Recognition wrap-up
&
Self-supervised representation learning

Kristen Grauman
UT-Austin
Wed Sept 20, 2017

Announcements

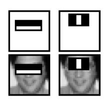
- Assignment 1 due Sept 22 11:59 pm on Canvas
- Hw2 is out and due Wed Oct 11

- Next week: CNN hands-on tutorial with Ruohan Gao and Tushar Nagarajan
  - Bring laptop
  - Set up your TACC portal account in advance

Outline

- Last time
  - Spatial verification for instance recognition
  - Recognizing categories
- Today
  - Wrap up on categories/classifiers
  - Self-supervised learning
  - External papers & assigned paper discussion
    - Shuffle and Learn (Yu-Chuan)
    - Colorization (Keivaun)
    - Curious Robot (Ginevra)
  - Experiment
    - Network dissection (Thomas and Wonjoon)

Last time: Three landmark case studies for image classification

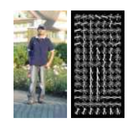


Boosting + face detection

Viola & Jones

NN + scene Gist classification

e.g., Hays & Efros



SVM + person detection

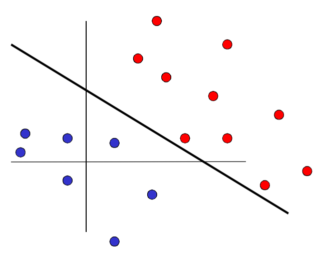
e.g., Dalal & Triggs

Slide credit: Kristen Grauman

Last time

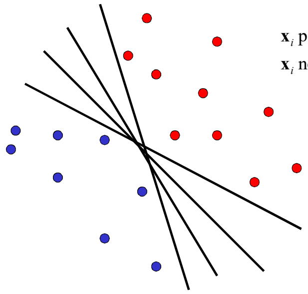
- Intro to categorization problem
- Object categorization as discriminative classification
  - Boosting + fast face detection example
  - Nearest neighbors + scene recognition example
  - Support vector machines + pedestrian detection example
    - Pyramid match kernels, spatial pyramid match
  - Convolutional neural networks + ImageNet example

Linear classifiers



Linear classifiers

- Find linear function to separate positive and negative examples

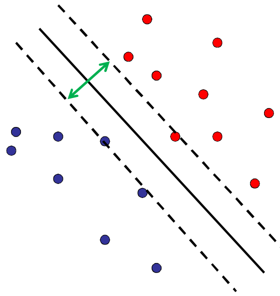


$\mathbf{x}_i$ positive : $\mathbf{x}_i \cdot \mathbf{w} + b \geq 0$

$\mathbf{x}_i$ negative : $\mathbf{x}_i \cdot \mathbf{w} + b < 0$

Which line is best?

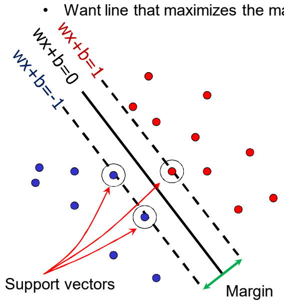
Support Vector Machines (SVMs)



- Discriminative classifier based on *optimal separating hyperplane*
- Maximize the *margin* between the positive and negative training examples

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

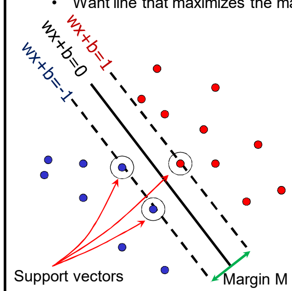
$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

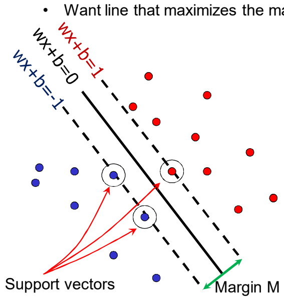
Distance between point and line: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

For support vectors:

$$\frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left|\frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|}\right| = \frac{2}{\|\mathbf{w}\|}$$

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and line: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

Finding the maximum margin line

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data points:

$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

Quadratic optimization problem:

Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to $y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1$

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

learned weight

Support vector

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ (for any support vector)

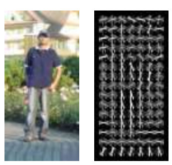
  $\mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$

- Classification function:

$$f(x) = \text{sign}\,(\mathbf{w} \cdot \mathbf{x} + \text{b})$$
$$= \text{sign}\left(\sum_i \alpha_i y_i \boxed{\mathbf{x}_i \cdot \mathbf{x}} + b\right)$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery,

Person detection with HoG's & linear SVM's



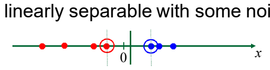
- Map each grid cell in the input window to a histogram counting the gradients per orientation.
- Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

Dalal & Triggs, CVPR 2005

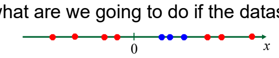
Code available: http://pascal.inrialpes.fr/soft/olt/

Non-linear SVMs

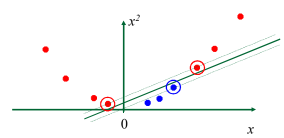
- Datasets that are linearly separable with some noise work out great:



- But what are we going to do if the dataset is just too hard?



