Machine Learning - Other supervision strategies (69152) Unsupervised - Intro

Master in Robotics, Graphics and Computer Vision

Ana C. Murillo

Universidad Zaragoza
1542

REMINDERS

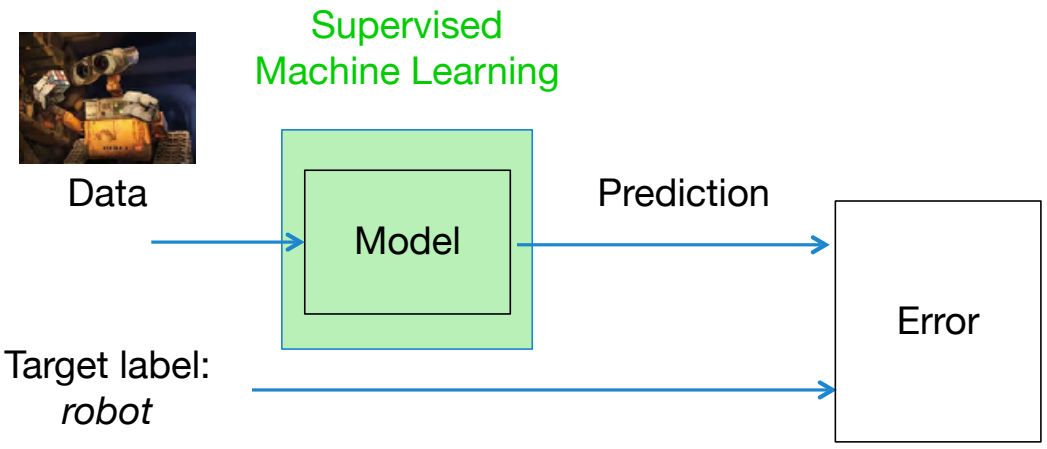
- **Tomorrow, second session Lab4**
  - you should have uploaded by last Friday (we have 20 out of 21 selected for that day)
  - review the instructions/advice given in moodle (**3 minutes to present the paper** and 2 minutes to answer questions)
- **Next Monday: guest talk**

