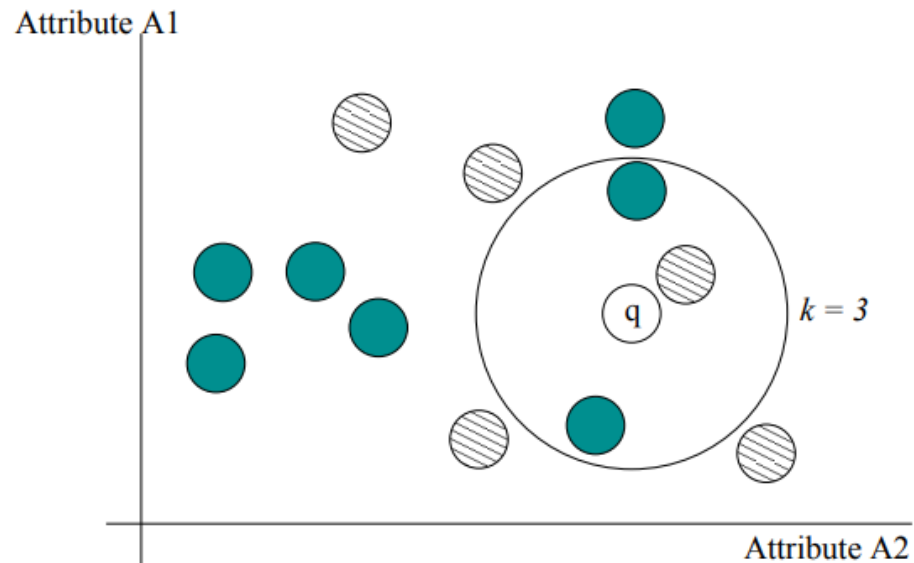
- Given training images with labels
- Learn a “classifier”
- Evaluate new images with the learned classifier



example of training images

Example: Nearest Neighbor Classifier

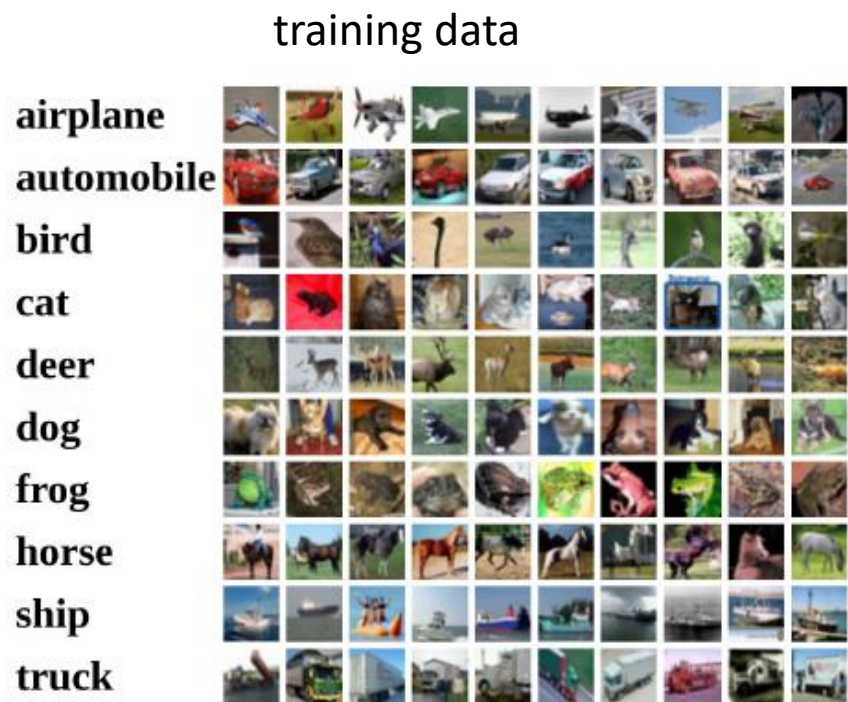
- Given training images with labels
- Learn a “classifier”: memorize all data (images + labels)
- Evaluate new images: find the most similar training image



Might be generalized to k -Nearest Neighbors:
Finding the k most similar training images, and let them vote

Example: KNN with CIFAR10

- 10 classes
- 50,000 training images
- 10,000 testing images



testing data with nearest neighbors



Distance between Images?

- Take each image as a matrix, and compute matrix distances (this would never work)
 - L1 distance: $d_1(I_1, I_2) = \sum_x |I_1(x) - I_2(x)|$
 - L2 distance: $d_2(I_1, I_2) = \sqrt{\sum_x |I_1(x) - I_2(x)|^2}$
- Better to compute a 'feature vector' for each image
 - And compute distance between 'feature vectors', e.g. L_1, L_2 , etc.

