Quick Introduction to Biological Modeling with Keras

JONATHAN SCHNEIDER

YORK UNIVERSITY

(PREVIOUSLY: POST DOC @ LEVINE LAB (UOFT) + TAYLOR LAB (GUELPH)

Proposed Schedule

- Today:
  - Quick Overview of Deep Learning
    - Basic concepts
  - Case study of using deep learning to learn something about fruit flies
  - Toy examples
- Saturday:
  - Group discussions about applications of deep learning
- Saturday/Sunday:
  - Playing around with toy networks (hopefully) applicable to your studies

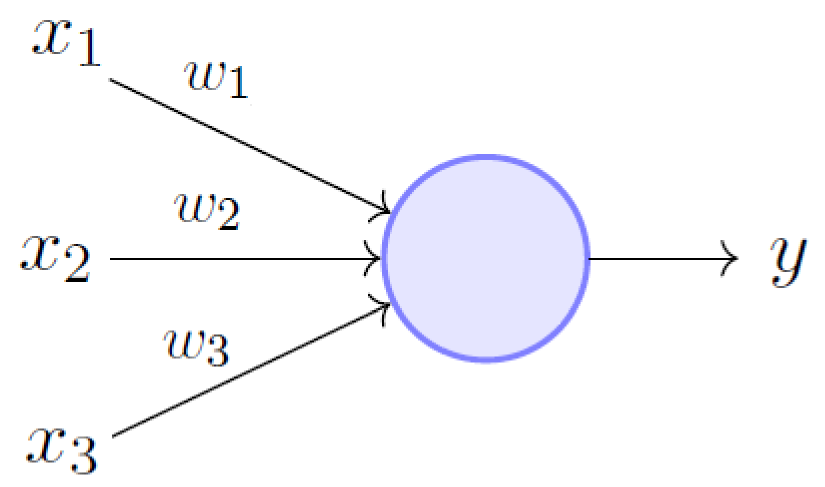
The Basics of Artificial Neural Networks

- Building blocks:
  - Neurons (activations and biases)
  - Multi-Layer Perceptrons (MLPs)
  - Cost Functions
  - Optimizers
  - Deeper Architectures (MLPs, Convolutions and Skip Connections, autoencoders)

[Coursera] Neural Networks for Machine Learning — Geoffrey Hinton
https://www.youtube.com/playlist?list=PLoRl3Ht4JOcdU872GhiYWf6jwrk_SNhz9

Perceptron

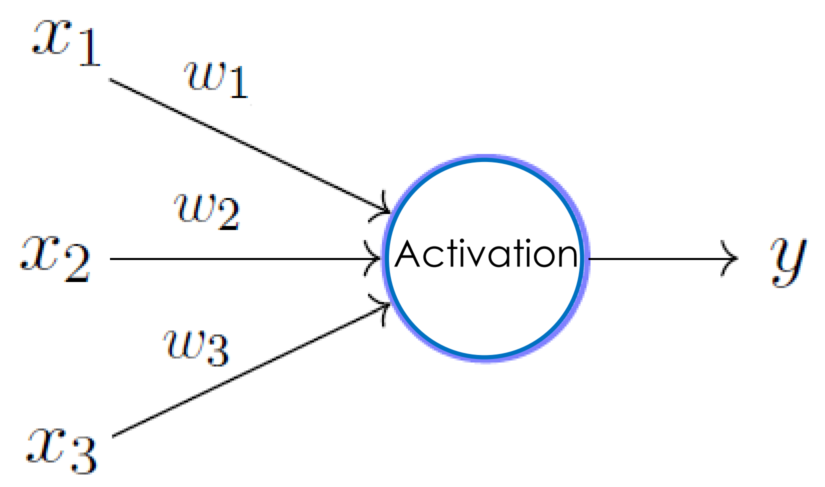
- Weighted inputs -> Output



Perceptron Model (Minsky-Papert in 1969)

Perceptron

- Weighted inputs -> Output



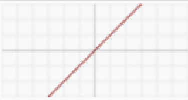
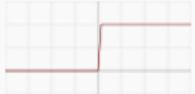
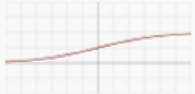
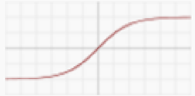
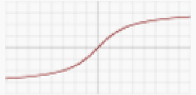
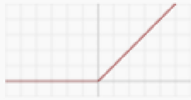
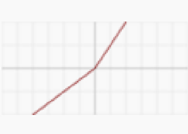
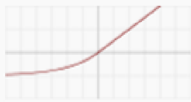
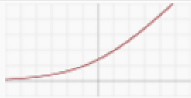
Perceptron Model (Minsky-Papert in 1969)

Activation Functions

- Virtually any function.
- Popular functions:
  - Linear (not useful in deep nets)
  - Sigmoid (good for probabilities since it ranges from 0->1)
    - Includes softmax
  - Tanh (Sigmoidal from -1 to 1)
  - ReLUs (and variants)
    - Non-linear
    - Sparse activation

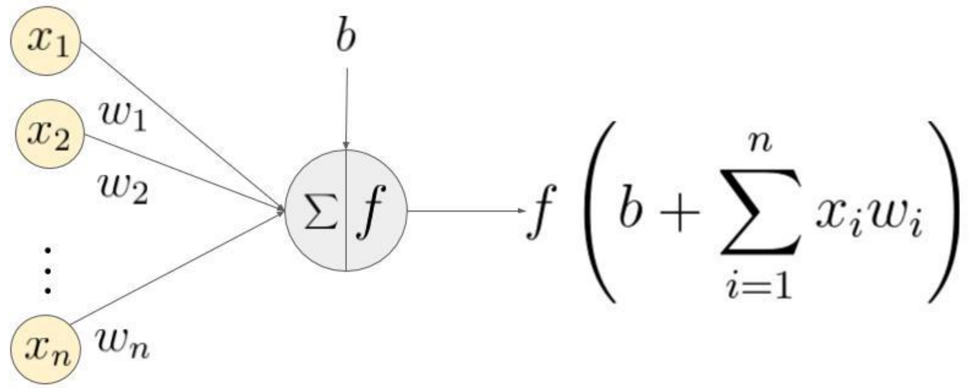
- Cheat Sheet

https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Artificial Neuron

- Can have any activation function
- Can have bias

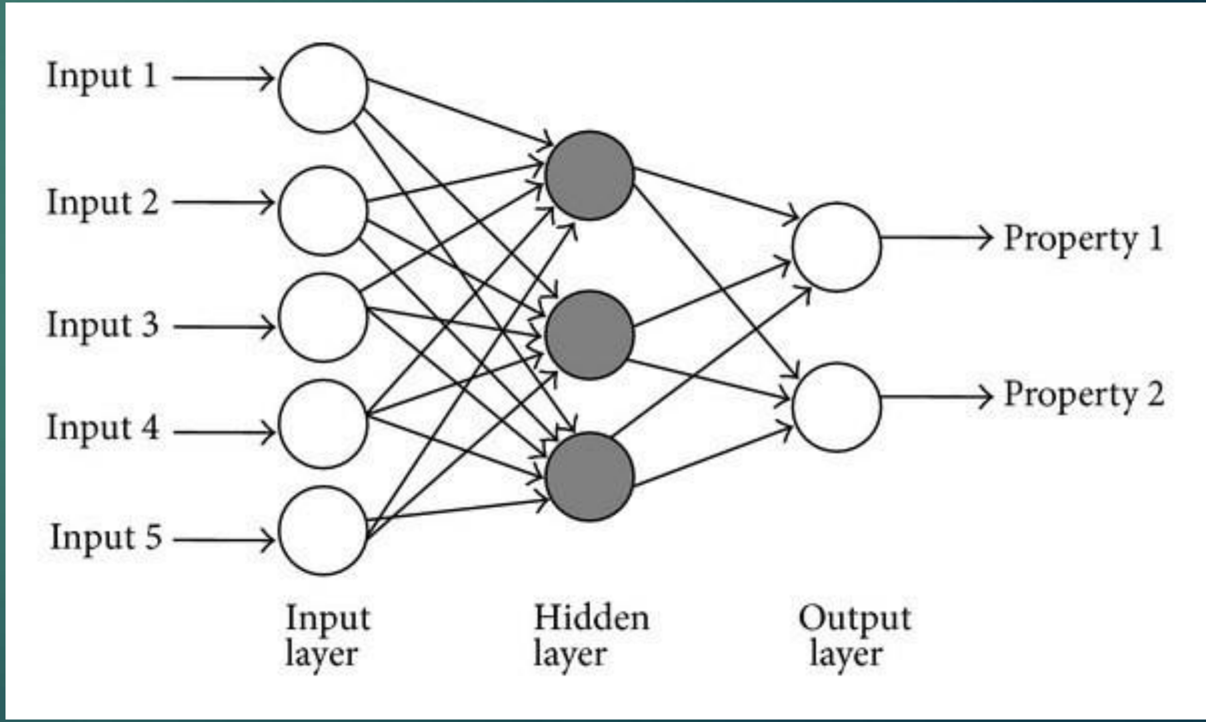


An example of a neuron showing the input (x_1 - x_n), their corresponding weights (w_1 - w_n), a bias (b) and the activation function f applied to the weighted sum of the inputs.

https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/

Multi Layer Perceptrons

- What happens when you link up a bunch of neurons in layers?
- What is it about ‘deep’ networks?



https://medium.com/pankajmathur/a-simple-multilayer-perceptron-with-tensorflow-3effe7bf3466

A Non-Linear Function Approximator

- We pass inputs into our network
- Our network computes each layers output
- We get an initial guess based on our initialization

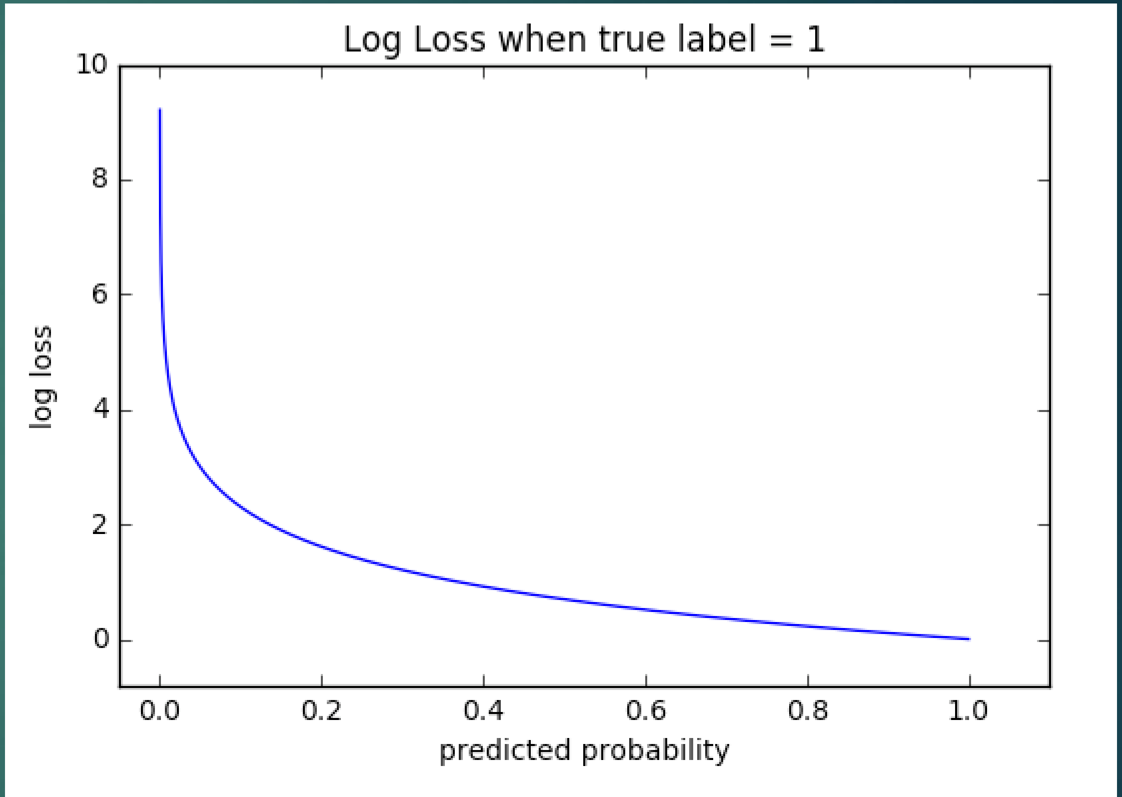
- … So how do we get it to get better?

Cost Functions

- How well is our network matching the desired output?
- Many different cost functions for different task:
  - Regression:
    - Absolute Error (more for diagnostics and evaluation)
    - Mean Squared Error
  - Categorization:
    - Cross Entropy

What Makes a Good Loss Function?

