deep imaging

Lecture 13: CNN visualization and example applications

Machine Learning and Imaging

BME 548L
Roarke Horstmeyer

Announcements

- Homework 1 should be graded soon
- Homework 3 assigned last Thursday 3/11 and is due Thursday 3/25
- Homework 4 assigned Thursday 3/25, will be due Thursday 4/8
- Homework 5 assigned Thursday 4/8, will be due Thursday 4/22 (last day of class)
- We'll also start to prepare for the final project….

Class project details

deep imaging

- Full details are here: https://deepimaging.github.io/proj-info/
- Can work alone or in a group (up to 4 people), required effort will scale with # of people
- Select a “base” dataset (online, or from a list I’ll make)
- Simulate parameters of a physical (imaging) system with base dataset
- Train deep neural net with simulated dataset
- Report results

What you’ll need to submit:

1) The project’s source code

2) A short research-style paper (3 pages minimum, 5 pages maximum) that includes an introduction, results, a discussion section, references and at least 2 figures

3) A completed web template containing the main results from the research paper

4) A 7-10 minute presentation that each student will deliver to the class

Class project – what are the first steps?

1. Think about it!
2. Discuss with your friends/others in the class (feel free to use Slack!) to form group
3. Schedule a short 15 meeting with me: https://calendly.com/rwh4/15min
    - **Monday March 22, 3pm – 5pm**
    - **Tuesday March 23, 9:30am – 11:30am**
4. Start to write-up a proposal
    - General aim: 1 paragraph, specifying physical layer or hardware analysis component
    - Discussion: (a) data source(s), (b) expected simulations, (c) expected CNN, (d) quantitative analysis of physical layer/physical component (comparison, plot, etc).

- Project proposal due date: **Friday March 26**, revisions after if needed
- Final project will be presented during final exam slot: **Thursday April 29, 9am-noon**
- (note: due to large class size, this may go a bit over 3 hours, can maybe split in 2 sessions)

Projects from prior semester of BME 548:

Finding Ultrasound Sub-apertures for Liver Vessel Segmentation
Single-Pixel, Single-Frequency Hand Gesture Recognition with a Dynamic Metasurfaces
Going Deeper: Depth Image Classification via simulated SPAD array images
Trained Blur Kernel for histology slide segmentation using a Deep Neural Network
Classification of Tuberculosis Bacilli With and Without Staining
A deep learning approach to improving ultrasonic plane wave imaging
Automated Image Focus Detecting Algorithm for Low-Cost Handheld Microscope
Optimal shift-variant point-spread function for improved classification
Deep Learning for Motion Tracking on the Micron Scale with Ultrasound
Sensor Multiplexing and Reconstruction for Color Images
Noise Reduction in Optical Coherence Tomography using a Deep Image Prior
Optimization of illumination for Unet-Base Cervix Segmentation
HDR image reconstruction with filters over pixels – What is the optimal design?
Detection of Lesions in Variably Noisy Ultrasound Images Using Machine Learning
Methods for Segmentation of Fine Structure in Rodent Histological Specimens
Direct reconstruction network for photoacoustic imaging with fewer measurements
Machine Learning for Ultrasound Lesion Mapping with Apodization Optimization
Resolution versus Precision in X-ray detection of Pneumonia
Optimizing illumination for overlapped image classification

Example project topics:

Can we design a new lens/transducer/antenna shape to improve classification of X?

What is the tradeoff between image resolution and accuracy for X (classification, segmentation, etc.)? What if we had access to n low-resolution cameras – how might we position them to get the best performance?

Can we determine an optimal set of colors to improve fluorophore distinguishability?

How does classification accuracy change with sensor bit depth, down to the 1-bit level for single-photon detectors?

If we just had a few sensors, how should be arrange them e.g. a mask to be able to predict the position of X?

Is there some optimal shift-variant blur that we can to use for a particular task?

Or, given a shift-variant blurry image, can we establish a good deconvolution using locally connected layers?

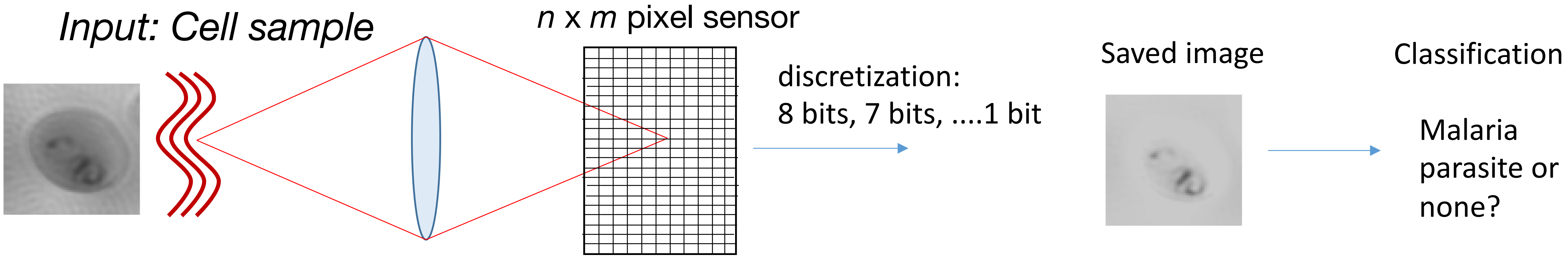
What is the optimal way to layout filters on a sensor to capture a color image for classification? Or an HDR image?

HDR image generation with filters over pixels – what is optimal design?

What if we could make a sensor with different sized pixels – how should they be laid out to achieve the best X?

How does classification accuracy change with sensor bit depth, down to the 1-bit level for single-photon detectors?

deep imaging



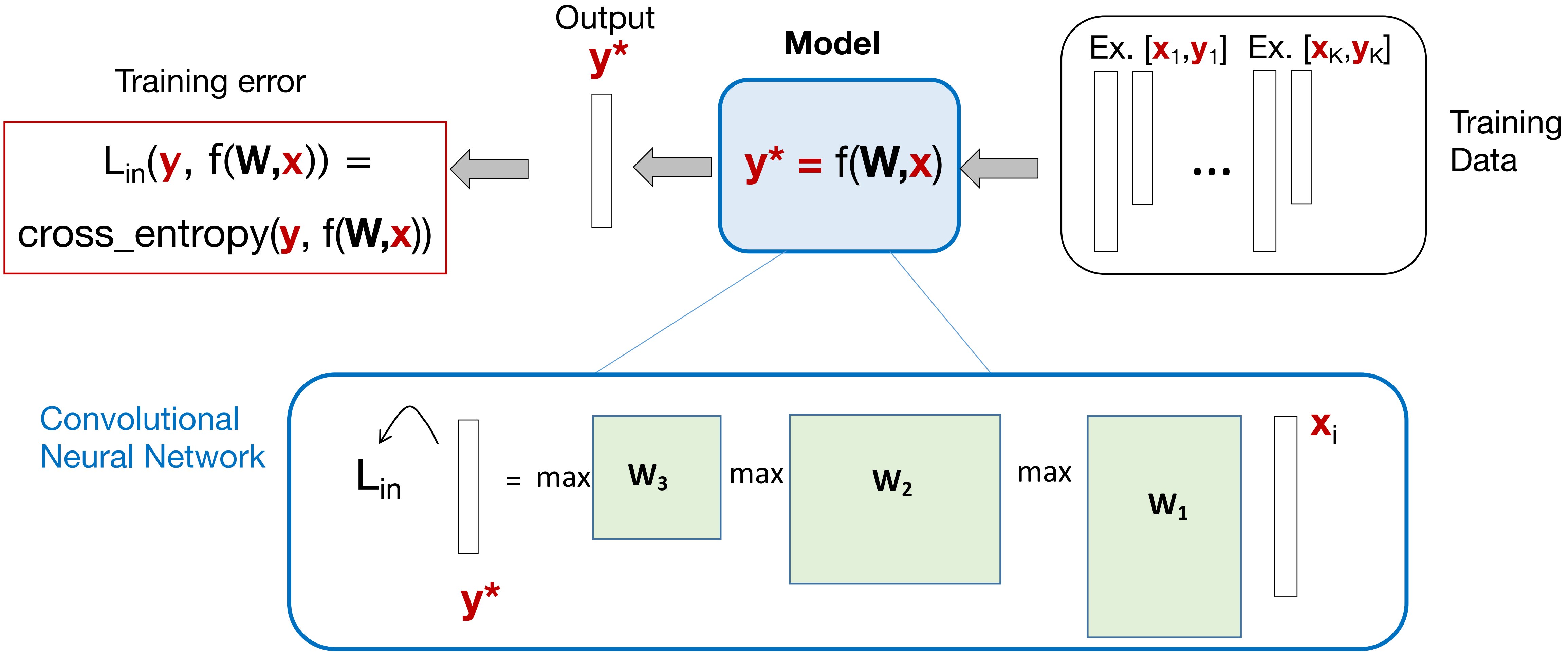
Physical layer test: per-pixel discretization (max. # bits/image)

I propose to test the classification performance of a microscope as a function of sensor bit depth (i.e., image discretization). I will plot average classification test accuracy as a function of number of sensor bits from 1 bit to 8 bits. I will additionally test whether the pixel discretization value can be optimized as a physical layer parameter. I will simulate a pixel discretization value, at each pixel, by multiplying the associated raw intensity value at each pixel by a weight, and will then using the max() operator to set a threshold. I will examine how classification accuracy varies with this additional constraint, and will attempt to draw insights into where the network prefers to have more bits/pixel.