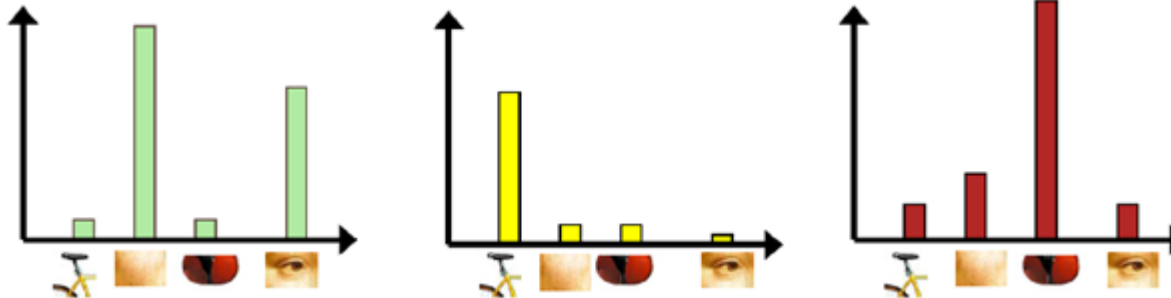


Bag-of-Words Features

images



feature vectors:
the frequency
of visual words



'visual words'



Analogy to documents features

Of all the sensory impressions proceeding to the brain, the visual experiences are the dominant ones. Our perception of the world around us is based essentially on the messages that reach the brain from the eyes. For a long time it was thought that the visual information was transmitted to the brain; the screen, so that the eye was the origin of the message. There is a course of events: impulses along the layers of the optic nerve have been able to determine the message about the image falling on the retina. The message undergoes a step-wise analysis in a system of nerve cells stored in columns. In this system each cell has its specific function and is responsible for a specific detail in the pattern of the retinal image.

sensory, brain, visual, perception, retinal, cerebral cortex, eye, cell, optical nerve, image Hubel, Wiesel

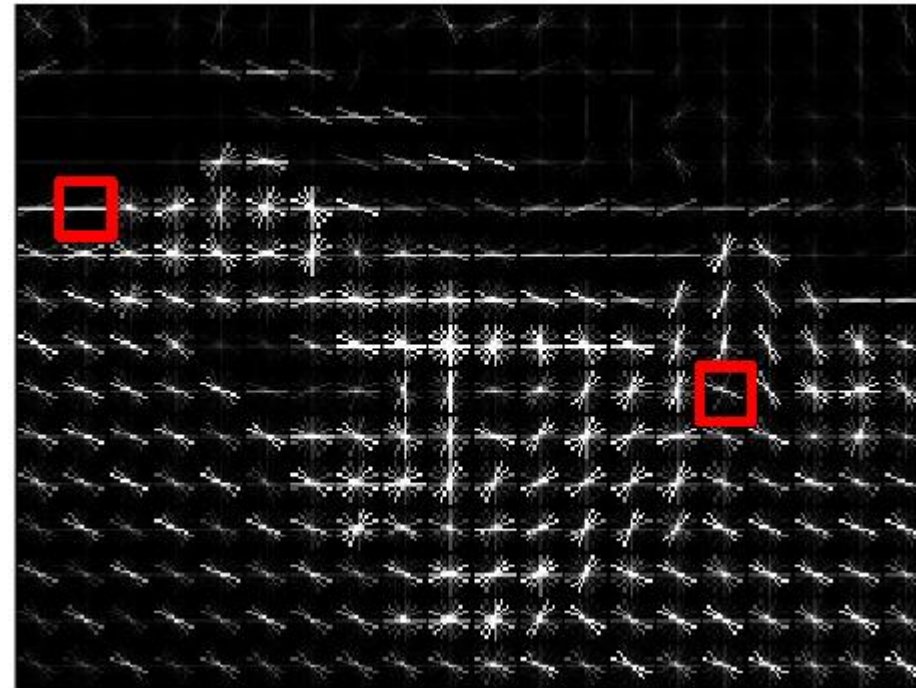
China is forecasting a trade surplus of \$90bn (£51bn) to \$100bn this year, a threefold increase on 2004's \$32bn. The Commerce Ministry said the surplus could be created by a predicted 3.5% increase in exports to \$750bn, compared with \$660bn in 2004. The US trade deficit with China was \$200bn in 2004, but the yuan is to be allowed to trade freely. However, Beijing has made it clear that it will take time and tread carefully before allowing the yuan to rise further in value.