- Needs to be appropriate for the task
- Can be beneficial to give greater error the more 'wrong' the network is.
- Needs to be differentiable



https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

Back-Propagation

- We want to know how to train the network
  - How do we adjust the weights and biases to give better (i.e. closer to our desired) output?
- We need to know how the error changes in respect to a given weight
  - We need to compute the partial derivative of the loss
  - We need to change the weights/biases to decrease the loss

Great walkthrough example can be found here: https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How do we Update the Weights?

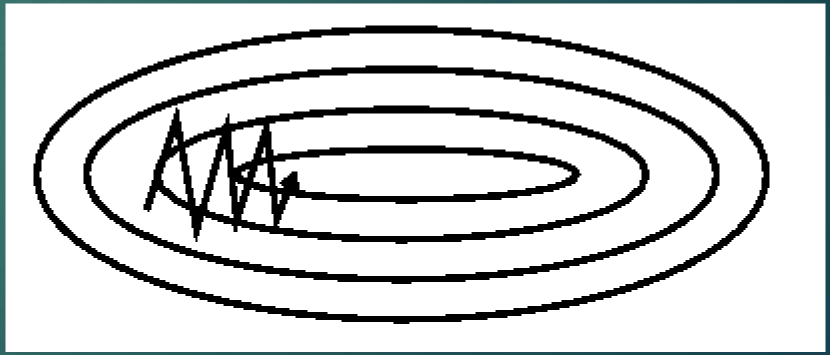
- Choose an optimizer:
  - SDG (Stochastic Gradient Descent)
  - Adam
  - RMSProp

Gradient Descent

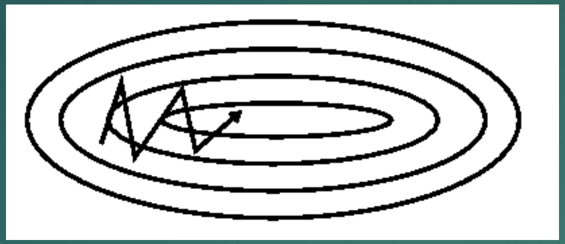
- Batch Gradient descent:
  - Compute the loss and derivatives over the entire dataset and perform one update
- Gradient descent:
  - Compute the loss for each example and perform one update
- Mini-Batch:
  - Compute and update over a batch of examples

Gradient Descent

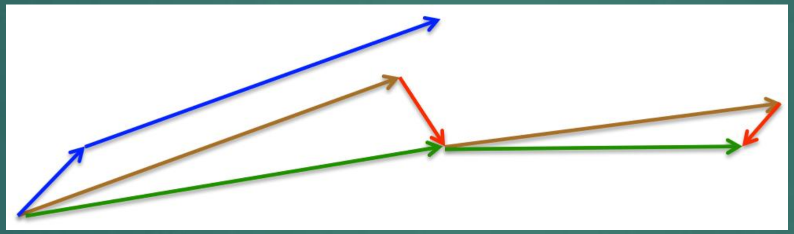
- Learning rate parameter defines how big a step we take in adjusting the network parameters.
  - Too high and the learning becomes difficult (possibly divergent)
  - Too low and learning is slow
- Can still get stuck in ‘local optima’



Extensions to Gradient Descent



- Momentum
- Nesterov Accelerated Momentum



- Special
  - Adagrad (different learning rate for each parameter)
  - AdaDelta/RMSProp (augments Adagrad)
  - Adam (Adaptive Moment Estimation)

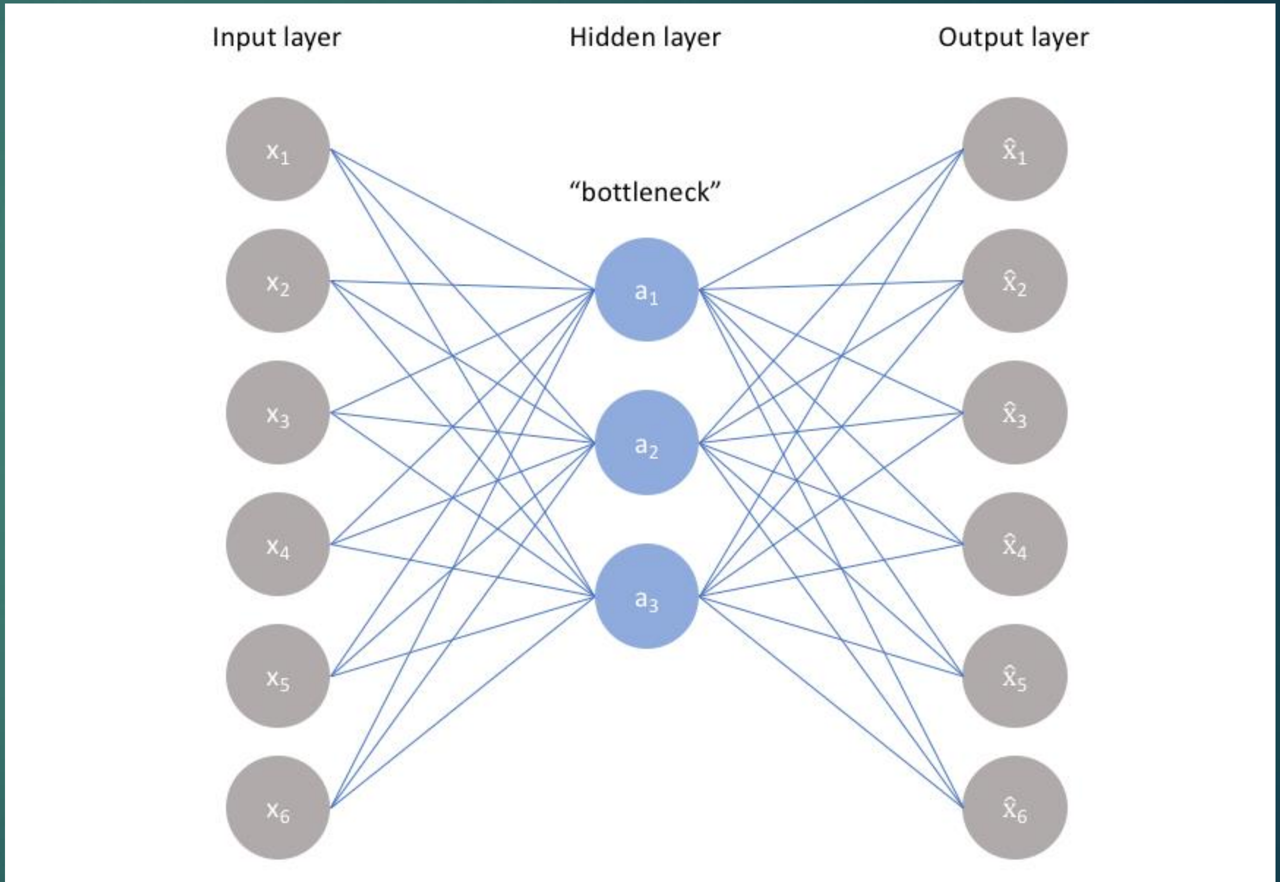
Great summary of these and more:
http://ruder.io/optimizing-gradient-descent/

Putting it All Together

- So how do we recognize an image:
  - Stack some Neurons (with chosen activations and biases)
  - Choose our Loss and Optimizer
  - Choose our hyperparameters (learning rate)
  - Train it with or without labelled data…

Auto-Encoder

- Unsupervised Training
- Several Types:
  - Sparse (penalize activations)
  - De-Noising

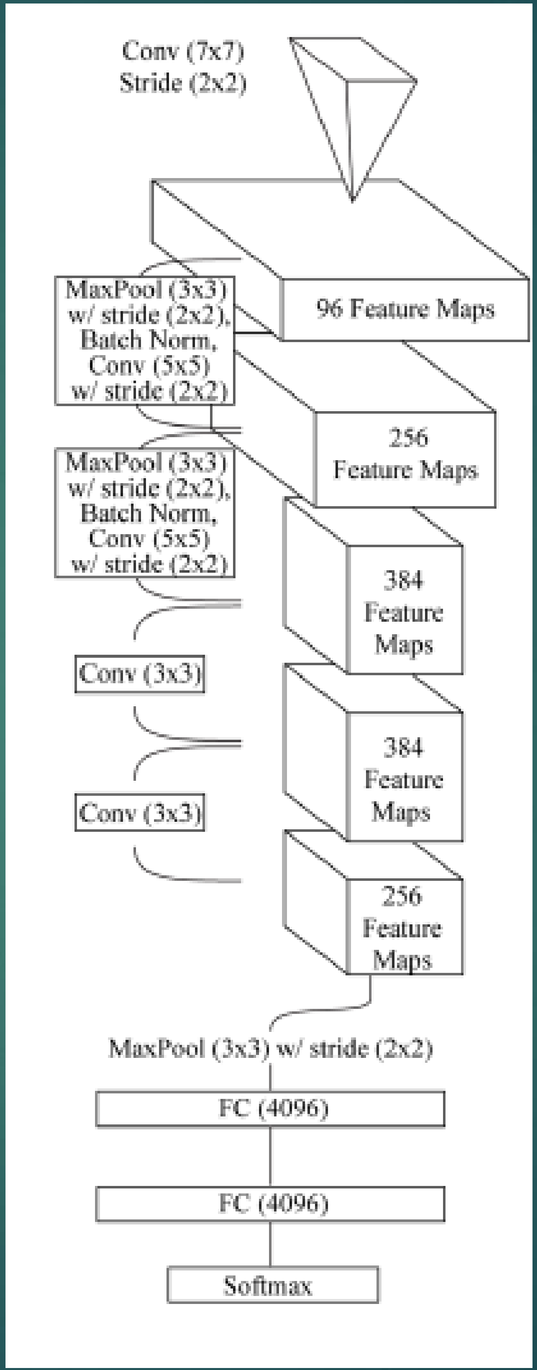


https://www.jeremyjordan.me/autoencoders/

Building Translation Invariance into the Network

- We want to be able to recognize patterns wherever they may be in the input.
- One way is to learn pattern filters that are moved across the input
  - Convolutional Networks

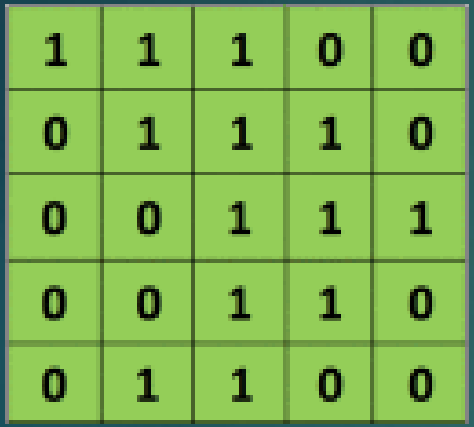
Example Convnet



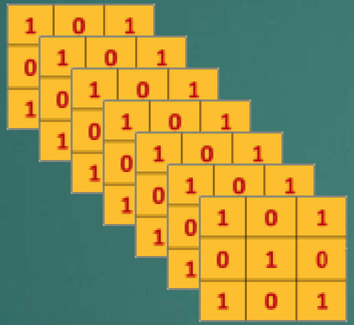
https://www.clarifai.com/technology

Filters

Image

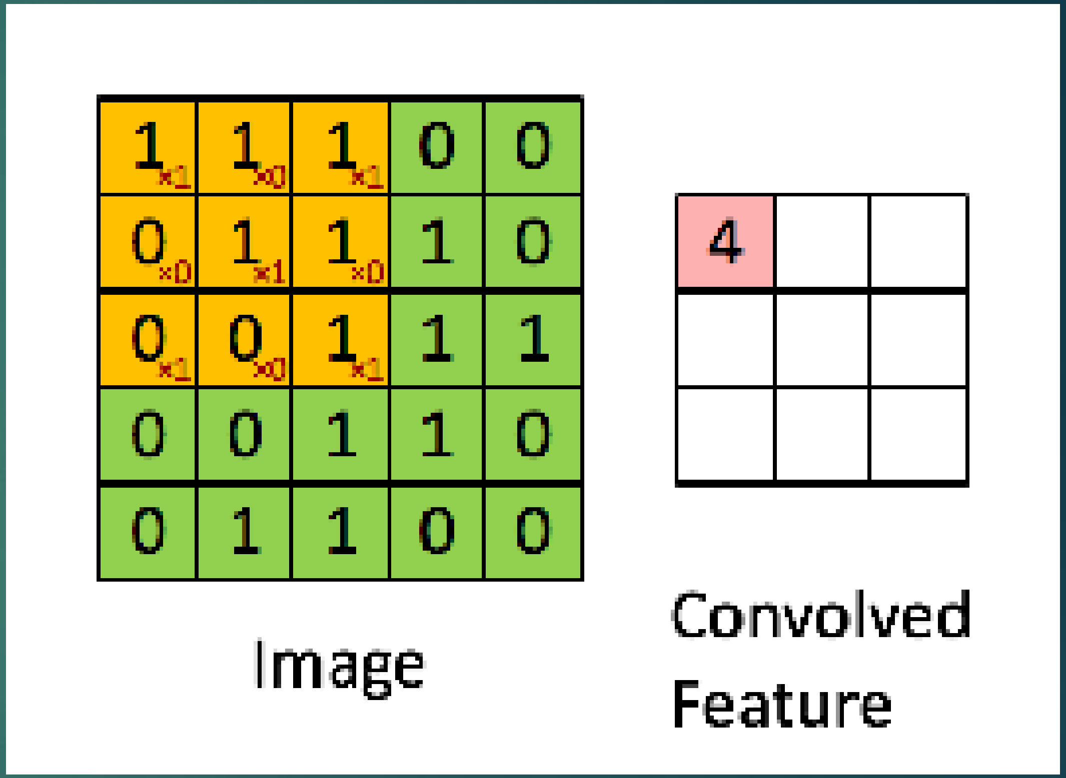


Filter (one of many)