Today

- Other supervision strategies:
  - **Unsupervised**

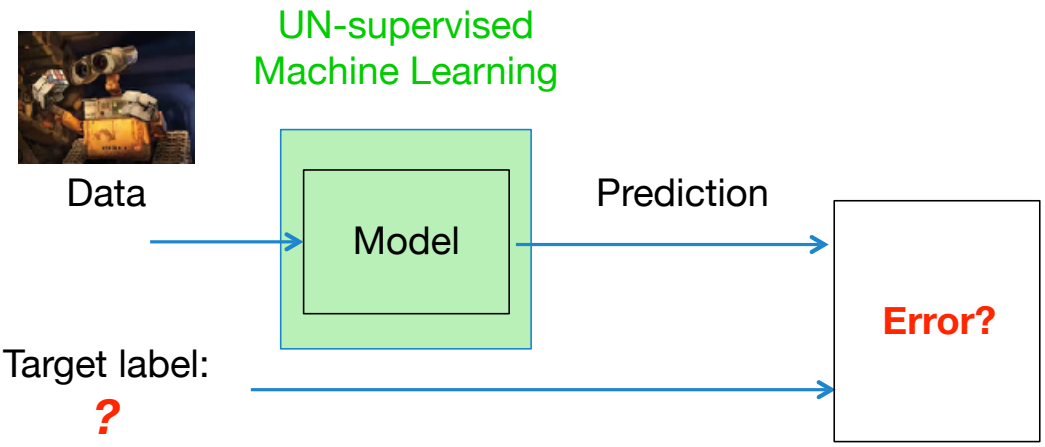
Supervised vs Unsupervised

So far —> supervised learning: I know the “target” output



Supervised vs Unsupervised

Unsupervised: no labeled data

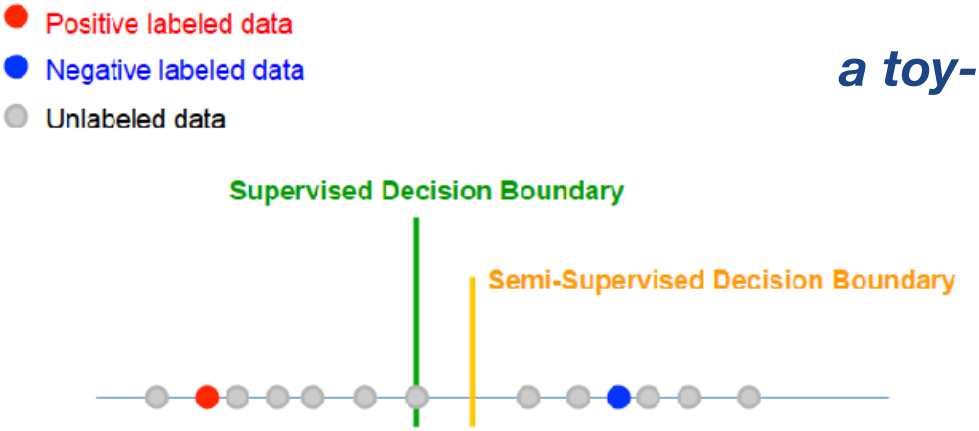


Motivation for *Unsupervised*

- Labeled data is hard to get
  - experts, money, time, …
- weakly or un-labeled data is very easy/cheap to get
  - Internet, Big-Data, …
- Un-labeled data can boost supervised methods …
  - semi-supervised learning
- … or be used on its own
  - unsupervised learning

Semi-supervised

- We can mix labeled and un-labeled data … How?



a toy-example

image from slides from Jerry Zhu. CMU.

- When? Does it always work?
  - Pre-requisite: that the distribution of examples is relevant for our problem.
  - Then, un-labeled data can help identify *boundary* more accurately
- It's getting a lot of attention in deep learning communities

Unsupervised

- Dimensionality reduction
- Clustering

Unsupervised: dims reduction

Decompose signals into components

—> reduce dimensionality

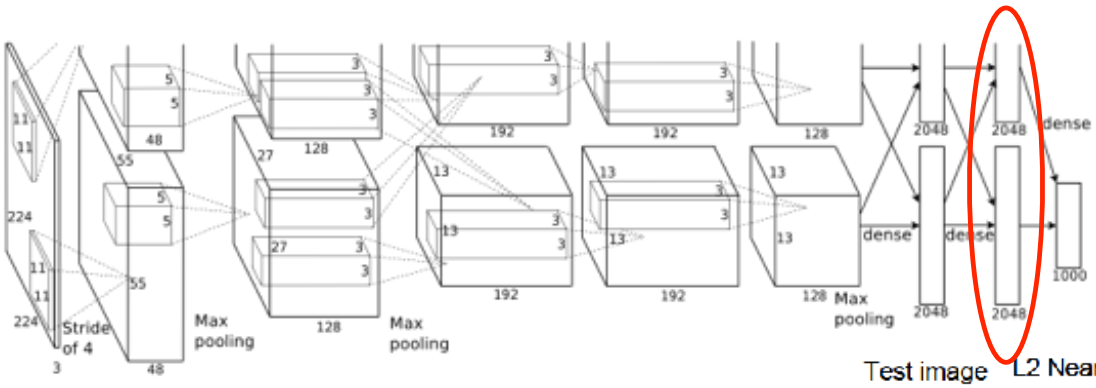
Why? When?

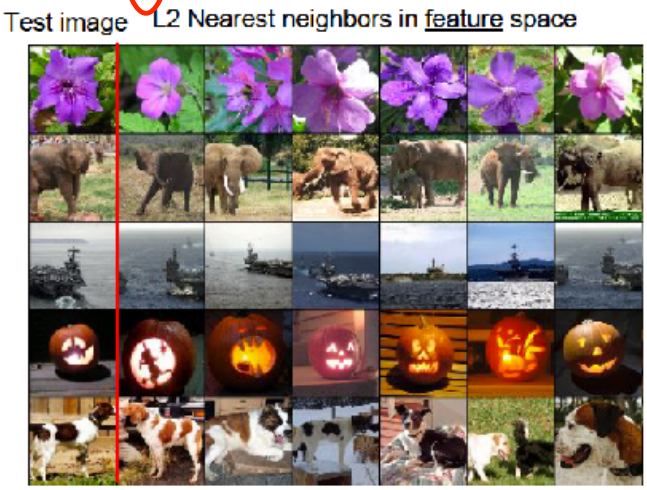
Visualization, efficiency, less noise, more generalization, pre-processing, …