- How about… mapping data to a higher-dimensional space:



Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\varphi(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$$

- This gives a nonlinear decision boundary in the original feature space:

$$\sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

Example

2-dimensional vectors $\mathrm{x}=[x_1\ \ x_2]$;

let $K(\mathrm{x_i},\mathrm{x_j})=(1 + \mathrm{x_i}^{\mathrm{T}}\mathrm{x_j})^2$

Need to show that $K(\mathrm{x_i},\mathrm{x_j})= \varphi(\mathrm{x_i})^{\mathrm{T}}\varphi(\mathrm{x_j})$:

$K(\mathrm{x_i},\mathrm{x_j})=(1 + \mathrm{x_i}^{\mathrm{T}}\mathrm{x_j})^2,$

$= 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}$

$= [1\ \ x_{i1}^2\ \ \sqrt{2}\, x_{i1}x_{i2}\ \ x_{i2}^2\ \ \sqrt{2}x_{i1}\ \ \sqrt{2}x_{i2}]^{\mathrm{T}}$

$[1\ \ x_{j1}^2\ \ \sqrt{2}\, x_{j1}x_{j2}\ \ x_{j2}^2\ \ \sqrt{2}x_{j1}\ \ \sqrt{2}x_{j2}]$

$= \varphi(\mathrm{x_i})^{\mathrm{T}}\varphi(\mathrm{x_j}),$

where $\varphi(\mathrm{x}) = [1\ \ x_1^2\ \ \sqrt{2}\, x_1x_2\ \ x_2^2\ \ \sqrt{2}x_1\ \ \sqrt{2}x_2]$

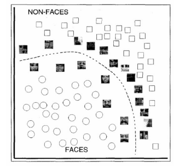
Examples of kernel functions

- Linear: $K(x_i, x_j) = x_i^T x_j$
- Gaussian RBF: $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$
- Histogram intersection:

$$K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k))$$

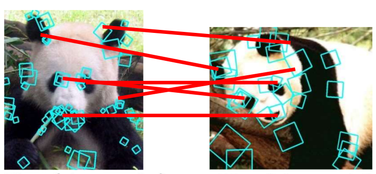
SVMs for recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples
4. Use this "kernel matrix" to solve for SVM support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.



Kristen Grauman

What about a *matching* kernel?

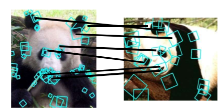


$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Local feature correspondence useful similarity measure for generic object categories

Kristen Grauman

Partially matching sets of features



$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Optimal match: O(m³)
Greedy match: O(m² log m)
Pyramid match: O(m)

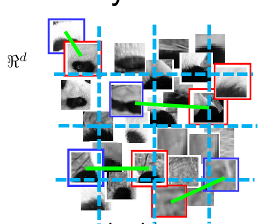
(**m**=num pts)

$\min_{\pi: \mathbf{X} \to \mathbf{Y}} \sum_{\mathbf{x}_i \in \mathbf{X}} ||\mathbf{x}_i - \pi(\mathbf{x}_i)||$ nate matching kernel that makes it practical to compare large sets of features based on their partial correspondences.

[Previous work: Indyk & Thaper, Bartal, Charikar, Agarwal & Varadarajan, …]

Kristen Grauman

Pyramid match: main idea

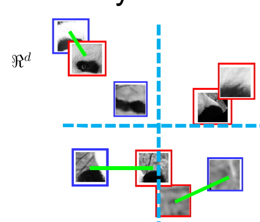


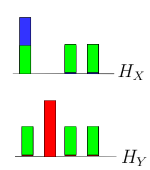
Feature space partitions serve to "match" the local descriptors within successively wider regions.

$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Kristen Grauman

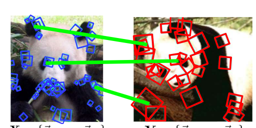
Pyramid match: main idea





$$\mathcal{I}(H_X, H_Y) = \sum_j \min(H_X(j), H_Y(j))$$
$$= 3$$

Histogram intersection counts number of possible matches at a given partitioning.



$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Kristen Grauman

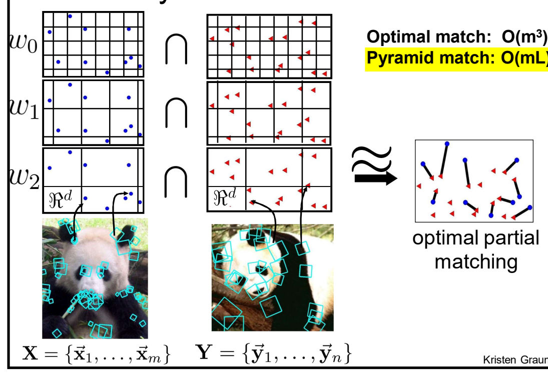
Pyramid match kernel

$$K_\Delta(X,Y)=\sum_{i=0}^{L}\underbrace{2^{-i}}_{\substack{\text{measures}\\ \text{difficulty of a}\\ \text{match at level } i}}\underbrace{\left(\mathcal{I}\left(H_X^{(i)},H_Y^{(i)}\right)-\mathcal{I}\left(H_X^{(i-1)},H_Y^{(i-1)}\right)\right)}_{\substack{\text{number of newly matched}\\ \text{pairs at level } i}}$$

- For similarity, weights inversely proportional to bin size (or may be learned)
- Normalize these kernel values to avoid favoring large sets

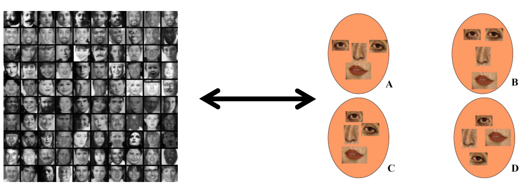
[Grauman & Darrell, ICCV 2005]

Pyramid match kernel



Kristen Grauman

Unordered sets of local features: **No** spatial layout preserved!