(also called kernels)

Convolved element-wise multiplication



http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

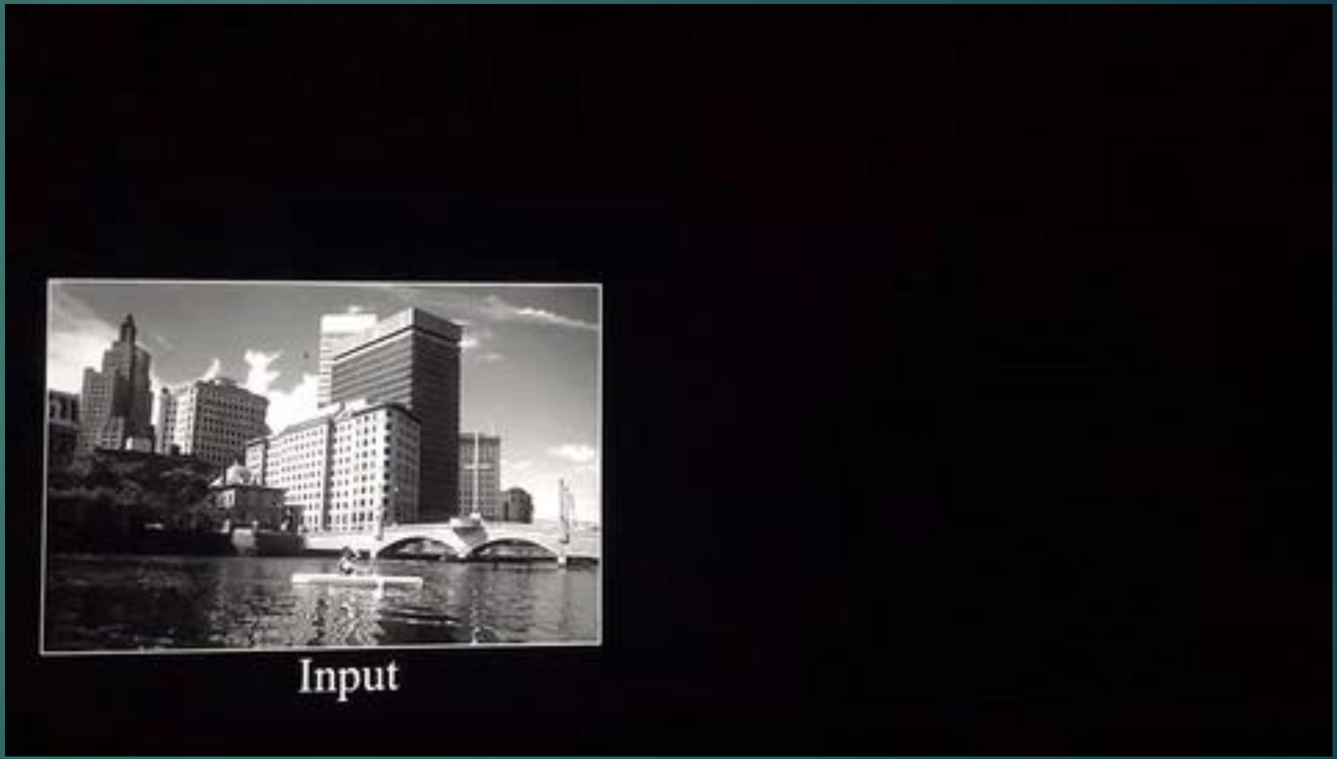
Feature 'Maps'

Some low-level examples:

- Edge detectors
- Corner detectors
- Dot detectors

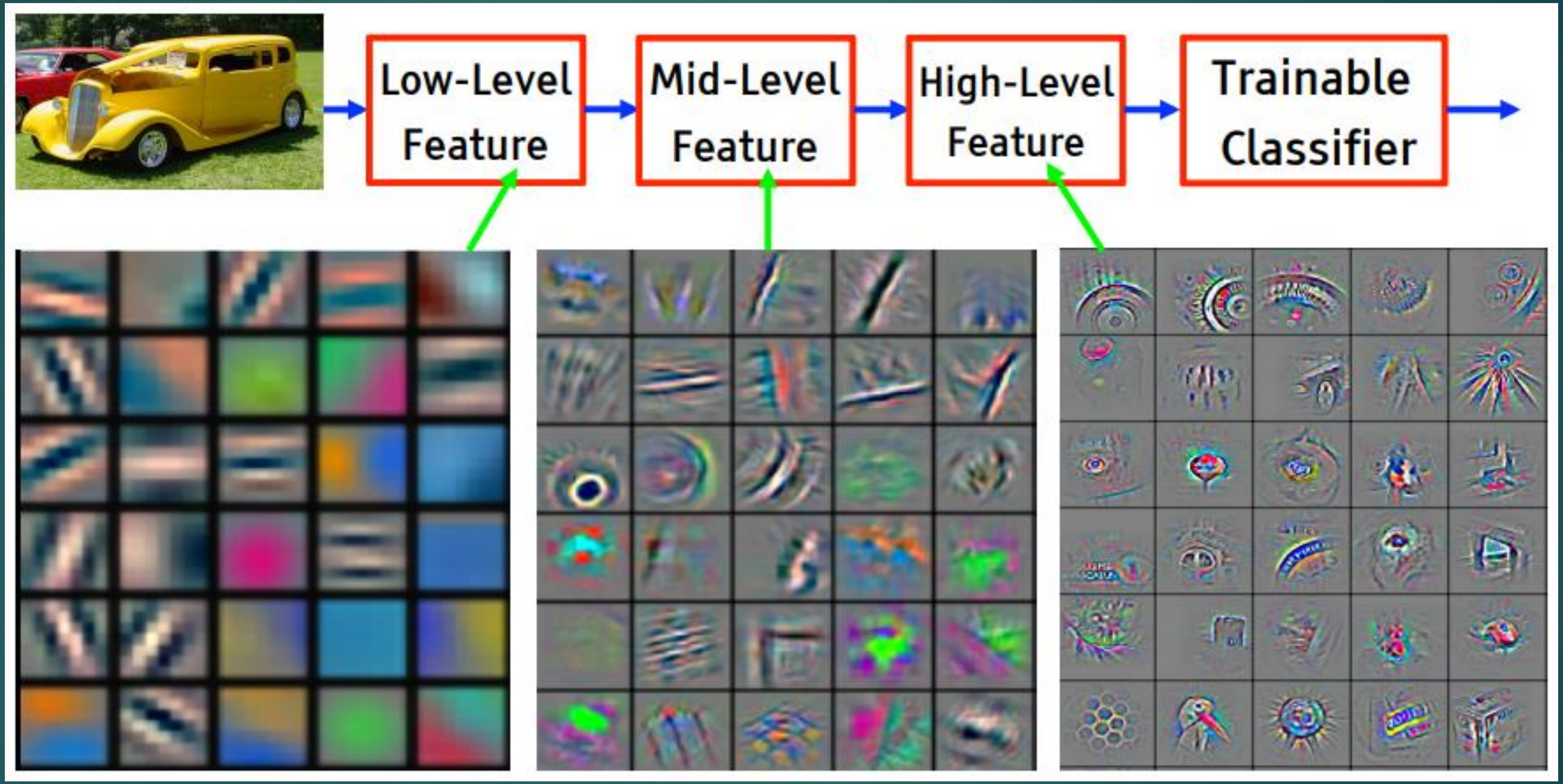
Some high-level examples:

- Eye detectors
- Face detectors



https://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/

Filters->Features->Classifiers



http://www.iro.umontreal.ca/~bengioy/talks/DL-Tutorial-NIPS2015.pdf

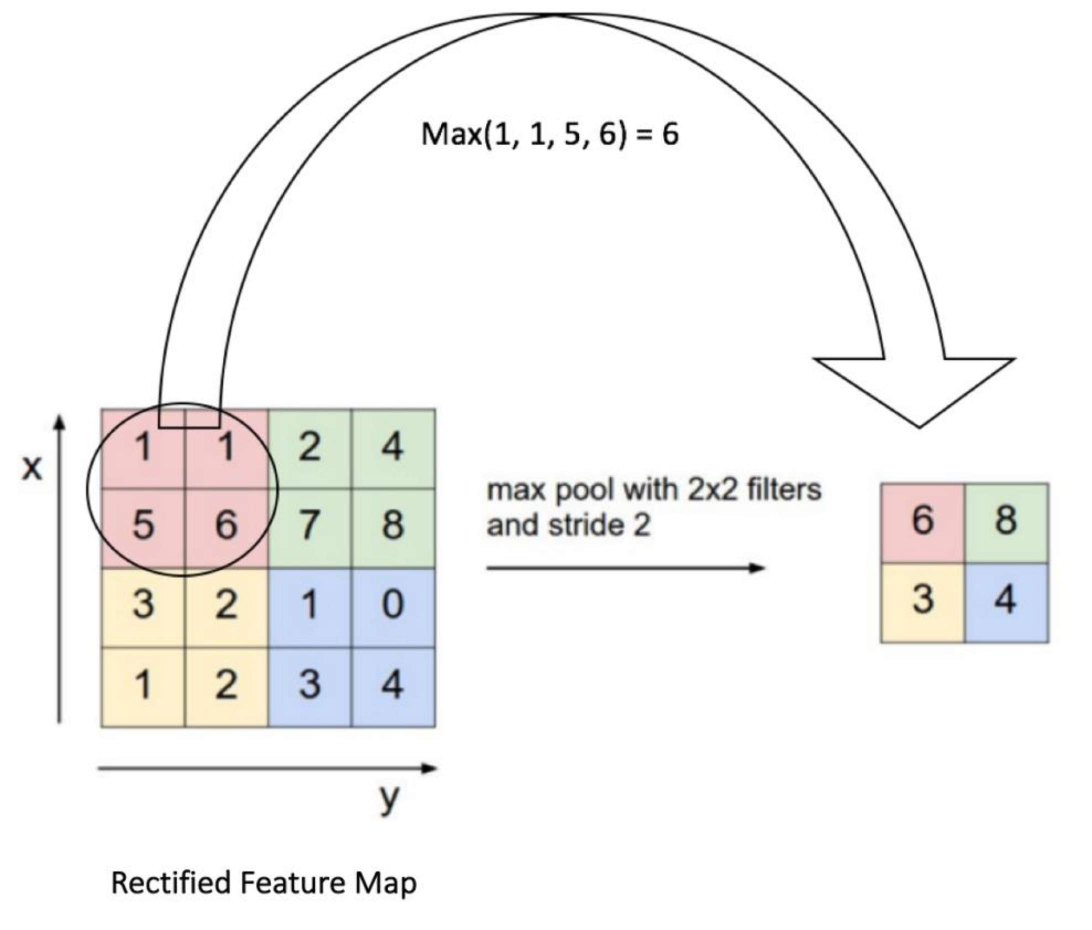
‘Pooling’ features

Two widely used types:

- Max pool
- Average pool

Help with ‘smoothing’ locations of features in a feature map.

Deals with lateral shift or skew.



