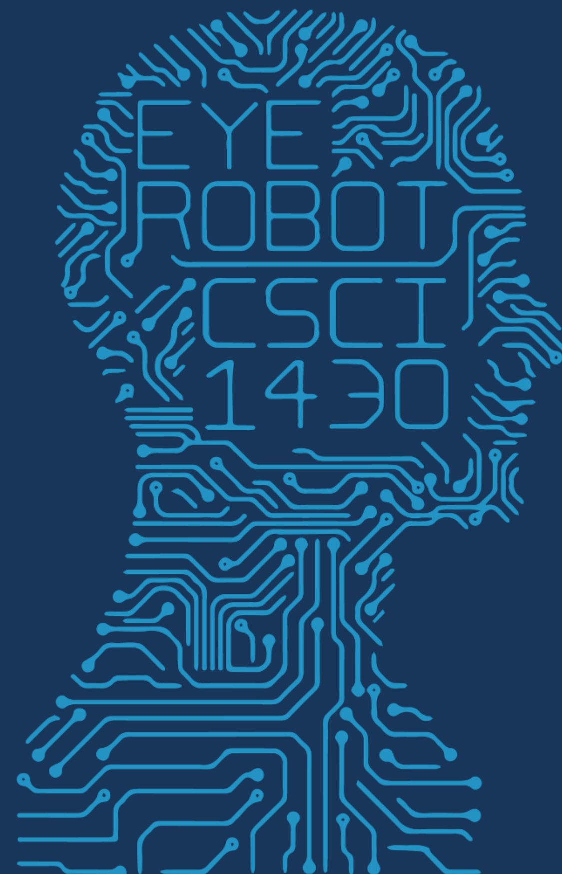
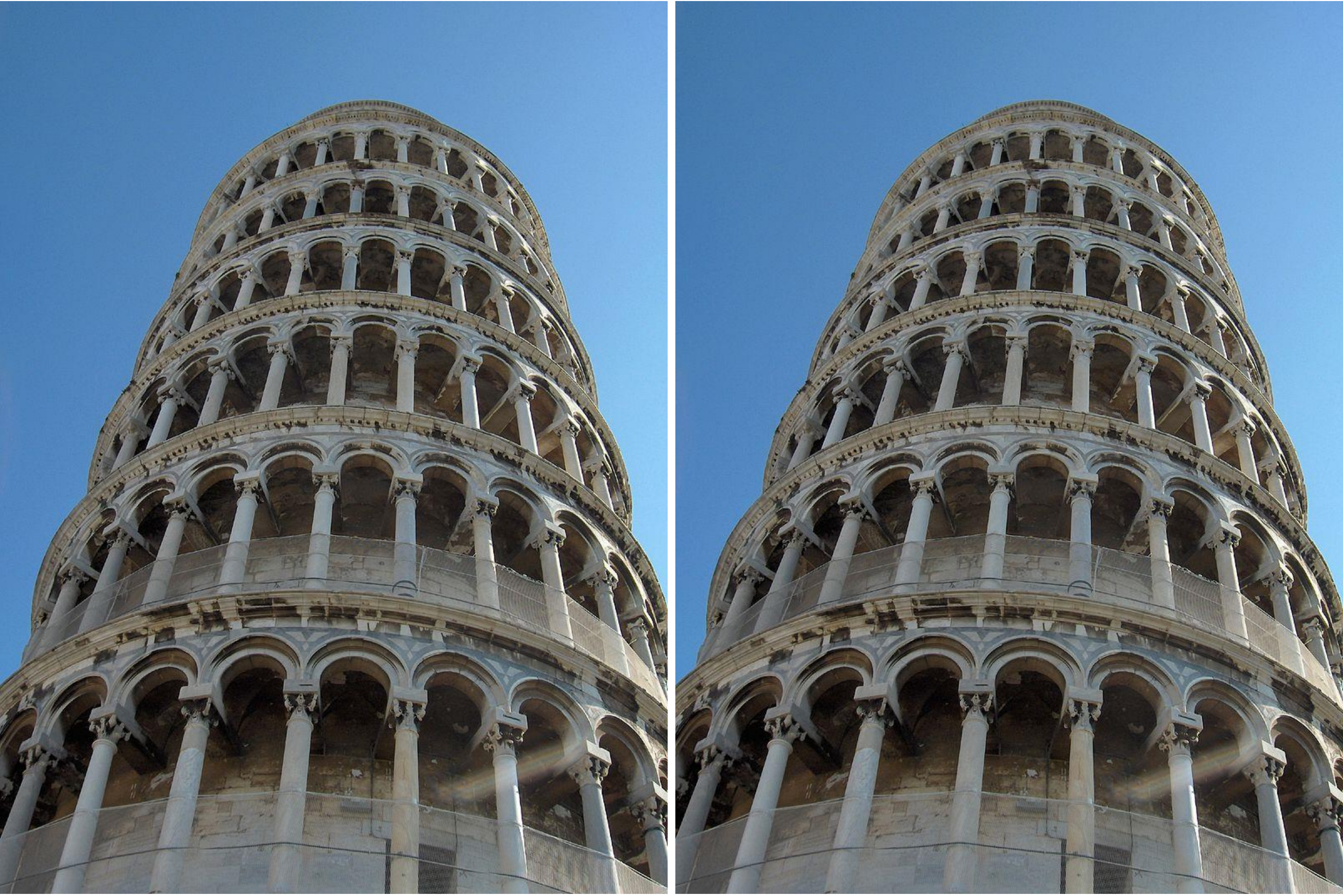
1950

Future Vision



2017 MWF 1pm 368

Computer Vision



Frederick Kingdom

Frederick Kingdom

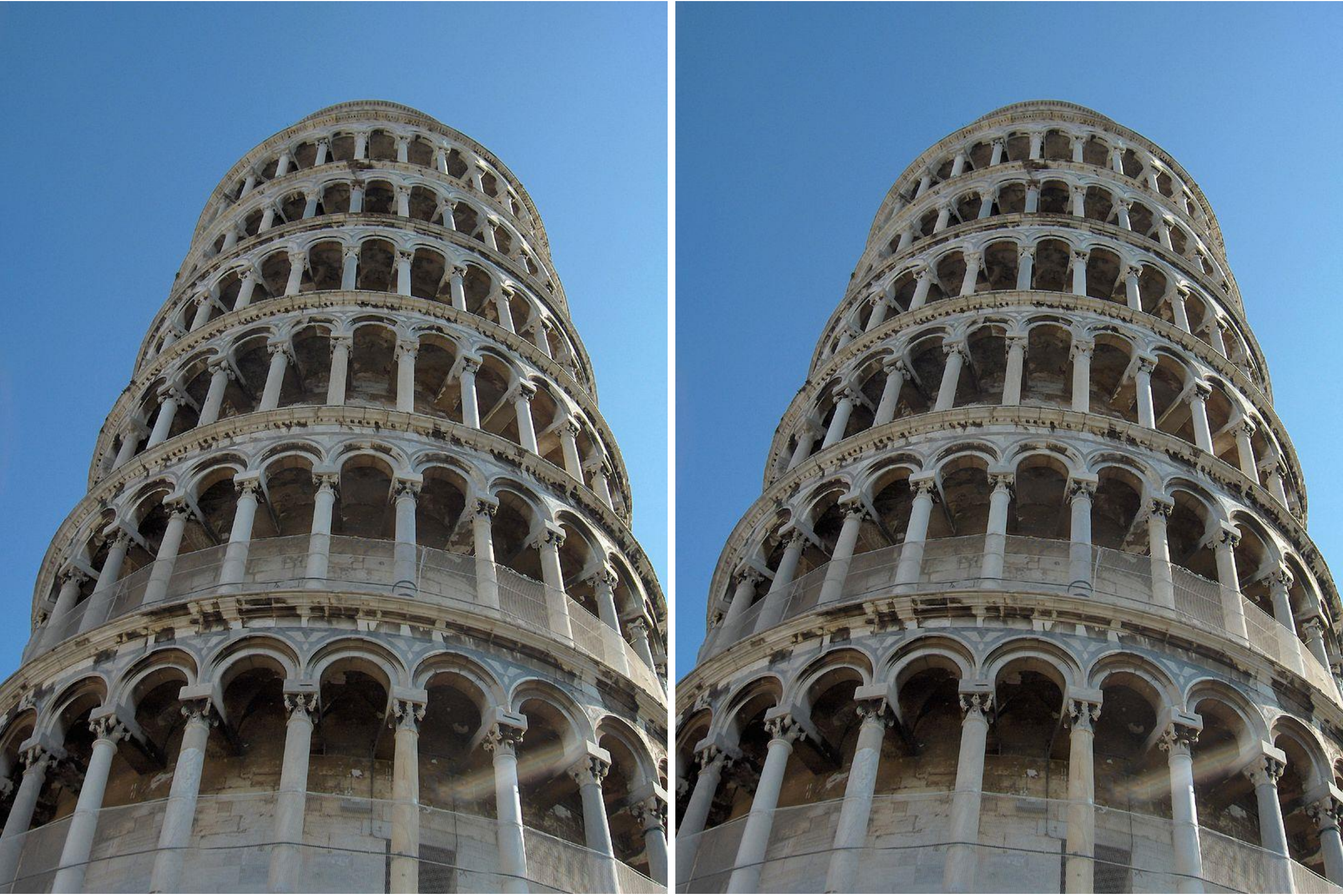
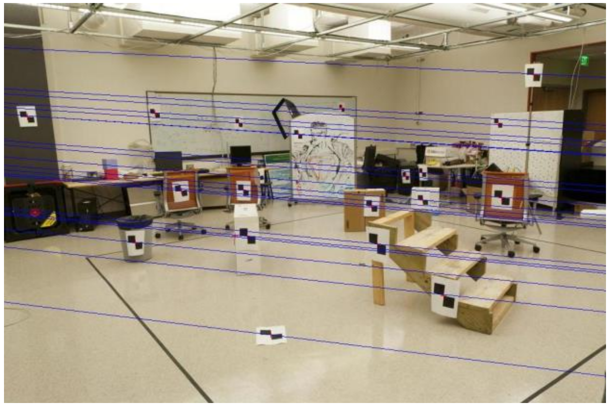
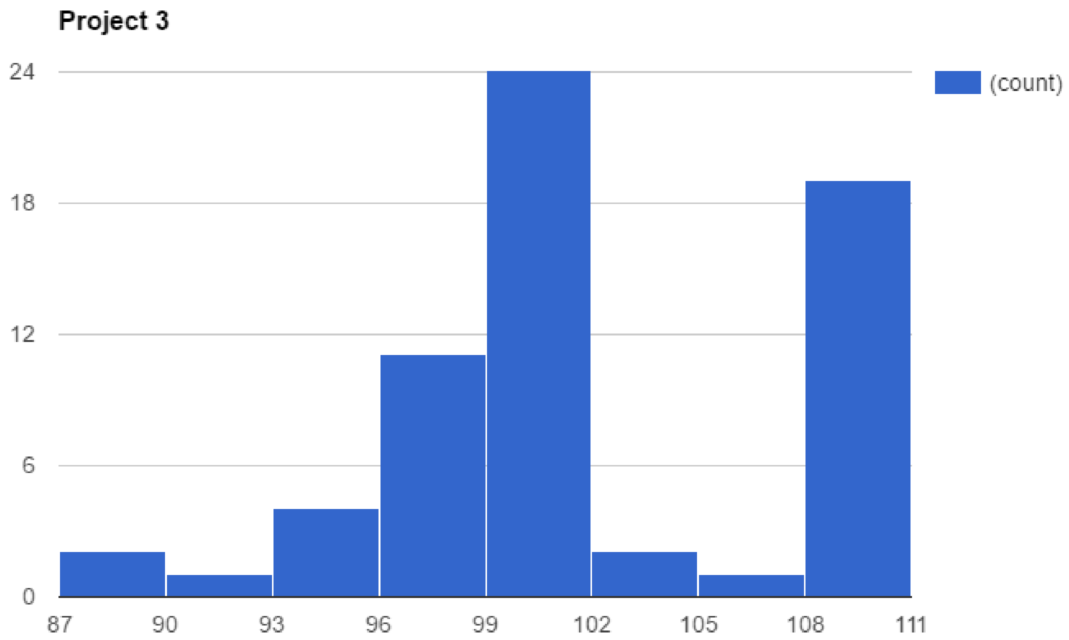