Dataset: 12,500 images of 4 types of blood cell https://www.kaggle.com/paultimothymooney/blood-cells

(Specify more details about simulation network, physical layer implementation and quantitative analysis)

Our very basic convolutional neural network



Forward pass: from $\mathbf{x}_i$ and current **W**'s, find L_{in}

deep imaging

How to examine and present your results: a few options at different stages

Options to examine your test data after processing:

- ROC curve, Precision-Recall
- Confusion matrix
- Sliding window visualization
- Layer visualizations
- Saliency maps etc.
- tSNE visualization

deep imaging

ROC curve and confusion matrix

- Can set threshold for f(x,W) wherever
- Leads to sliding window between FN and FP rate
- Need to summarize both statistics as a function of sliding window

Estimated label

f(x, W)

Actual label y	+1	-1
+1	True positive	False negative
-1	False positive	True negative

Missed an event

Predict event when there isn't one

ROC curve and confusion matrix

TP Rate =

Sensitivity = TP / (TP + FN) = TP / Actual positives

False Positive Rate = FP / (TN + FP) = FP/ Actual negatives

Specificity = TN / (TN + FP) = TN / Actual negatives
= 1 – False Positive Rate

Estimated label

$f(x, W)$

Missed an event

Actual label y		Estimated +1	Estimated -1
	+1	True positive	False negative
	-1	False positive	True negative

Predict event when there isn't one

ROC curve and confusion matrix

TP Rate =

Sensitivity = TP / (TP + FN) = TP / Actual positives

False Positive Rate = FP / (TN + FP) = FP/ Actual negatives

Specificity = TN / (TN + FP) = TN / Actual negatives
= 1 – False Positive Rate

Estimated label

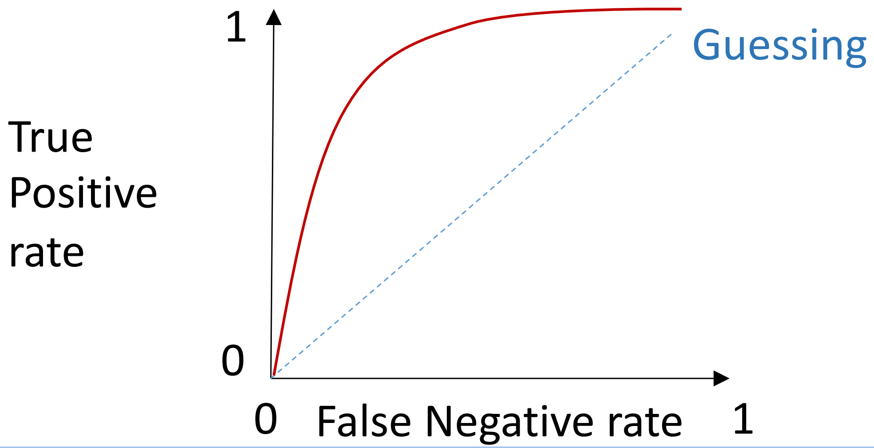
f(x, W)

Missed an event

Actual label y	+1	-1
+1	True positive	False negative
-1	False positive	True negative

Predict event when there isn't one

Receiver-Operator Curve



ROC curve and confusion matrix

TP Rate =

Sensitivity = TP / (TP + FN) = TP / Actual positives

False Positive Rate = FP / (TN + FP) = FP/ Actual negatives

Specificity = TN / (TN + FP) = TN / Actual negatives
= 1 – False Positive Rate

Estimated label

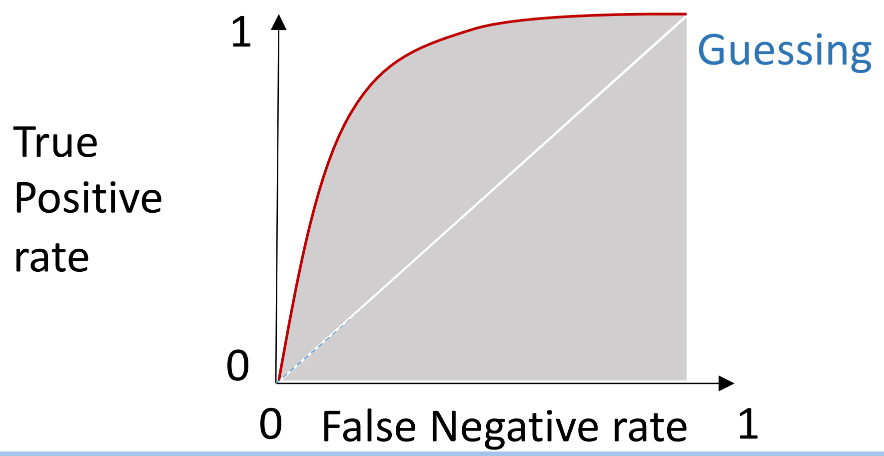
$f(x, W)$

Missed an event

Actual label y	+1	-1
+1	True positive	False negative
-1	False positive	True negative

Predict event when there isn't one

Receiver-Operator Curve



Area under the curve (AUC): Integral of ROC curve

ROC curve and confusion matrix

Recall =
Sensitivity = TP / (TP + FN) = TP / Actual positives

Estimated label

f(x, W)

Missed an event

Actual label y	+1	-1
+1	True positive	False negative
-1	False positive	True negative

Predict event when there isn't one

Precision = TP / (TP + FP) = TP / Estimated positives

- Sometimes, you don't care about true negatives (just want to find events)
- In this case, use Precision and Recall

ROC curve and confusion matrix

Recall =
Sensitivity = TP / (TP + FN) = TP / Actual positives

Estimated label

$f(x, W)$

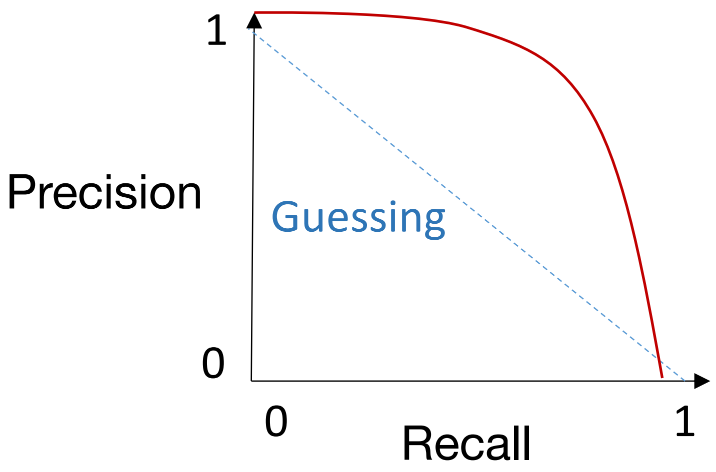
Missed an event

Actual label y		+1	-1
	+1	True positive	False negative
	-1	False positive	True negative

Predict event when there isn't one

Precision = TP / (TP + FP) = TP / Estimated positives

Precision-Recall curve



F1 Metric: $(1/\text{precision} + 1/\text{recall})^{-1}$

ROC curve and confusion matrix

deep imaging

Just 2 categories

Estimated label

$f(x, W)$

y		+1	-1
	+1	True positive	False negative
	-1	False positive	True negative

Confusion Matrix: 2+ categories

Estimated label

$f(x, W)$

y

Actual ↓	State1 (Predicted)	State2 (Predicted)	State3 (Predicted)	State4 (Predicted)	State5 (Predicted)	State6 (Predicted)	State7 (Predicted)	State8 (Predicted)
State1 (Actual)	90.12 %	0.00 %	9.88 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
State2 (Actual)	0.00 %	100.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
State3 (Actual)	0.00 %	0.00 %	92.66 %	0.00 %	0.00 %	7.34 %	0.00 %	0.00 %
State4 (Actual)	0.00 %	0.00 %	0.00 %	100.00 %	0.00 %	0.00 %	0.00 %	0.00 %

Other performance metrics

- Overlap between segmented areas: Jaccard similarity coefficient

  J = |R1 ∩ R2| / |R1 U R2|

- MSE, PSNR

- Structural Similarity (SSIM)

$$\mathrm{SSIM}(x,y)=\frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)}$$

with:

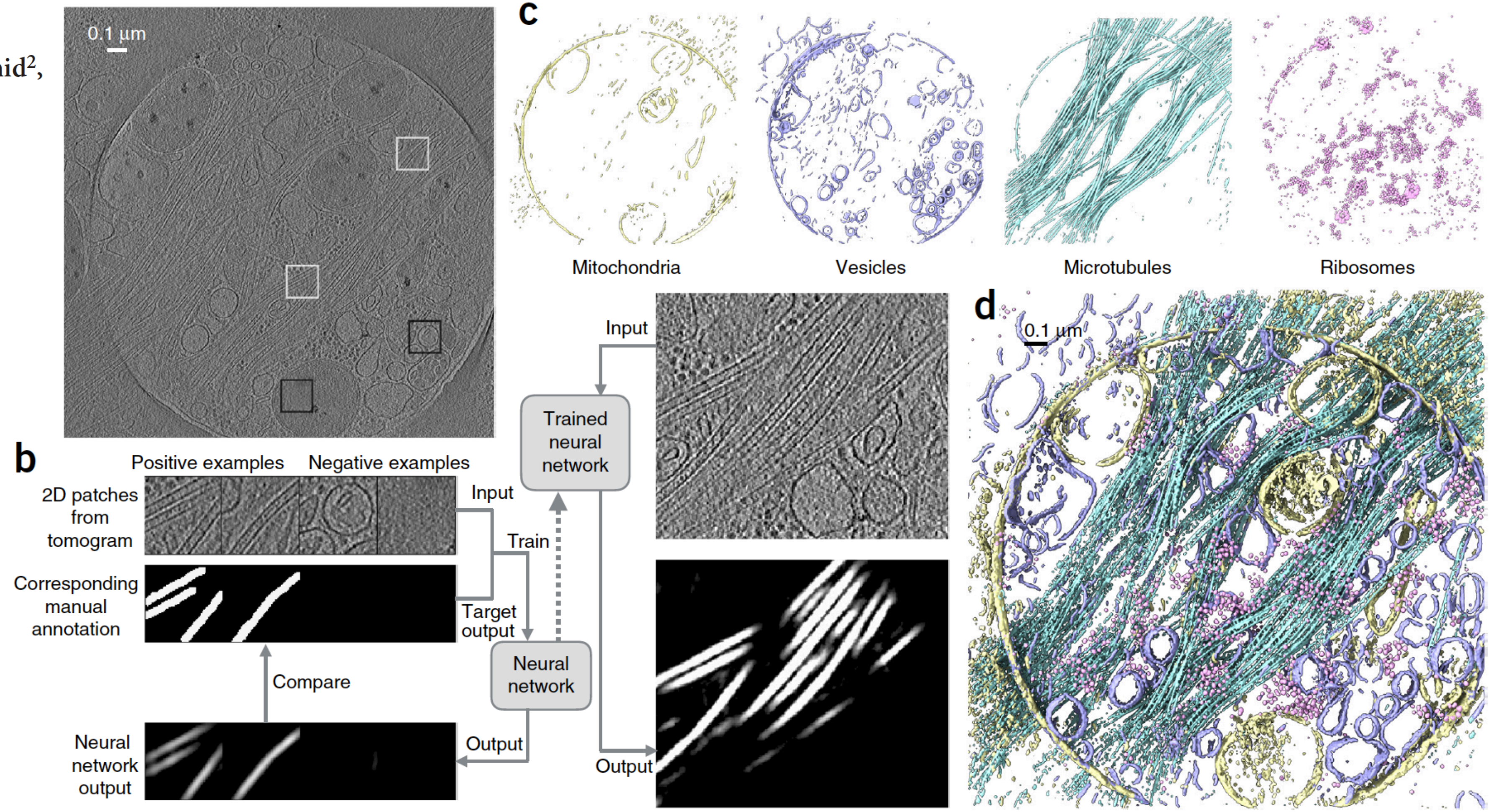
- μ_x the average of x;
- μ_y the average of y;
- σ_x^2 the variance of x;
- σ_y^2 the variance of y;
- σ_{xy} the covariance of x and y;
- $c_1=(k_1L)^2$, $c_2=(k_2L)^2$ two variables to stabilize the division with weak denominator;
- L the dynamic range of the pixel-values (typically this is $2^{\#bits\ per\ pixel}-1$);
- $k_1=0.01$ and $k_2=0.03$ by default.

deep imaging

Examples of CNN's for biomedical image analysis

Convolutional neural networks for automated annotation of cellular cryo-electron tomograms

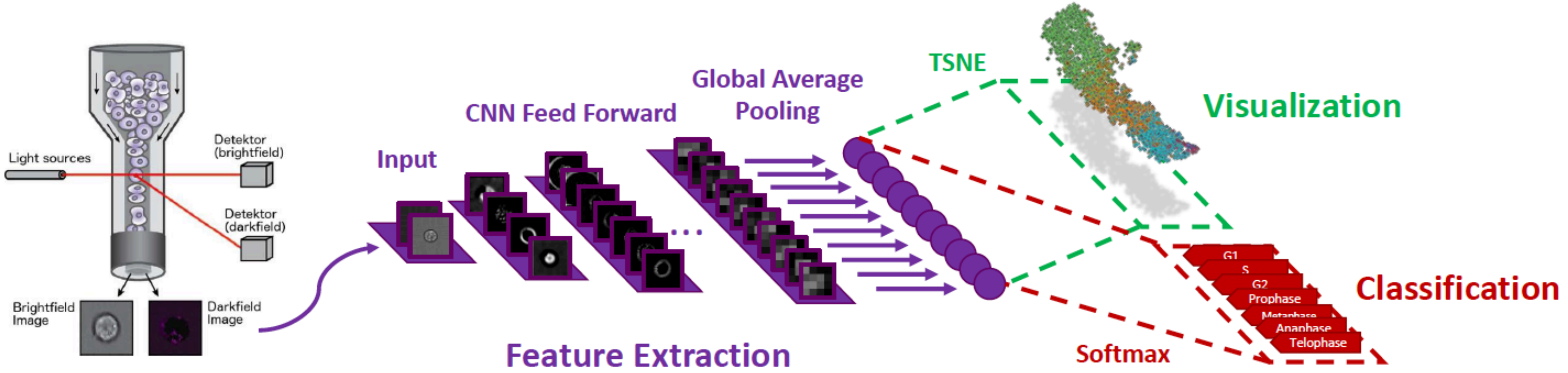
Muyuan Chen[1,2], Wei Dai[2,4], Stella Y Sun[2], Darius Jonasch[2], Cynthia Y He[3], Michael F Schmid[2], Wah Chiu[2] & Steven J Ludtke[2]