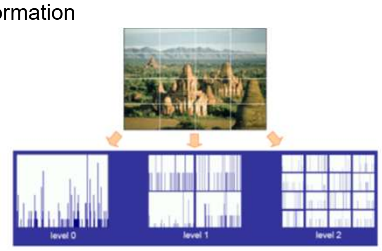


Too much?

Too little?

Spatial pyramid match

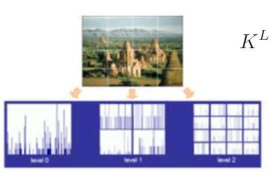
- Make a pyramid of bag-of-words histograms.
- Provides some loose (global) spatial layout information



[Lazebnik, Schmid & Ponce, CVPR 2006]

Spatial pyramid match

- Make a pyramid of bag-of-words histograms.
- Provides some loose (global) spatial layout information



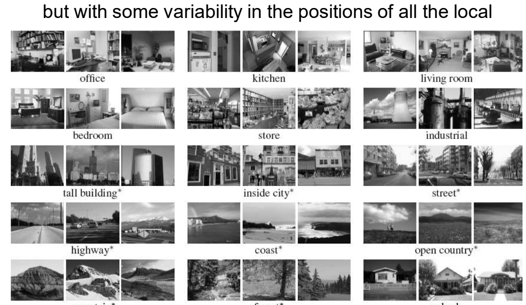
$$K^L(X,Y)=\sum_{m=1}^{M}\kappa^L(X_m,Y_m)$$

Sum over PMKs computed in *image coordinate* space, one per word.

[Lazebnik, Schmid & Ponce, CVPR 2006]

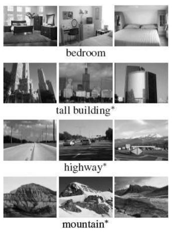
Spatial pyramid match

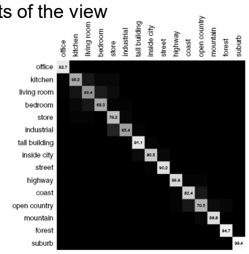
- Can capture **scene** categories well---texture-like patterns but with some variability in the positions of all the local



Spatial pyramid match

- Can capture **scene** categories well---texture-like patterns but with some variability in the positions of all the local pieces.
- Sensitive to global shifts of the view





Confusion table

SVMs: Pros and cons

- Pros
  - Kernel-based framework is very powerful, flexible
  - Often a sparse set of support vectors – compact at test time
  - Work very well in practice, even with very small training sample sizes
- Cons
  - No "direct" multi-class SVM, must combine two-class SVMs
  - Can be tricky to select best kernel function for a problem
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

Adapted from Lana Lazebnik

Recall: Evolution of methods

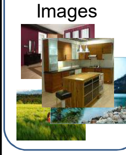
- Hand-crafted models
- 3D geometry
- Hypothesize and align

- Hand-crafted features
- Learned models
- Data-driven

- "End-to-end" learning of features and models*,**

Traditional Image Categorization: Training phase

Training Images



Training

Training Labels

Image Features → Classifier Training → Trained Classifier

Slide credit: Jia-Bin Huang

Traditional Image Categorization: Testing phase

Training Images

Training

Training Labels

Image Features → Classifier Training → Trained Classifier

Testing

Test Image

Image Features → Trained Classifier → Prediction **Outdoor**

Slide credit: Jia-Bin Huang

Learning a Hierarchy of Feature Extractors

- Each layer of hierarchy extracts features from output of previous layer
- All the way from pixels → classifier
- Layers have the (nearly) same structure

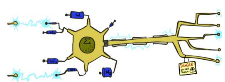
Image/video → Layer 1 → Layer 2 → Layer 3 → Labels

- Train all layers jointly

Slide: Rob Fergus

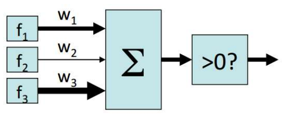
Neuron: Linear Perceptron

- Inputs are feature values
- Each feature has a weight
- Sum is the activation



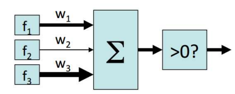
$$\text{activation}_w(x) = \sum_i w_i \cdot f_i(x) = w \cdot f(x)$$

- If the activation is:
  - Positive, output +1
  - Negative, output -1



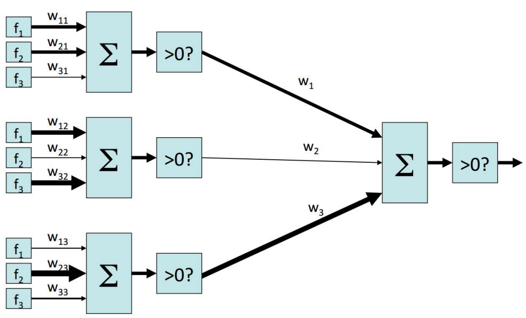
Slide credit: Pieter Abeel and Dan Klein