Photo: Georges Jansoon. Illusion: Frederick Kingdom, Ali Yoonessi and Elena Gheorghiu

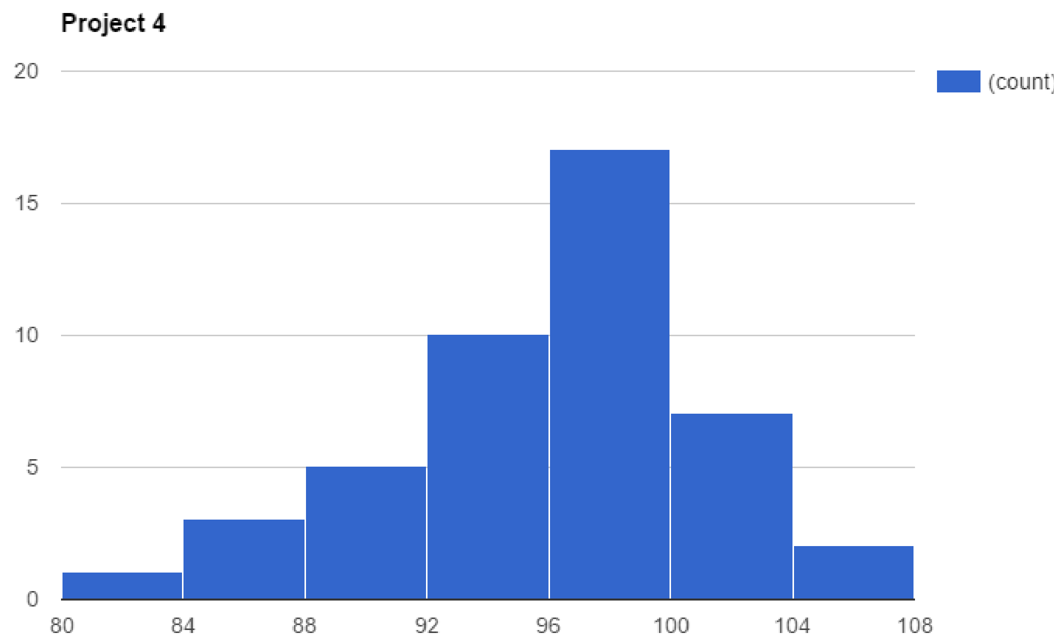
Frederick Kingdom

Project 3: Camera calibration





Project 4: Scene Recognition



Normalization – why?

Required by some underlying property of the learning mechanism.

- E.G., removing hyperplane bias in SVM to aid fitting.

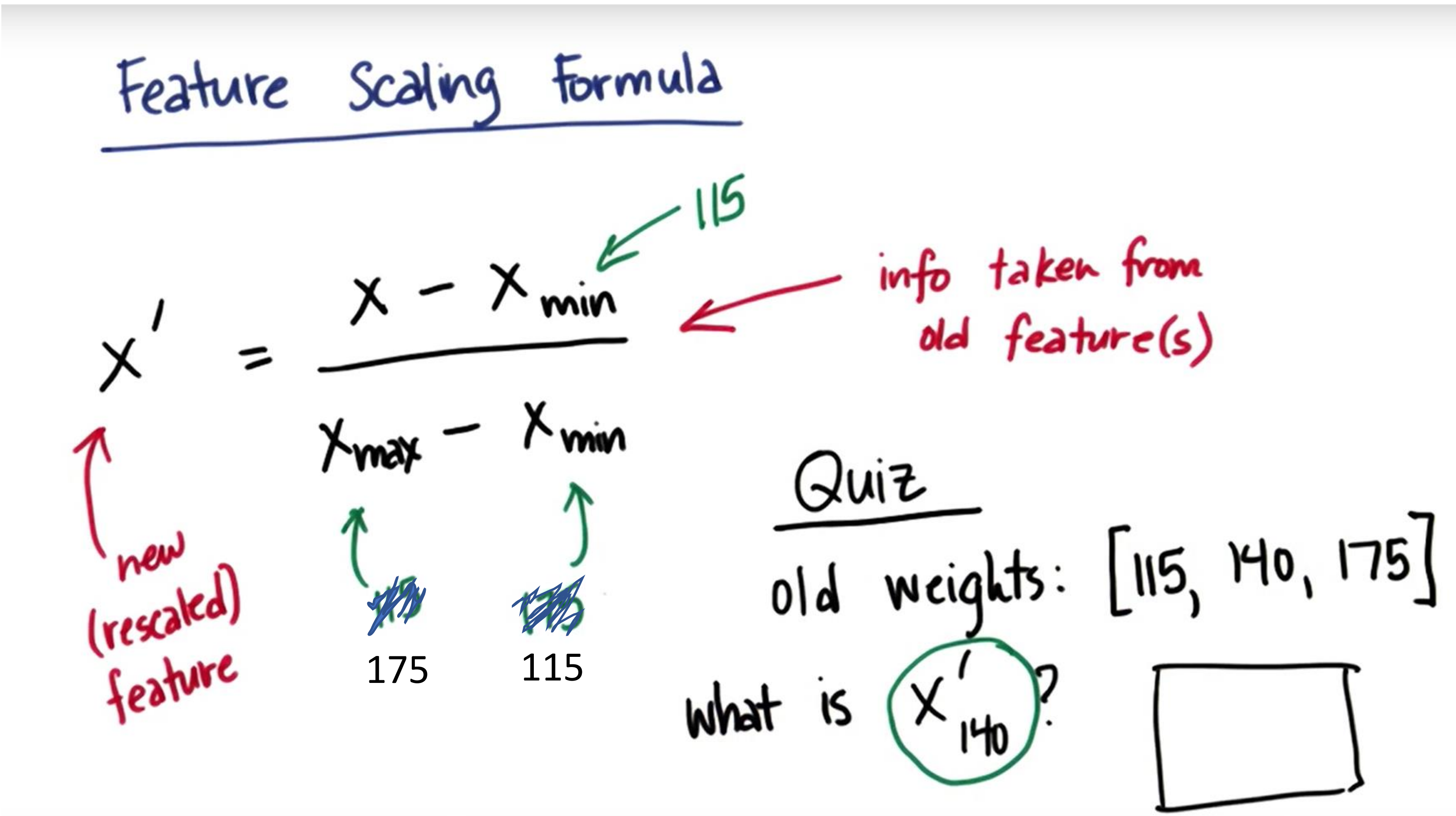
Also called 'feature scaling'.

- Many methods, e.g.,

$$x' = \frac{x - \bar{x}}{\sigma} \qquad\qquad x' = \frac{x}{||x||}$$

Normalization can be implemented *wrt. other data points,* and sometimes *wrt. other features.*

Wrt. other data points – human weight



Tiny Image as a feature vector

- Each pixel is treated as a different feature.
- Feature vector is matrix reshaped into an array.

Normalization – how?

```
function image_feats = get_tiny_images( image_paths )

size = 16;
N = size(image_paths, 1);
image_feats = zeros(N, size * size);

for i = 1:N
   img = im2double( imread(image_paths{i, 1}) );
   img = imresize( img, [size, size] );
   fv = reshape( img, [1, size * size] );

   fv = fv - mean( fv );
   image_feats(i, :) = fv ./ norm( fv );
end
```

Normalization – how?

```
function image_feats = get_tiny_images( image_paths )

size = 16;
N = size(image_paths, 1);
image_feats = zeros(N, size * size);

for i = 1:N
   img = im2double( imread(image_paths{i, 1}) );
   img = imresize( img, [size, size] );
   fv = reshape( img, [1, size * size] );

   fv = fv - mean( fv );
   image_feats(i, :) = fv ./ norm( fv );
end
```

Mean across *features*

```
function image_feats = get_tiny_images( image_paths )

size = 16;
N = size(image_paths, 1);
image_feats = zeros(N, size * size);

for i = 1:N
   img = im2double( imread(image_paths{i, 1}) );
   img = imresize( img, [size, size] );
   fv = reshape( img, [1, size * size] );

   fv = fv - mean( fv );              % Mean across features (pixels) per data point
   image_feats(i, :) = fv ./ norm( fv );
end
```

Mean across *data points*

```
function image_feats = get_tiny_images( image_paths )

size = 16;
N = size(image_paths, 1);
image_feats = zeros(N, size * size);

for i = 1:N
   img = im2double( imread(image_paths{i, 1}) );
   img = imresize( img, [size, size] );
   fv = reshape( img, [1, size * size] );
end

mean_img = mean(image_feats);   % Mean of each feature (pixel) across data points
var_img = std(image_feats);

for i = 1:N
   image_feats(i,:) = ( image_feats(i,:) - mean_img ) ./ var_img;
end
```

Friday: CV for Social ~~Good~~ Bad

- We saw how dataset bias can introduce error.

- But what about the underlying feature representation itself?

- How might I discover why my CV system is bad?
- How might I explain the failure?

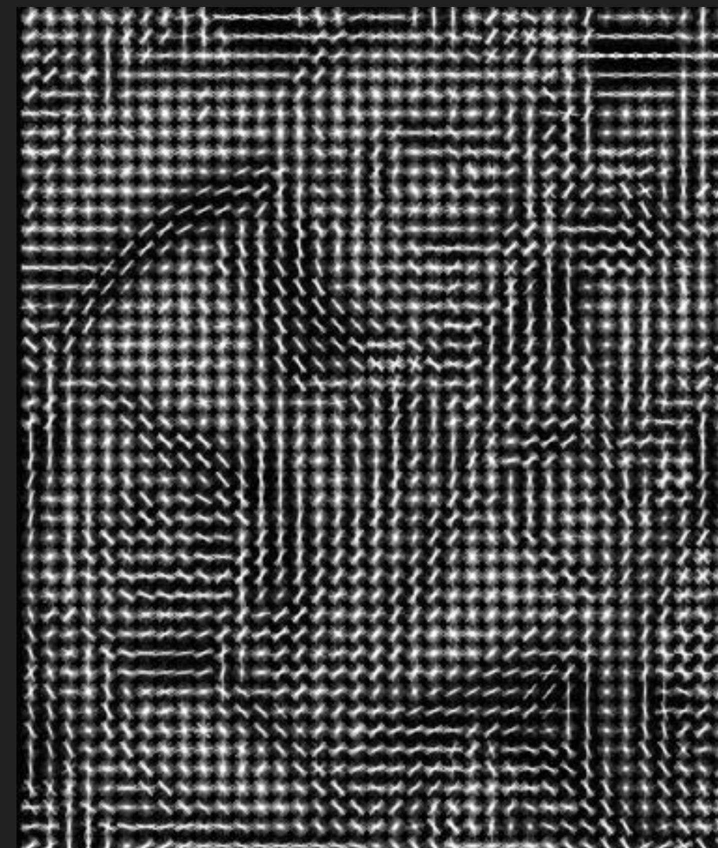
Vondrick et al.