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K}$$

Transfer Learning: features

- Features from last layer: 4096 dims. vector





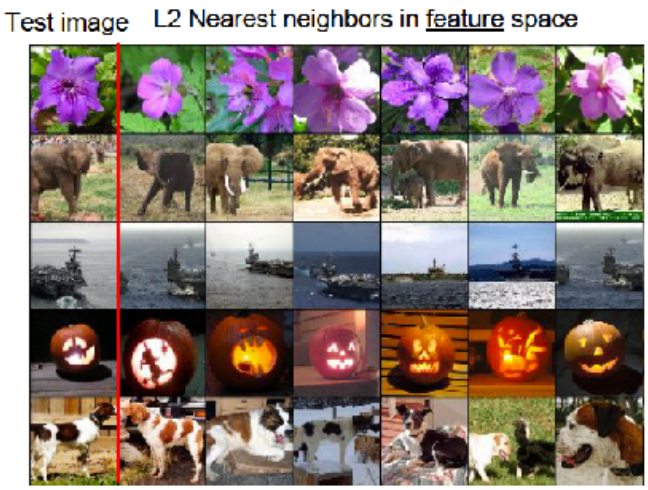
How to visualize or analyze this feature space?

ImageNet Classification with Deep Convolutional Neural Networks
Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. NIPS 2012

Transfer Learning: features

- If you want to “transfer” from certain model …
  - What’s more representative for your data? (if any)
  - How do you “decide” that?
  - Visualization? **Problems?**

What about more compact features?



Unsupervised: dims reduction

One of the most known …

Unsupervised: dims reduction

- Principal Components Analysis (**PCA**)
  - project to **different sub-space**
  - find how much **variation of the data is represented in each *axis*** —> we can keep only the *top* ones

Intuition [edit]

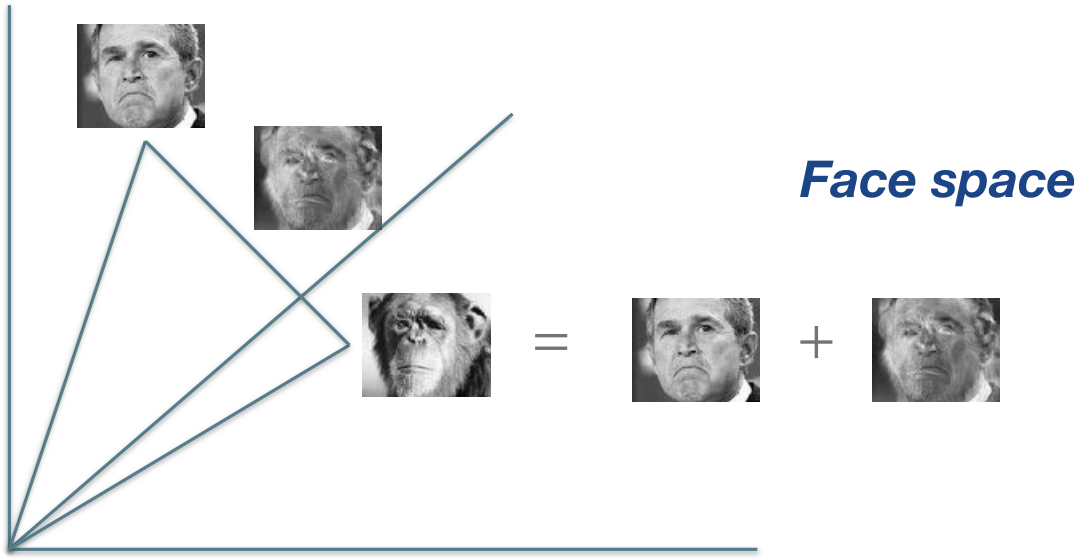
PCA can be thought of as fitting an *n*-dimensional ellipsoid to the data, where each axis of the ellipsoid represents a principal component. If some axis of the ellipsoid is small, then the variance along that axis is also small, and by omitting that axis and its corresponding principal component from our representation of the dataset, we lose only a commensurately small amount of information.