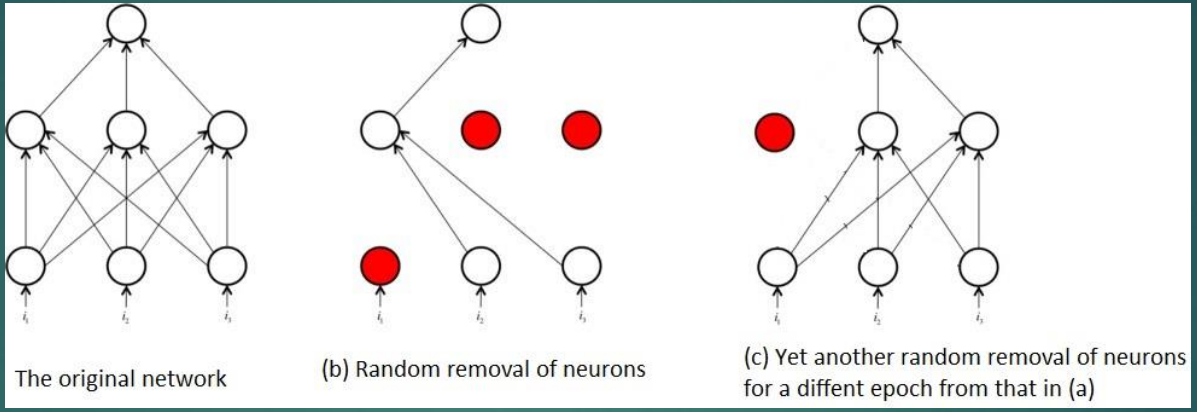
http://cs231n.github.io/convolutional-networks/

Tricks to Help Learning and Avoid Vanishing Gradients

- Input Level:
  - Normalization
  - Data Augmentation
- Network Level:
  - ‘Drop-out’
  - ‘Batch-Norm’
  - Skip Connections

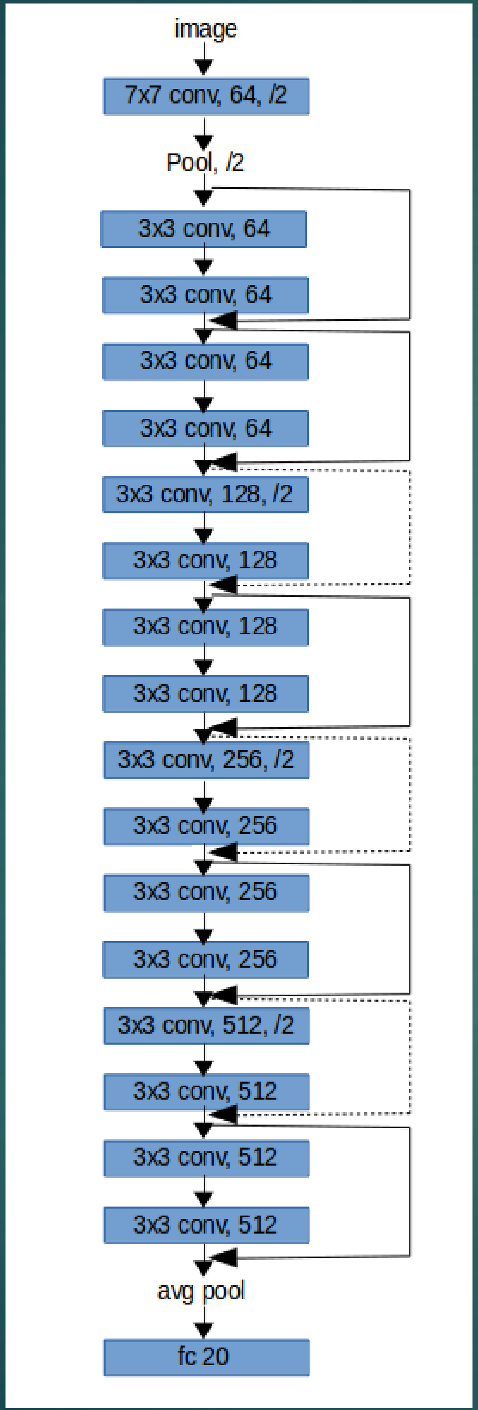
DropOut and BatchNorm

- DropOut:



Srivastava, Nitish, et al. "Dropout: A simple way to prevent neural networks from overfitting." The Journal of Machine Learning Research 15.1 (2014): 1929-1958

- BatchNorm: Normalization for layers.
  - Normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation



Skip Connection Architecture Example: Residual Networks (ResNet18)

He, Kaiming, et al. (2016) "Deep residual learning for image recognition"

All of These can be Mixed and Matched

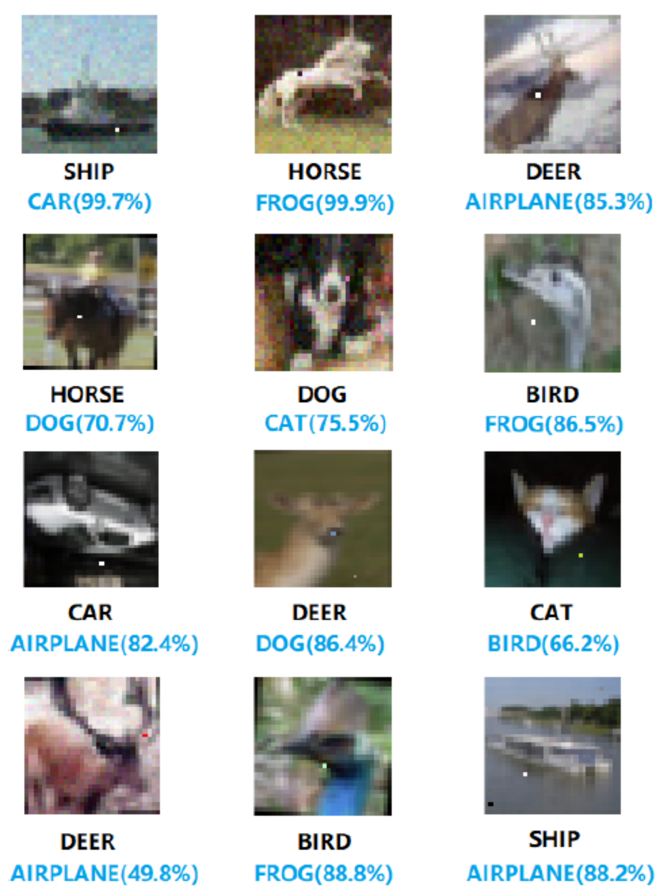
So What?

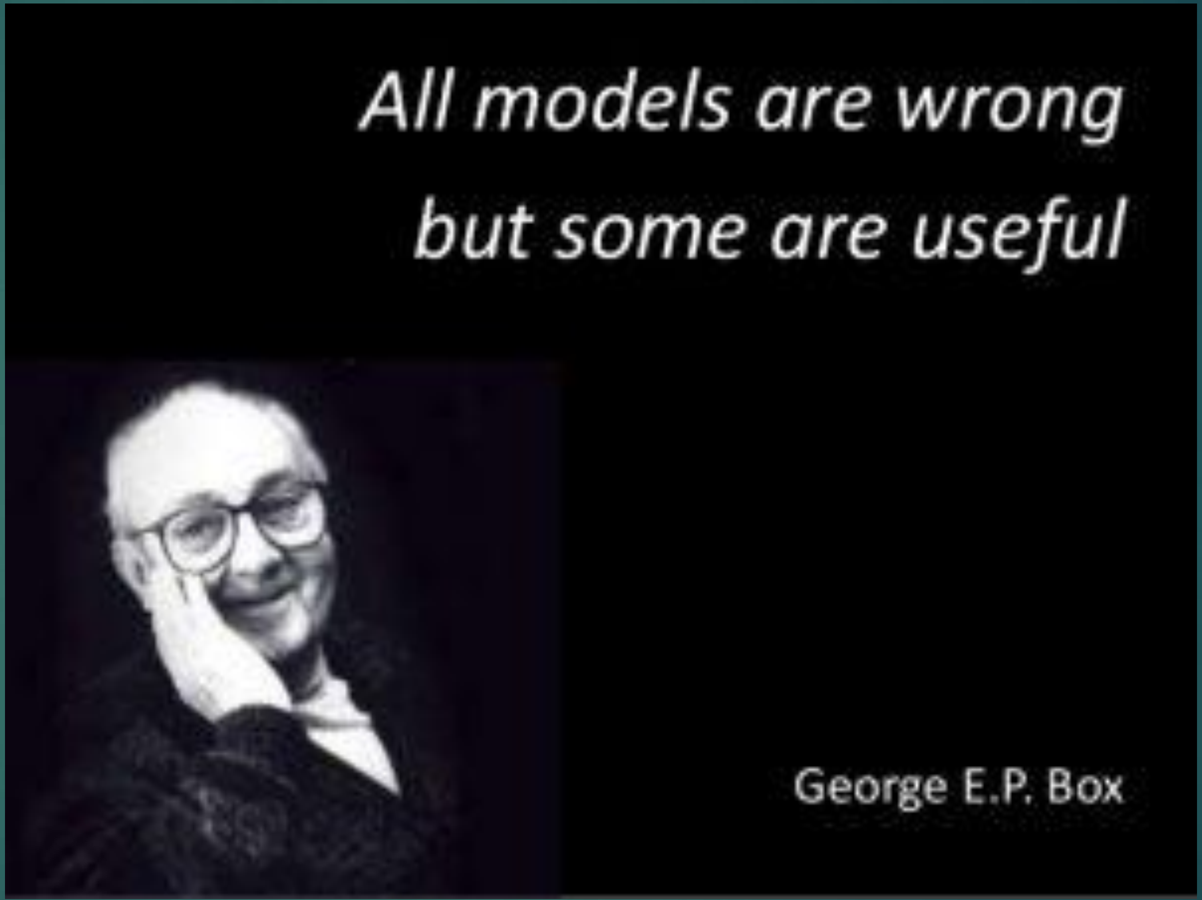
- Simplified model
  - Biologically implausible:
    - Learning (backprop through the network)
    - Sequential direction (no connections to lower layers)
  - Not as ‘general’ as previously hoped:
    - Adversarial examples demonstrate lack of ‘generalizable’ features

Adversarial Examples

One Pixel Attack for Fooling Deep Neural Networks

Jiawei Su*, Danilo Vasconcellos Vargas* and Kouichi Sakurai





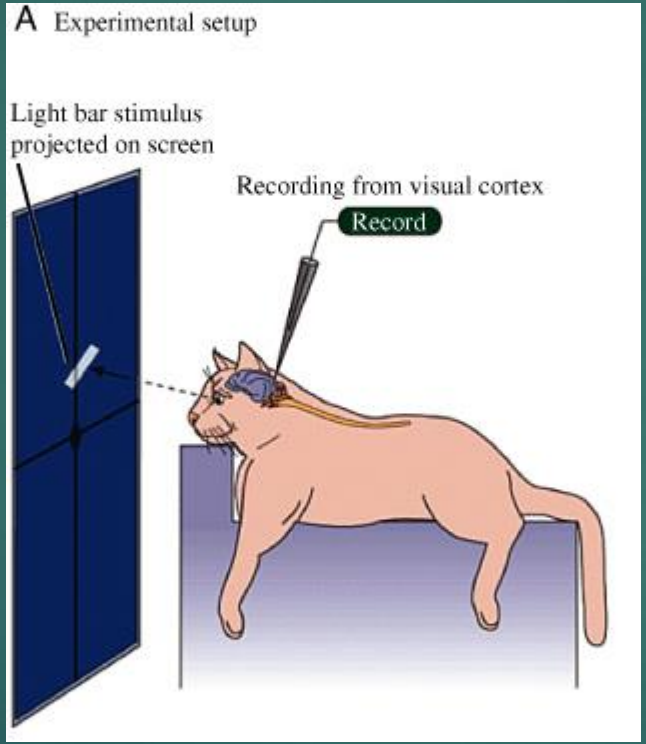
https://www.lacan.upc.edu/admoreWeb/2018/05/all-models-are-wrong-but-some-are-useful-george-e-p-box/

Are we getting too far away from Biology?

Single cell 'feature'

Hubel and Wiesel (1962) sedated a cat and recorded neuronal responses from a single cell.

- Cell responds only to a very defined width/angle of light stimulation
- Neurons arranged in a columnar architecture



http://www.informit.com/articles/article.aspx?p=1431818

Case study

- Can networks tell us something that we don’t know?
  - Can they do something we can’t?
- What can they model?

Teach and learn with *Drosophila* and convnets

2. Re-identifying *Drosophila* across days

Results:

- Architectures used
- Accuracy
- What is the network *seeing*?