Vondrick et al.

Vondrick et al.

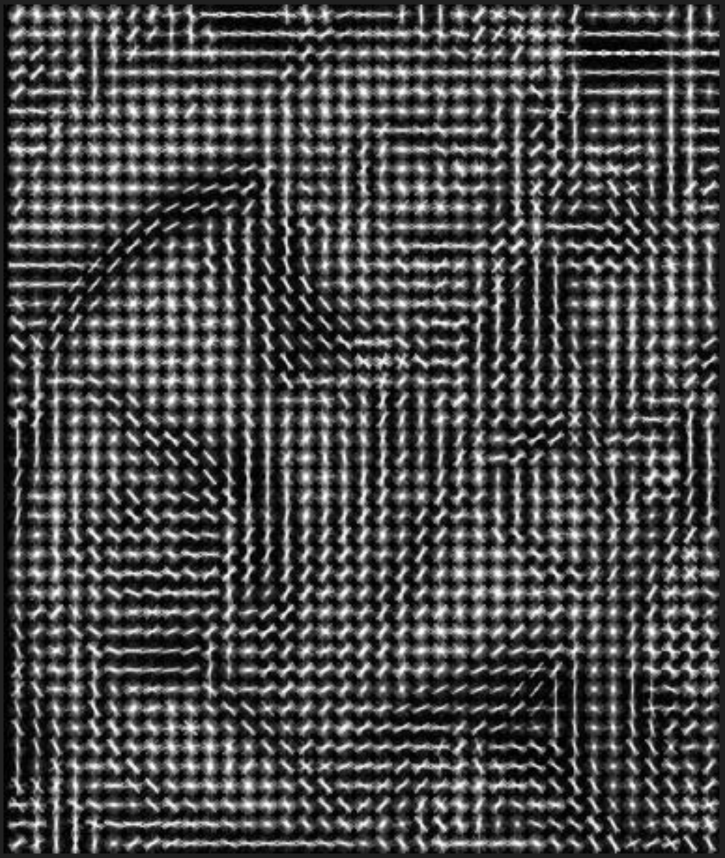
Vondrick et al.

What information is lost?



Vondrick et al.

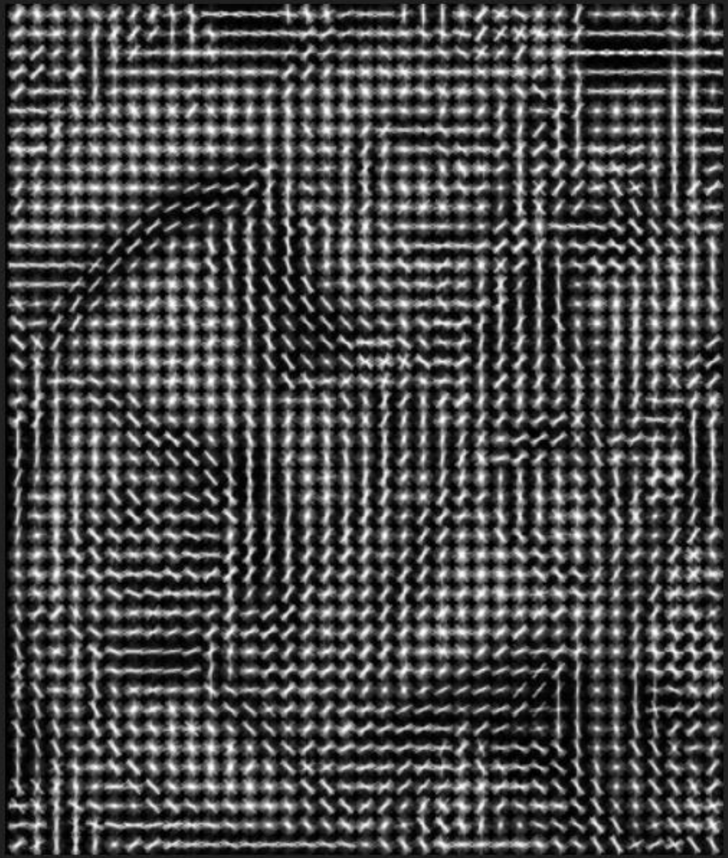
What information is lost?



Vondrick et al.

What information is lost?

$$\min_{x \in \mathbb{R}^d} ||\phi(x) - y||_2^2$$



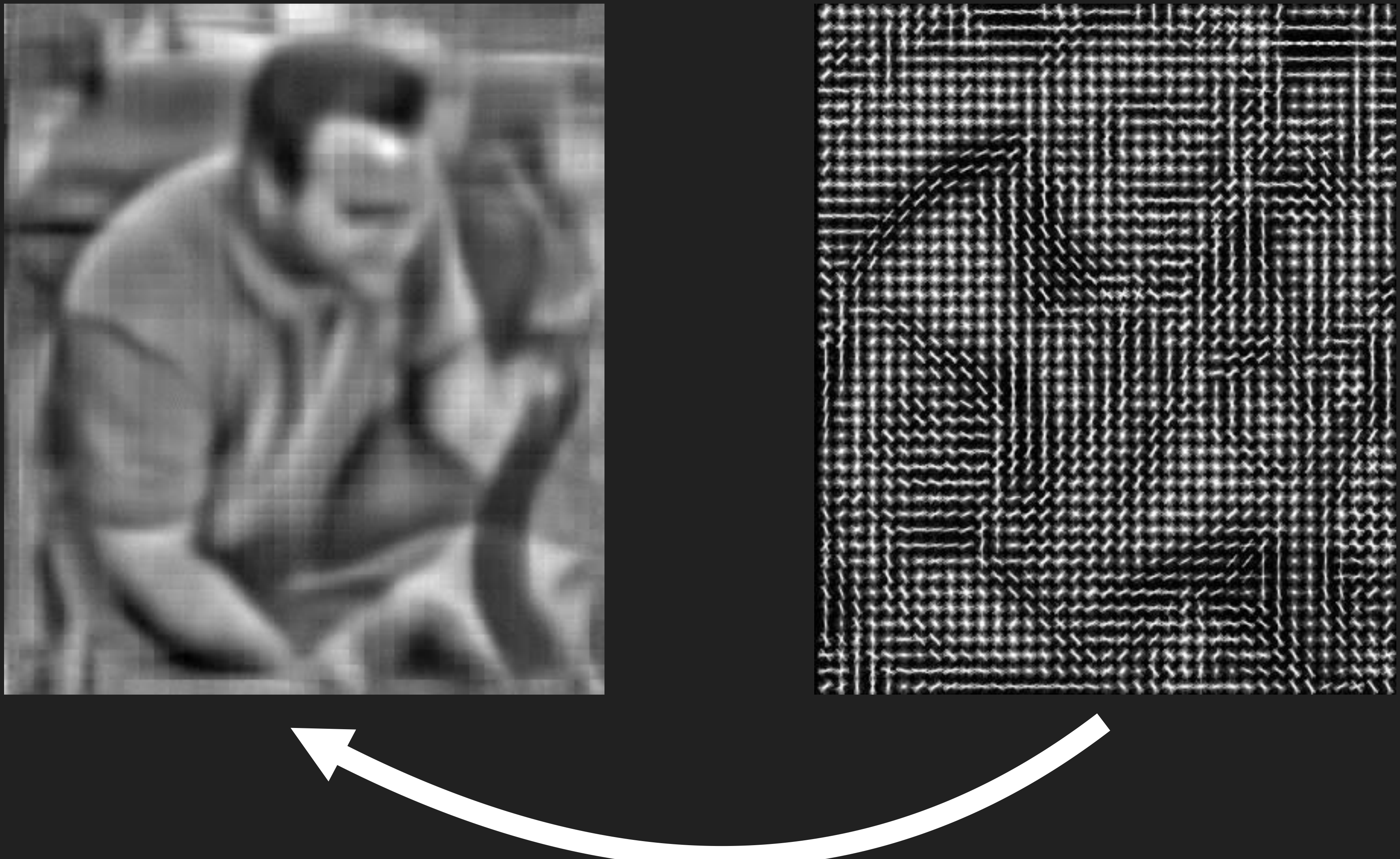
HOG = ϕ

Many-to-one function

No inverse

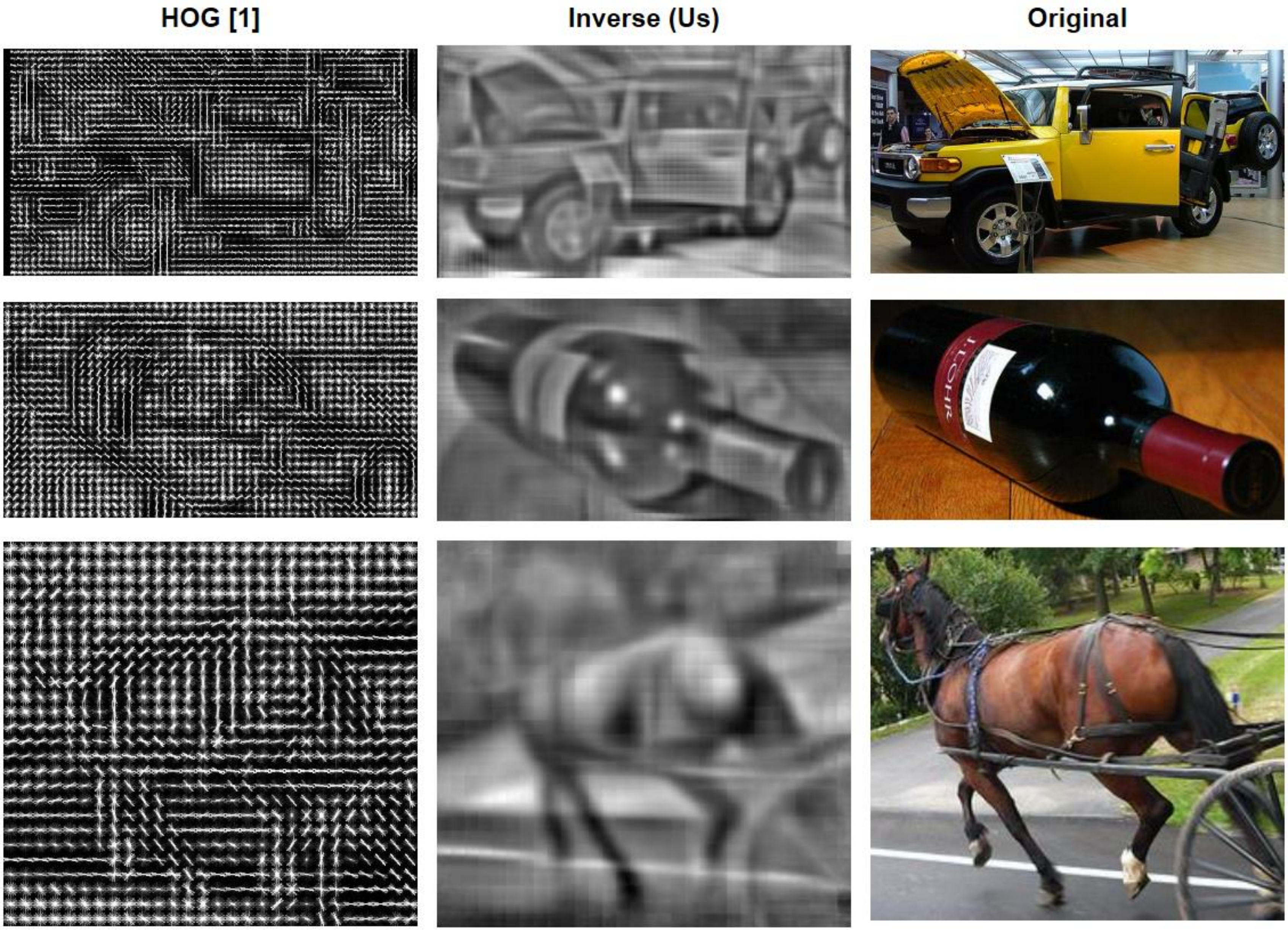
Vondrick et al.

