# Image Classification

D.A. Forsyth

University of Illinois at Urbana Champaign

# Core idea

- Label image by:
  - Insert image into encoder
  - apply multiclass classifier
- Variants:
  - complexity and training process for encoder
  - how encoder is trained
    - using labels from your dataset is one of many options
  - what kinds of label are predicted

- Right now, one label for the image
  - but the label might have quite complex structure…

# Types of label

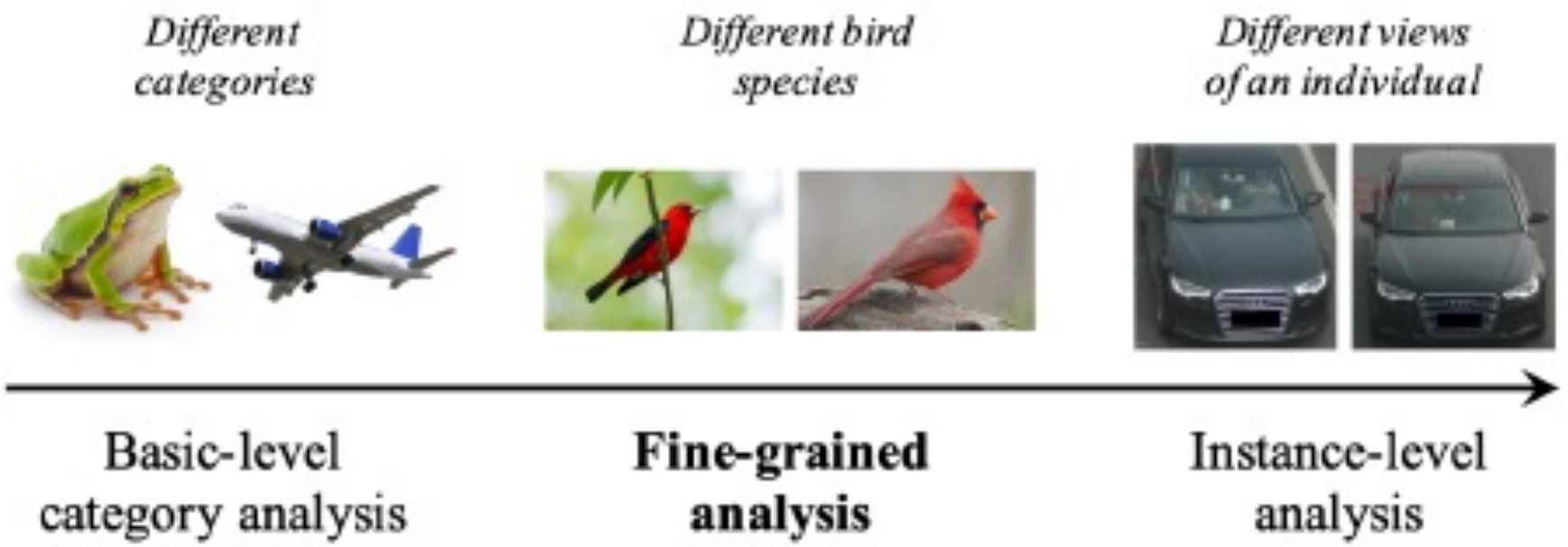
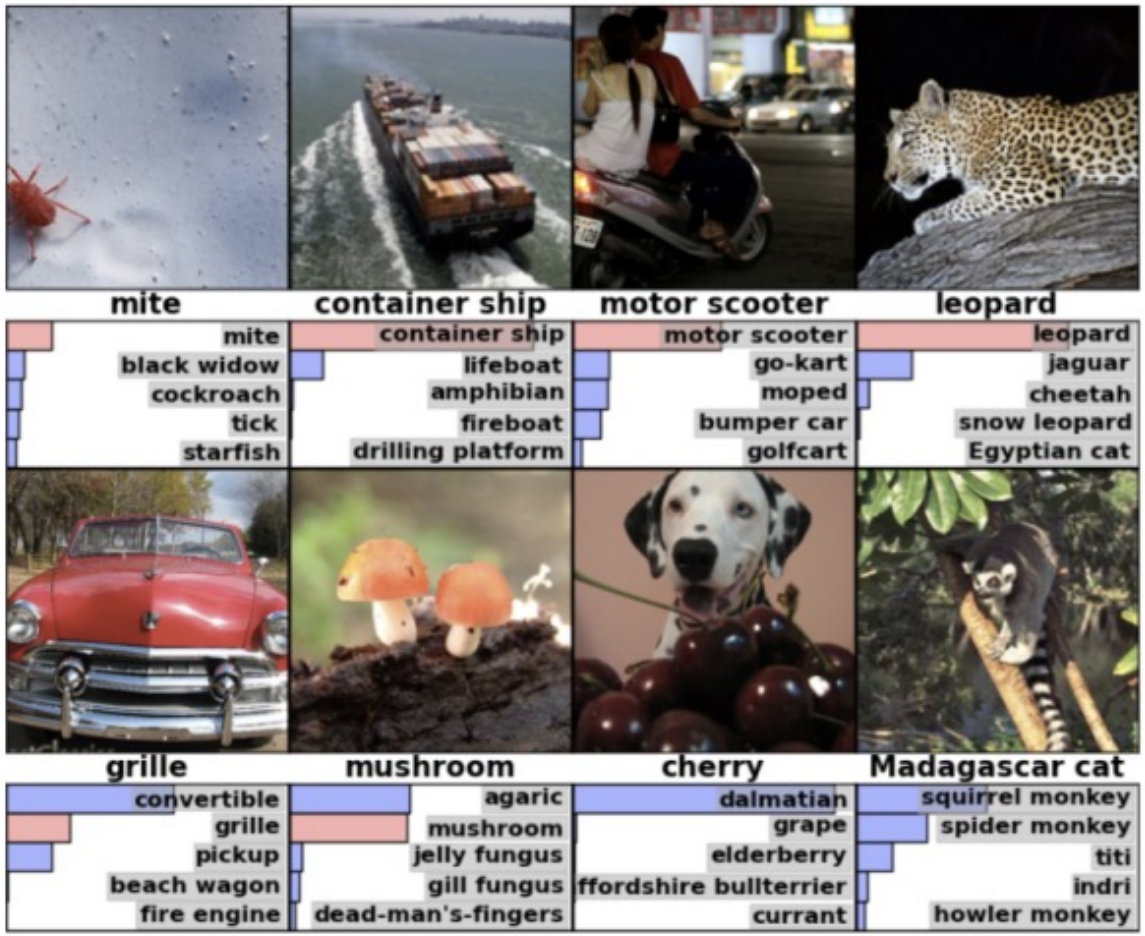