Dataset

Flies:

- 10 Males
- 10 Females
- Filmed alone 15min every day for 3 consecutive days
- 14,400 images/day/fly
- Grayscale
- Filmed 3 sets over 3 weeks (n=3)

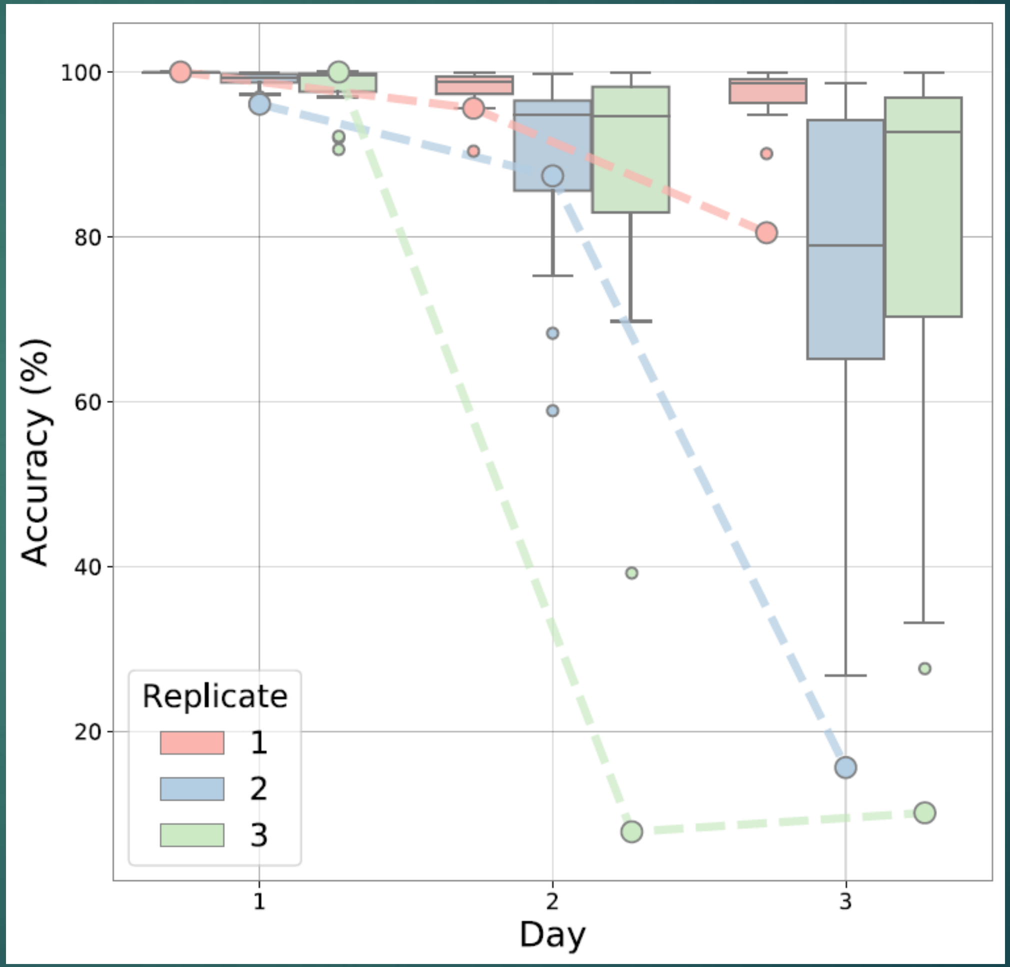
Pre-processing:

- Tracked (Ctrax)
- Re-oriented to face “North”
- Cropped to 181 x 181 pixels

“Off-the-shelf” Results

Dataset	Validation Accuracy (Day-1)	Test Accuracy (Day-2)	Test Accuracy (Day-3)
Replicate-1	99.61 ± 0.03	97.74 ± 0.16	96.72 ± 0.20
Replicate-2	98.95 ± 0.07	89.67 ± 0.31	74.52 ± 1.18
Replicate-3	98.29 ± 0.08	84.52 ± 0.39	76.65 ± 0.43

Non-Uniform Decrease in Accuracy



General conclusions so far:

- Flies are visually distinguishable
  - Can re-identify the majority of flies with >90% accuracy without data augmentation.
  - Can do this despite our inability to do so.
- Flies visually differentiate between days
  - Some flies become impossible to reliably re-identify after 24hours

Different Domains

Torralba and Efros (2011) "Unbiased look at dataset bias."

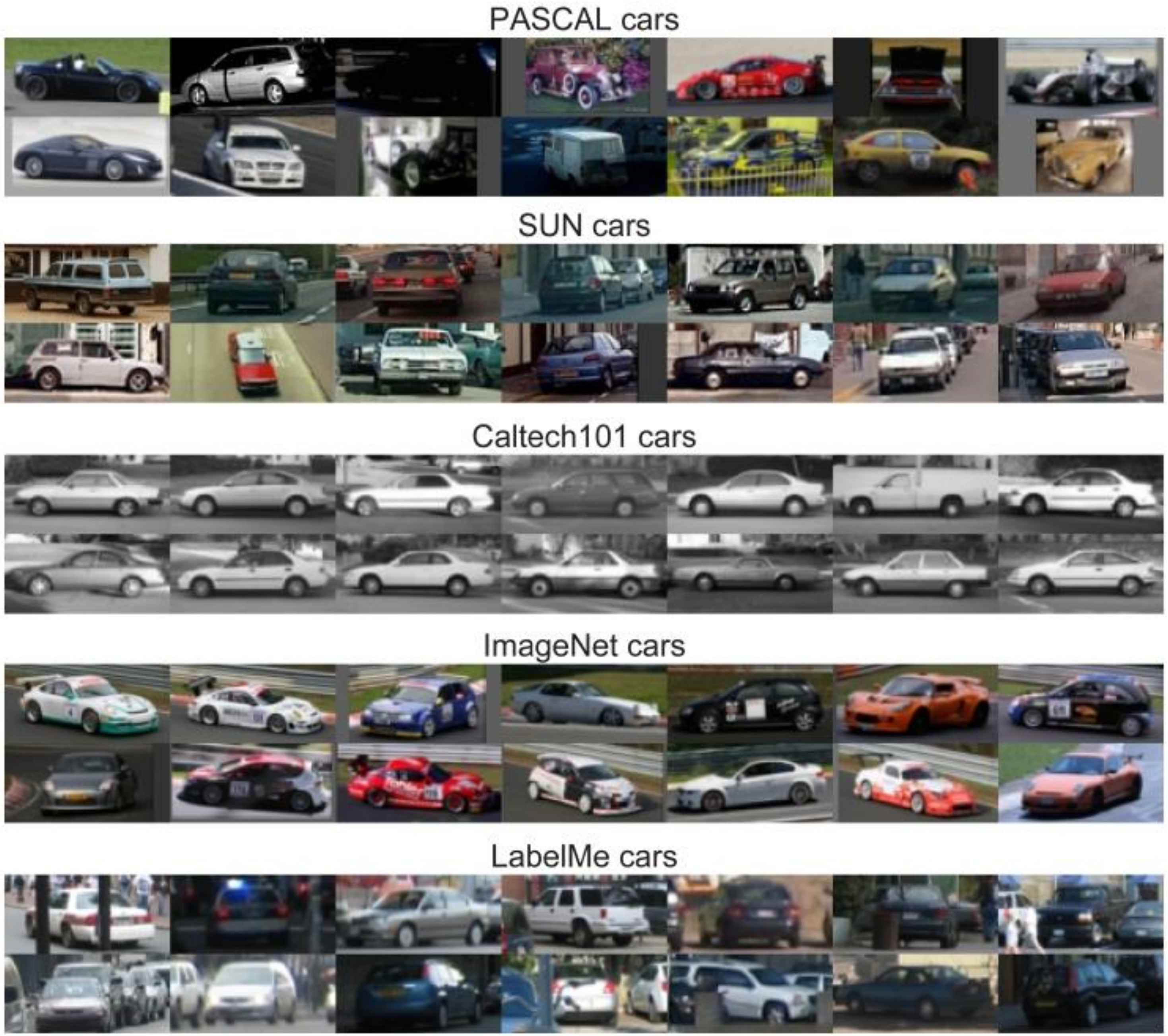
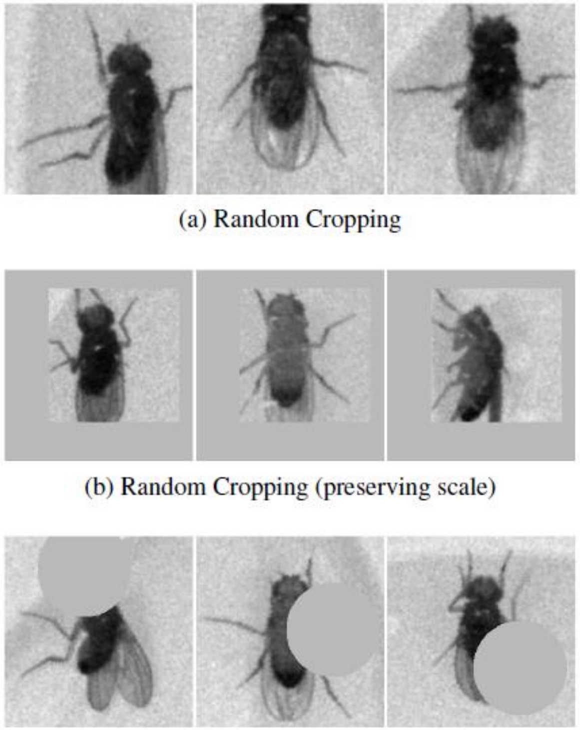


Figure 4. Most discriminative cars from 5 datasets

Two Approaches

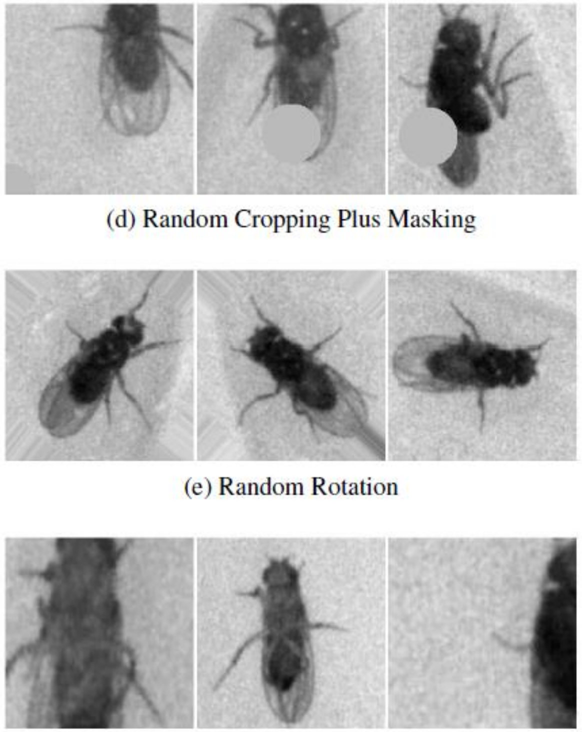
- Data Augmentation
  - Prevent it from learning the features specific to days
- Network Re-Structuring
  - Explicitly task the network with ignoring features specific to days



(a) Random Cropping

(b) Random Cropping (preserving scale)