China, trade, surplus, commerce, exports, imports, US, yuan, bank, domestic, foreign, increase, trade, value

HOG Features



Divide image into 8x8 pixel regions
Within each region quantize edge
direction into 9 bins



Example: 320x240 image gets divided
into 40x30 bins; in each bin there are
9 numbers so feature vector has
 $30 \times 40 \times 9 = 10,800$ numbers

Recap: Classification

- Given training images with labels
- Compute a feature vector (e.g. BoW, HOG, etc) for each image
- Learn a classifier (e.g. K-NN)
- Apply the classifier to unseen testing images

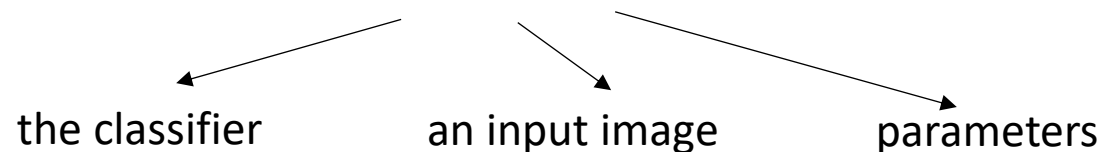
Questions?



Parametric Classifiers

- A function (with parameters) to predict the category label

$$\hat{y}_i = f(x_i, W)$$



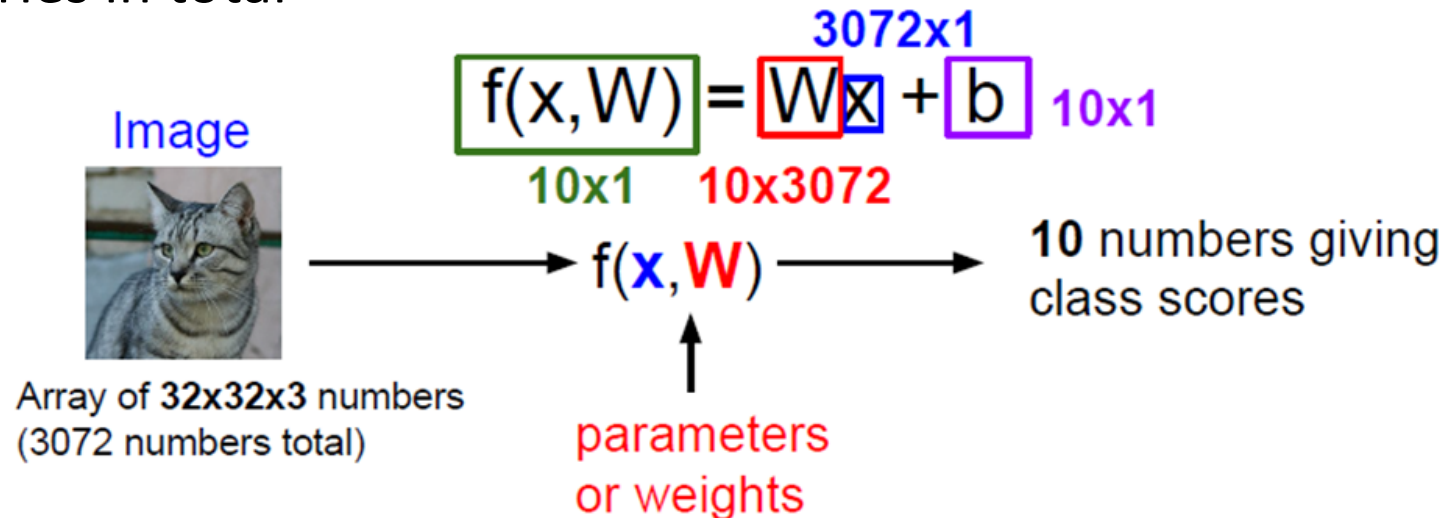
- Training:
 - Given a set of training data $\{(x_i, y_i)\}$
 - Estimate parameters W , such that $\hat{y}_i = y_i$ on the training data
- There are different choices for **function f** , **parameter estimation method**, etc.

Linear Classifier

- A simple example is linear classifier, where $f(\cdot, \cdot)$ is a linear function
- For example:

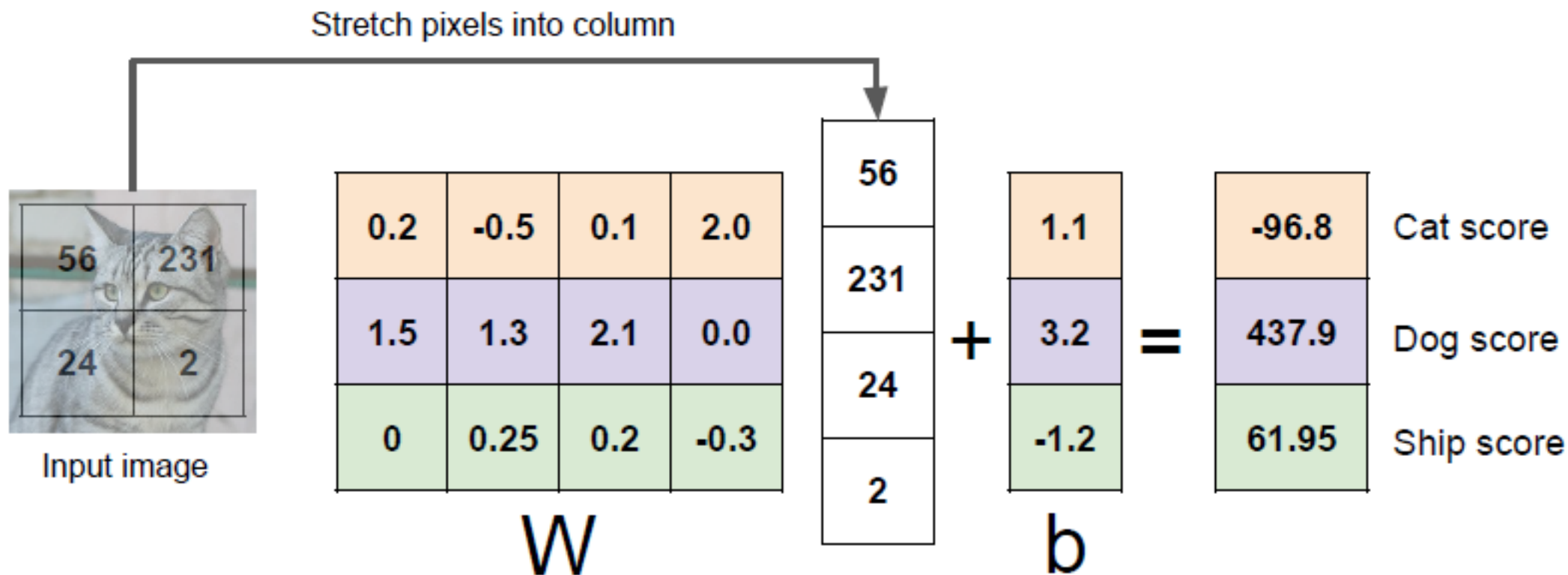
$$s = f(x, W) = Wx + b$$

- Input: x is a vector formed by concatenating all pixel values
- Output: s is a vector giving class scores, e.g., 10×1 vector if there are 10 categories in total



Linear Classifier: An Example

Example with an image with 4 pixels, and 3 classes (cat/dog/ship)



Parameter Estimation

- To estimate parameters W from training data $\{(x_i, y_i)\}$
- Need to measure the consistency between prediction and label
 - by a loss function $L_i(y_i, s_i)$ – larger when y_i is inconsistent with $s_i = f(x_i, W)$

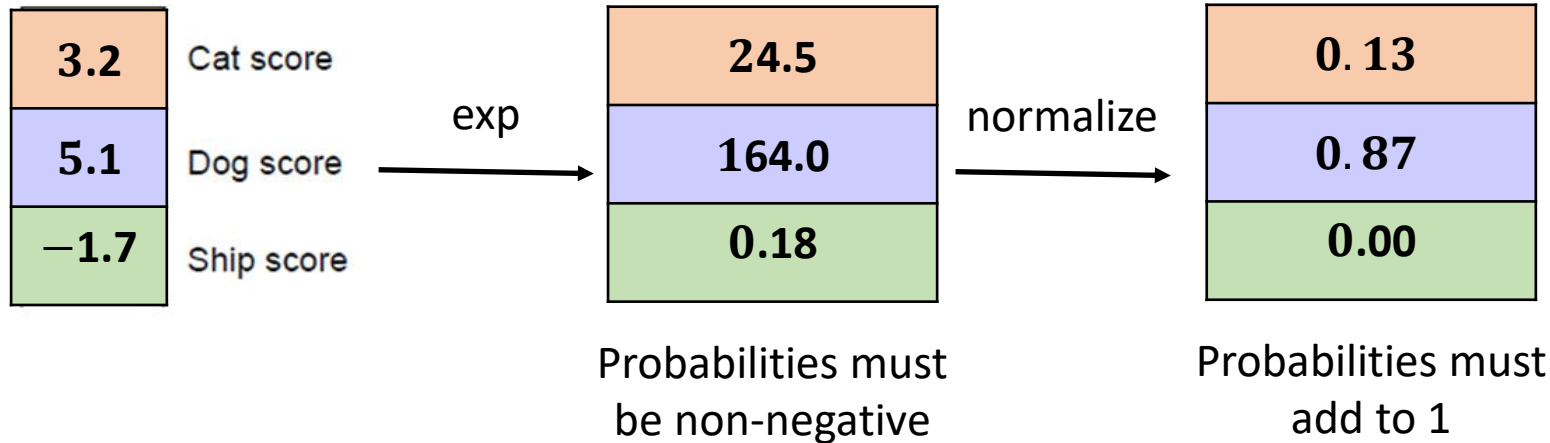
Note s_i is a vector contain scores for all categories