Image credit: Figure 3 of *Fine-Grained Image Analysis with Deep Learning: A Survey*

# Types of label:  category

- predict the category of the “main object”



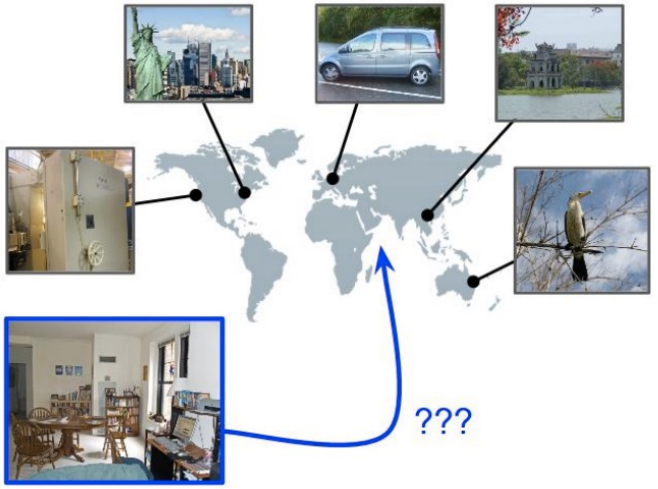
# Types of label: attribute


A.Farhadi, I. Endres, D. Hoiem, D.A. Forsyth, “Describing objects by their attributes”, CVPR 2009

# Other types of label…

Date prediction

Vittayakorn et al. (2017)




Vo et al. (2017)

# The encoder

- Architectures
  - ResNet
  - Transformer
  - Something else
- Training
  - Labels only
  - Pretrained and fine-tune

# Recipes for cases

- Cheap and cheerful
  - obtain small encoder
    - likely, small ResNet, possibly pre-trained on ImageNet
  - Train this on your labels
    - possibly fine-tuning encoder
      - small learning rate for encoder, bigger for linear classifier
- Luxury high performance
  - obtain large encoder
    - very likely transformer, heavily pretrained
  - fine-tune on your labels
- Large specialized
  - eg fine-grained recognition
  - architectures and processes quite varied