Two-layer perceptron network



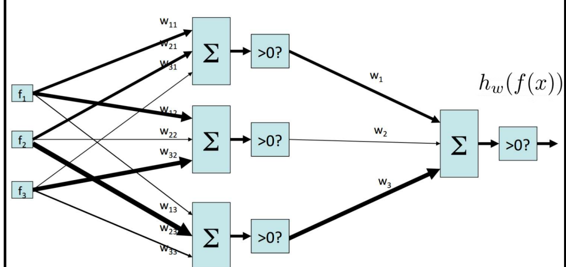
Slide credit: Pieter Abeel and Dan Klein

Two-layer perceptron network



Slide credit: Pieter Abeel and Dan Klein

Two-layer perceptron network



Slide credit: Pieter Abeel and Dan Klein

Convolutional Neural Networks (CNN, ConvNet, DCN)

- CNN = a multi-layer neural network with
  - **Local** connectivity:
    - Neurons in a layer are only connected to a small region of the layer before it
  - **Share** weight parameters across spatial positions:
    - Learning shift-invariant filter kernels

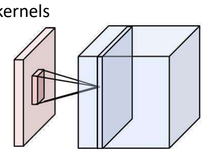
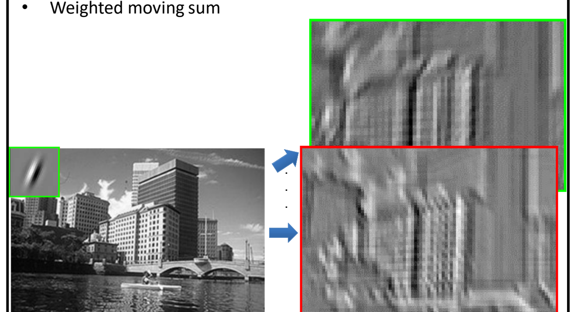


Image credit: A. Karpathy

Jia-Bin Huang and Derek Hoiem, UIUC

What is a Convolution?

- Weighted moving sum

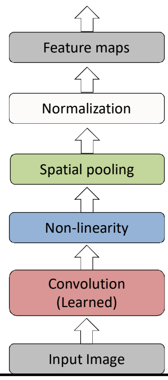


Input

Feature Activation Map

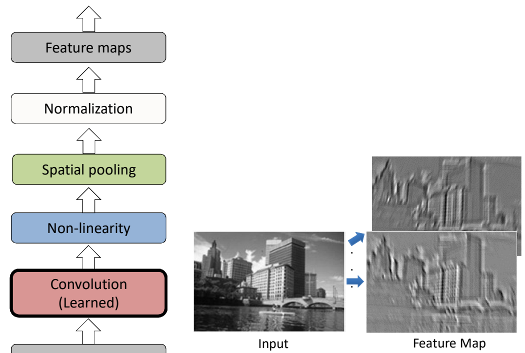
slide credit: S. Lazebnik

Convolutional Neural Networks



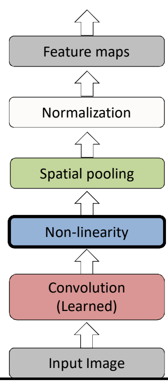
slide credit: S. Lazebnik

Convolutional Neural Networks

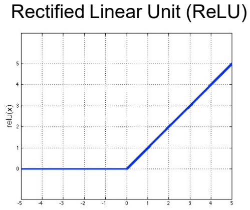


slide credit: S. Lazebnik

Convolutional Neural Networks

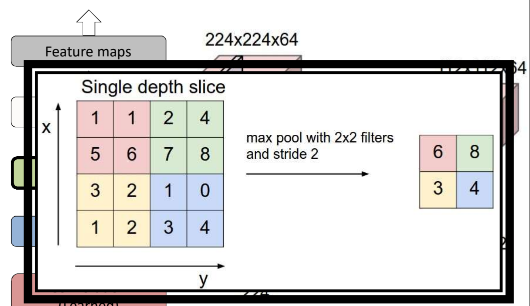


Rectified Linear Unit (ReLU)



slide credit: S. Lazebnik

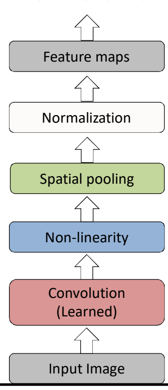
Convolutional Neural Networks



Provide *translation invariance*

slide credit: S. Lazebnik

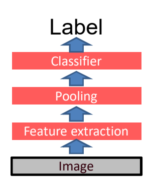
Convolutional Neural Networks

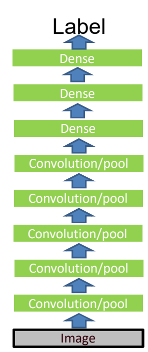


slide credit: S. Lazebnik

Engineered vs. learned features

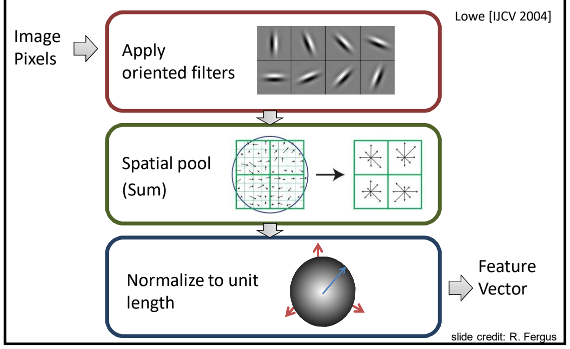
Convolutional filters are trained in a supervised manner by back-propagating classification error