What information is lost?

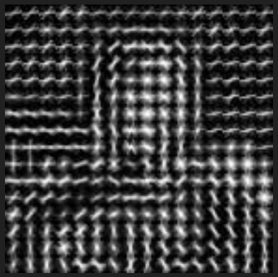


Vondrick et al.

HOGgles (Vondrick et al. ICCV 2013)

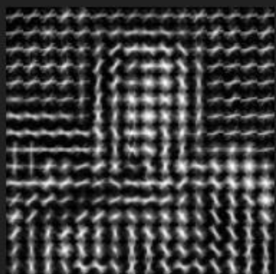
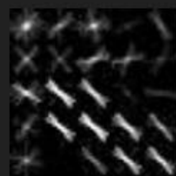
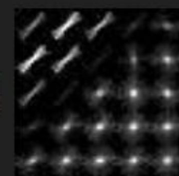
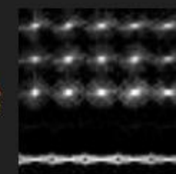


Method: Paired Dictionary



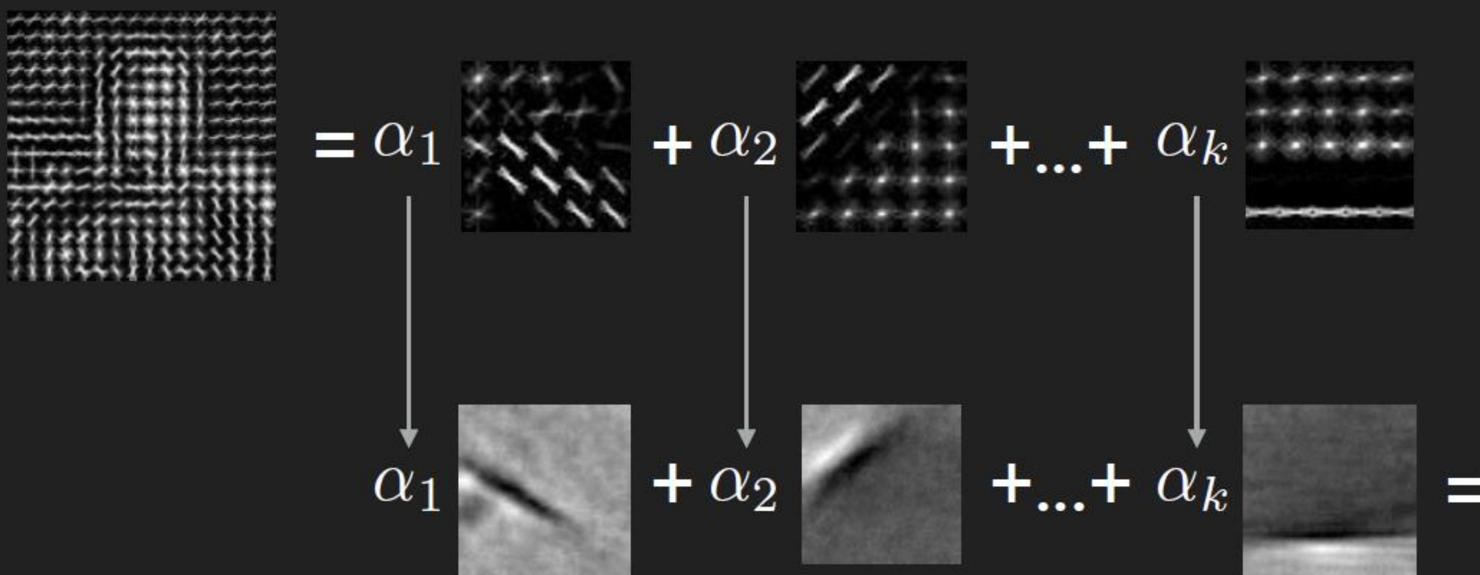
Vondrick et al.

Method: Paired Dictionary

 = α_1  + α_2  +...+ α_k 

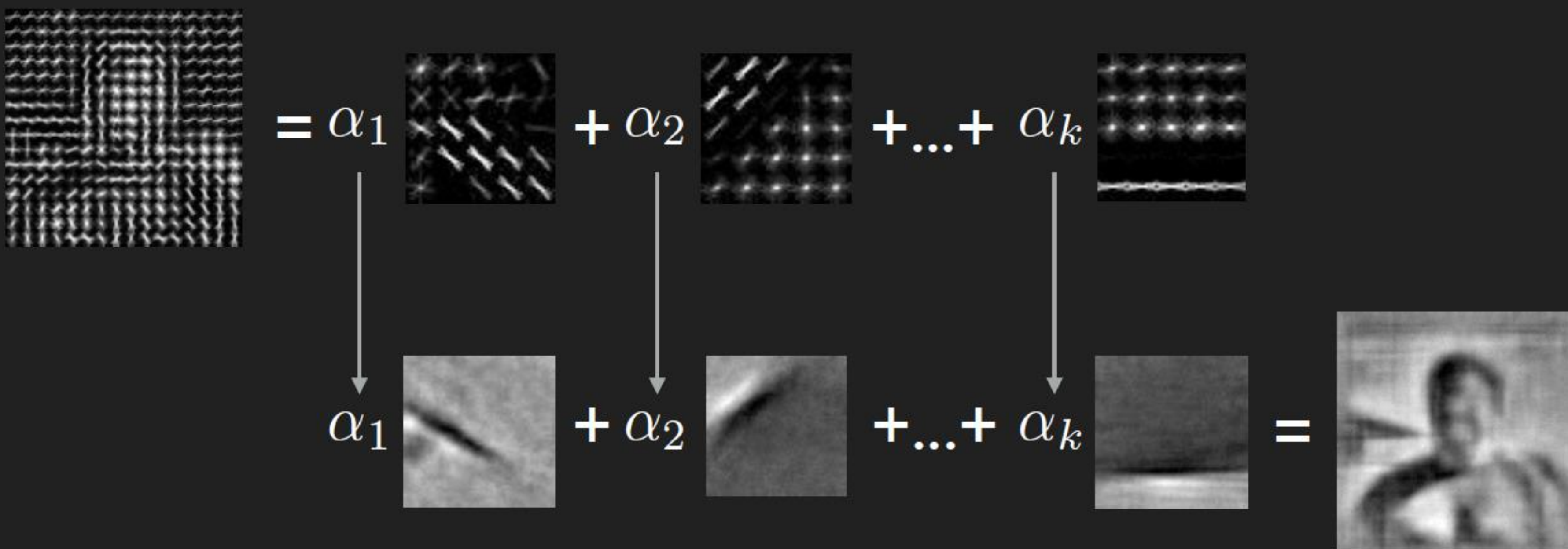
Vondrick et al.

Method: Paired Dictionary

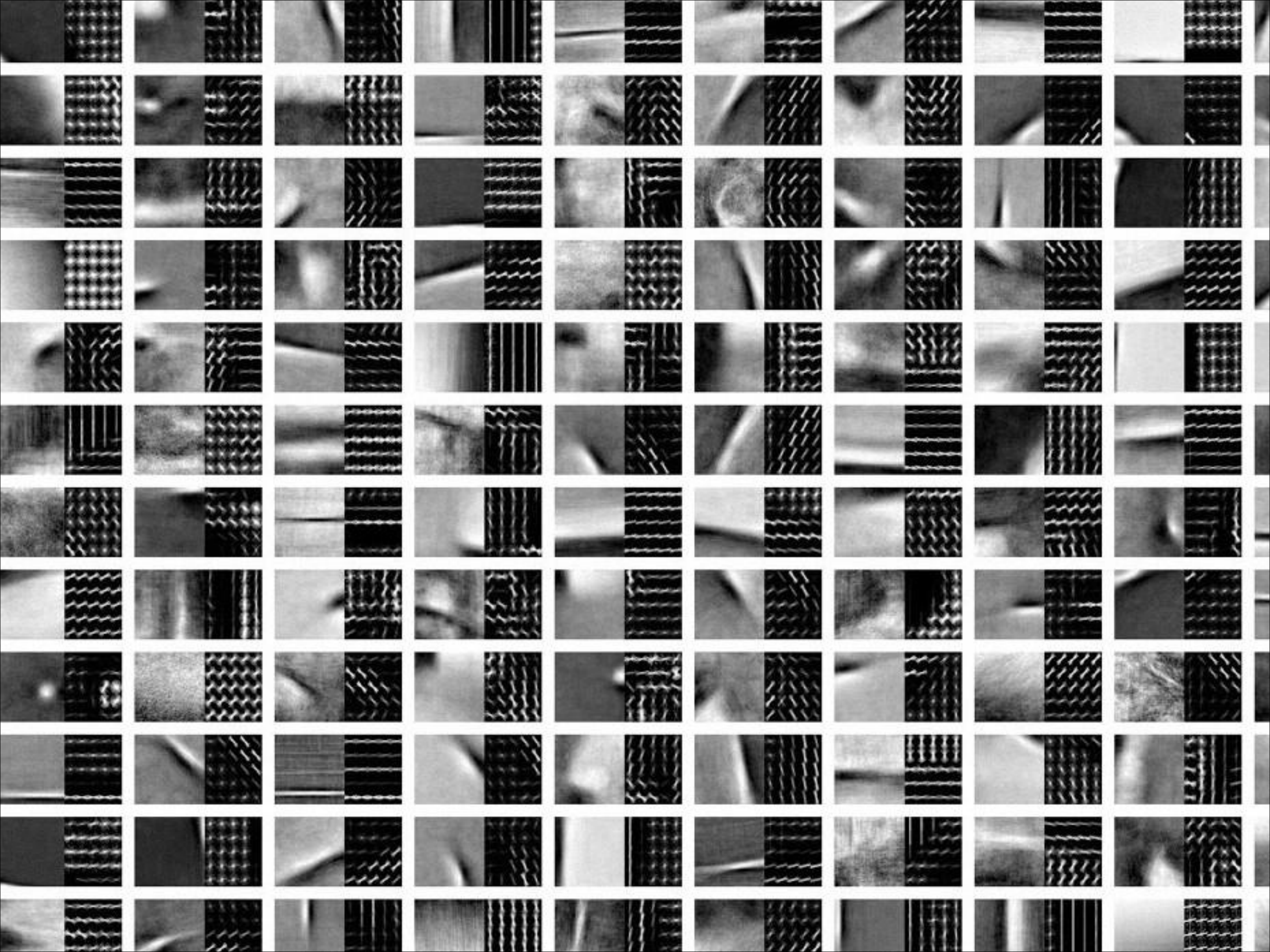


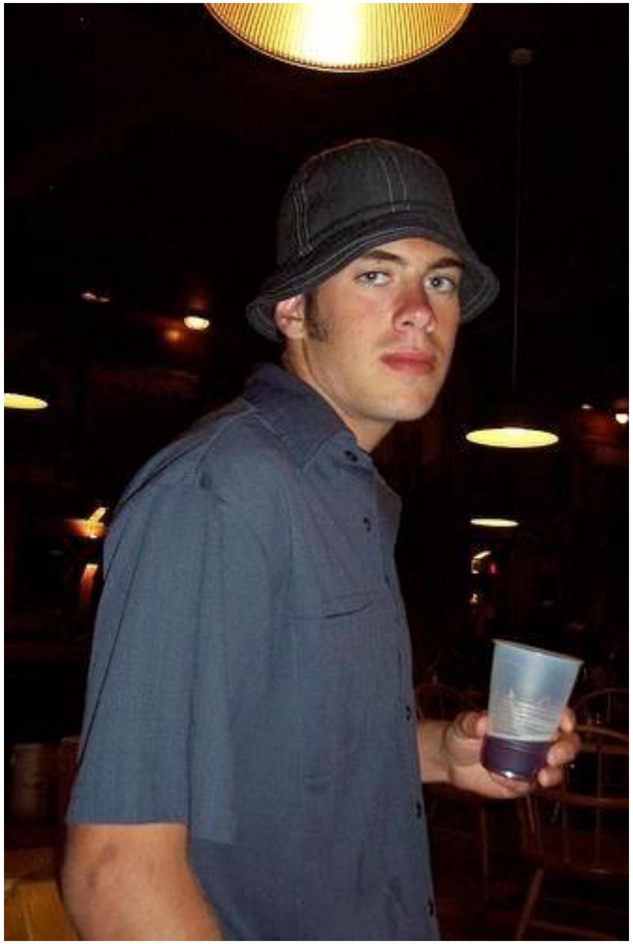
Vondrick et al.