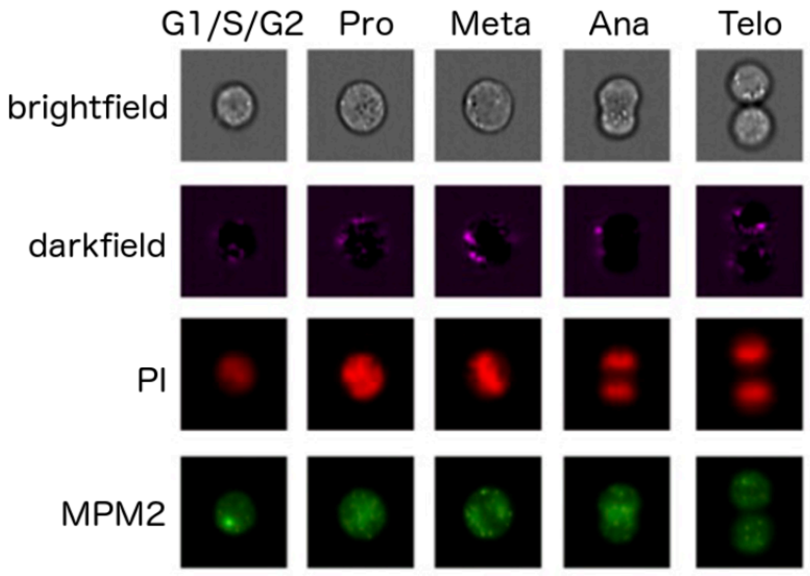


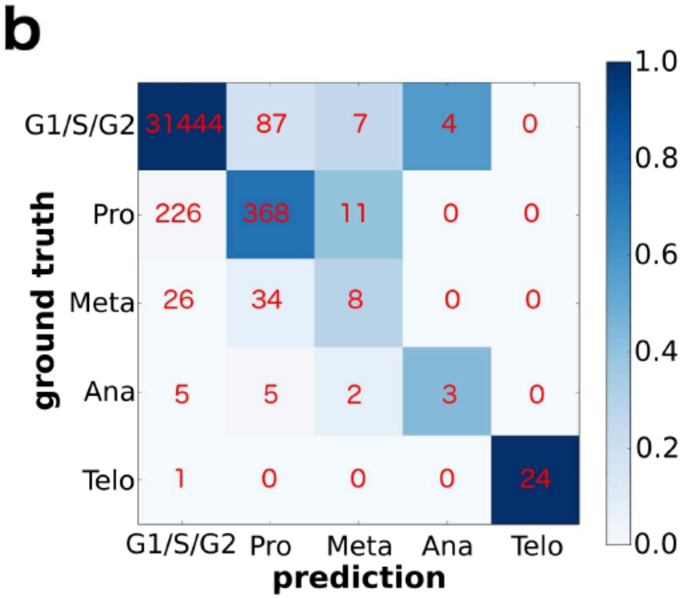
deep imaging

P. Eulenberg et al., “Reconstructing cell cycle and disease progression using deep learning”

deep imaging

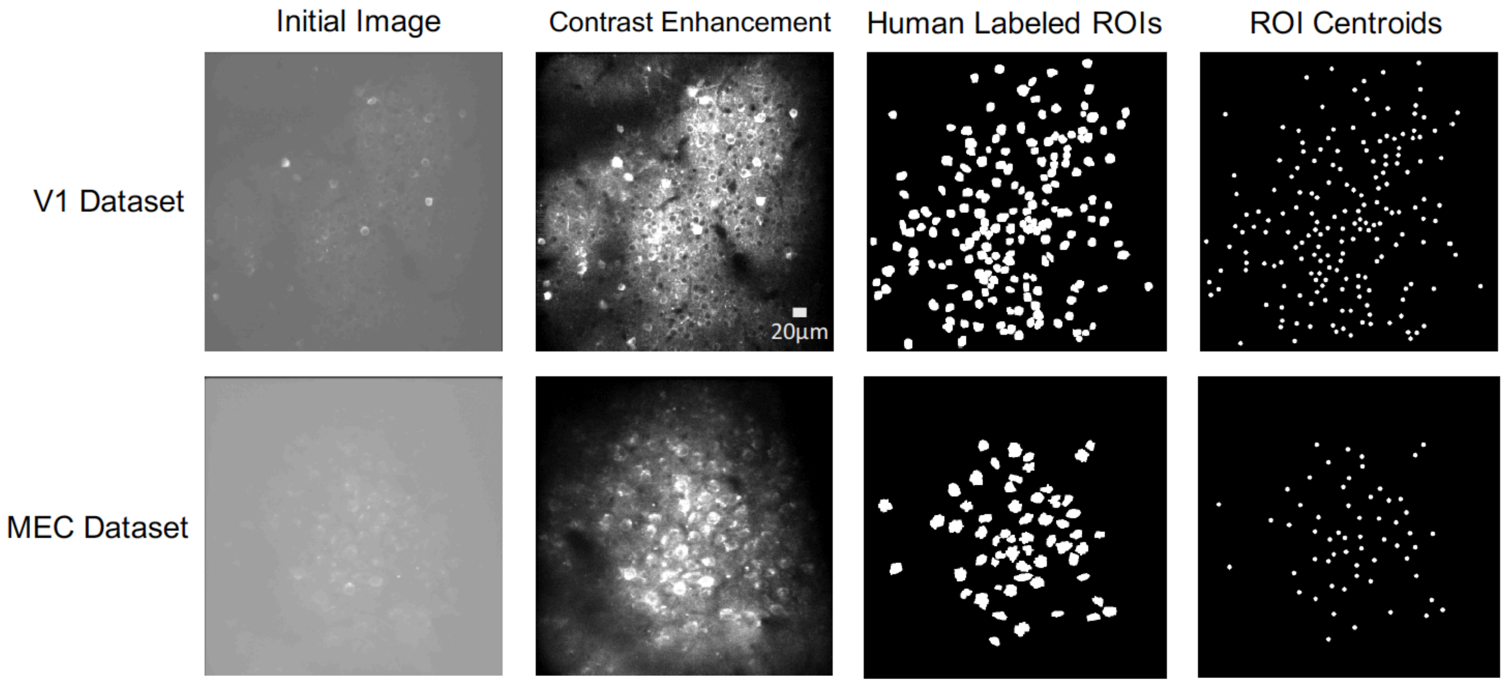


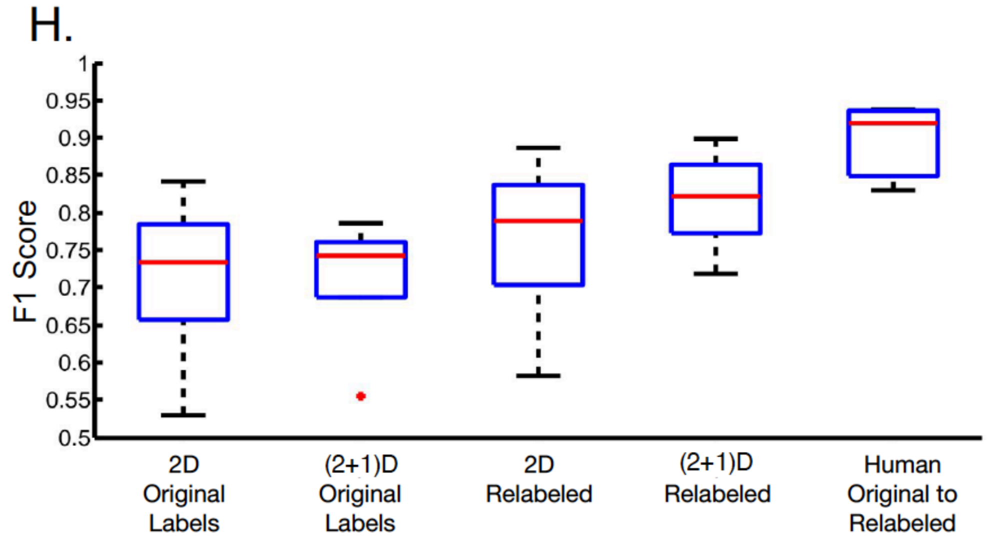


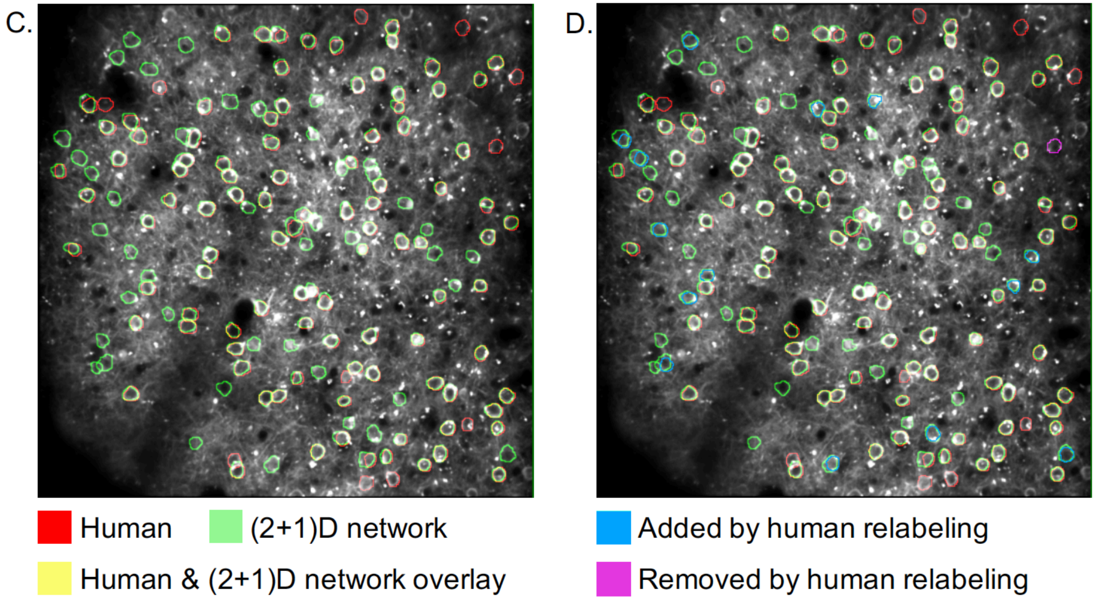


Automatic Neuron Detection in Calcium Imaging Data Using Convolutional Networks

Noah J. Apthorpe[1*] Alexander J. Riordan[2*] Rob E. Aguilar[1] Jan Homann[2]
Yi Gu[2] David W. Tank[2] H. Sebastian Seung[12]
[1]Computer Science Department [2]Princeton Neuroscience Institute
Princeton University