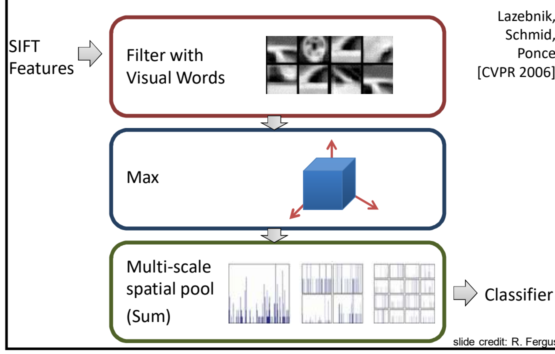


Jia-Bin Huang and Derek Hoiem, UIUC

SIFT Descriptor

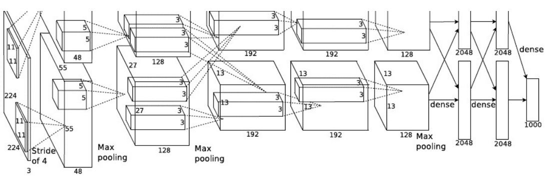


Spatial Pyramid Matching



AlexNet

- Similar framework to LeCun'98 but:
  - Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
  - More data (10^6 vs. 10^3 images)
  - GPU implementation (50x speedup over CPU)
    - Trained on two GPUs for a week

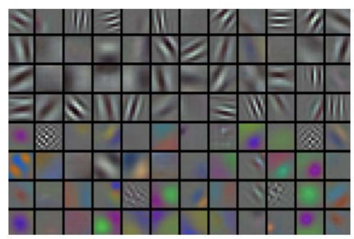


A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

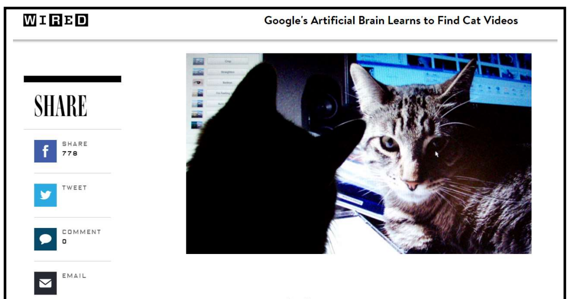
Jia-Bin Huang and Derek Hoiem, UIUC

Visualizing what was learned

- What do the learned filters look like?



Typical first layer filters



BY LIAT CLARK, *Wired UK*

When computer scientists at Google's mysterious X lab built a neural network of 16,000 computer processors with one billion connections and let it browse YouTube, it did what

https://www.wired.com/2012/06/google-x-neural-network/

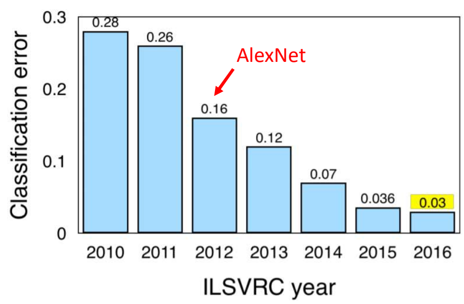
Application: ImageNet

[Deng et al. CVPR 2009]

- ~14 million labeled images, 20k classes
- Images gathered from Internet
- Human labels via Amazon Turk

https://sites.google.com/site/deeplearningcvpr2014 Slide: R. Fergus

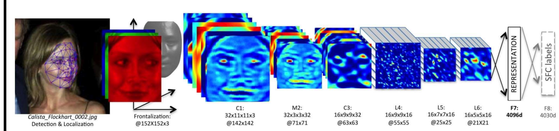
ImageNet Classification Challenge



http://image-net.org/challenges/talks/2016/ILSVRC2016_10_09_clsloc.pdf

Industry Deployment

- Used in Facebook, Google, Microsoft
- Image Recognition, Speech Recognition, ….
- Fast at test time



Taigman et al. DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CVPR'14

Slide: R. Fergus

Beyond classification

- Detection
- Segmentation
- Regression
- Pose estimation
- Matching patches
- Synthesis

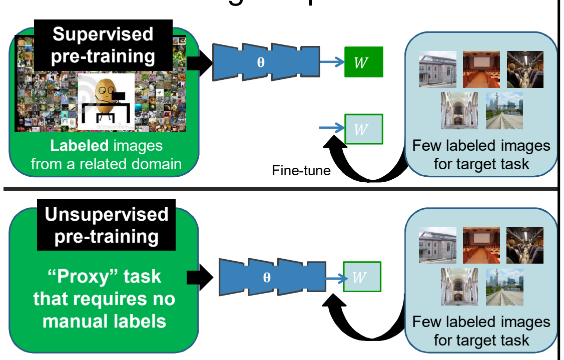
and many more…

Jia-Bin Huang and Derek Hoiem, UIUC

Recap

- Neural networks / multi-layer perceptrons
  - View of neural networks as learning hierarchy of features
- Convolutional neural networks
  - Architecture of network accounts for image structure
  - "End-to-end" recognition from pixels
  - Together with big (labeled) data and lots of computation → major success on benchmarks, image classification and beyond

Pre-training a representation



New forms of self-supervision

- **What can be our "proxy" or "pretext" task?**
- *Temporal coherence in video*
  - *Mobahi et al. 2009, Wang & Gupta 2015, Wang et al. 2016, Gao et al. 2016,…*
- *Audio channel – ambient sounds*
  - *Owens et al. 2016, Arandjelovic & Zisserman 2017*
- *Ego-motion*
  - *Jayaraman et al. 2015, Agrawal et al. 2015*
- *Spatial context, patch layout*
  - *Doersch et al. 2015, Noroozi & Favaro 2016*
- *In-painting missing pixels*
  - *Pathak et al. 2016*
- *Colorization*
  - *Larsson et al. 2016, Zheng et al. 2016*
- *Temporal order of frames*
  - *Misra et al. 2016*

Evaluation of self-supervised rep

How to test quality of unsupervised pre-training?