# Pretraining with labels

- Procedure:
  - minimize loss by descent
    - rich collection of training tricks, can’t do here
      - multi-GPU training; managing batch sizes; learning rate schedules;etc, etc.
- ImageNet
  - the Big Beast
    - 1000 categories, millions of training images
  - BUT
    - categories might not line up with what you want
    - there are always more unlabelled than labelled images
- TinyImageNet
  - 200 classes, 100K images, smaller images

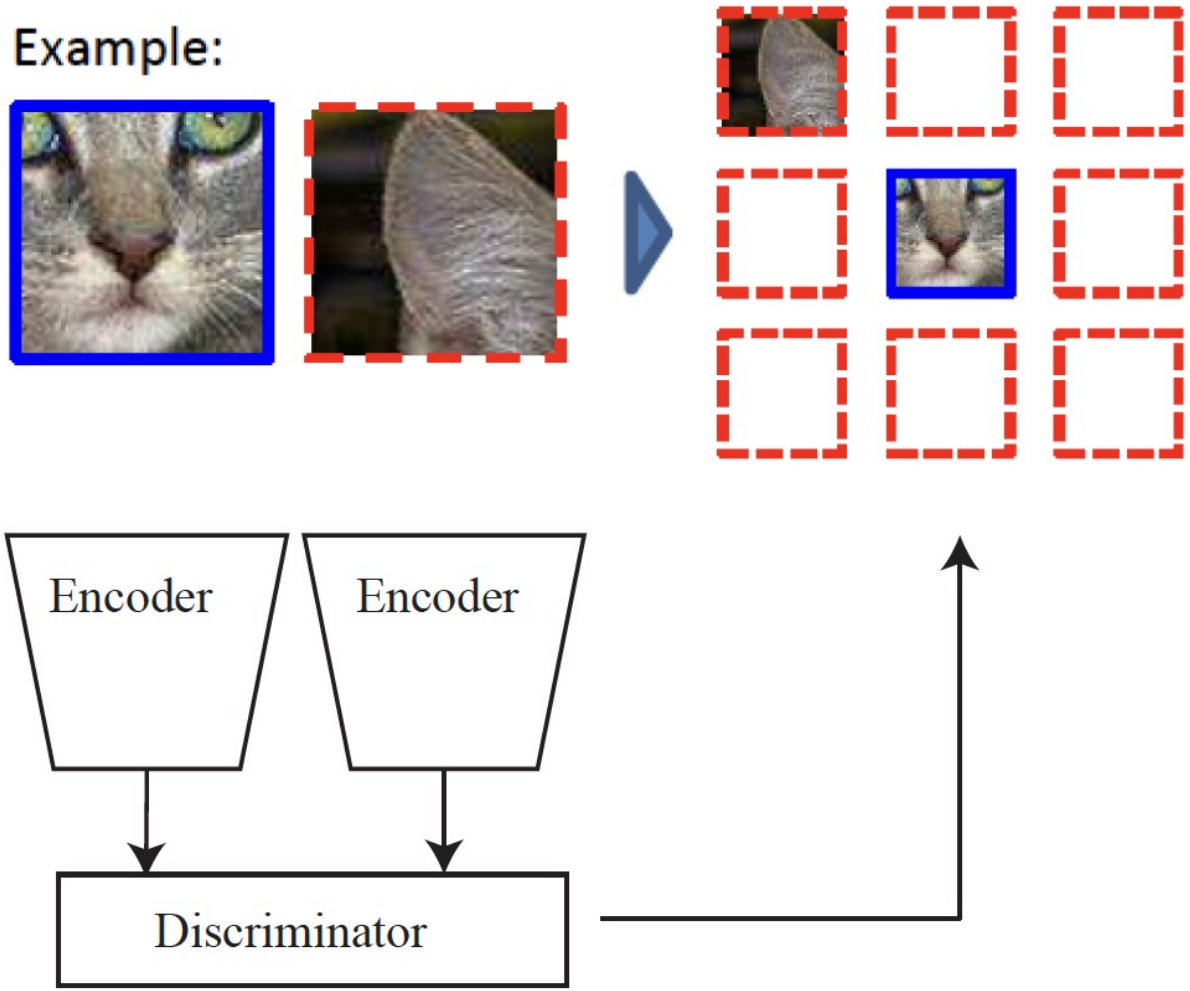
# Pretraining without labels

- Strategies:
  - Predicting location
  - Inpainting
  - Contrastive learning

# Predicting location

- Idea:
  - if you can predict relative location of two patches
    - you must have a good image encoding
- Procedure:
  - pass reference, query patch through encoder
  - predict location of query wrt reference using discriminator
    - manage a number of effects that lead to trivial solutions
      - chromatic aberration, etc.