To find the axes of the ellipsoid, we must first subtract the mean of each variable from the dataset to center the data around the origin. Then, we compute the covariance matrix of the data, and calculate the eigenvalues and corresponding eigenvectors of this covariance matrix. Then we must normalize each of the orthogonal eigenvectors to become unit vectors. Once this is done, each of the mutually orthogonal, unit eigenvectors can be interpreted as an axis of the ellipsoid fitted to the data. This choice of basis will transform our covariance matrix into a diagonalised form with the diagonal elements representing the variance of each axis . The proportion of the variance that each eigenvector represents can be calculated by dividing the eigenvalue corresponding to that eigenvector by the sum of all eigenvalues.

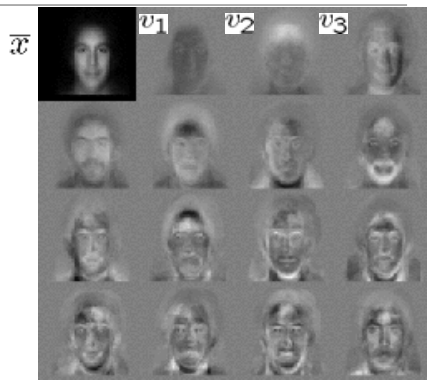
This procedure is sensitive to the scaling of the data, and there is no consensus as to how to best scale the data to obtain optimal results.

PCA example: *Eigenfaces*

- An image is a point in a high dimensional space
- Define vectors in this space as we do in 2D

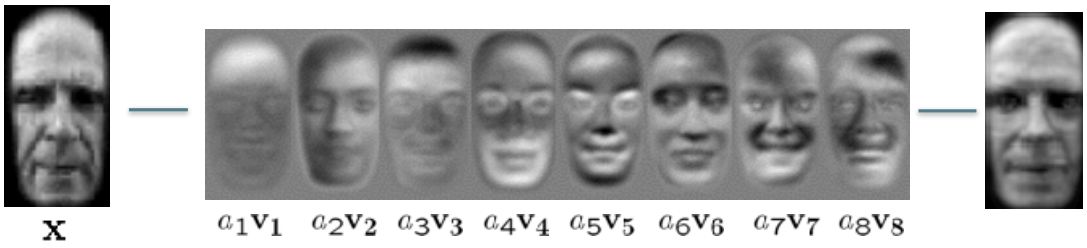


PCA example: *Eigenfaces*



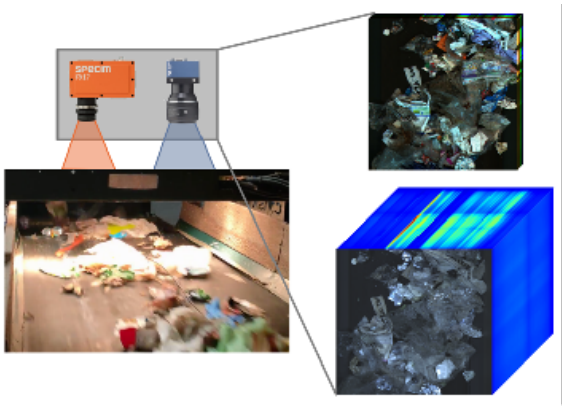
- PCA extracts the eigenvectors of A
- Gives a set of vectors v1, v2, v3, ...
- Each vector is a direction in face space
- Each face is converted to eigenface coordinates

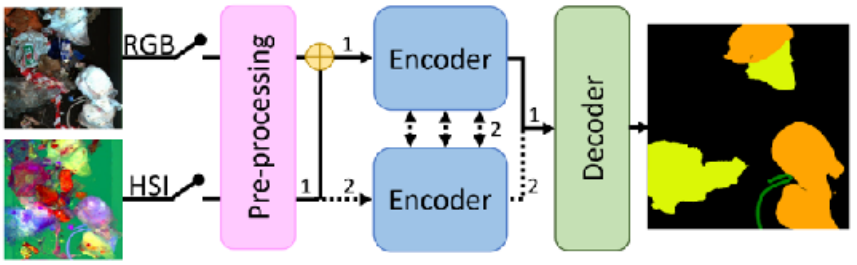
$$\mathbf{x} \approx \bar{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K}$$



PCA example: *very large inputs*

- PCA reduces the input dimensionality minimising the information that is “lost” with the *compression*