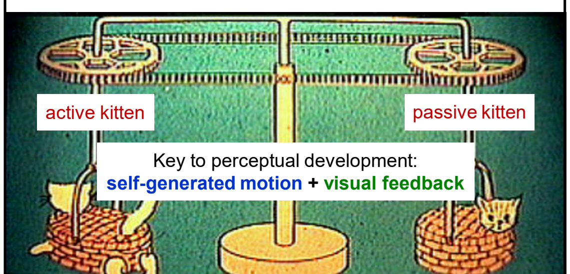
Comparisons against
- Equally supervised, but without unsup pretrain
- Fully supervised pre-training (ImageNet)
- Same network with random weights
- Counting “object-selective units” (Owens et al.)

Raw representation, +/- fine-tuning to a task

(Ego)motion for self-supervision

Dinesh Jayaraman and Kristen Grauman
Department of Computer Science
University of Texas at Austin
THE UNIVERSITY OF TEXAS AT AUSTIN

The kitten carousel experiment
[Held & Hein, 1963]



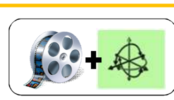
Big picture goal: Embodied vision

Status quo:
Learn from “disembodied” bag of labeled snapshots.

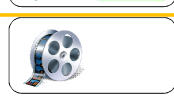
Goal:
Learn in the context of acting and moving in the world.

Two formulations

1. Learning representations tied to ego-motion



2. Learning representations from unlabeled video



Our idea: Ego-motion ↔ vision

Goal: Teach computer vision system the connection: “how I move” ↔ “how my visual surroundings change”

Ego-motion motor signals

\+

Unlabeled video

[Jayarman & Grauman, ICCV 2015]

Our idea: Ego-motion ↔ vision

Goal: Teach computer vision system the connection:
"how I move" ↔ "how my visual surroundings change"

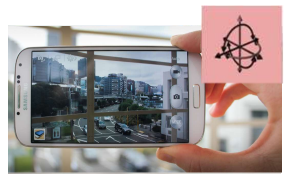
\+

Ego-motion motor signals **Unlabeled video**

[Jayaraman & Grauman, ICCV 2015]

Our idea: Ego-motion ↔ vision

Goal: Teach computer vision system the connection:
"how I move" ↔ "how my visual surroundings change"



\+

Ego-motion motor signals **Unlabeled video**

[Jayaraman & Grauman, ICCV 2015]

Ego-motion ↔ vision: view prediction

After moving:

Approach idea: Ego-motion equivariance

Invariant features: unresponsive to some classes of transformations

$$\mathbf{z}(g\mathbf{x}) \approx \mathbf{z}(\mathbf{x})$$

Simard et al, Tech Report, '98
Wiskott et al, Neural Comp '02
Hadsell et al, CVPR '06
Mobahi et al, ICML '09
Zou et al, NIPS '12
Sohn et al, ICML '12
Cadieu et al, Neural Comp '12
Goroshin et al, ICCV '15
Lies et al, PLoS computation biology '14
…

Approach idea: Ego-motion equivariance

Invariant features: unresponsive to some classes of transformations

$$\mathbf{z}(g\mathbf{x}) \approx \mathbf{z}(\mathbf{x})$$

Equivariant features: *predictably* responsive to some classes of transformations, through simple mappings (e.g., linear)

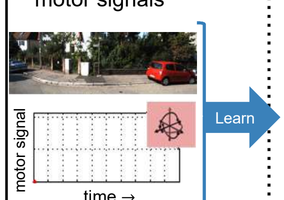
"equivariance map"

$$\mathbf{z}(g\mathbf{x}) \approx M_g \mathbf{z}(\mathbf{x})$$

Invariance *discards* information;
equivariance *organizes* it.

Approach idea: Ego-motion equivariance

Training data
Unlabeled video +
motor signals



Equivariant embedding
organized by ego-motions

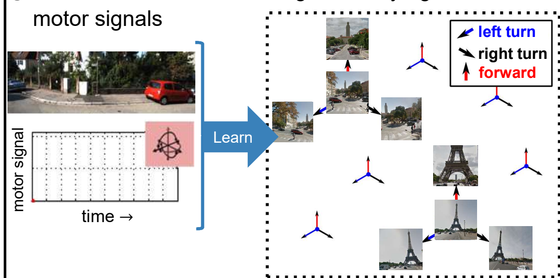
Pairs of frames related by similar ego-motion should be related by same feature transformation

[Jayaraman & Grauman, ICCV 2015]