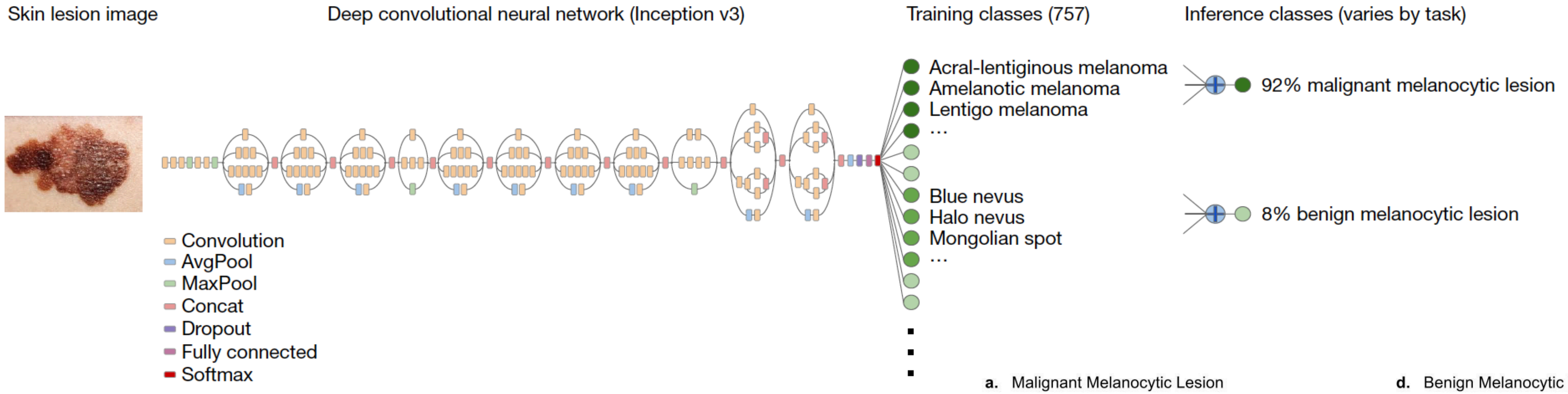


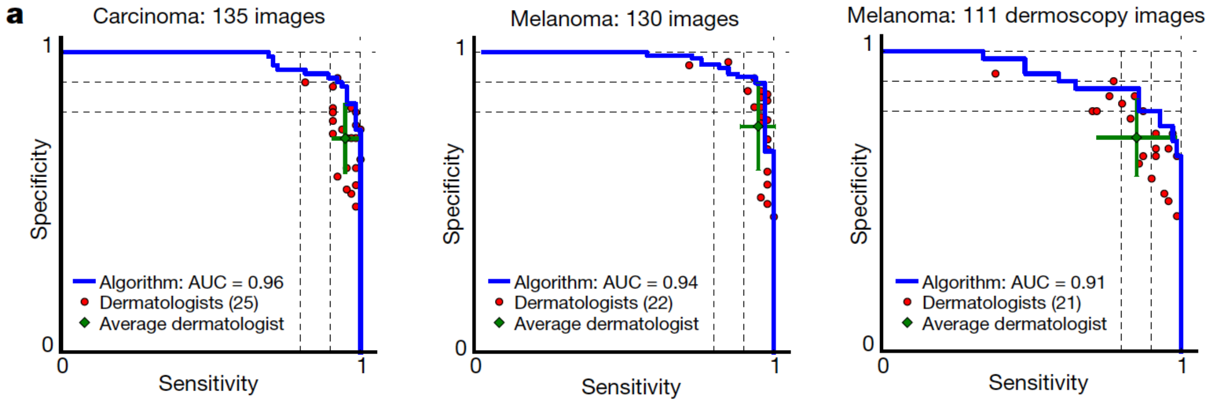


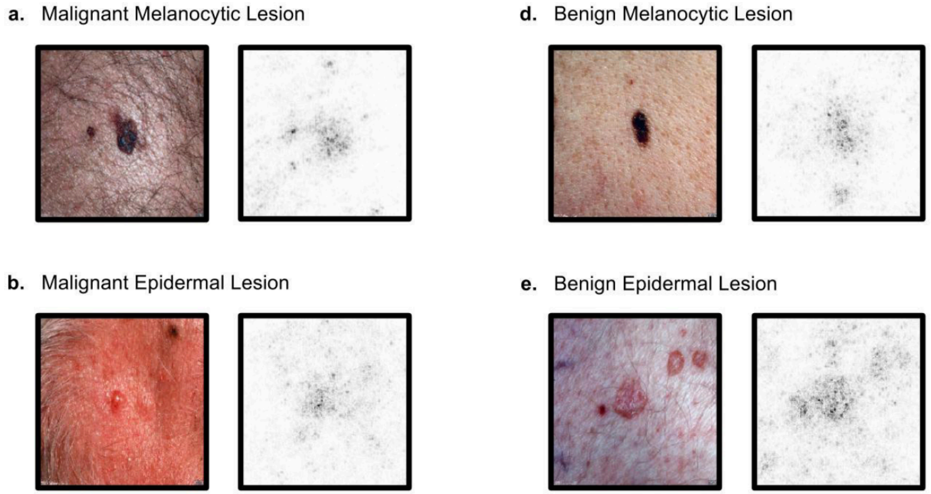


Dermatologist-level classification of skin cancer with deep neural networks

Andre Esteva[1]*, Brett Kuprel[1]*, Roberto A. Novoa[2,3], Justin Ko[2], Susan M. Swetter[2,4], Helen M. Blau[5] & Sebastian Thrun[6]

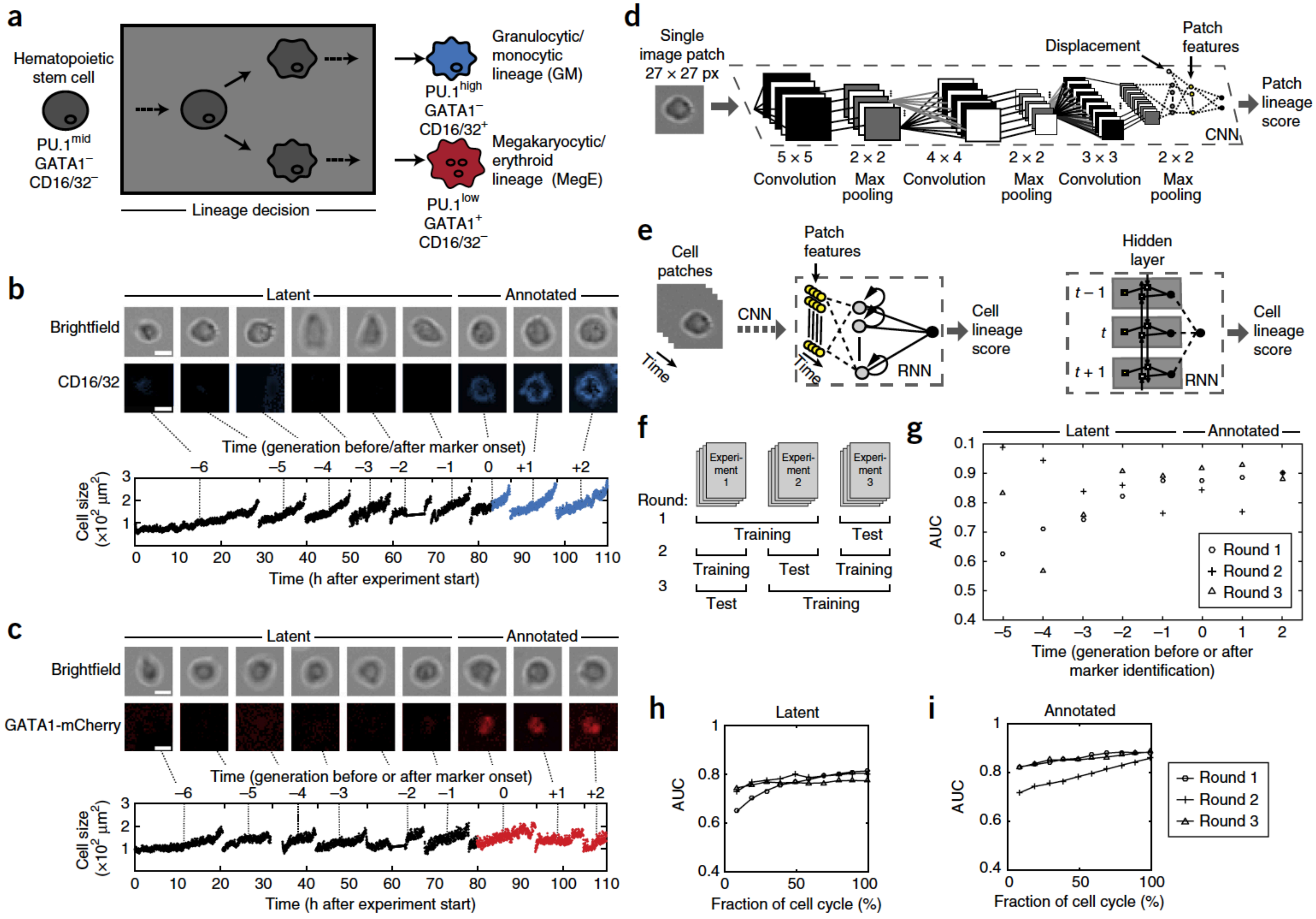






Prospective identification of hematopoietic lineage choice by deep learning

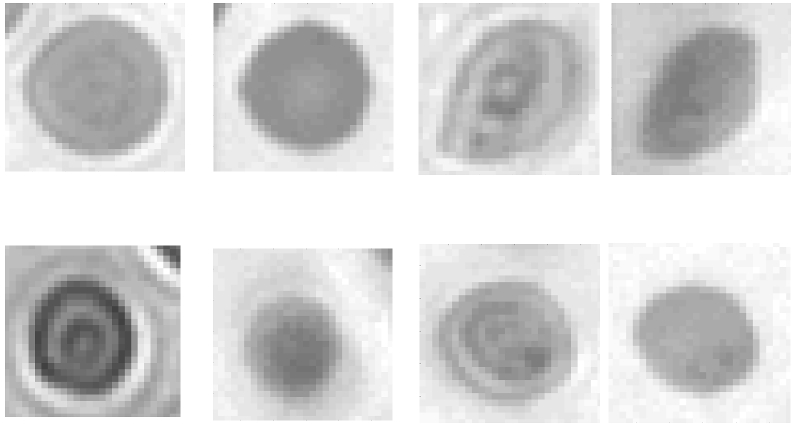
Felix Buggenthin[1,6], Florian Buettner[1,2,6],
Philipp S Hoppe[3,4], Max Endele[3], Manuel Kroiss[1,5],
Michael Strasser[1], Michael Schwarzfischer[1],
Dirk Loeffler[3,4], Konstantinos D Kokkaliaris[3,4],
Oliver Hilsenbeck[3,4], Timm Schroeder[3,4],
Fabian J Theis[1,5] & Carsten Marr[1]



Beyond statistics, how can we visualize performance for classification?