Method: Paired Dictionary



Vondrick et al.





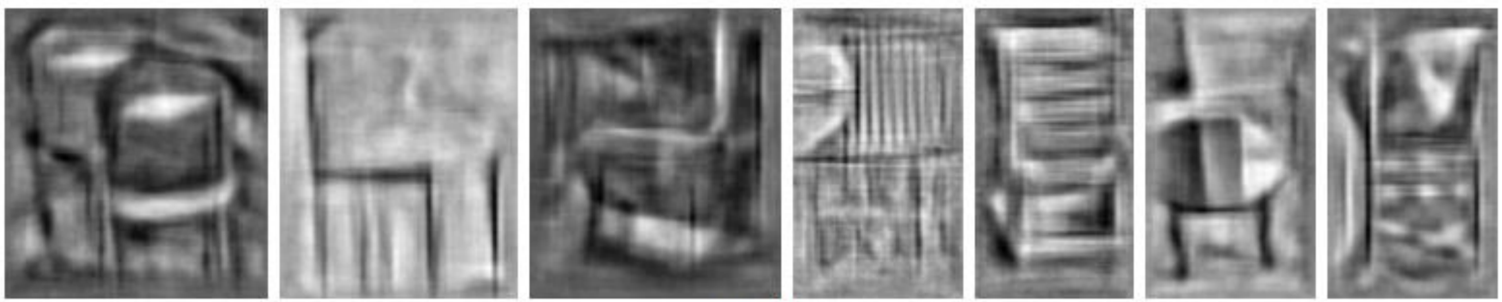
HumanVision HOGVision

Vondrick et al.

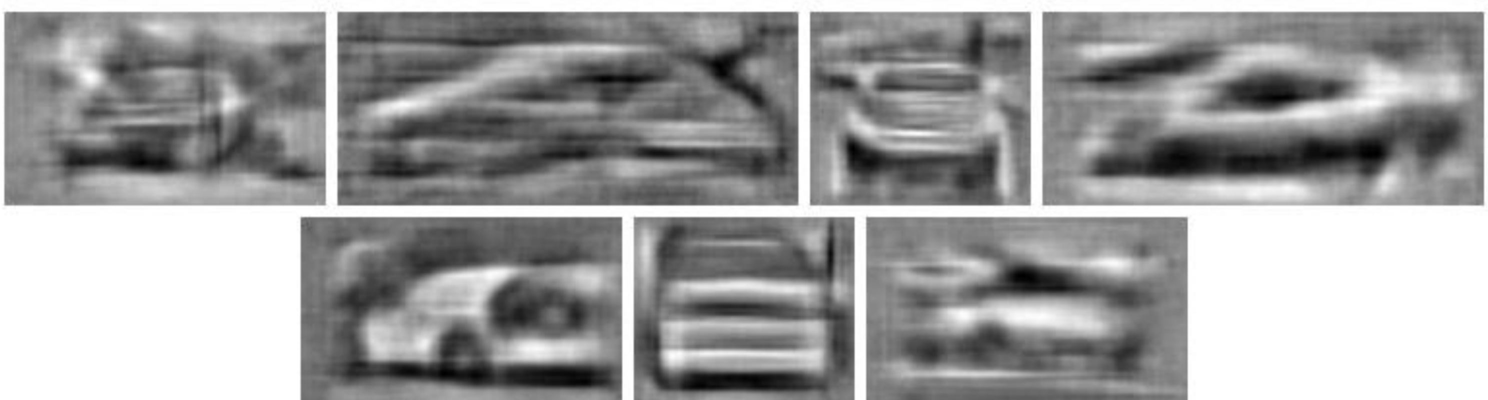
Visualizing Top Detections

We have visualized some high scoring detections from the deformable parts model. Can you guess which are false alarms? Click on the images below to reveal the corresponding RGB patch. You might be surprised!

Person



Chair



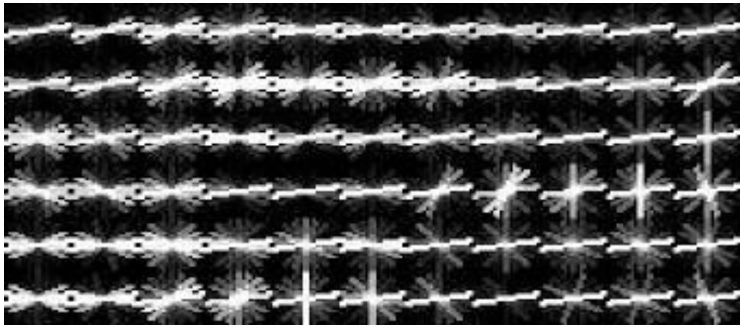
Car

Vondrick et al.

Why did the detector fail?

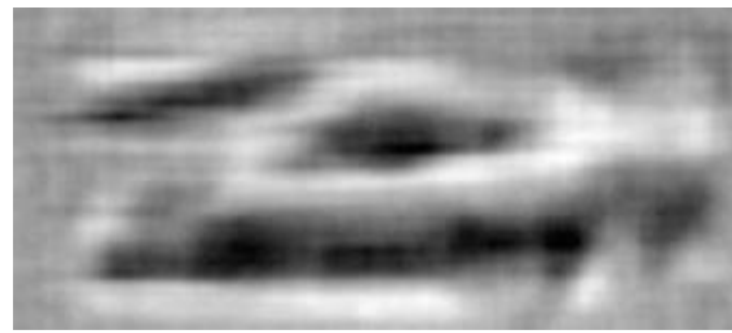
Vondrick et al.

Why did the detector fail?



Vondrick et al.

Why did the detector fail?



Vondrick et al.

Code Available

Try it on your project 5!

http://web.mit.edu/vondrick/ihog/

```
ihog = invertHOG(feat);
```

Friday: CV for Social ~~Good~~ Bad

- We saw how dataset bias can introduce error.
- …and how features can be ambiguous.
- What about label bias errors?
- How might I move towards a more flexible label system?

Describing Objects by their Attributes

Ali Farhadi, Ian Endres,
Derek Hoiem, David Forsyth

CVPR 2009


U.S. AIR FORCE

What do we want to know about this object?

What do we want to know about this object?

Object recognition expert:
“Dog”

What do we want to know about this object?

Object recognition expert:
"Dog"

Person in the Scene:
"Big pointy teeth", "Can move fast", "Looks angry"

Goal: Infer Object Properties

Can I **draw with it**?

Can I **put stuff in it**?

Is it **alive**?

What **shape** is it?

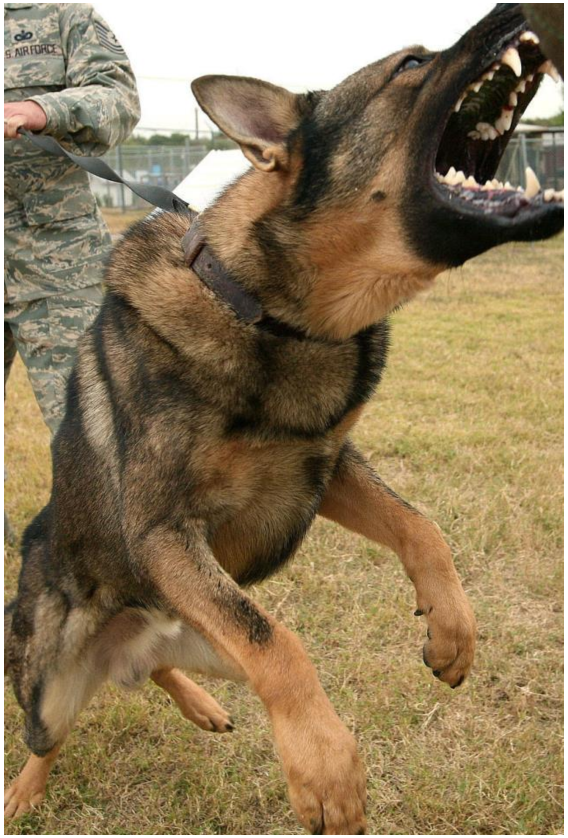
Is it **soft**?

Does it have a **tail**?

Will it **blend**?

Why Infer Properties

1. We want detailed information about objects



"Dog"
vs.
"Large, angry animal with pointy teeth"

Why Infer Properties

2. We want to be able to infer something about unfamiliar objects – “zero shot learning”

Familiar Objects

New Object

Why Infer Properties

2. We want to be able to infer something about unfamiliar objects – “zero shot learning”

If we can infer category names…

Familiar Objects

New Object

Cat

Horse

Dog

???

Why Infer Properties

2. We want to be able to infer something about unfamiliar objects – “zero shot learning”

If we can infer properties…

Familiar Objects

Has Stripes
Has Ears
Has Eyes
….

Has Four Legs
Has Mane
Has Tail
Has Snout
….

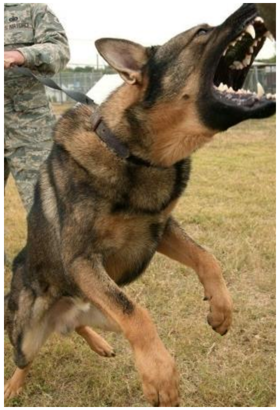
Brown
Muscular
Has Snout
….

New Object

Has Stripes (like cat)
Has Mane and Tail (like horse)
Has Snout (like horse and dog)

Why Infer Properties

3. We want to make comparisons between objects or categories



What is unusual about this dog?

What is the difference between horses and zebras?

Strategy 1: Category Recognition

Object Image

classifier →

Category

“Car”

associated properties →

Has Wheels
Used for Transport
Made of Metal
Has Windows
…

Category Recognition: PASCAL 2008
Category → Attributes: ??