Unsupervised: dims reduction

Another strategy, very interesting for deep nets …

Unsupervised: dims reduction

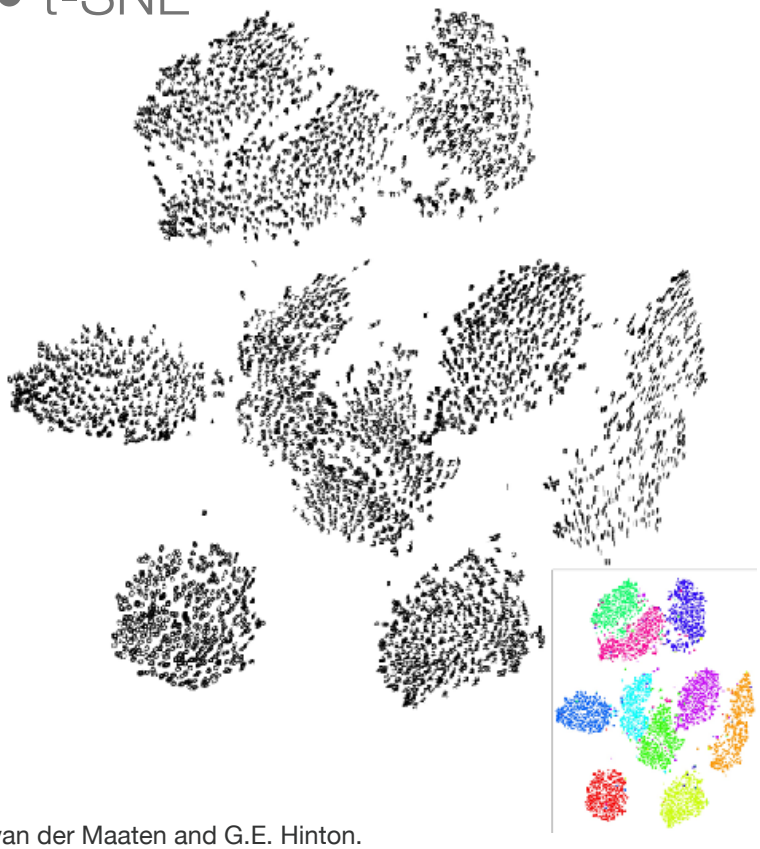
- **t-SNE** *(t-distributed stochastic neighbor embedding)*
  - represent each point by 2|3 dim. point
  - **similar** objects are modeled by **nearby** points (modelling as good as possible local structure)
  - **dissimilar** objects are modeled by **distant** points (often “too” far, but that’s ok)

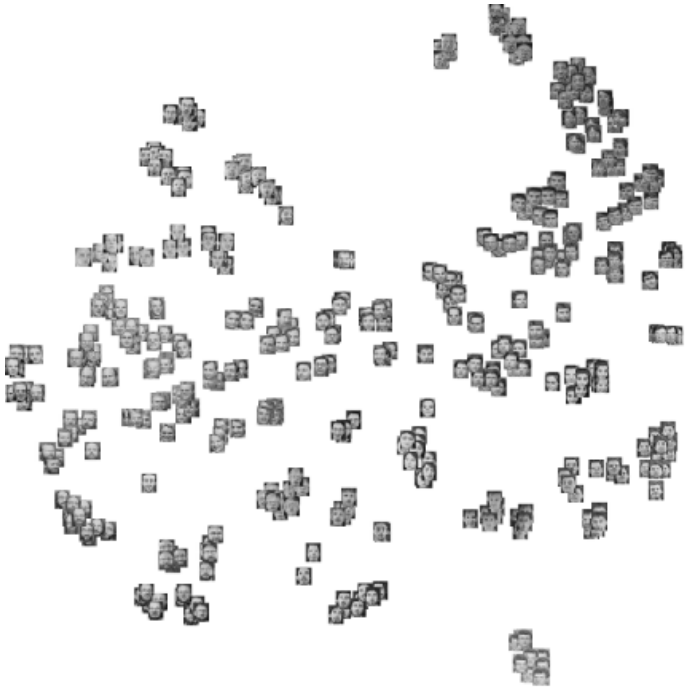
L.J.P. van der Maaten and G.E. Hinton.
Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research.* 2008