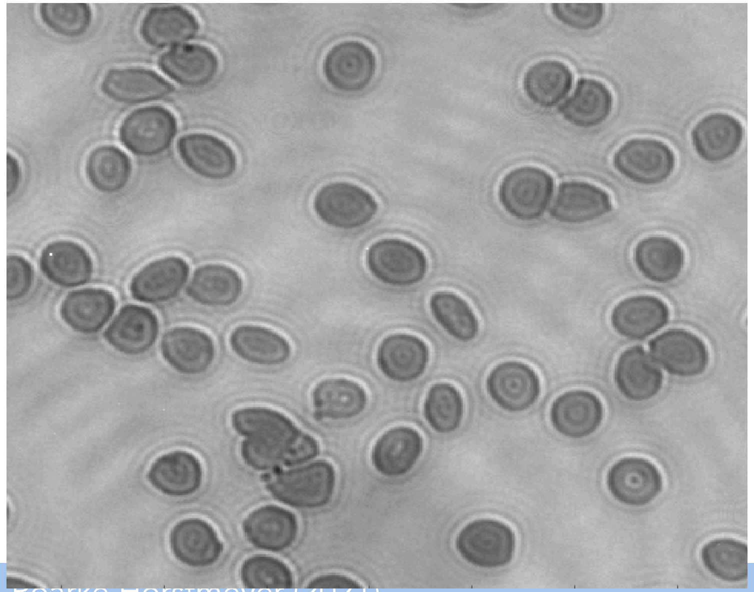
deep imaging

Training dataset



Trained CNN

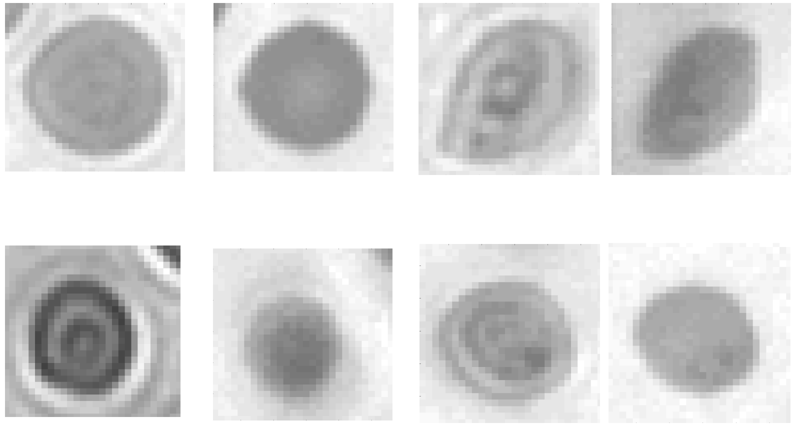
Real data is big…what to do??



Beyond statistics, how can we visualize performance for classification?

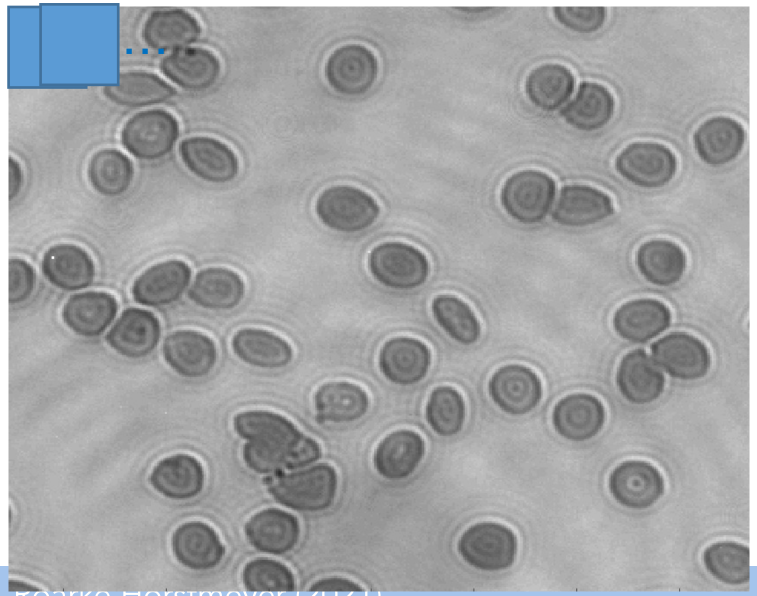
deep imaging

Training dataset



Trained CNN

Real data is big…what to do??

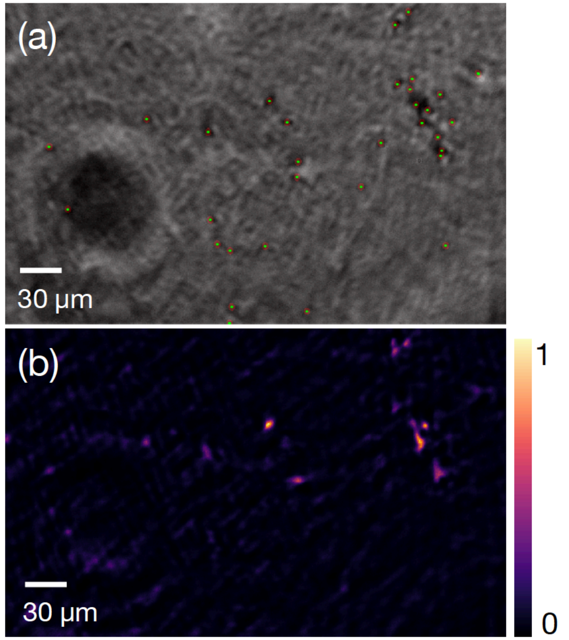


Use sliding window!

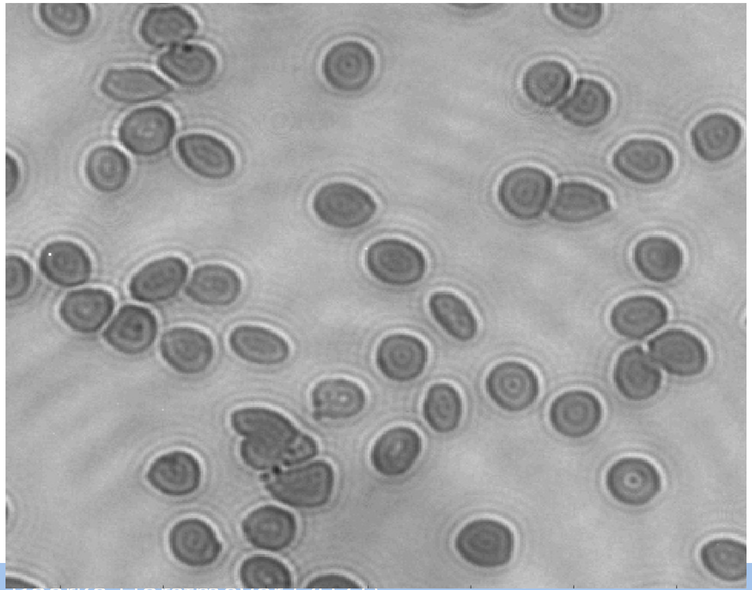
Beyond statistics, how can we visualize performance for classification?

deep imaging

Trained CNN



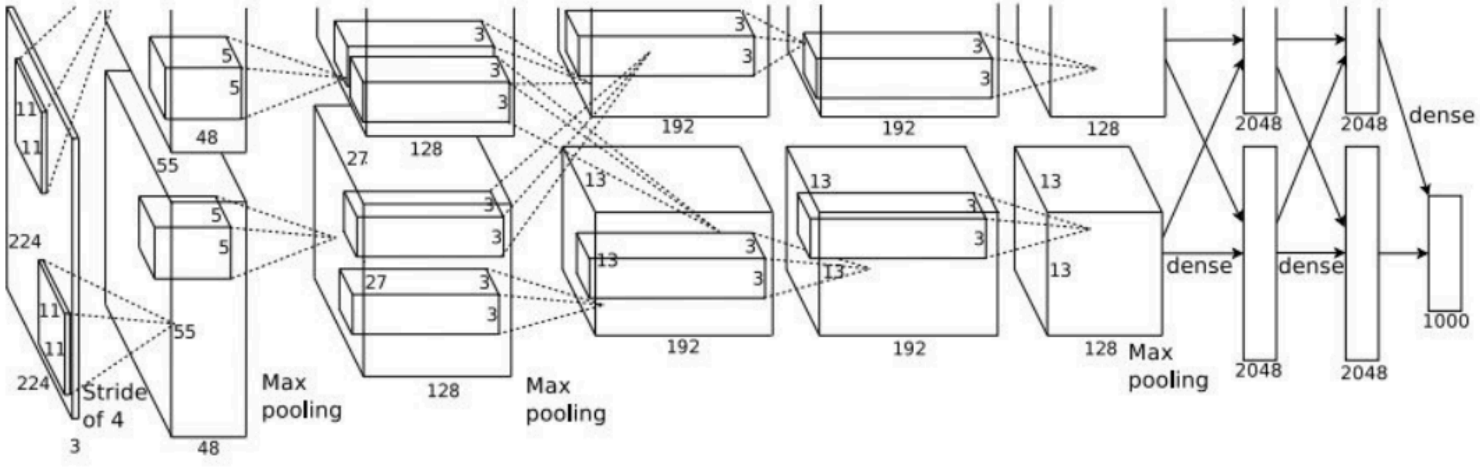
Use sliding window



How can we visualize what's in the network?

This image is CC0 public domain

Input Image:
3 x 224 x 224

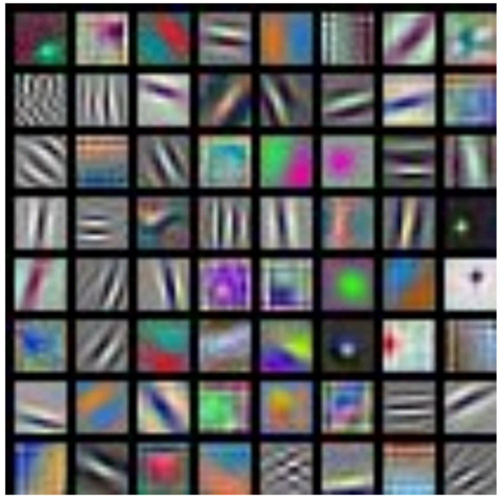


Class Scores:
1000 numbers

What are the intermediate features looking for?