# Predicting location




Part of Fig1 of *Unsupervised Visual Representation Learning by Context Prediction*

# Masked autoencoder

- Knock out high percentage of image patches
  - masked image
- train encoder/decoder to predict
  - image from masked image
  - drop decoder, keep encoder
- This requires a strong (transformer) encoder

# Masked autoencoder

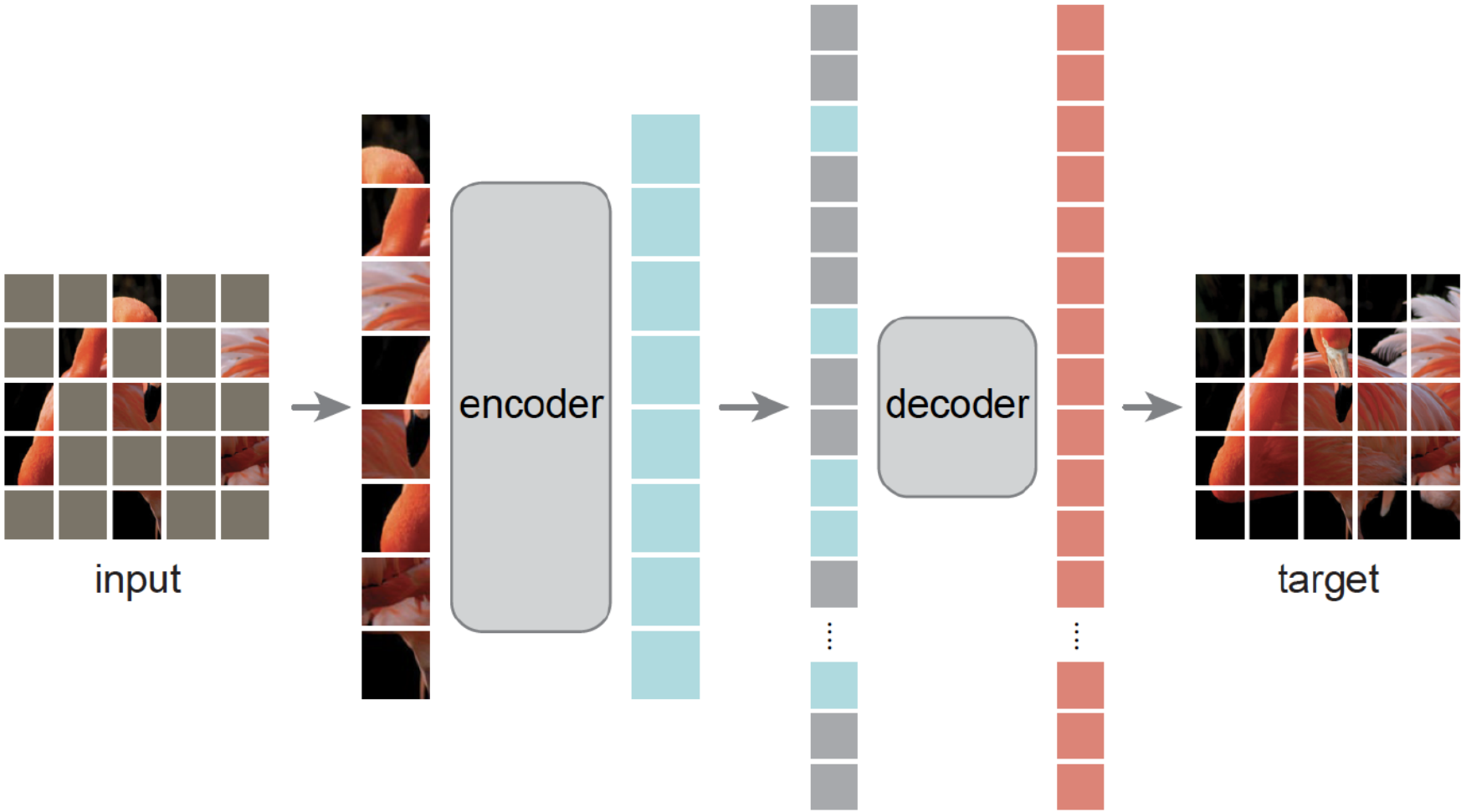



Figure 1 from *Masked Autoencoders Are Scalable Vision Learners*

# Pretraining: contrastive learning

- An encoder yields an image embedding
  - for this, use embeddings on to unit sphere
- Contrastive learning
  - (one of many variants!)
  - Exploit data augmentation
    - take image and
      - crop+resize; adjust colormap; etc
    - Adjust encoder so that
      - encode(A) and encode( augment(A)) are close
      - Images spread out
        - encode(A) and encode(B) are far
        - which is why we need the sphere

# Fine-grained recognition

- Applications:
  - identify birds
  - identify animals
  - identify parts
  - etc.
- General issue:
  - some categories are very similar
  - may need to know where to look to find the difference