(c) Random Masking

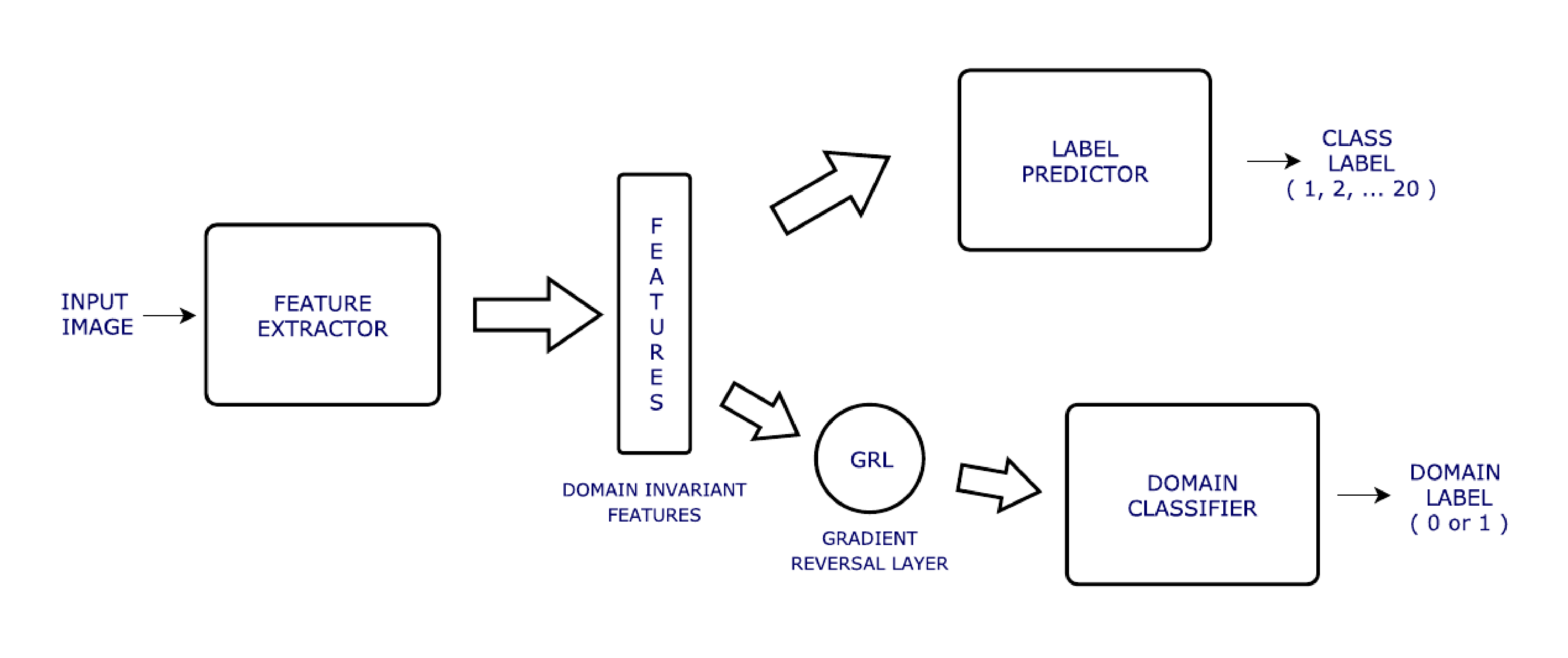


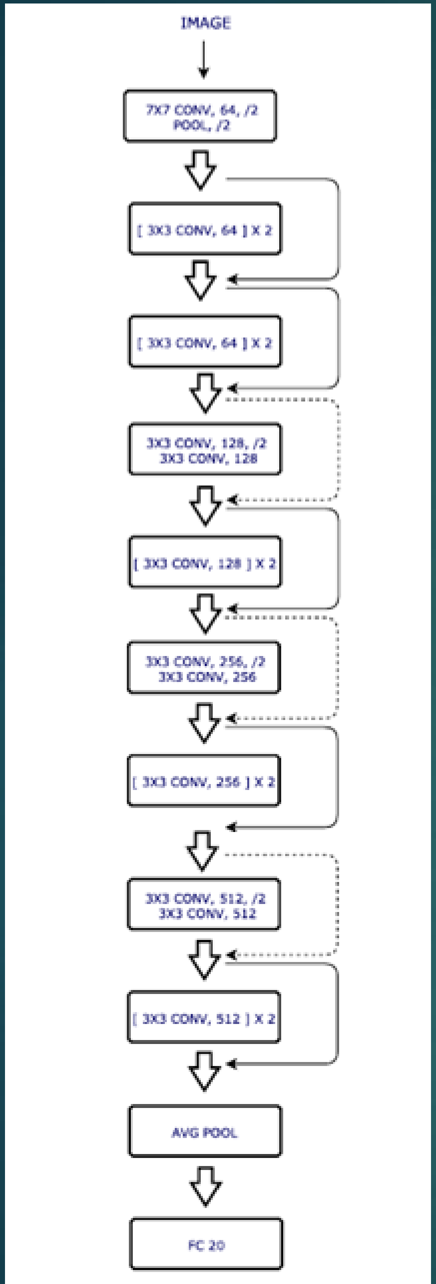
(d) Random Cropping Plus Masking

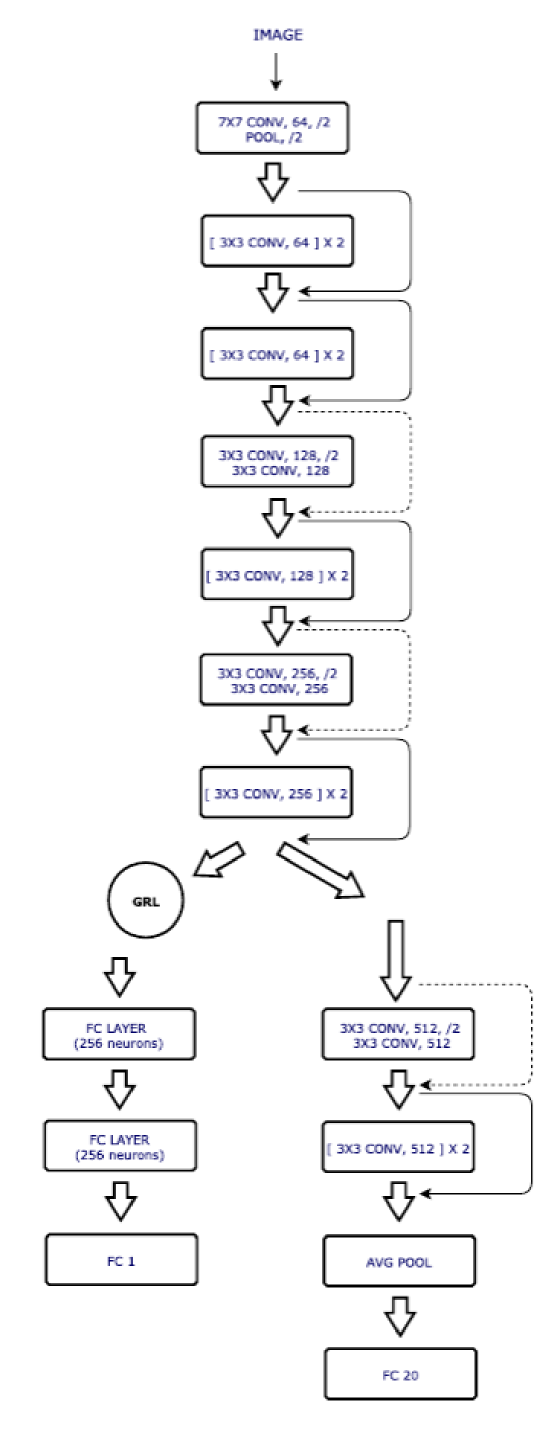
(e) Random Rotation

(f) Random-Sized Cropping

Domain Adversarial Architecture







Adapting Resnet18 to have Adversarial Domain Adaptation

Overall Results and Final Accuracies

Table 3: Different combinations of domain adaptation and data augmentation techniques compared to the baseline methods. Test accuracy (day-3) on all three replicates is reported here. Experiments shown are: SDT — Single Day Training; DDT — Double Day Training; R-MASK — Random Masking; DANN — Domain Adversarial Neural Network.

Experiment	trained on	replicate-1	replicate-2	replicate-3
SDT	Day-1	96.72 ± 0.20	74.52 ± 1.18	76.65 ± 0.43
DDT	Days-1,2	99.40 ± 0.03	90.06 ± 0.32	98.34 ± 0.06
DDT + R-MASK	Days-1,2	$\mathbf{99.51 \pm 0.08}$	93.20 ± 0.33	$\mathbf{98.61 \pm 0.05}$
DANN	Days-1,2	98.45 ± 0.08	92.21 ± 0.38	96.87 ± 0.16
DANN + R-MASK	Days-1,2	99.07 ± 0.02	$\mathbf{95.51 \pm 0.42}$	97.90 ± 0.16

*Domain Adaptation can accurately classify (>80% minimum) 59/60 flies studied.

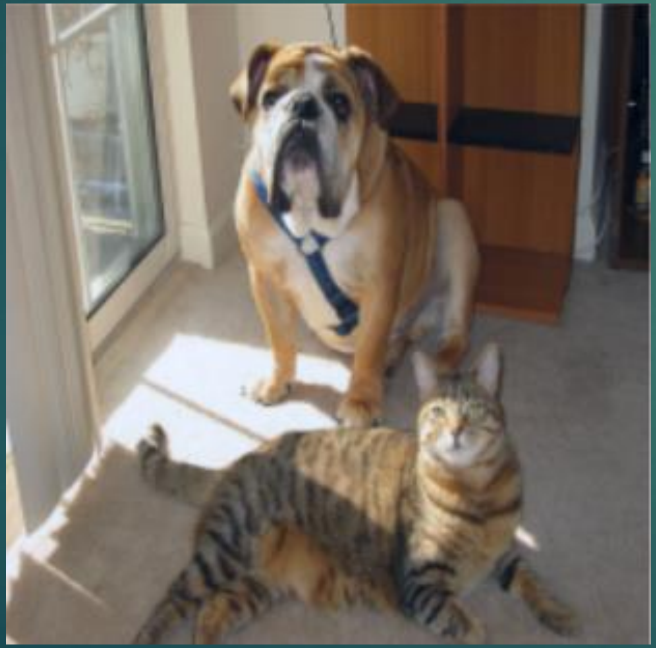
Overall Results and Final Accuracies

- What are the networks *seeing*?
  - What 'features' of the fly is it using to identify individuals?

- Convnets (with or without pooling) are generally thought to be 'black boxes'
  - Some efforts underway to understand *what* the network uses to classify

Guided Backpropagation

Original Image

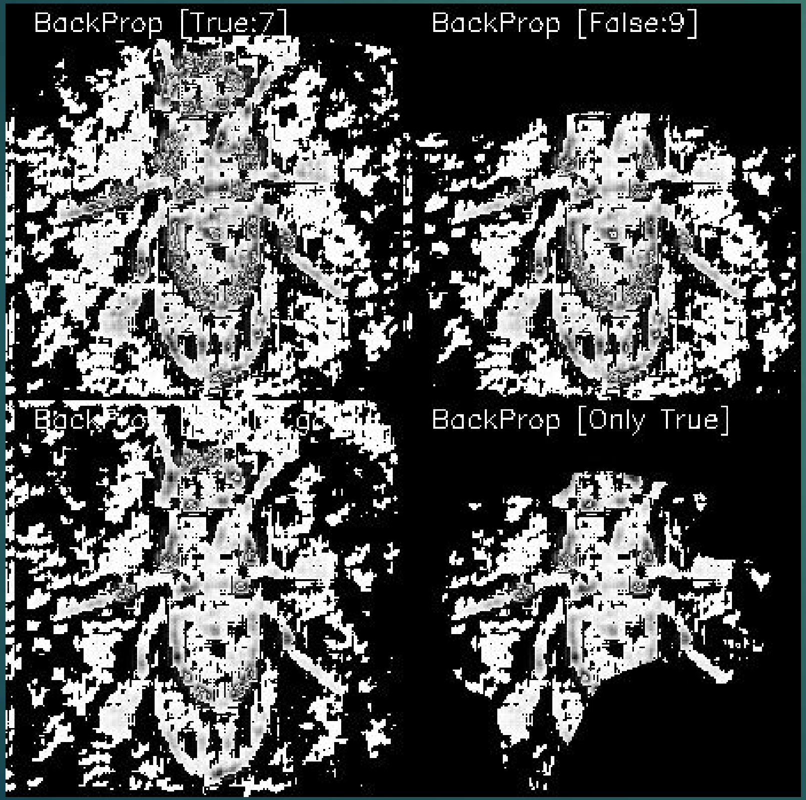


Guided Backpropagation

Selvaraju *et al.* 2016 “Grad-CAM: Why did you say that?”

Guided Backpropagation… was a bust

Generated Image



Why? Low level filters…


Trained on flies


Trained on CIFAR10

Class Activation Mapping (CAM)

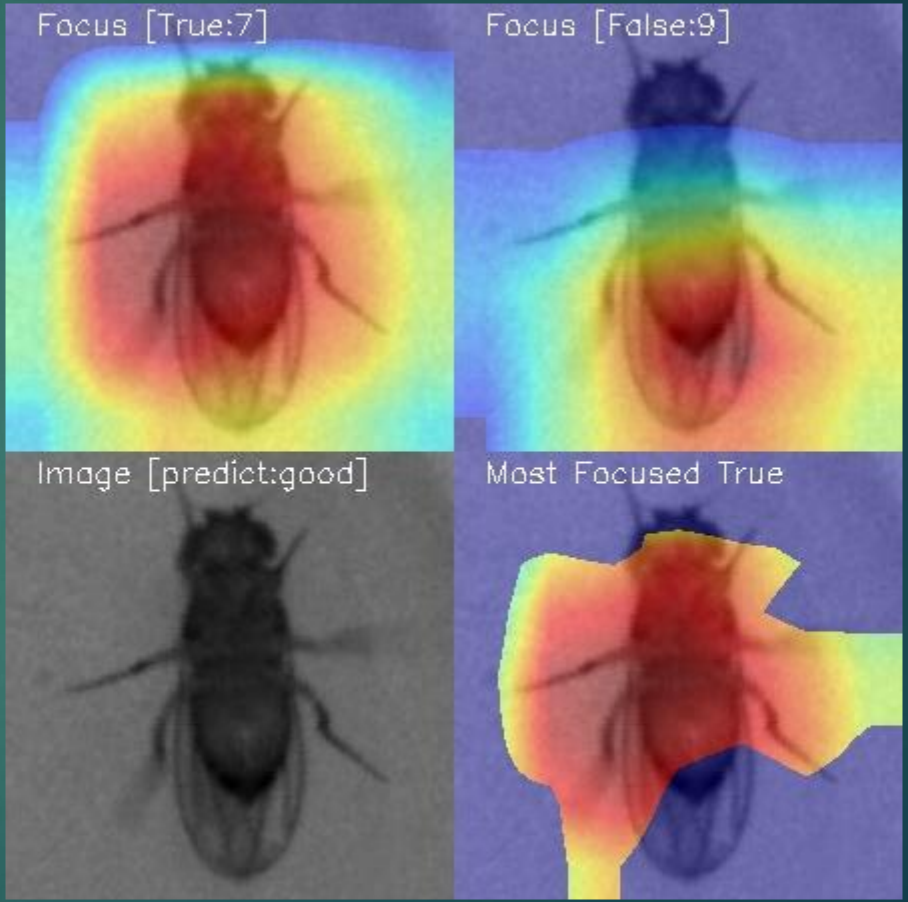
What part of the image corresponds to the 'dog' class?