Approach idea: Ego-motion equivariance

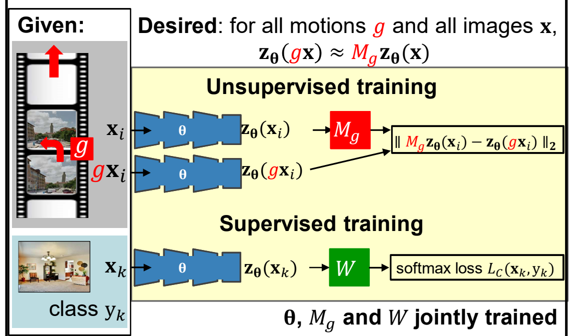
Training data
Unlabeled video + motor signals

Equivariant embedding
organized by ego-motions



[Jayaraman & Grauman, ICCV 2015]

Ego-motion equivariant feature learning



[Jayaraman & Grauman, ICCV 2015]

Results: Recognition

Learn from **unlabeled car video** (KITTI)

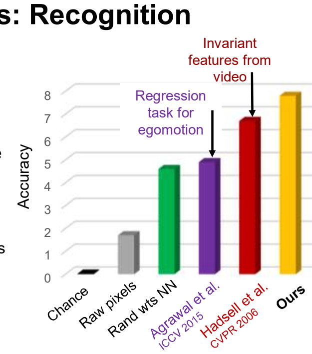
Geiger et al, IJRR '13

Exploit features for **static scene classification**
(SUN, 397 classes)

Xiao et al, CVPR '10

Results: Recognition

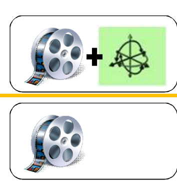
- Purely unsupervised feature learning
- *k*-nearest neighbor scene classification task in learned feature space
  - Unlabeled video: KITTI
  - Images: SUN, 397 classes
  - 50 labels per class



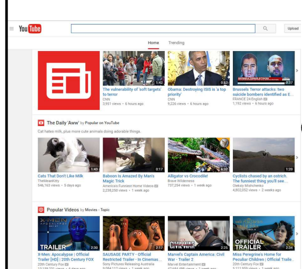
Agrawal, Carreira, Malik, Learning to see by moving. ICCV 2015
Hadsell et al., Dimensionality Reduction by Learning an Invariant Mapping, CVPR 2006

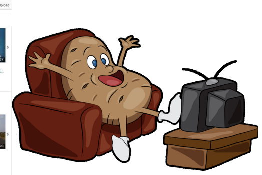
Two formulations

1. Learning representations tied to ego-motion
2. Learning representations from unlabeled video



Learning from arbitrary unlabeled video?



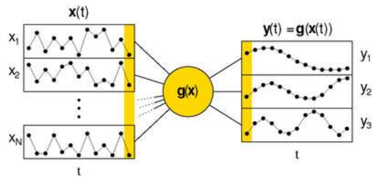


Unlabeled video

Background: Slow feature analysis

[Wiskott & Sejnowski, 2002]

Find functions **g(x)** that map



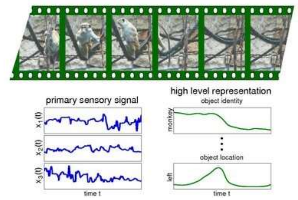
quickly varying input signal **x**(t) → slowly varying features **y**(t)

Figure: Laurenz Wiskott, http://www.scholarpedia.org/article/File:SlowFeatureAnalysis-OptimizationProblem.png

Background: Slow feature analysis

[Wiskott & Sejnowski, 2002]

Find functions **g(x)** that map

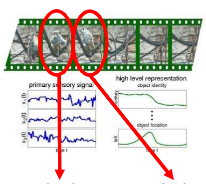


quickly varying input signal **x**(t) → slowly varying features **y**(t)

Figure: Laurenz Wiskott, http://www.scholarpedia.org/article/File:SlowFeatureAnalysis-OptimizationProblem.png

Background: Slow feature analysis

[Wiskott & Sejnowski, 2002]



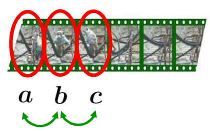
$$\mathbf{z}(a) \approx \mathbf{z}(b)$$

in learned embedding

- Existing work exploits "slowness" as temporal coherence in video → learn invariant representation

  [Hadsell et al. 2006; Mobahi et al. 2009; Bergstra & Bengio 2009; Goroshin et al. 2013; Wang & Gupta 2015,…]
- Fails to capture *how* visual content changes over time

Our idea: Steady feature analysis



- Higher order temporal coherence in video → learn equivariant representation

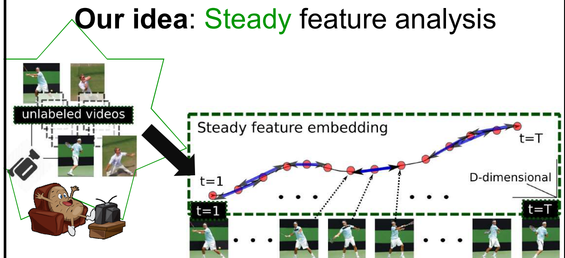
Second order slowness operates on frame triplets:

$$\mathbf{z}(b) - \mathbf{z}(a) \approx \mathbf{z}(c) - \mathbf{z}(b)$$

in learned embedding

[Jayaraman & Grauman, CVPR 2016]

Our idea: Steady feature analysis

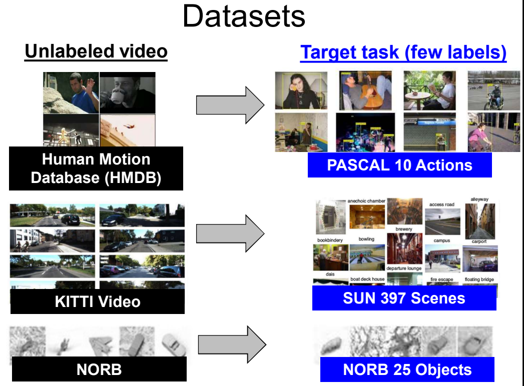


Equivariance ≈ "steadily" varying frame features!