- The parameters are estimated by minimizing the loss

$$W^* = \arg \min_W \sum_i L_i(y_i, s_i)$$

SoftMax Loss

- We can interpret the category score s_i as probabilities
- That requires some manipulations of the original scores
 - $P(Y = k|X = x_i) = \frac{e^{s_i(k)}}{\sum_j e^{s_i(j)}}$ (called softmax function)



SoftMax Loss



1
0
0

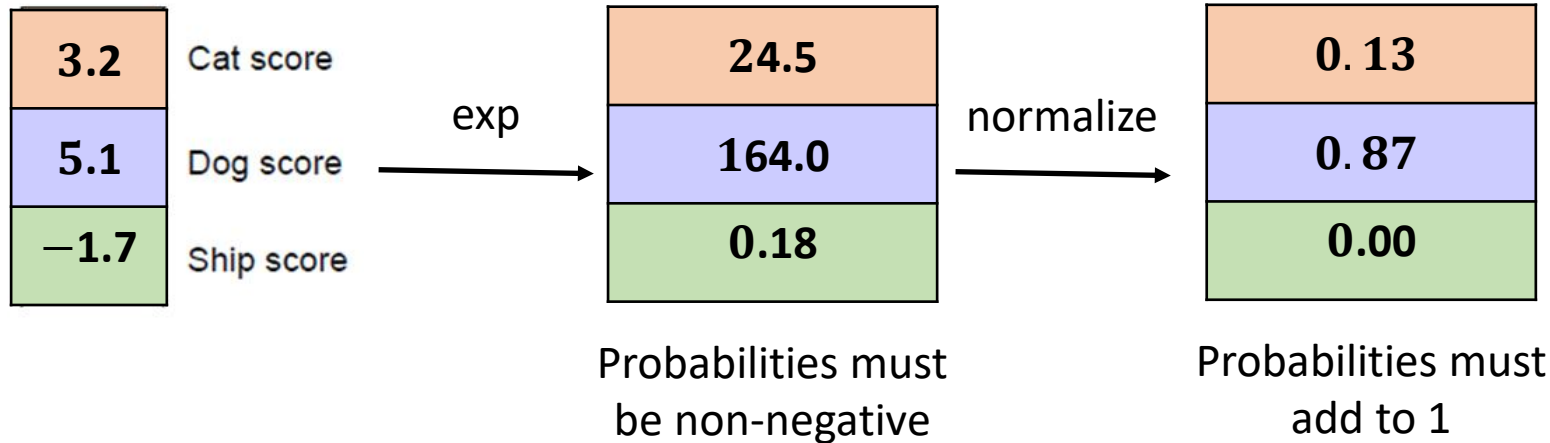
- Evaluates how well the current s_i is

$$L_i(y_i, s_i) = -\log s_i(y_i)$$

the probability vector

the ground truth
class label

- It maximizes the likelihood of the truth class



$$\begin{aligned} L_i &= -\log P(Y = y_i | X = x_i) \\ &= -\log 0.13 = 0.89 \end{aligned}$$

Maximum Likelihood Estimation
Maximizing the likelihood of the correct label.

SoftMax Loss

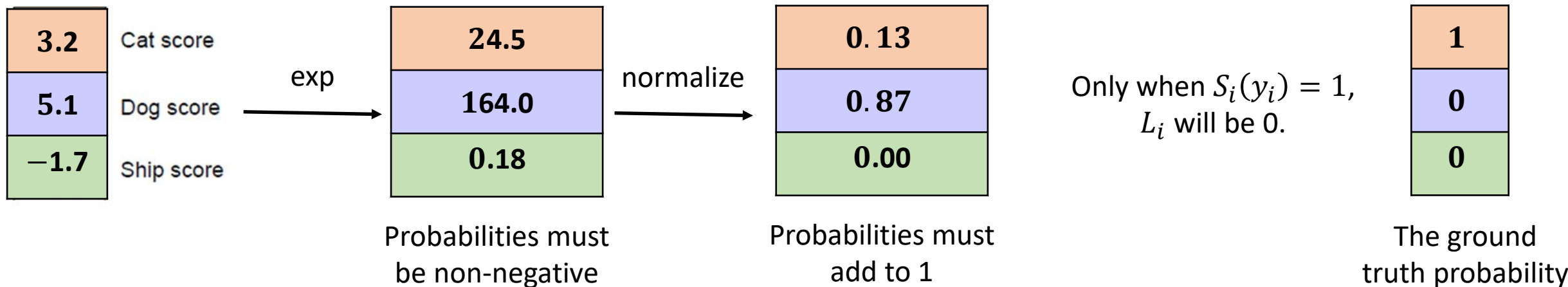
- Evaluates how well the current s_i is

$$L_i(y_i, s_i) = -\log s_i(y_i)$$

the probability vector

the ground truth
class label

- It maximizes the likelihood of the truth class



SVM Loss

- Instead of normalizing the original scores to probabilities
- It requires the score of the true category to be the largest
- So a loss function can be defined as (for a cat training image):
 $\max(0, \text{Dog Score} + 1 - \text{Cat Score}) + \max(0, \text{Ship Score} + 1 - \text{Cat Score})$