What part of the image corresponds to the 'cat' class?

Selvaraju *et al.* 2016 "Grad-CAM: Why did you say that?"

What (if any) features allow identification?

- Global size/shape and patterns
- Head seems relatively not important



Teach and learn with *Drosophila* and convnets

2. Re-identifying *Drosophila* across days

Results:

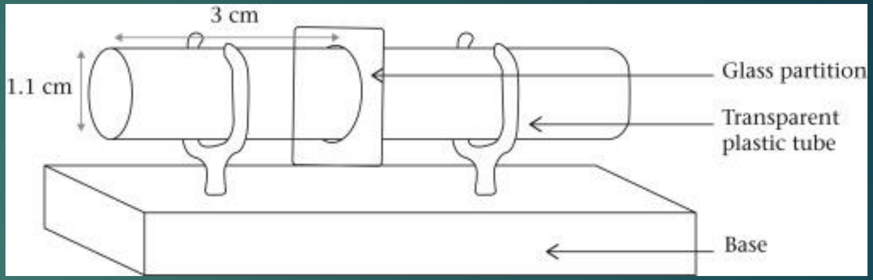
- Architectures used
- Accuracy
- What is the network *seeing*?

3. Build a fly-eye

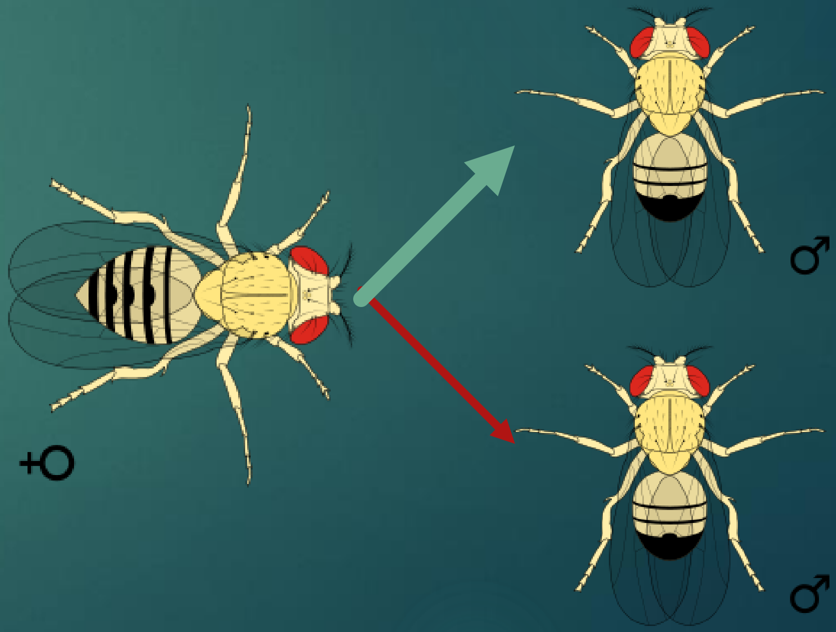
Having fun with convnets

Can *Drosophila* even see each other?

Some semi-controversial studies



- Acquiring mate preference for color-dusted males following visual observation of successful copulation (Mery *et al.* (2009))
- Observing behavior of wasp-exposed flies 'dialect' through glass changes egg-laying (Kacsoh *et al.* (2017) *BioRxiv*)



‘Effective’ Inter-Ommatidial Angle

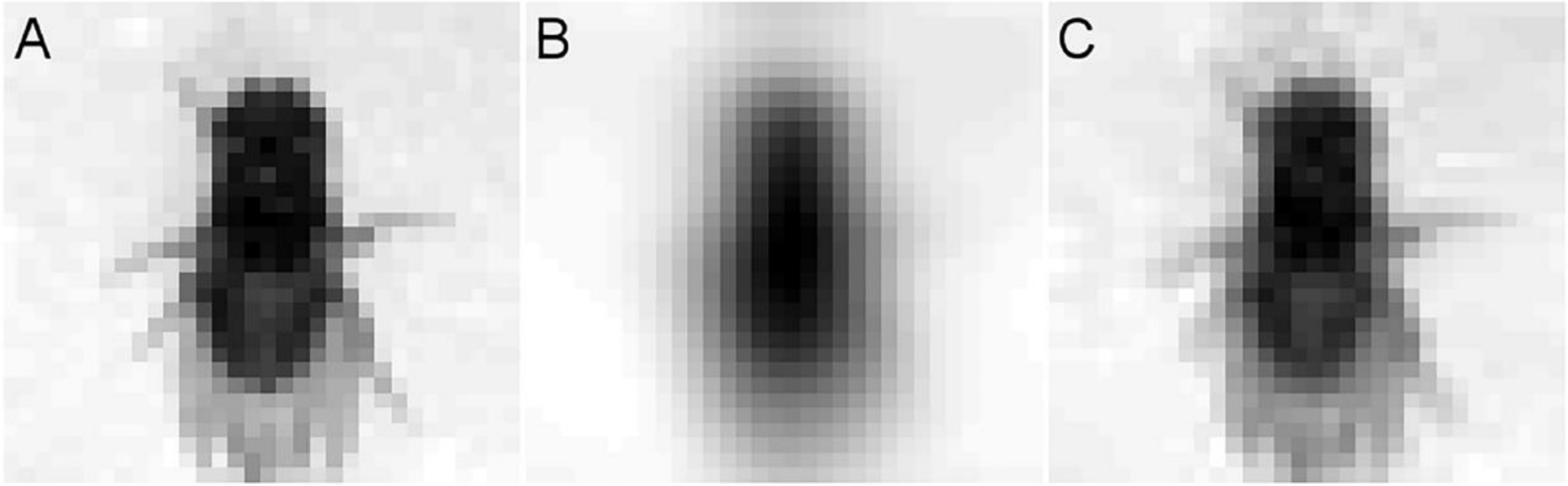
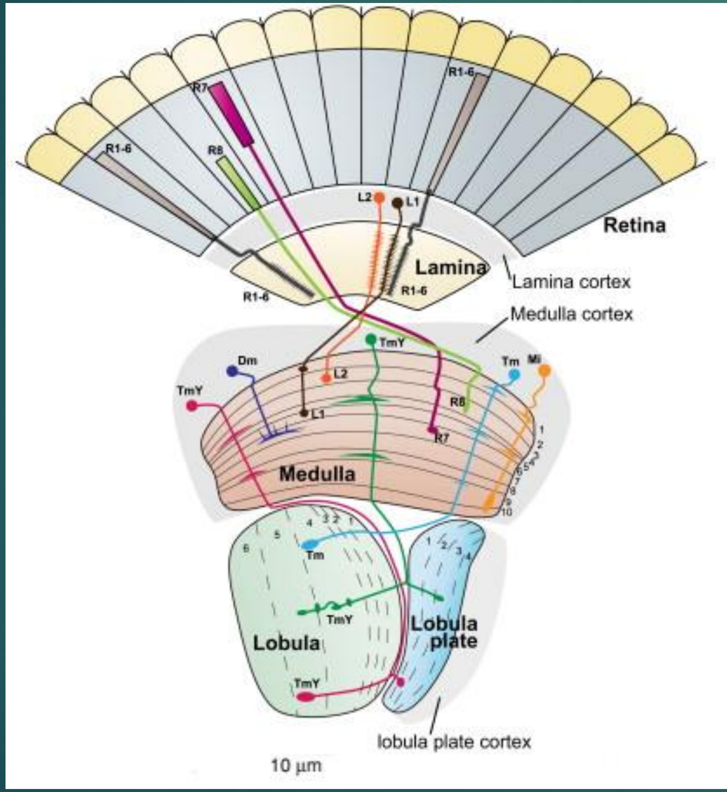
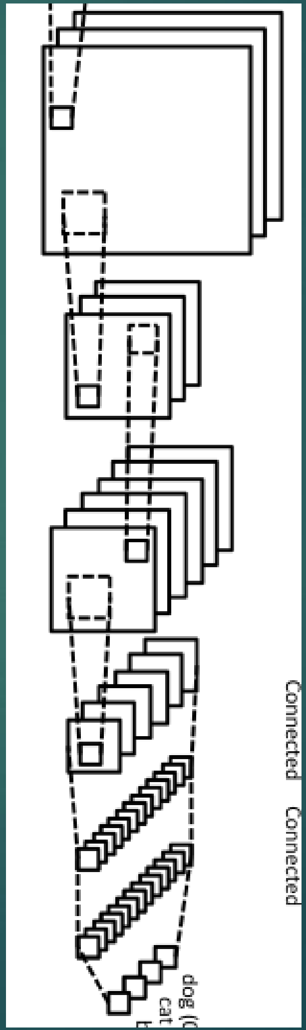


Fig 1. Theoretical visual acuity of *Drosophila melanogaster*. Image of *Drosophila melanogaster* represented after various theoretical bottlenecks. A: Image of a female *D. melanogaster* re-sized through a 32×32 bottleneck. B: The same image, but adjusted using AcuityView [4] for a viewing distance of 3 body lengths using the inter-ommatidial angle of 4.8˚ [5]. C: The same image and distance, but using a conservative estimate of the effective acuity determined by Juusola *et al*. [6] of approximately 1.5˚.

Biological convolutions and feature maps?





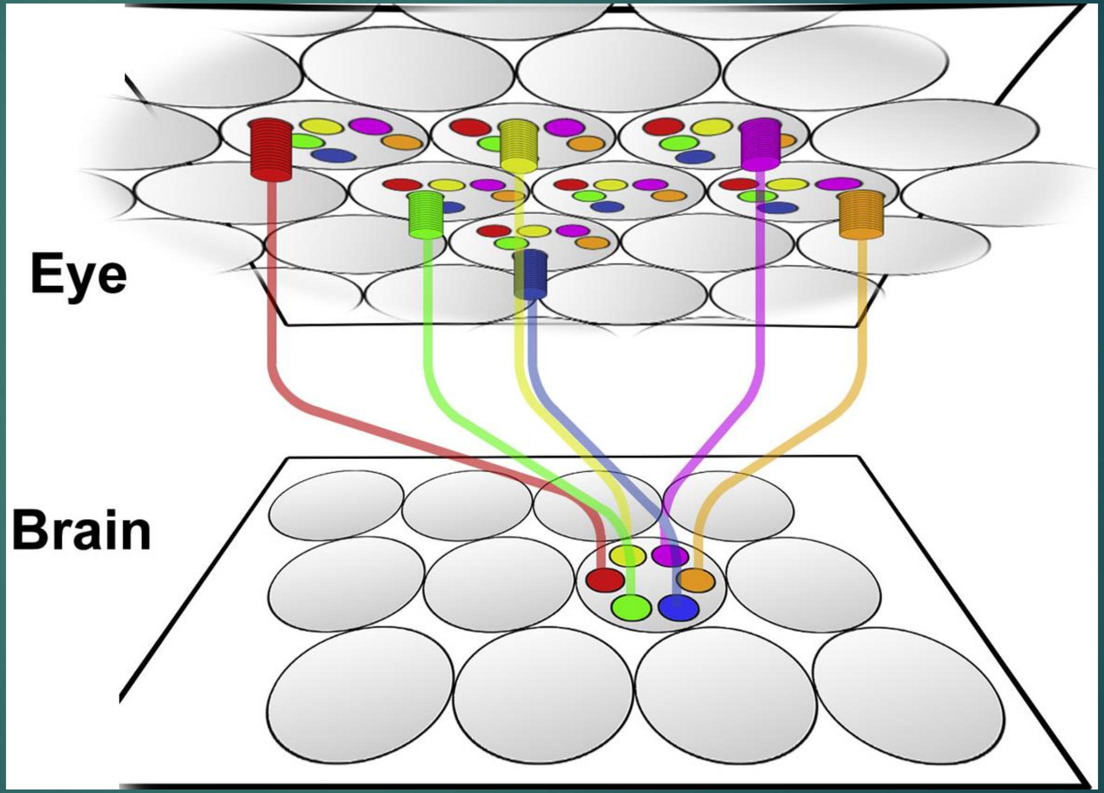
Build a simplified fly-eye net

Each laminal cartridge gets input from 6 surrounding cells (from R1-R6 photoreceptors)

Each photoreceptor could be 'tuned'

This is very similar to a 6 pixel low level 'filter'