http://lvdmaaten.github.io/tsne/

Unsupervised: dims reduction

- t-SNE



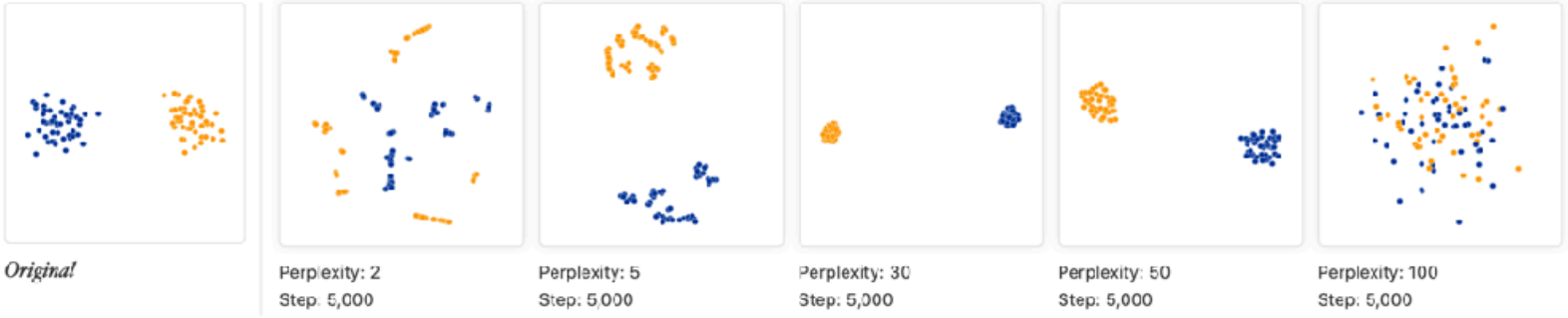


http://lvdmaaten.github.io/tsne/

L.J.P. van der Maaten and G.E. Hinton.
Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research.* 2008

Unsupervised: dims reduction

- t-SNE, not so easy to tune and interpret
  - *hyperparameters influence a lot*
  - *perplexity* recommended between 5-50 ()



https://distill.pub/2016/misread-tsne/

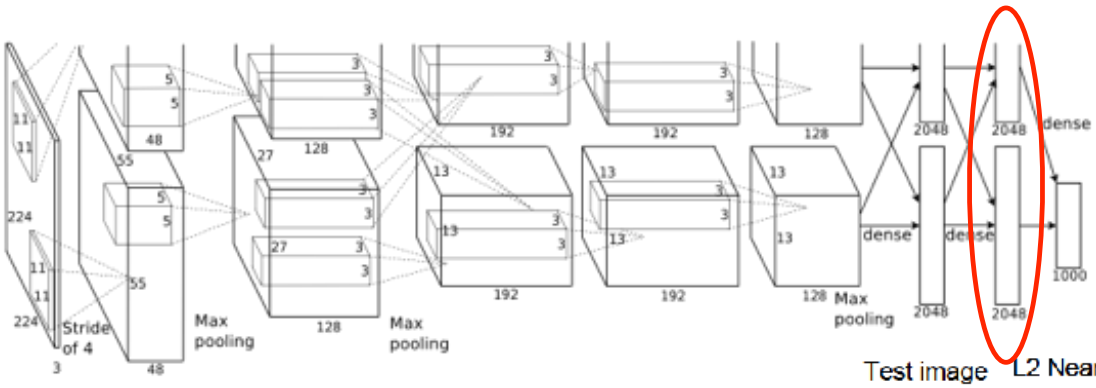
Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. http://doi.org/10.23915/distill.00002