make sure the cat score is larger for a sufficient margin

- Formally, the loss is:

$$L_i(y_i, s_i) = \sum_{j \neq y_i} \max(0, s_i(j) + 1 - s_i(y_i))$$

3.2	Cat score
5.1	Dog score
-1.7	Ship score

$$\begin{aligned}
 & \max(0, 5.1 + 1 - 3.2) + \max(0, -1.7 + 1 - 3.2) \\
 &= \max(0, 2.9) + \max(0, -3.9) \\
 &= 2.9 + 0 = 2.9
 \end{aligned}$$

Sum Over All Samples

- The final loss is summed over all training data

$$\sum_i L_i(y_i, s_i)$$

- An example: suppose we have the following class scores $s_i = Wx_i$



Evaluate the loss (say the SVM loss) for each sample

$$L_i = \sum_{j \neq y_i} \max(0, s_i(j) + 1 - s_i(y_i))$$

And sum over all samples

cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1

$$\text{Cat: } \max(0, 5.1 + 1 - 3.2) + \max(0, -1.7 + 1 - 3.2) = 2.9$$

$$\text{Car: } \max(0, 1.3 + 1 - 4.9) + \max(0, 2.0 + 1 - 4.9) = 0$$

$$\text{Frog: } \max(0, 2.2 + 1 - (-3.1)) + \max(0, 2.5 + 1 - (-3.1)) = 12.9$$

$$\text{Total Loss} = 2.9 + 0 + 12.9 = 15.8$$

Regularizer

- Suppose a coefficient W makes all loss to be 0
- Is W unique? Usually no!
 - For example, the SVM loss, comparing the relative scores of different classes
 - If we scale W , scores of all classes will be scaled the same way
 - If W makes $L = 0$, $2W$ also makes $L = 0$
- The optimization algorithm will waggle between solutions
 - We need regularizers to favor some solution than the others
 - So that the optimizer won't be confused

Regularizer

$$W^* = \arg \min_W \sum_i L_i(y_i, s_i) + \lambda R(W)$$

data loss regularization strength regularizer

- Some common regularizers:
 - L2 regularization: $R(W) = \sum_{k,l} W_{k,l}^2$
 - L1 regularization: $R(W) = \sum_{k,l} |W_{k,l}|$

Questions?