# Fine grained category recognition

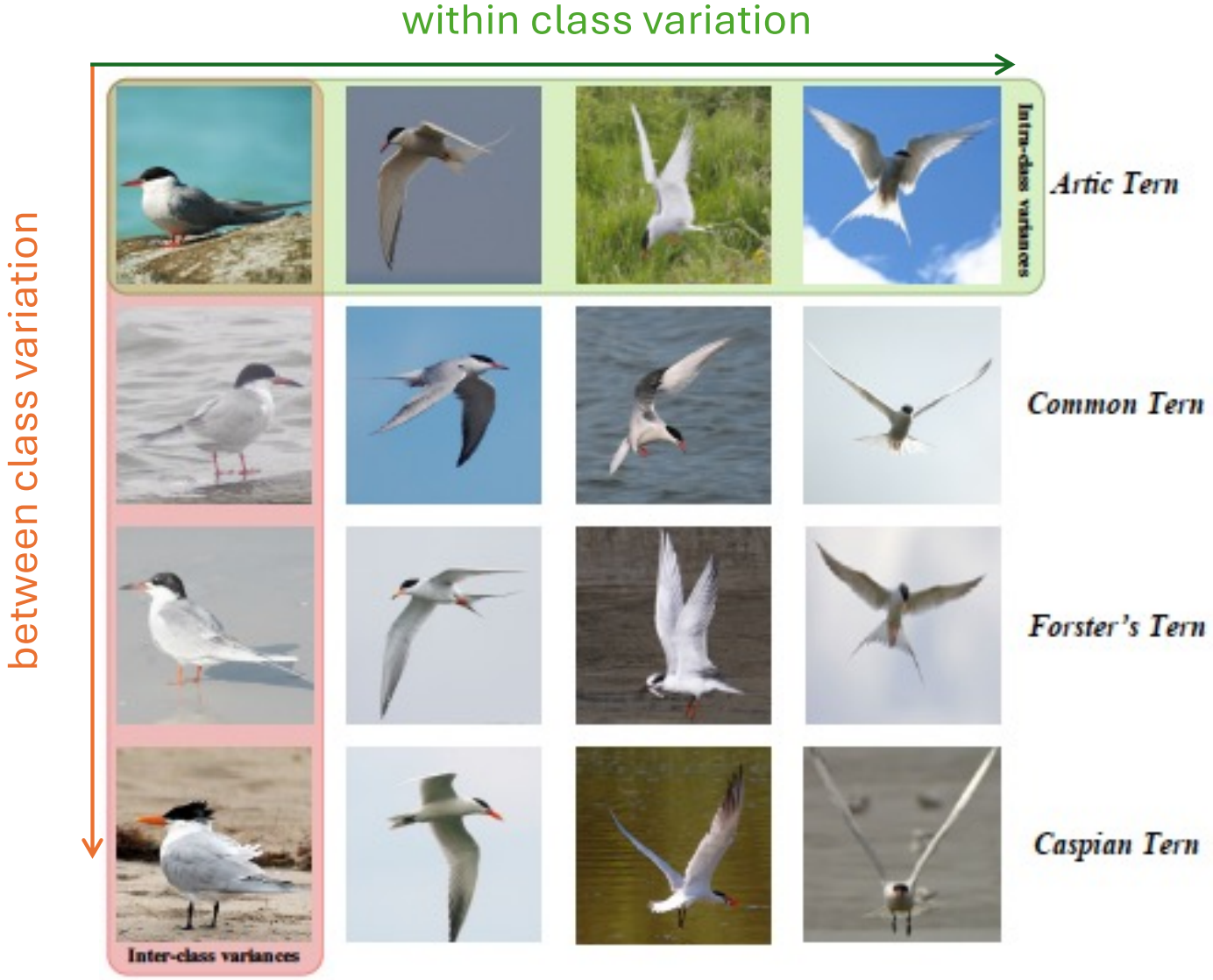
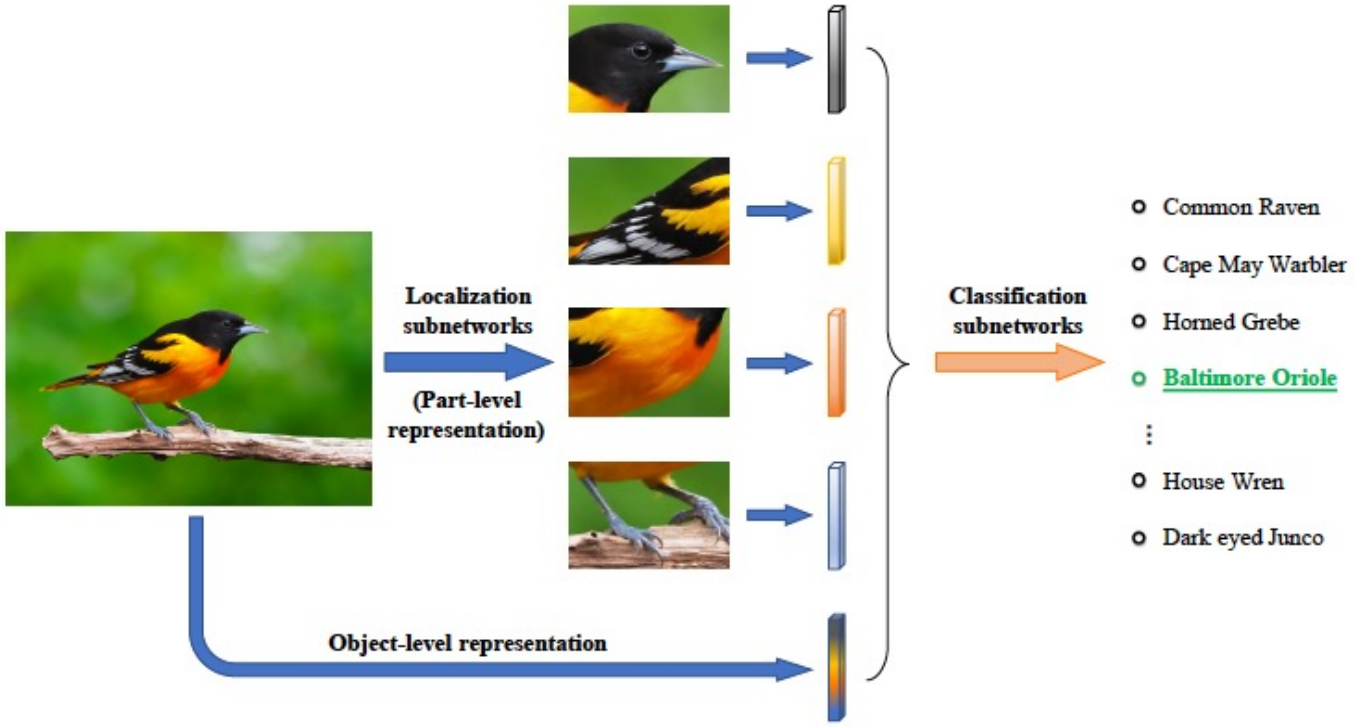



Image credit: Figure 4 of *Fine-Grained Image Analysis with Deep Learning: A Survey*

# Strategy: localize then describe




- Issue:
  - fierce demands on labelled data

Image credit: Figure 8 of *Fine-Grained Image Analysis with Deep Learning: A Survey*