Will be our first feature layer in the 'fleye-net'



Langen *et al.* (2015) "The Developmental Rules of Neural superposition in *Drosophila*"

Laminal cartridge as simple filter

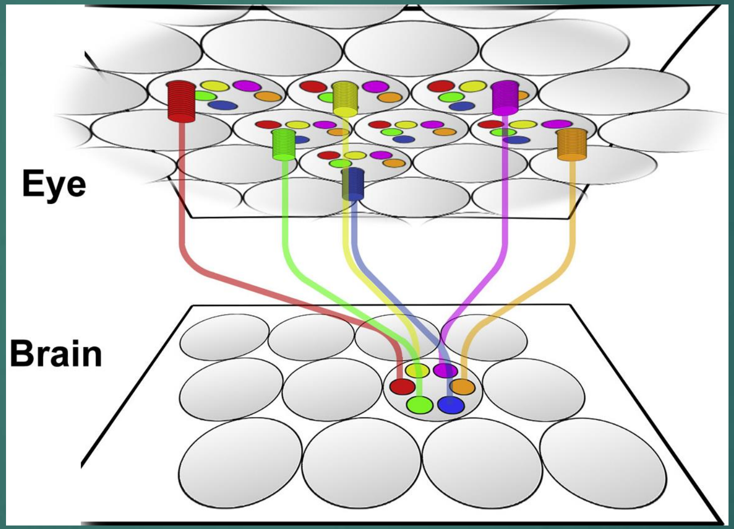
Remember: ‘simple’ filters adequate to ID individual *Drosophila*

Trained on flies

Trained on CIFAR10

Build a simplified fly-eye net

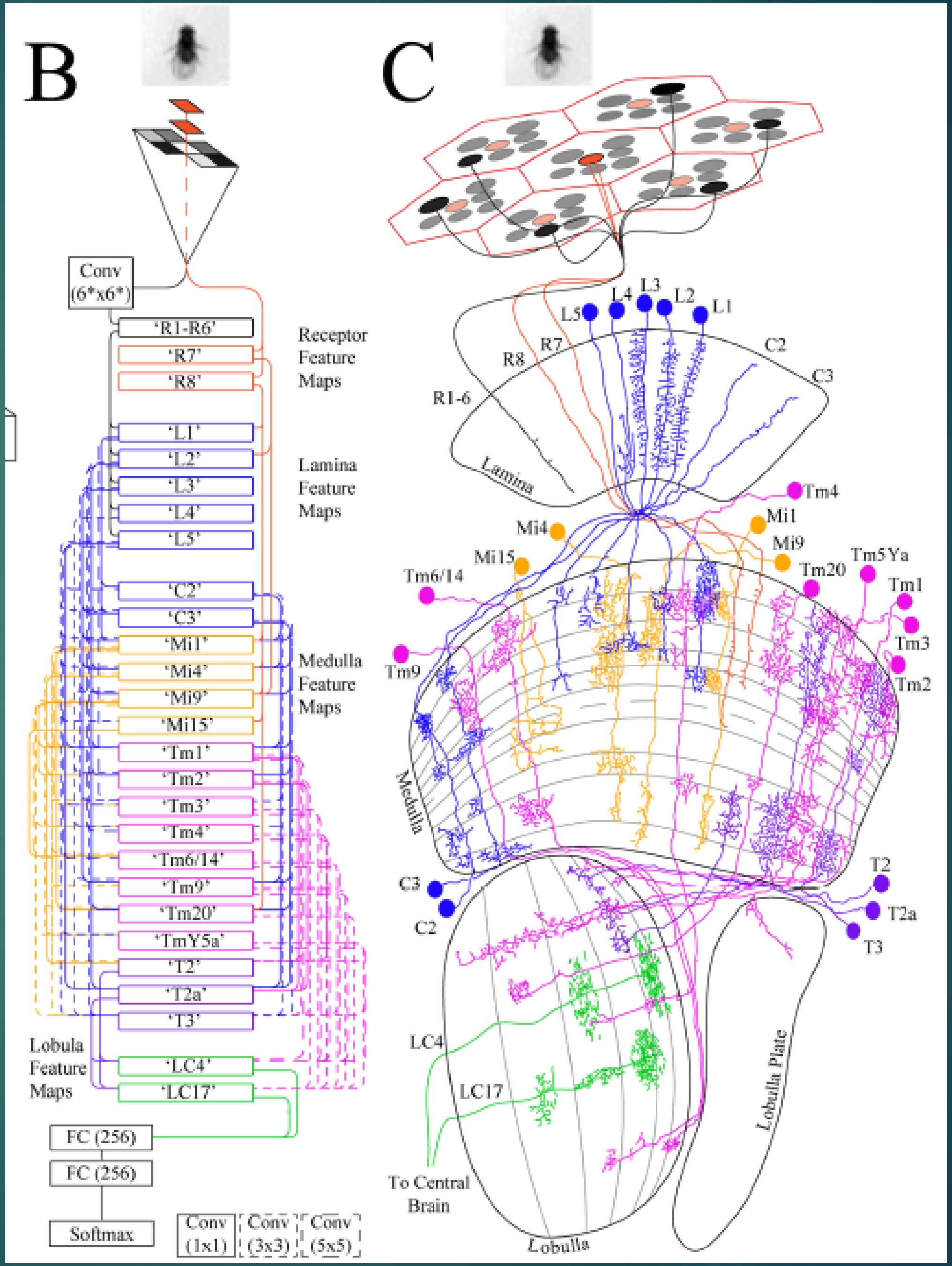
- Our first filter layer will be an 6-pixel donut, with learnable weights that will generate the first feature layer.
- R7, R8 neurons are simple weighted pass-through neurons (1x1 filters).



Langen *et al.* (2015) “The Developmental Rules of Neural superposition in *Drosophila*”

- The architecture we used:
  - Modified 1st convolutions
  - Convolutions (filter size based on arborization)
  - Skip connections

https://github.com/j-schneider/fly_eye



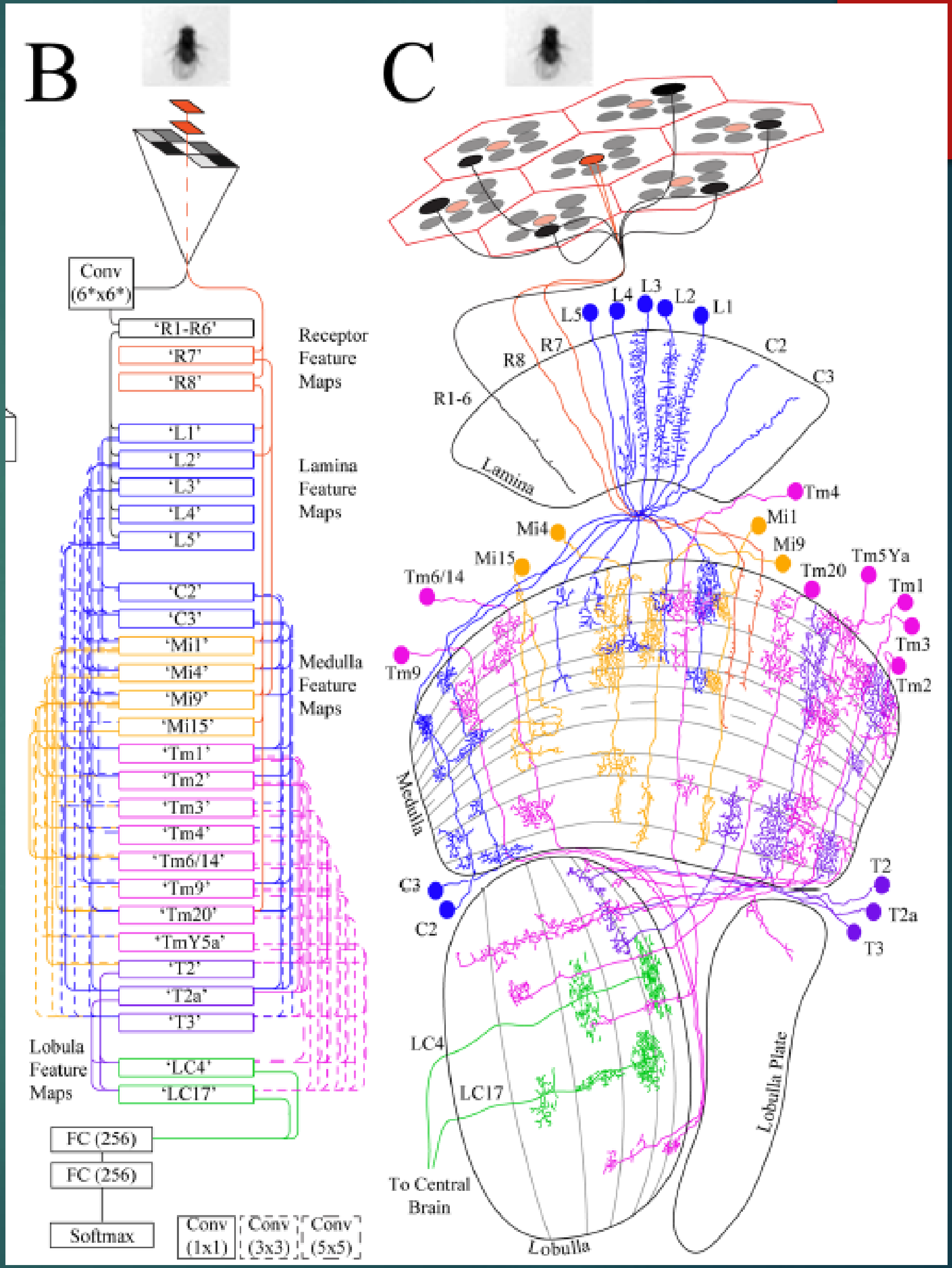
- Better than humans!

Table 1. Performance on *D. melanogaster* re-identification.

Model Name	Resolution[1] (pixels)	Accuracy (F_1 Score)[2]
ResNet18 [19]	158×158	0.9426 ± 0.0358
Zeiler and Fergus [13]	158×158	0.9373 ± 0.0365
Human Performance	158×158	0.1309
Zeiler and Fergus [13]	29×29	0.8549 ± 0.0778
ResNet18 [19]	29×29	0.8357 ± 0.0909
Our fly-eye	29×29	**0.7548 ± 0.1141**
Our fly-eye w/ random zoom[3]	29×29	**0.5486 ± 0.1316**
Human Performance	29×29	0.0829
Random Chance		0.05

What's next

- Some lobula columnar neurons (like LC11) seem specialized for high-acuity small object motion detection (Keleş MF, Frye MA. Object-detecting neurons in Drosophila. Current Biology. 2017; 27(5):680–687)
- Other LC neurons (like LC17), when stimulated, seem to provoke social-context dependent behaviours (Wu M, Nern A, Williamson WR, Morimoto MM, Reiser MB, Card GM, et al. Visual projection neurons in the Drosophila lobula link feature detection to distinct behavioral programs. Elife. 2016; 5. https://doi.org/10.7554/eLife.21022).



Other Examples

Deep Neural Networks Rival the Representation of Primate IT Cortex for Core Visual Object Recognition

Charles F. Cadieu[1*], Ha Hong[1,2], Daniel L. K. Yamins[1], Nicolas Pinto[1], Diego Ardila[1], Ethan A. Solomon[1], Najib J. Majaj[1], James J. DiCarlo[1]

1 Department of Brain and Cognitive Sciences and McGovern Institute for Brain Research, Massachusetts Institu
of America, 2 Harvard–MIT Division of Health Sciences and Technology, Institute for Medical Engineering and
Massachusetts, United States of America

A Connectome Based Hexagonal Lattice Convolutional Network Model of the Drosophila Visual System

Fabian David Tschopp
Institute of Neuroinformatics
University of Zurich and ETH Zurich
tschopfa@student.ethz.ch

Michael B. Reiser
Janelia Research Campus
Howard Hughes Medical Instutute
reiserm@janelia.hhmi.org

Srinivas C. Turaga
Janelia Research Campus
Howard Hughes Medical Instutute
turagas@janelia.hhmi.org

Do Neural Networks Show Gestalt Phenomena? An Exploration of the Law of Closure

Been Kim[1] **Emily Reif**[1] **Martin Wattenberg**[1] **Samy Bengio**[1]
beenkim@google.com, Google

All Models are Wrong

- Steps are being taken to make them progressively ‘better’
  - Ex. Spiking Neural Networks

- But ultimately it is not whether they are ‘wrong’ or right, but if they are useful to *you*.

Acknowledgements

UNIVERSITY of GUELPH

UNIVERSITY OF TORONTO MISSISSAUGA

CIFAR
CANADIAN INSTITUTE for ADVANCED RESEARCH

NSERC CRSNG

CIHR IRSC
Canadian Institutes of Health Research
Instituts de recherche en santé du Canada