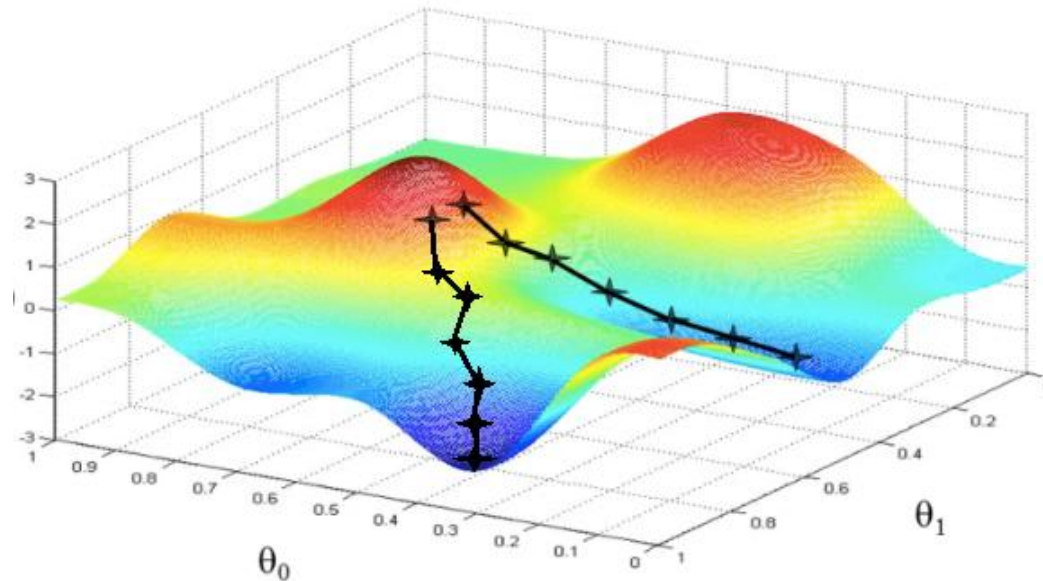


Parameter Estimation

- Estimate parameters from training data

$$W^* = \arg \min_W \sum_i L_i(y_i, s_i) + \lambda R(W)$$

- Energy minimization by gradient descent

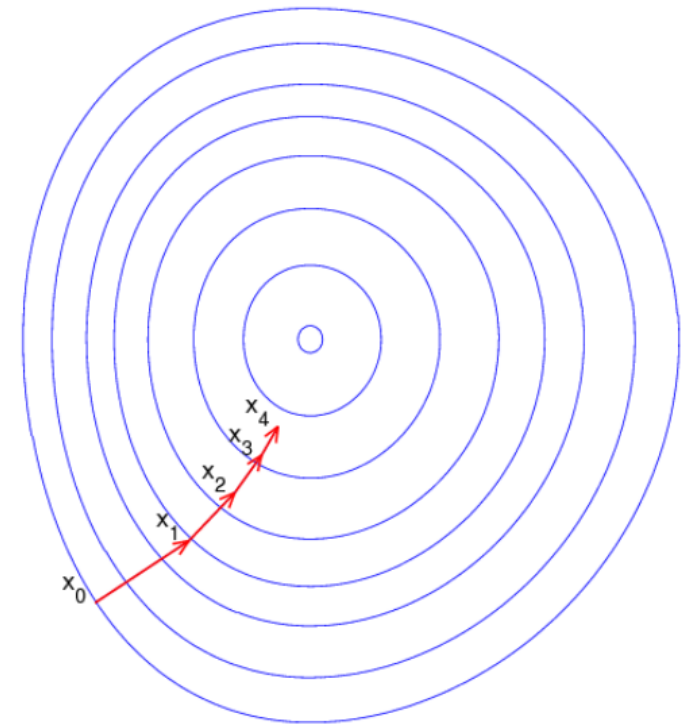


Gradient Descent

- Start from some initial parameters W_0
- Iterate the following two steps:
 - Compute the gradient vector $\frac{\partial L}{\partial W}$
 - Update the parameters

$$W_{n+1} = W_n + \alpha \frac{\partial L}{\partial W}$$

α is called learning rate



Gradient of Multi-Variable Functions

- In 1D, the derivative of a function is:

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

- In multiple dimensions, the gradient is a vector of partial derivatives along each dimension

$$\frac{\partial f}{\partial \mathbf{x}} = \begin{pmatrix} df/dx_1 \\ df/dx_2 \\ \vdots \\ df/dx_n \end{pmatrix}$$

Stochastic Gradient Descent (SGD)

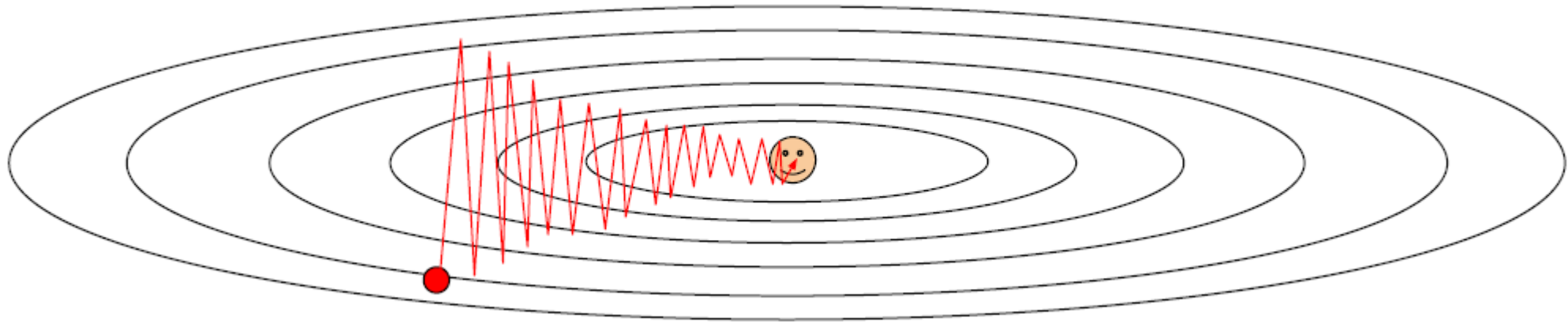
- Evaluating the Loss (and gradient) over all training data is expensive