Strategy 2: Exemplar Matching

Object Image

similarity function

Similar Image

associated properties

Has Wheels
Used for Transport
Made of Metal
Old
…

Malisiewicz Efros 2008
Hays Efros 2008
Efros et al. 2003

Strategy 3: Infer Properties Directly

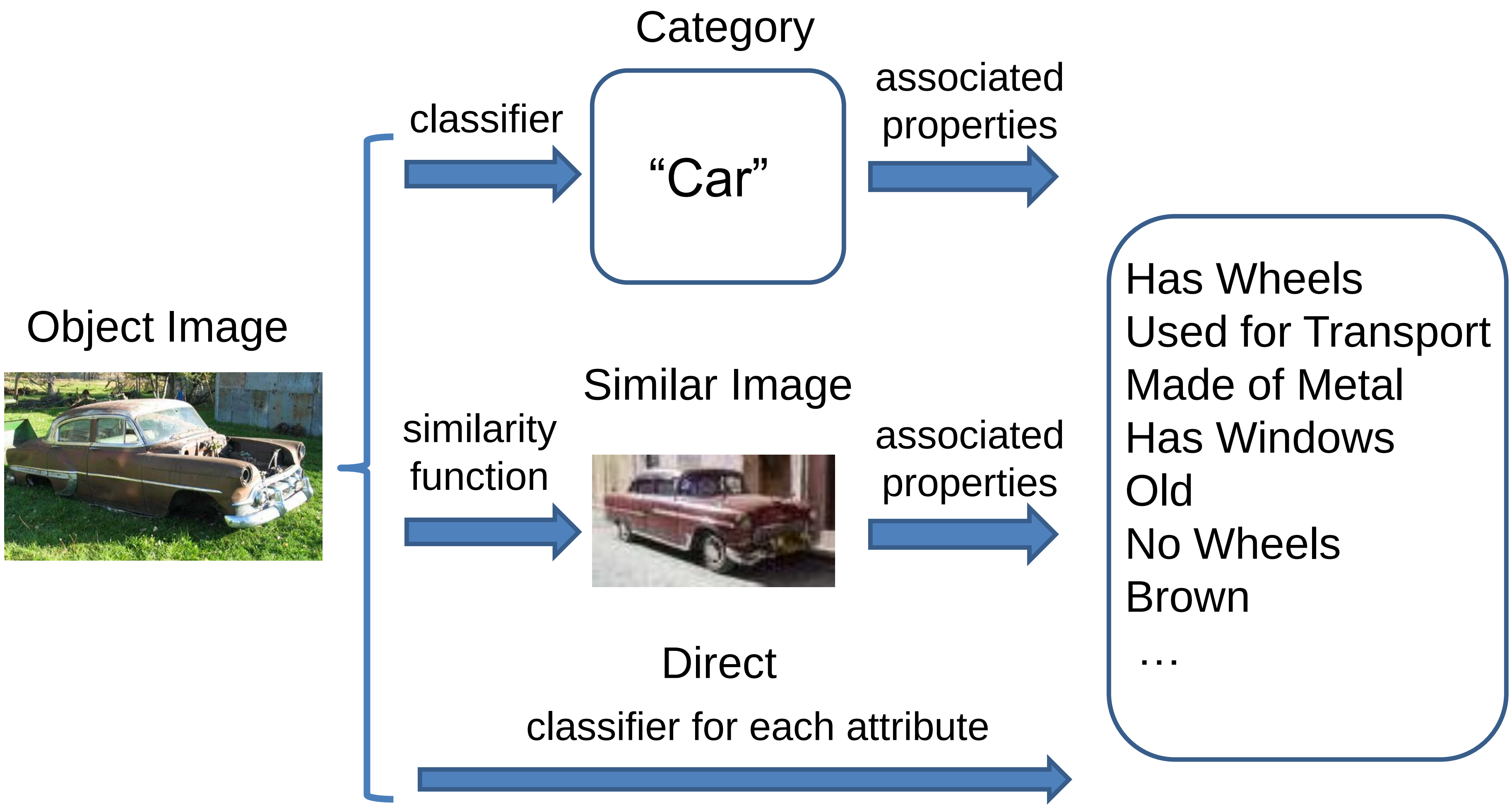
Object Image

classifier for each attribute

No Wheels
Old
Brown
Made of Metal
…

See also Lampert et al. 2009
Gibson's affordances

The Three Strategies



Candidate attributes

- Visible parts: “wheels”, “snout”, “eyes”
- Visible materials or material properties: “made of metal”, “shiny”, “clear”, “made of plastic”
- Shape: “3D boxy”, “round”

Attribute Examples

Shape: Horizontal Cylinder
Part: Wing, Propeller, Window, *Wheel*
Material: *Metal*, Glass

Shape:
Part: Window, *Wheel*, Door, Headlight, Side Mirror
Material: *Metal*, Shiny

Attribute Examples

Shape:
Part: Head, Ear, Nose, Mouth, Hair, Face, Torso, Hand, Arm
Material: Skin, Cloth

Shape:
Part: Head, Ear, Snout, Eye
Material: Furry

Shape:
Part: Head, Ear, Snout, Eye, Torso, Leg
Material: Furry

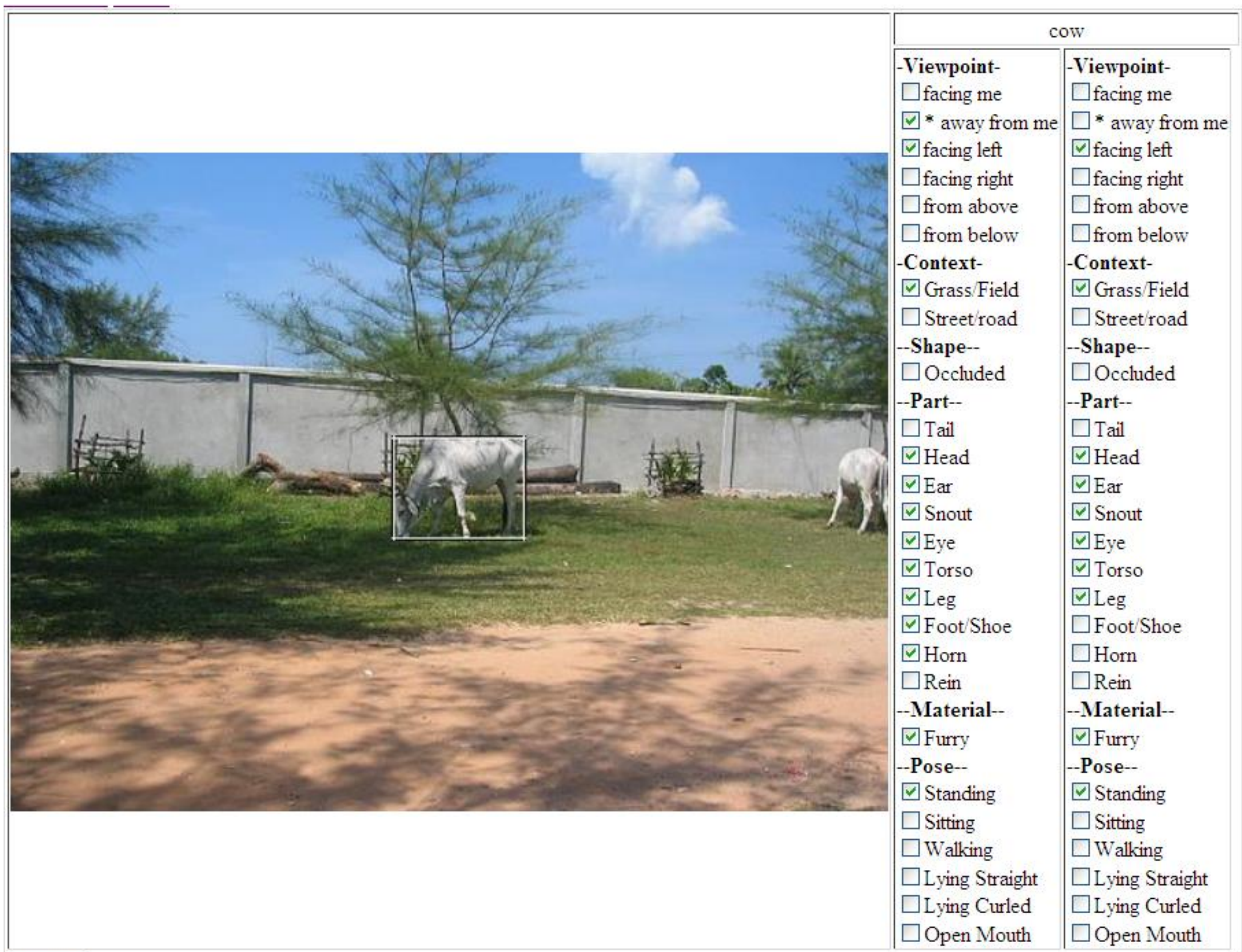
Datasets

- a-Pascal
  - 20 categories from PASCAL 2008 trainval dataset (10K object images)
    - airplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, sofa, train, tv monitor
  - ‘Ground truth’ for 64 attributes
  - Annotation via Amazon’s Mechanical Turk

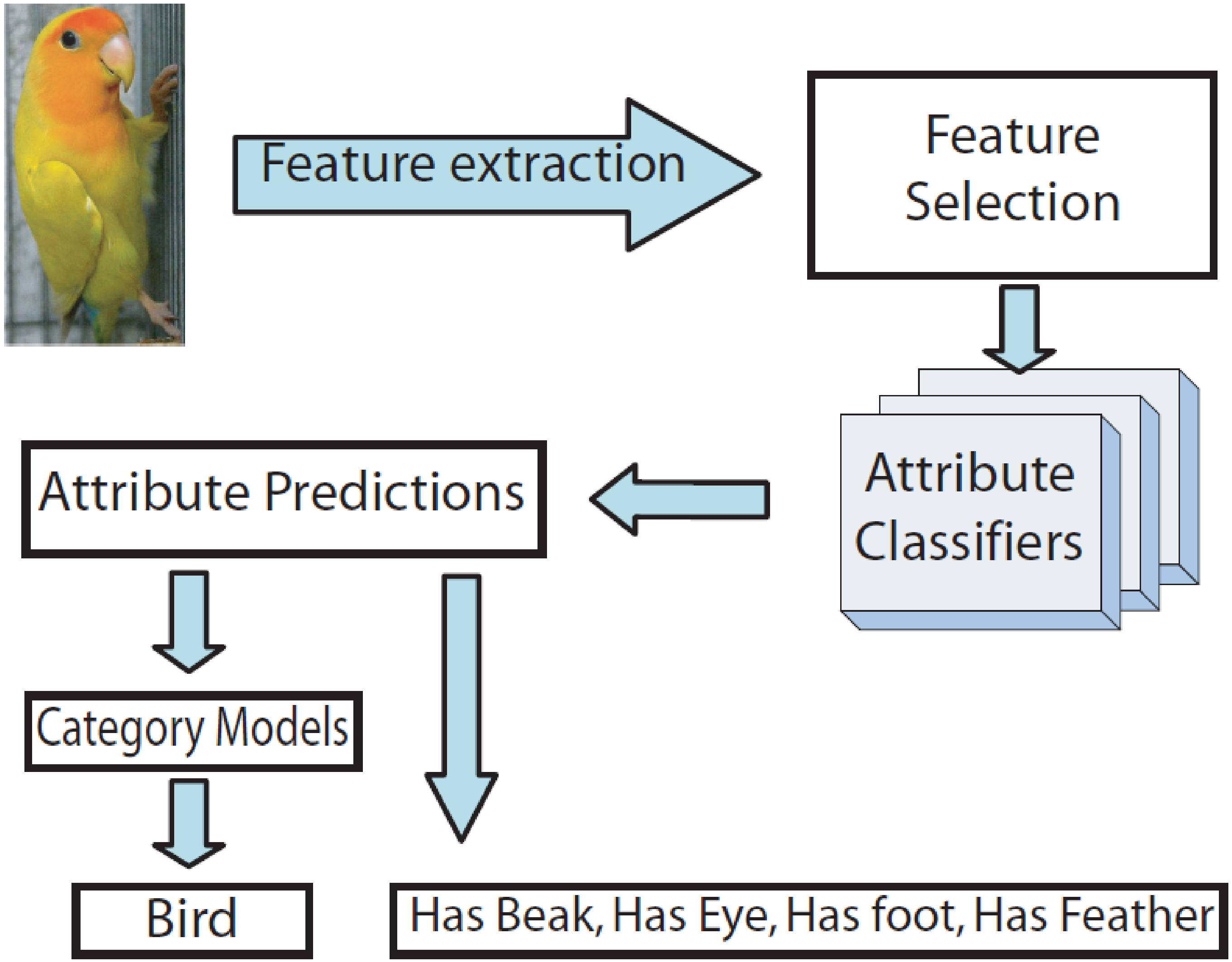
- a-Yahoo
  - 12 new categories from Yahoo image search
    - bag, building, carriage, centaur, donkey, goat, jet ski, mug, monkey, statue of person, wolf, zebra
  - Categories chosen to share attributes with those in Pascal

- Attribute labels are somewhat ambiguous
  - Agreement among “experts” 84.3
  - Between experts and Turk labelers 81.4
  - Among Turk labelers 84.1

Annotation on Amazon Turk



Approach


Feature extraction
Feature Selection
Attribute Classifiers
Attribute Predictions
Category Models
Bird
Has Beak, Has Eye, Has foot, Has Feather

Features + classifiers

Spatial pyramid histograms of quantized

- Color and texture for **materials**
- Histograms of gradients (HOG) for **parts**
- Canny edges for **shape**

Learn presence / absence of attribute.