Stanford CS231n: http://cs231n.stanford.edu/

First Layer: Visualize Filters

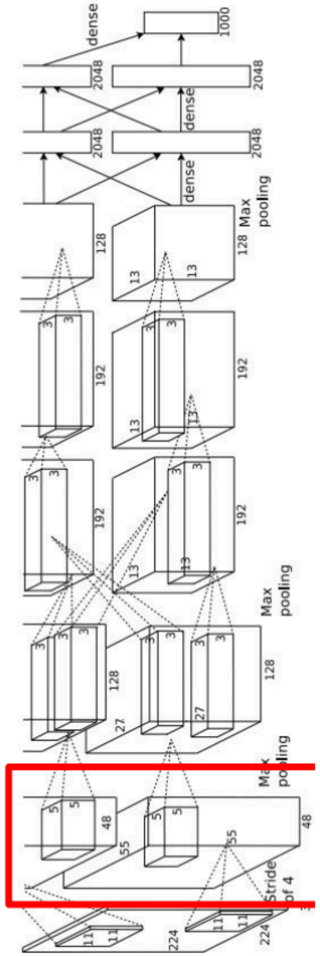


AlexNet:
64 x 3 x 11 x 11

ResNet-18:
64 x 3 x 7 x 7

ResNet-101:
64 x 3 x 7 x 7

DenseNet-121:
64 x 3 x 7 x 7



Krizhevsky, “One weird trick for parallelizing convolutional neural networks”, arXiv 2014
He et al, “Deep Residual Learning for Image Recognition”, CVPR 2016
Huang et al, “Densely Connected Convolutional Networks”, CVPR 2017

Stanford CS231n: http://cs231n.stanford.edu/

Visualize the filters/kernels (raw weights)

We can visualize filters at higher layers, but not that interesting

(these are taken from ConvNetJS CIFAR-10 demo)

layer 1 weights

16 x 3 x 7 x 7

layer 2 weights

20 x 16 x 7 x 7

layer 3 weights

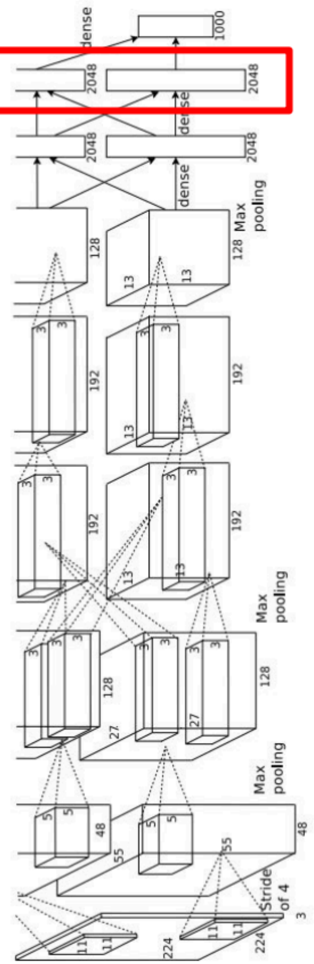
20 x 20 x 7 x 7

Stanford CS231n: http://cs231n.stanford.edu/

deep imaging

Last Layer

FC7 layer



4096-dimensional feature vector for an image
(layer immediately before the classifier)

Run the network on many images, collect the feature vectors

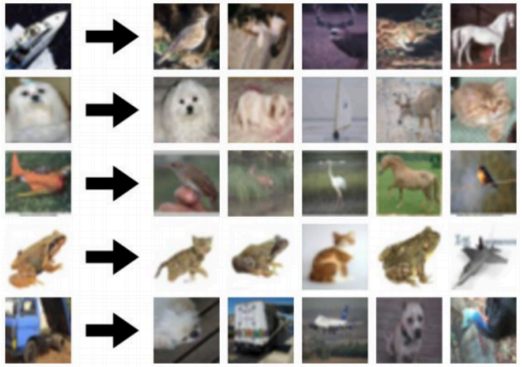
Stanford CS231n: http://cs231n.stanford.edu/

Last Layer

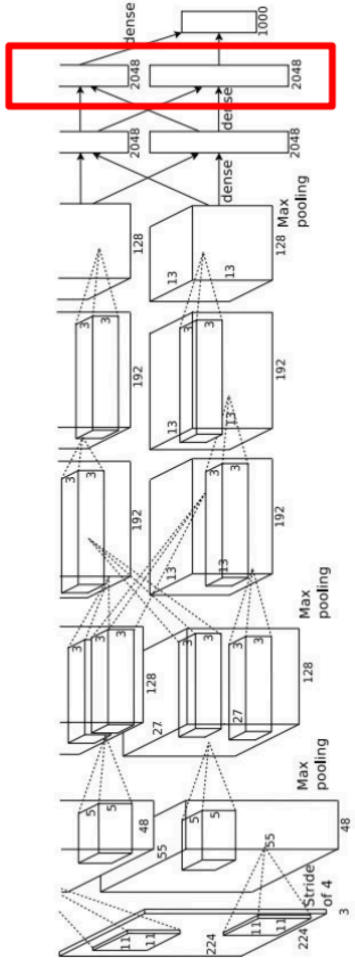
Nearest Neighbors

FC7 layer

Recall: Nearest neighbors in pixel space



Test image   L2 Nearest neighbors in feature space



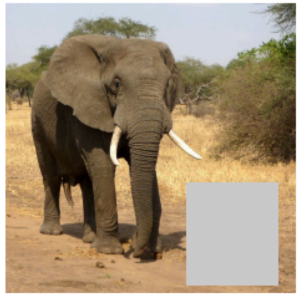
Krizhevsky et al, "ImageNet Classification with Deep Convolutional Neural Networks", NIPS 2012.
Figures reproduced with permission.

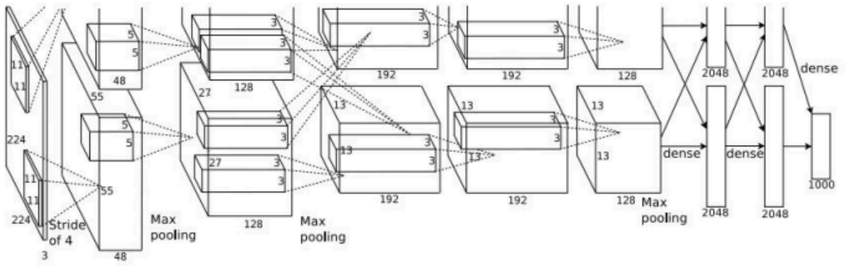
Stanford CS231n: http://cs231n.stanford.edu/

deep imaging

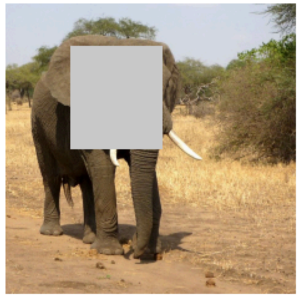
Which pixels matter: Saliency vs Occlusion

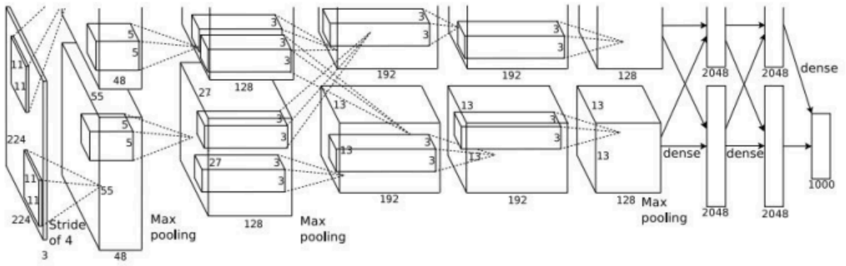
Mask part of the image before feeding to CNN, check how much predicted probabilities change