$$W^* = \arg \min_W \sum_i L_i(y_i, s_i) + \lambda R(W)$$

- Approximate sum using a mini-batch of training data, e.g. 32/64/128

Problems in SGD

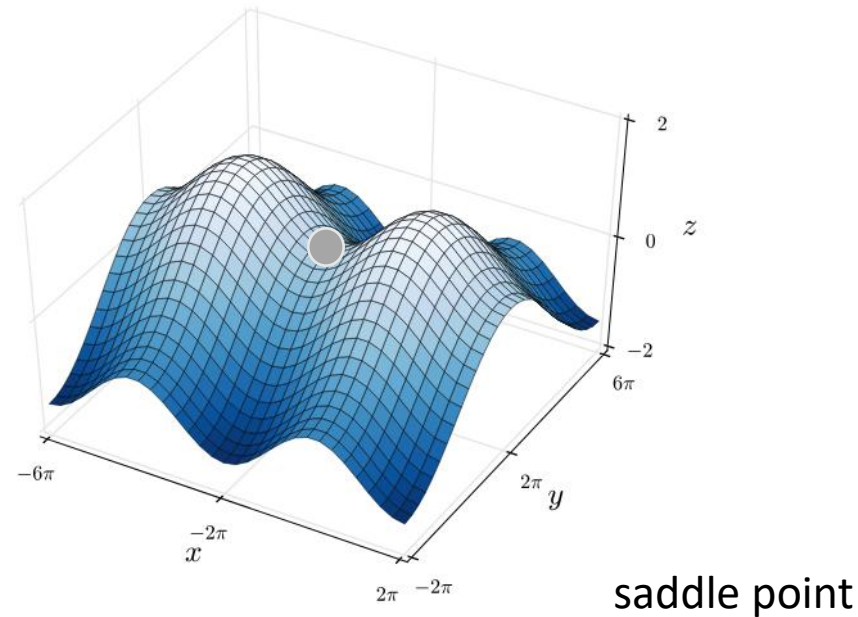
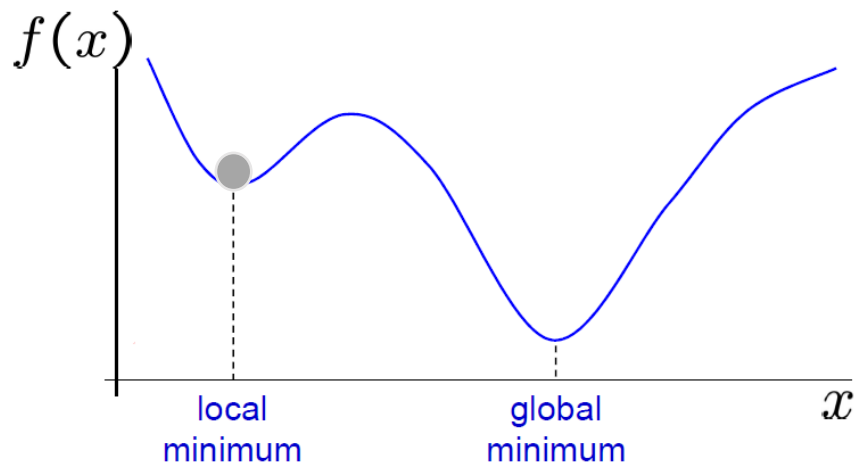
- What if the loss function changes quickly in one dimension but slowly in another?
- What does gradient descent do?
- **Very slow progress along shallow dimension, jitter along steep direction**



The gradient direction is always perpendicular to the local contours

Problems in SGD

- What if the loss function has a local minima or saddle point?



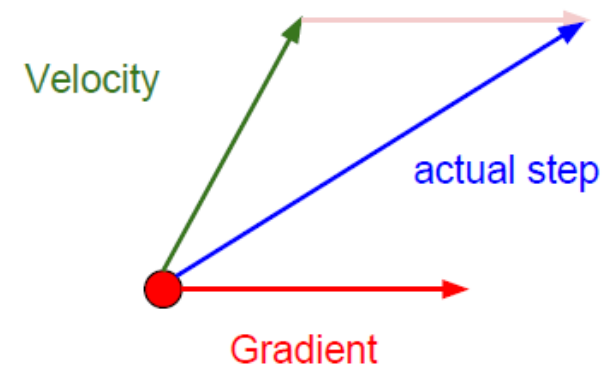
Momentum

- Build up velocity as a running mean of gradient
- Compute the update vector as combination of gradient and velocity

$$v_{t+1} = \rho v_t + \nabla f(x_t)$$

$$x_{t+1} = x_t - \alpha v_{t+1}$$

```
vx = 0
while True:
    dx = compute_gradient(x)
    vx = rho * vx + dx
    x -= learning_rate * vx
```



Recap: Learning Parametric Classifiers

- Choose a parametric function $s_i = f(x_i, W)$
- Choose a loss function $L(y_i, s_i)$
- Minimize the loss over all training data to learn W

$$W^* = \arg \min_W \sum_i L_i(y_i, s_i)$$

- Regularizers are helpful
- Minimize by gradient descent
- Use stochastic gradient descent
- Use momentum

Questions?