$d^2\mathbf{z}_\theta(\mathbf{x}t)/dt^2 \approx \mathbf{0}$

[Jayaraman & Grauman, CVPR 2016]

Datasets



32 x 32 images or 96 x 96 images

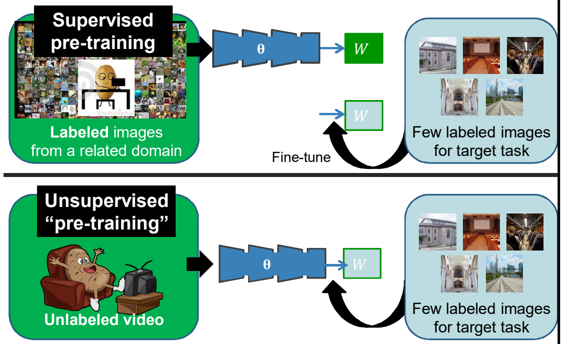
Results: Steady feature analysis

Task type→	Objects	Scenes		Actions
Datasets→	NORB→NORB	KITTI→SUN		HMDB→PASCAL-10
Methods↓	[25 cls]	[397 cls]	[397 cls, top-10]	[10 cls]
random	4.00	0.25	2.52	10.00
UNREG	24.64±0.85	0.70±0.12	6.10±0.67	15.34±0.28
SFA-1 [30]*	37.57±0.85	1.21±0.14	8.24±0.25	19.26±0.45
SFA-2 [14]**	39.23±0.94	1.02±0.12	6.78±0.32	19.04±0.24
SSFA (ours)	**42.83±0.33**	**1.65±0.04**	**9.19±0.10**	**20.95±0.13**

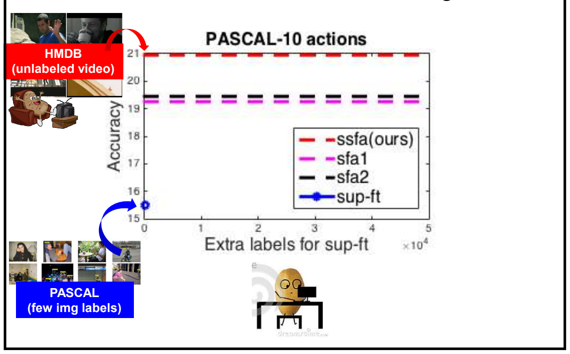
Multi-class recognition accuracy

*Hadsell et al., Dimensionality Reduction by Learning an Invariant Mapping, CVPR'06
**Mobahi et al., Deep Learning from Temporal Coherence in Video, ICML'09

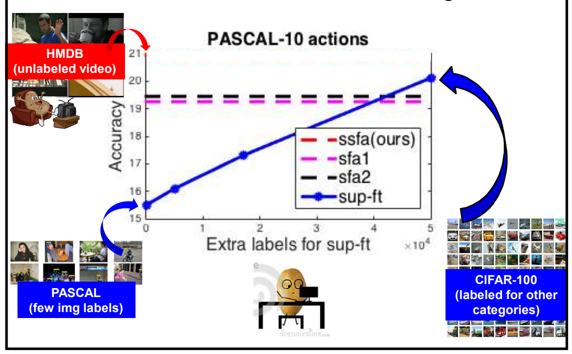
Pre-training a representation



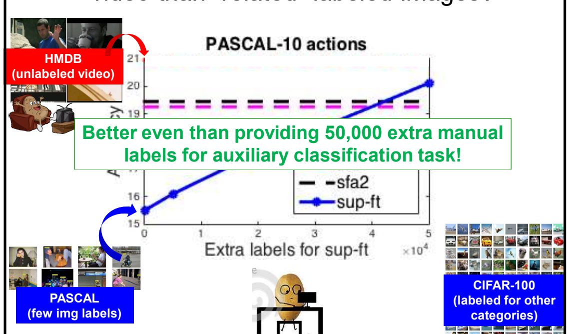
Results: Can we learn *more* from unlabeled video than "related" labeled images?



Results: Can we learn *more* from unlabeled video than "related" labeled images?



Results: Can we learn *more* from unlabeled video than "related" labeled images?



Summary

- Visual learning benefits from
  - context of action and motion in the world
  - continuous self-acquired feedback
- New ideas:
  - "Embodied" feature learning using both visual and motor signals
  - Feature learning from unlabeled video via higher order temporal coherence

Papers

- **Learning Image Representations Tied to Ego-Motion**. D. Jayaraman and K. Grauman. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, Dec 2015.
- **Slow and Steady Feature Analysis: Higher Order Temporal Coherence in Video**. D. Jayaraman and K. Grauman. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, June 2016.