# Strategy: higher order features

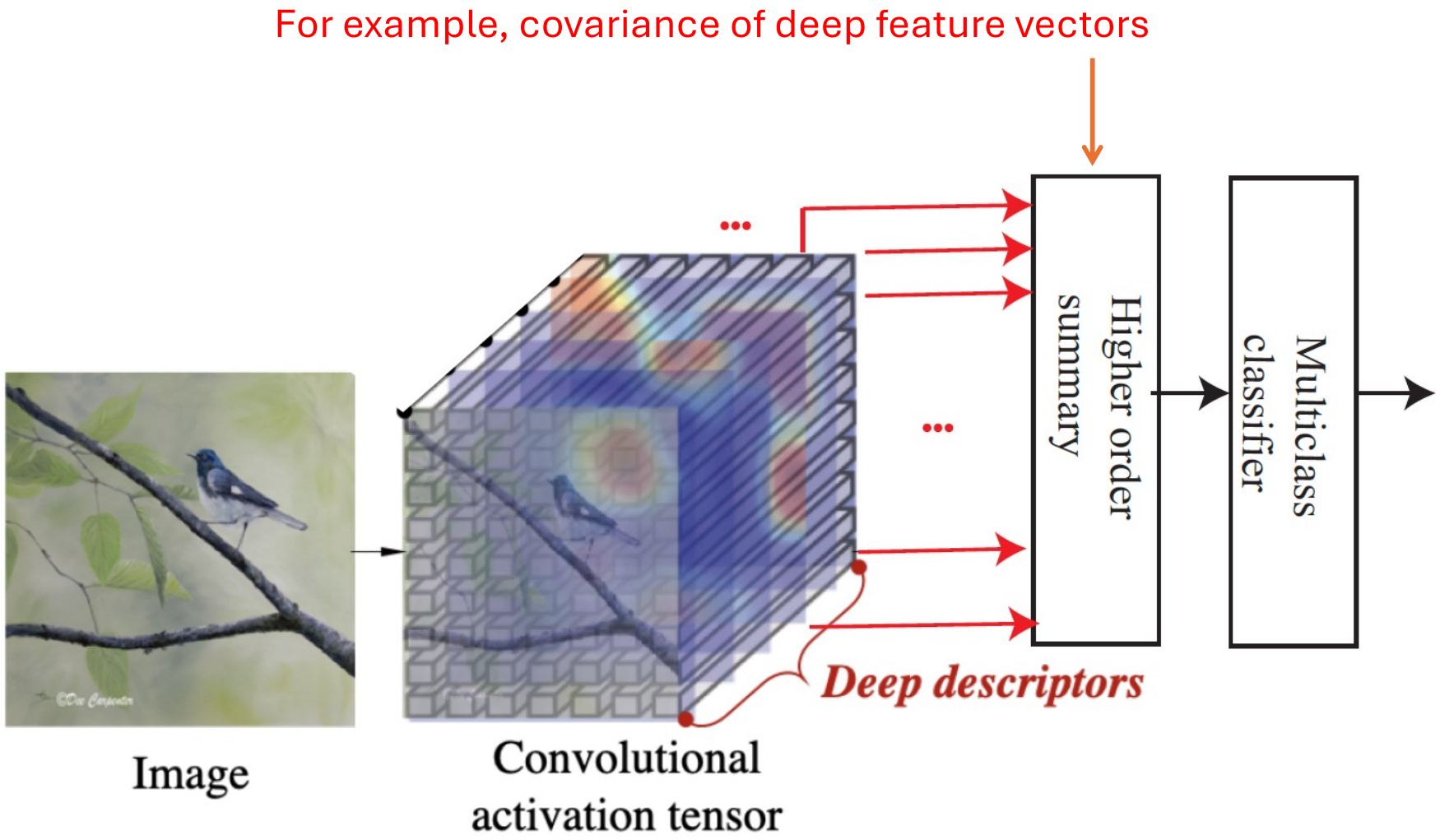



Image credit: uses Fig 9 of *Fine-Grained Image Analysis with Deep Learning: A Survey*

# SOTA - rough summary

- Very high accuracy with 1000's of classes
  - Using
    - very deep residual networks
    - alternative feature construction methods
- Challenges
  - tough with little training data
    - use a pretrained encoder and a classifier
    - link to language (later)
  - sufficient change in dataset presents problems