P(elephant) = 0.95





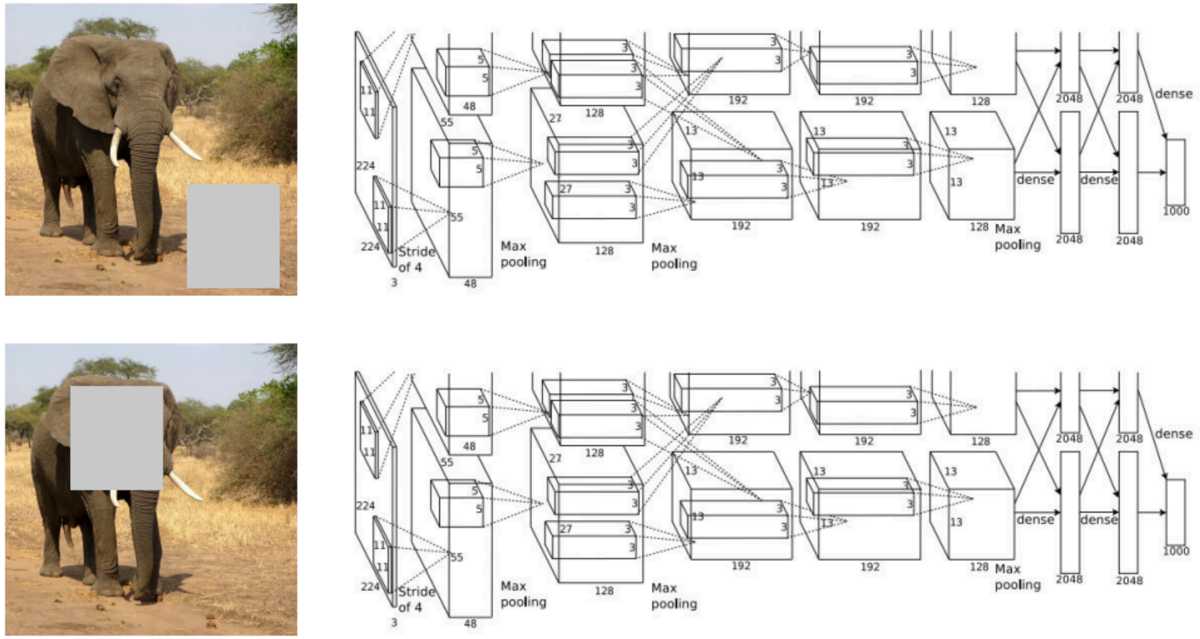
P(elephant) = 0.75

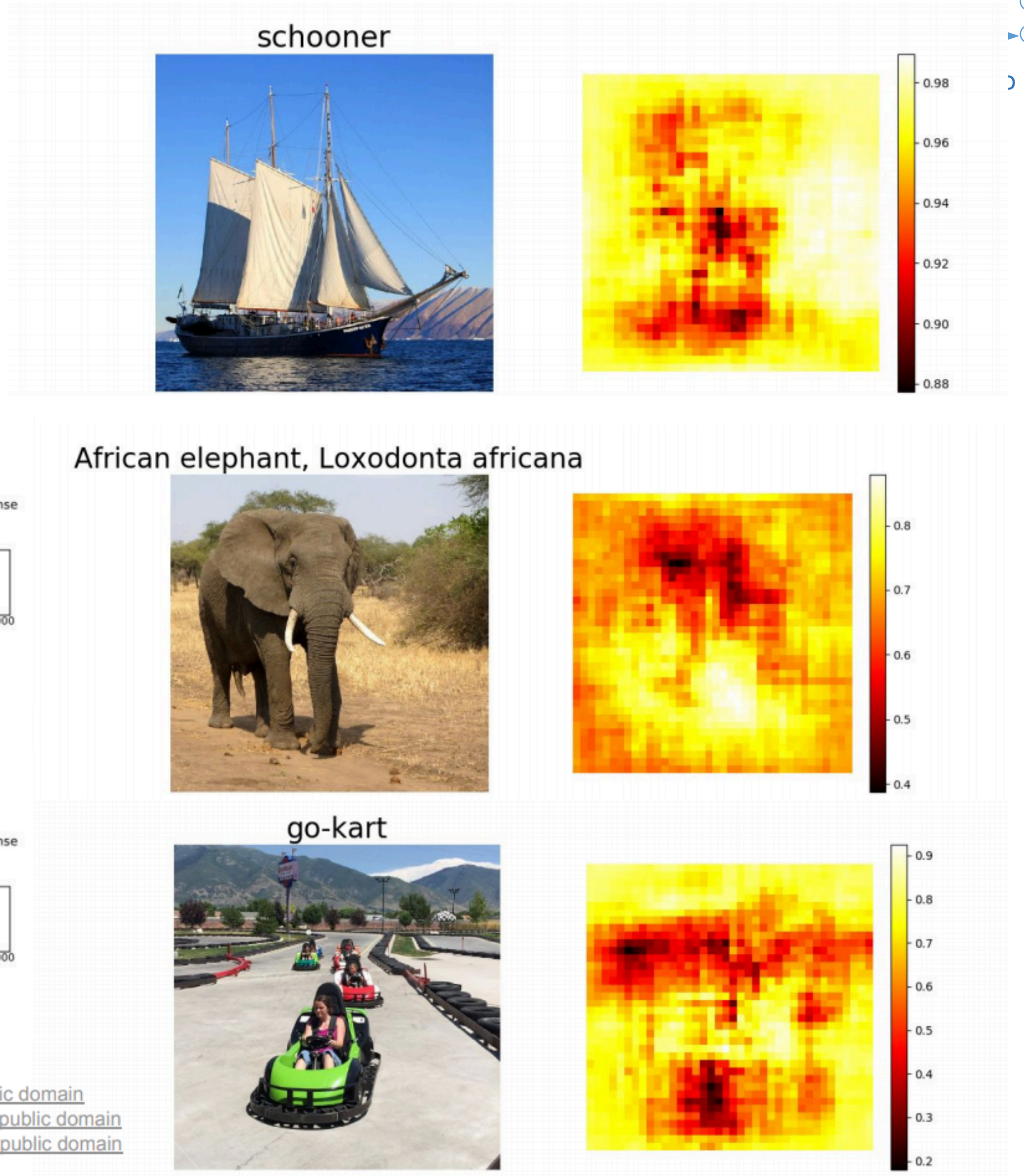
Zeiler and Fergus, "Visualizing and Understanding Convolutional Networks", ECCV 2014

Stanford CS231n: http://cs231n.stanford.edu/

Which pixels matter: Saliency vs Occlusion

Mask part of the image before feeding to CNN, check how much predicted probabilities change

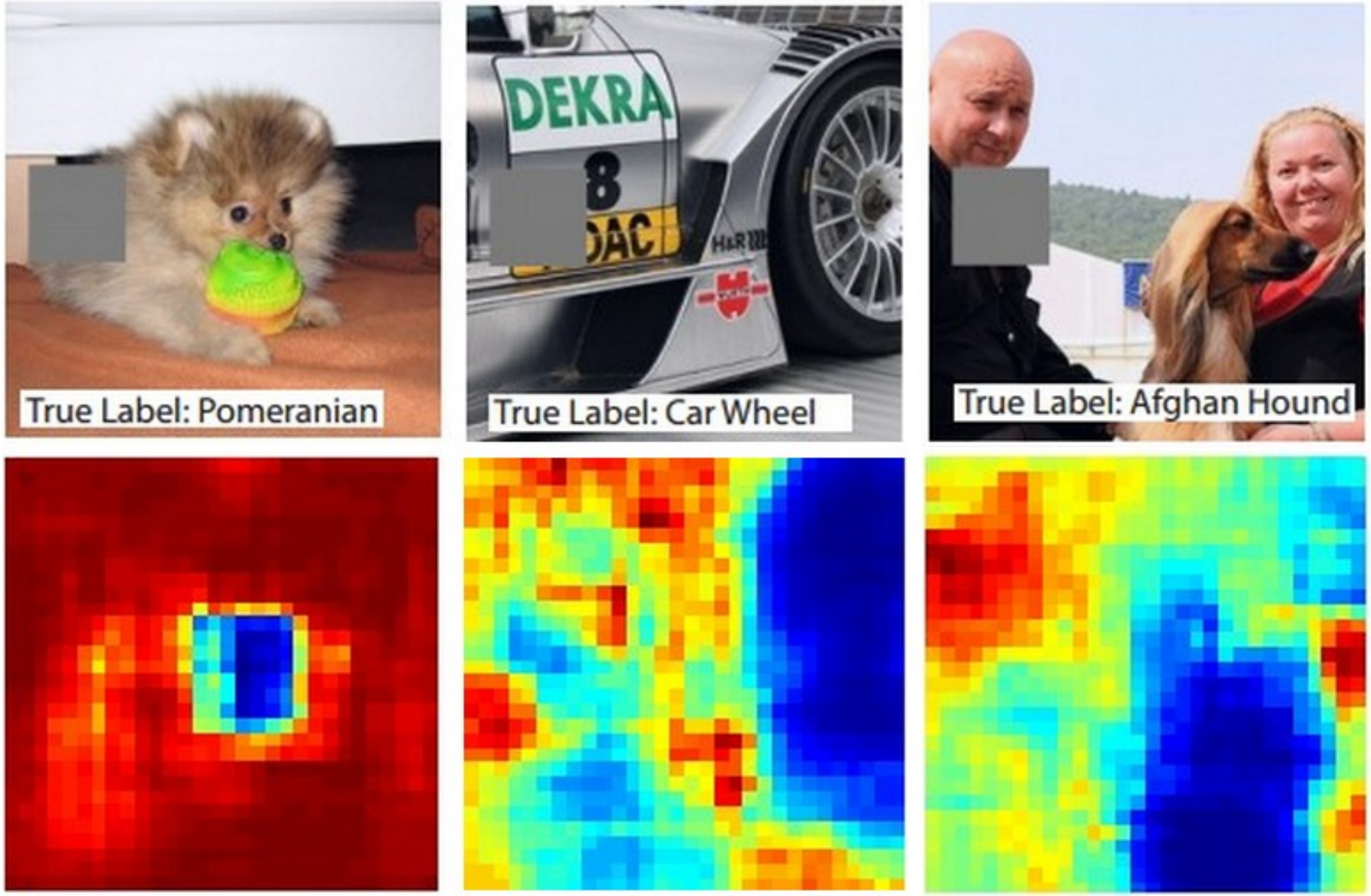




Zeiler and Fergus, "Visualizing and Understanding Convolutional Networks", ECCV 2014

Boat image is CC0 public domain
Elephant image is CC0 public domain
Go-Karts image is CC0 public domain

Stanford CS231n: http://cs231n.stanford.edu/



Stanford CS231n: http://cs231n.stanford.edu/