- Train one classifier (linear SVM) per attribute

Average ROC Area

Trained on a-PASCAL objects

Test Objects	Parts	Materials	Shape
a-PASCAL	0.794	0.739	0.739
a-Yahoo	0.726	0.645	0.677

Describing Objects by their Attributes

No examples from these object categories were seen during training

Describing Objects by their Attributes

No examples from these object categories were seen during training

Category Recognition

- Semantic attributes not enough
  - 74% accuracy even with ground truth attributes
- Introduce discriminative attributes
  - Trained by selecting subset of classes and features
    - Dogs vs. sheep using color
    - Cars and buses vs. motorbikes and bicycles using edges
  - Train 10,000 and select 1,000 most reliable, according to a validation set

Attributes not big help when sufficient data

- Use attribute predictions as features
- Train linear SVM to categorize objects

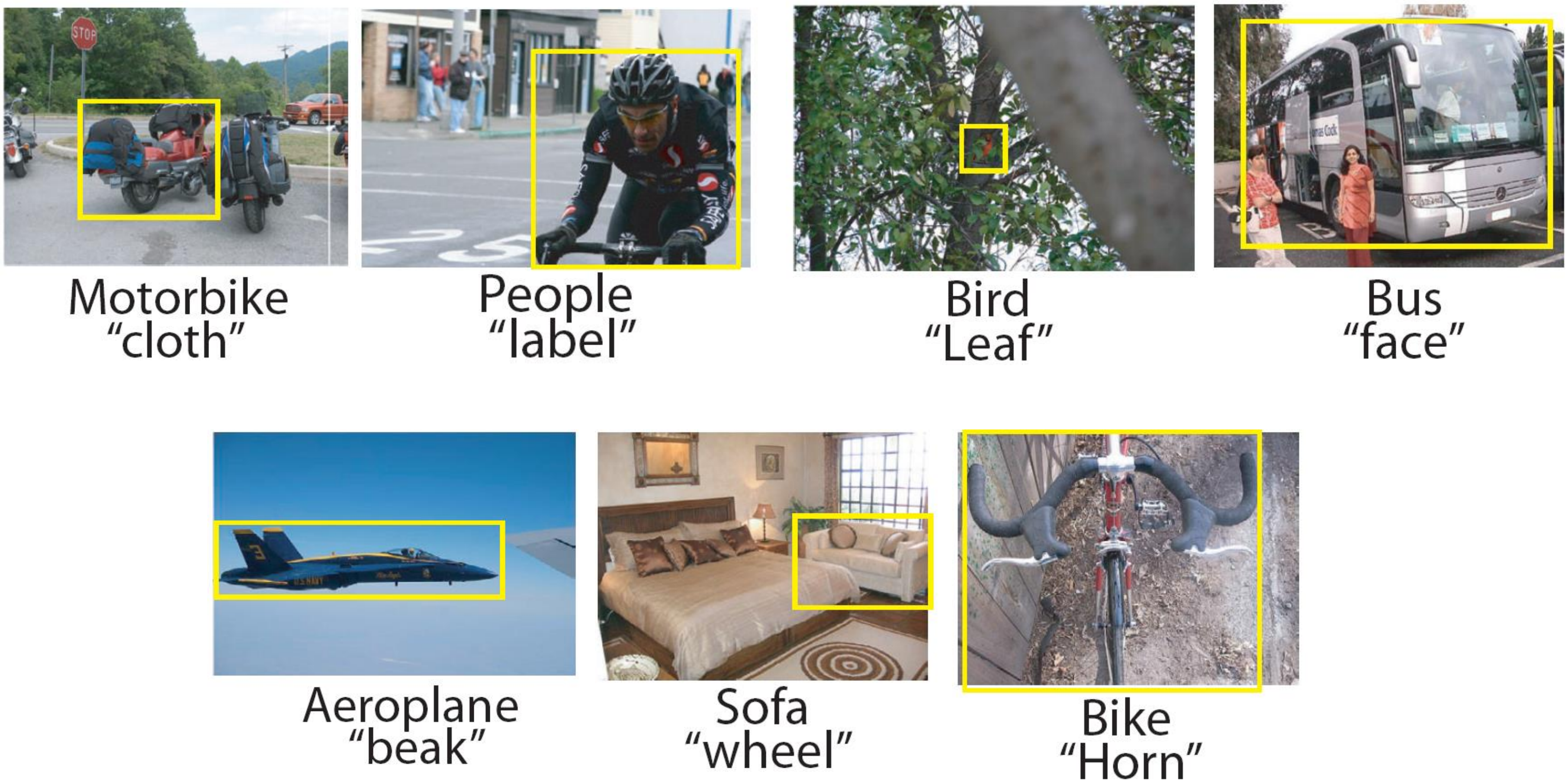
PASCAL 2008	Base Features	Semantic Attributes	All Attributes
Classification Accuracy	58.5%	54.6%	**59.4%**
Class-normalized Accuracy	35.5%	28.4%	**37.7%**

Absence of typical attributes

752 reports

68% are correct

Presence of atypical attributes



951 reports

47% are correct

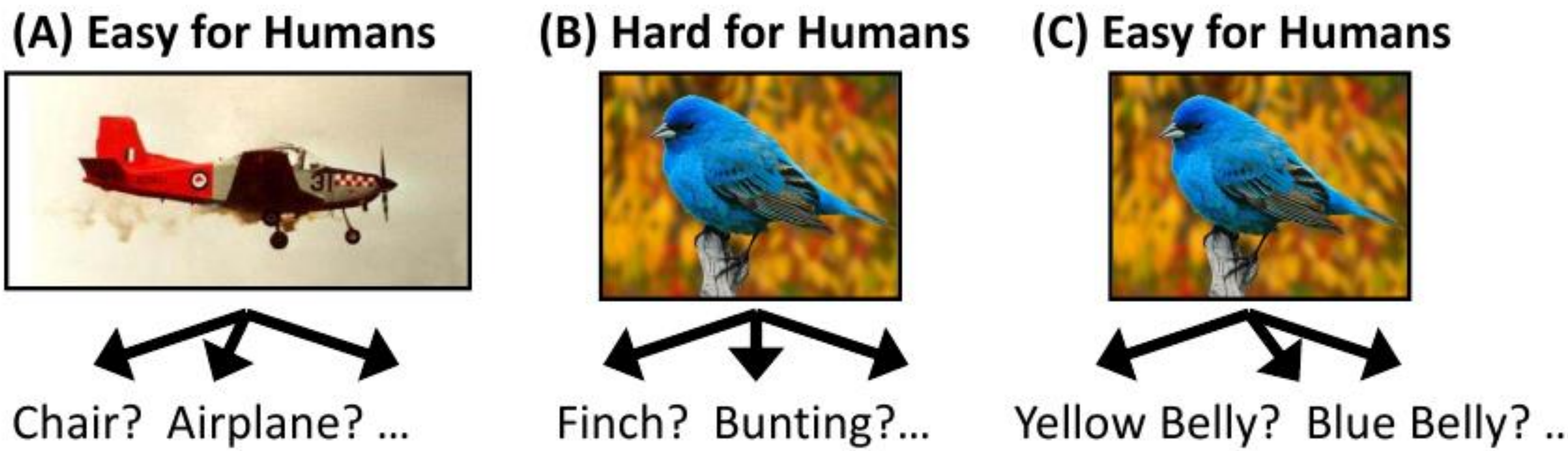
Visual Recognition with Humans in the Loop

Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, Serge Belongie

Part of the [Visipedia project]()

Slides from Brian O'Neil

Introduction:



Computers starting to get good at this.

If it's hard for humans, it's probably too hard for computers.

Semantic feature extraction difficult for computers.

Combine strengths to solve this problem.

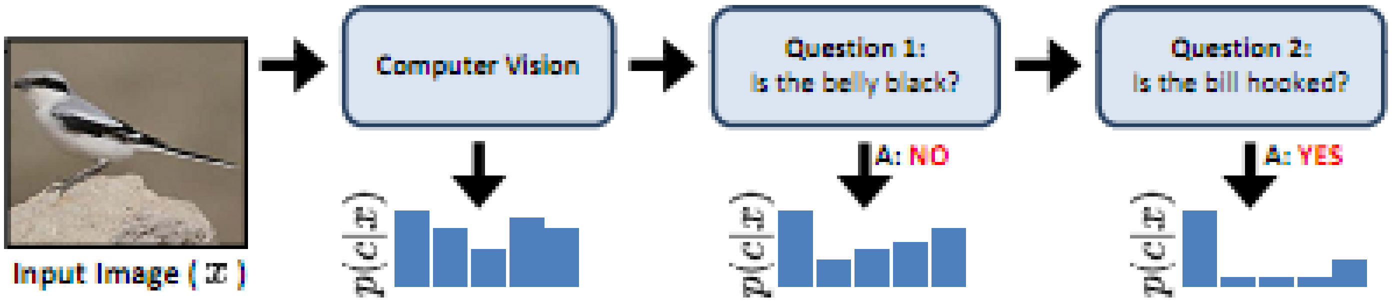
The Approach: What is progress?

- Supplement visual recognition with the human capacity for visual feature extraction to tackle difficult (fine-grained) recognition problems.
- Typical progress is viewed as increasing data difficulty while maintaining full autonomy

- Reduction in human effort on difficult data.

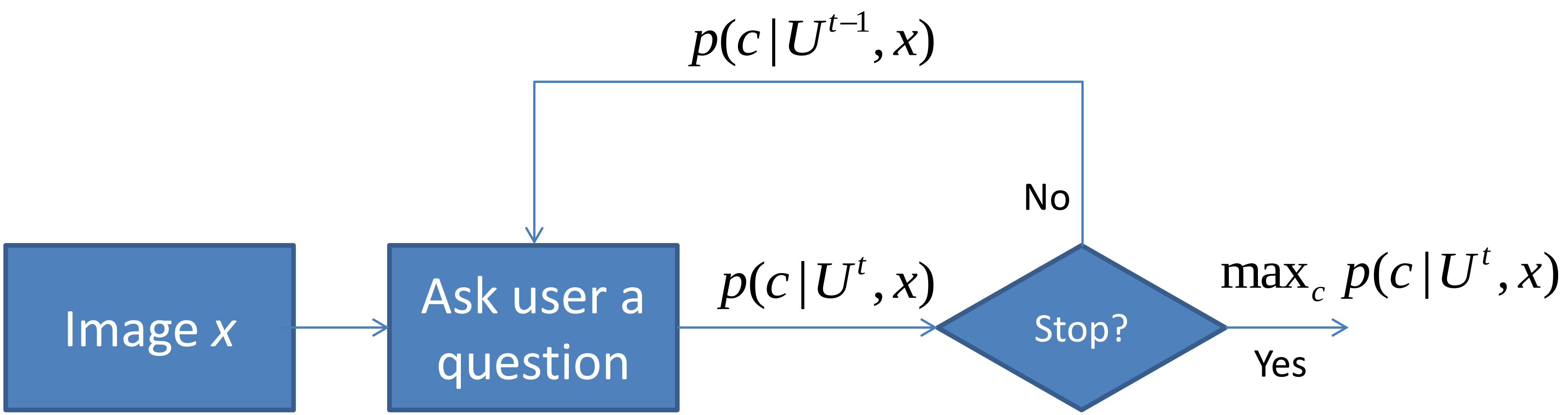
The Approach: 20 Questions

- Ask the user a series of discriminative visual questions to make the classification.



Which 20 questions?

- At each step, exploit the image itself and the user response history to select the most informative question to ask next.



Which question to ask?

- The question that will reduce entropy the most, taking into consideration the computer vision classifier confidences for each category.

The Dataset: Birds-200

- 6033 images of 200 species

Implementation

amazonmechanical turk

- Assembled 25 visual questions encompassing 288 visual attributes extracted from www.whatbird.com
- Mechanical Turk users asked to answer questions and provide confidence scores.

User Responses.

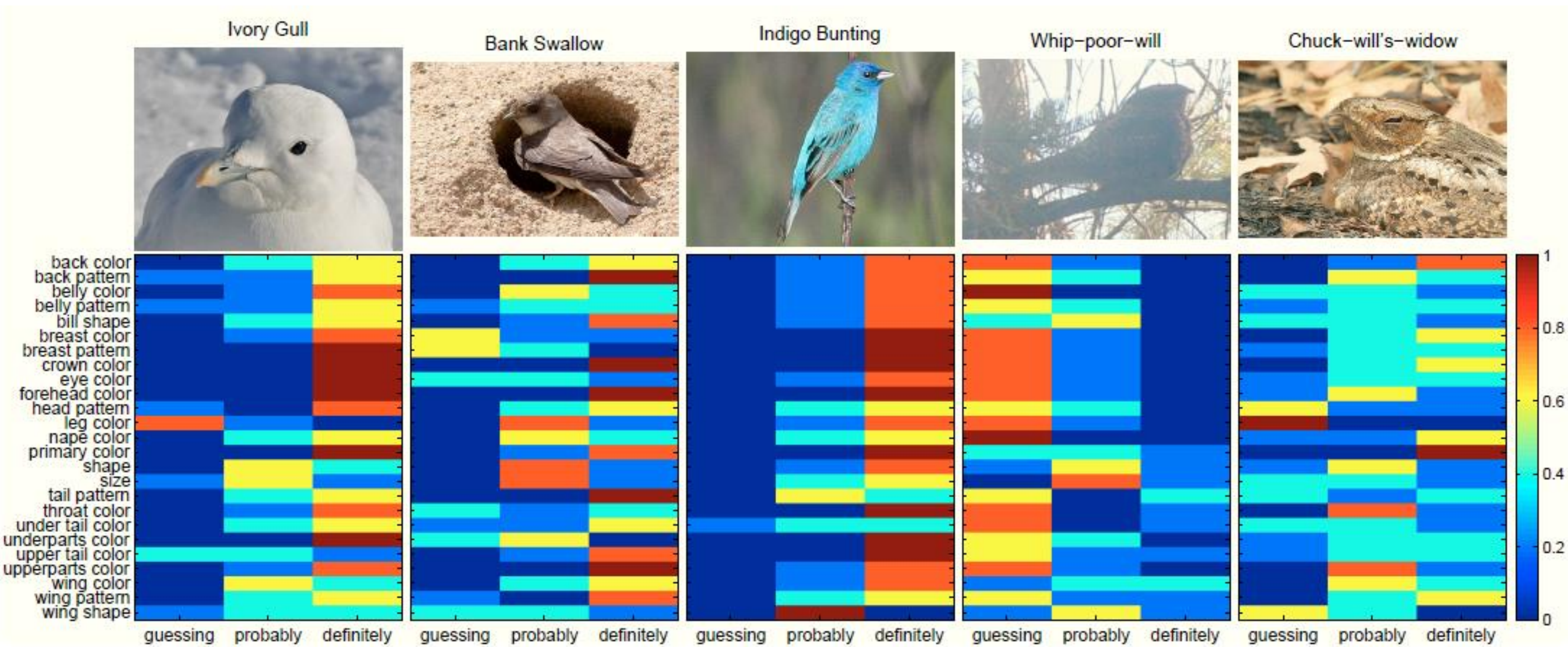
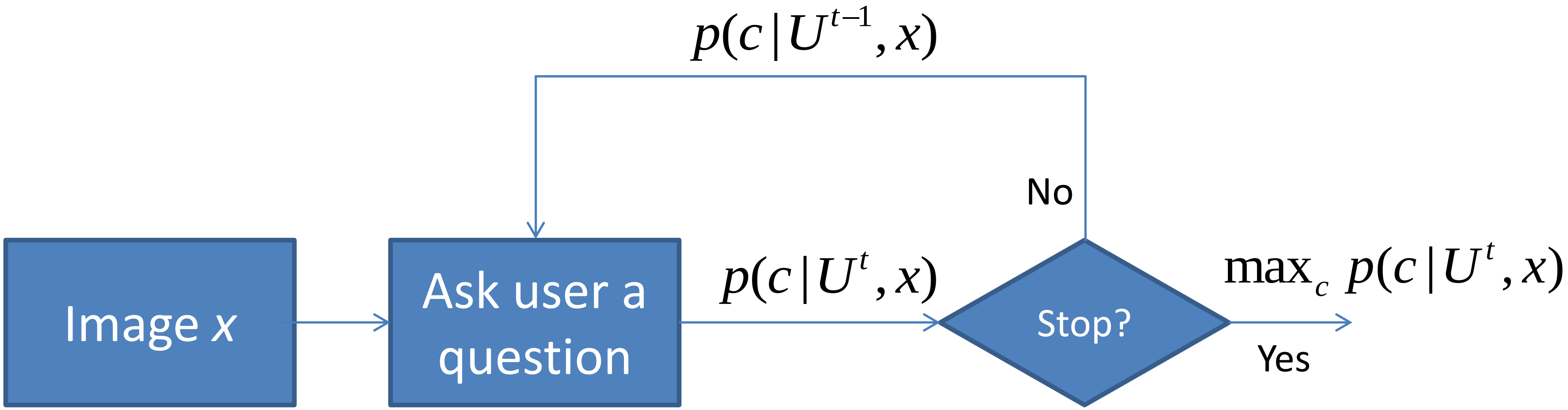


Fig. 4. Examples of user responses for each of the 25 attributes. The distribution over $\{Guessing, Probably, Definitely\}$ is color coded with blue denoting 0% and red denoting 100% of the five answers per image attribute pair.

Visual recognition

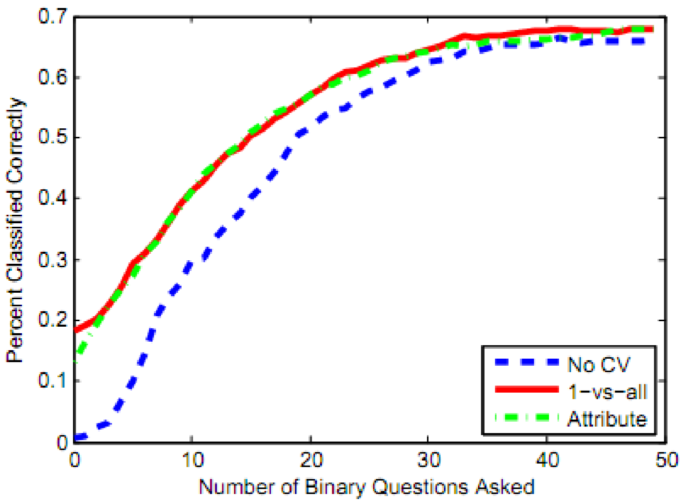
- Any vision system that can output a probability distribution across classes will work.
- Authors used Andrea Vedaldis's code.
  - Color/gray SIFT
  - VQ geometric blur
  - 1 v All SVM
- Authors added full image color histograms and VQ color histograms

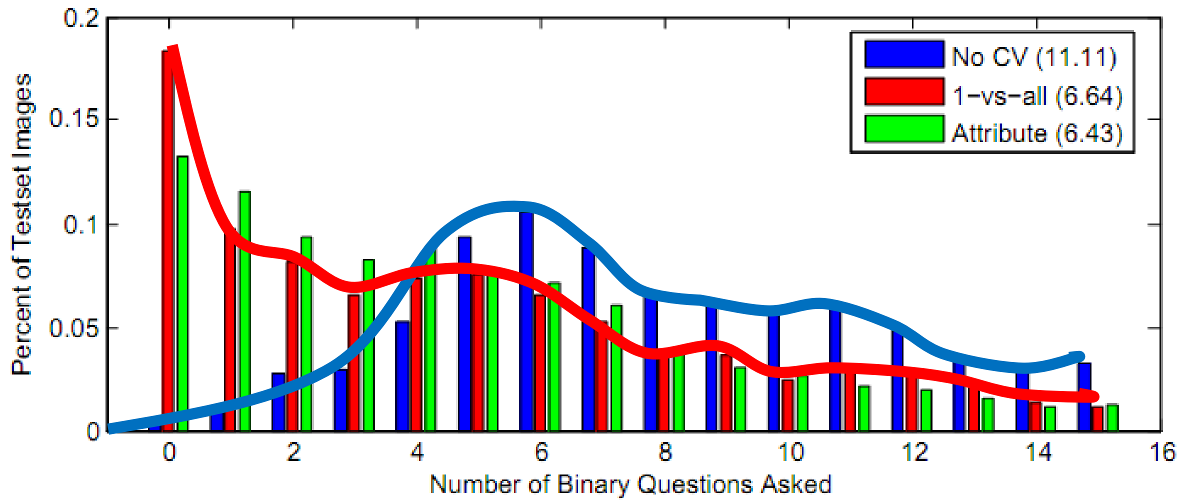
Experiments



- 2 Stop criteria:
  - Fixed number of questions – evaluate accuacy
  - User stops when bird identified – measure number of questions required.

Results





- Average number of questions to make ID reduced from 11.11 to 6.43
- Method allows CV to handle the easy cases, consulting with users only on the more difficult cases.

Key Observations

- Visual recognition reduces labor over a pure "20 Q" approach.
- Visual recognition improves performance over a pure "20 Q" approach. (69% vs 66%)
- User input dramatically improves recognition results. (66% vs 19%)

Strengths and weaknesses

- Handles very difficult data and yields excellent results.
- Plug-and-play with many recognition algorithms.
- Requires significant user assistance
- Reported results assume humans are perfect verifiers
- Is the reduction from 11 questions to 6 really that significant?