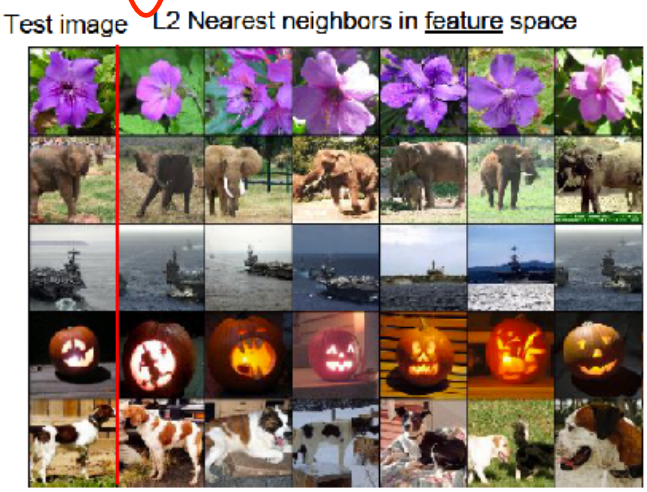
Unsupervised: dims reduction

- Interactive visualisation
  - http://projector.tensorflow.org
- Plenty of implementations
  - https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html
  - http://lvdmaaten.github.io/tsne/

How are we going to use this?

- Features from last layer: 4096 dims. vector



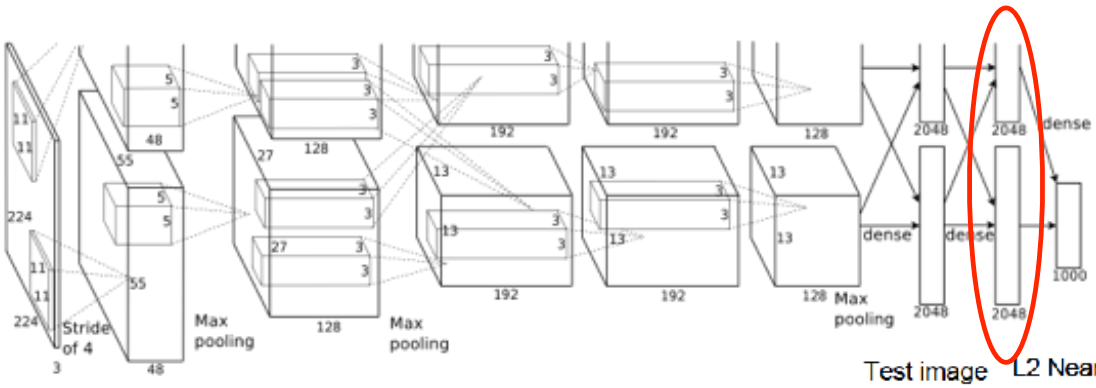


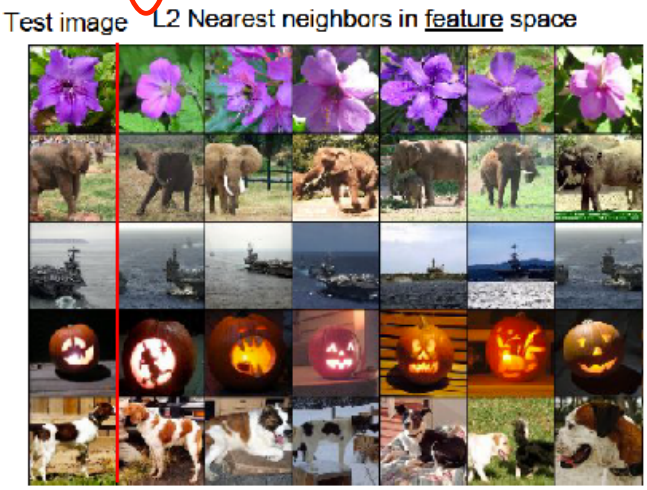
How to visualize or analyze this feature space?

ImageNet Classification with Deep Convolutional Neural Networks
Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. NIPS 2012

How are we going to use this?

- Features from last layer: 4096 dims. vector



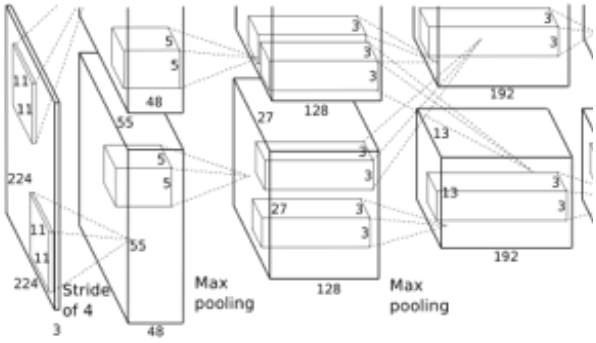


How to visualize or analyze this feature space? —> **reduce the 4096 to 2 dimensions to "plot"**

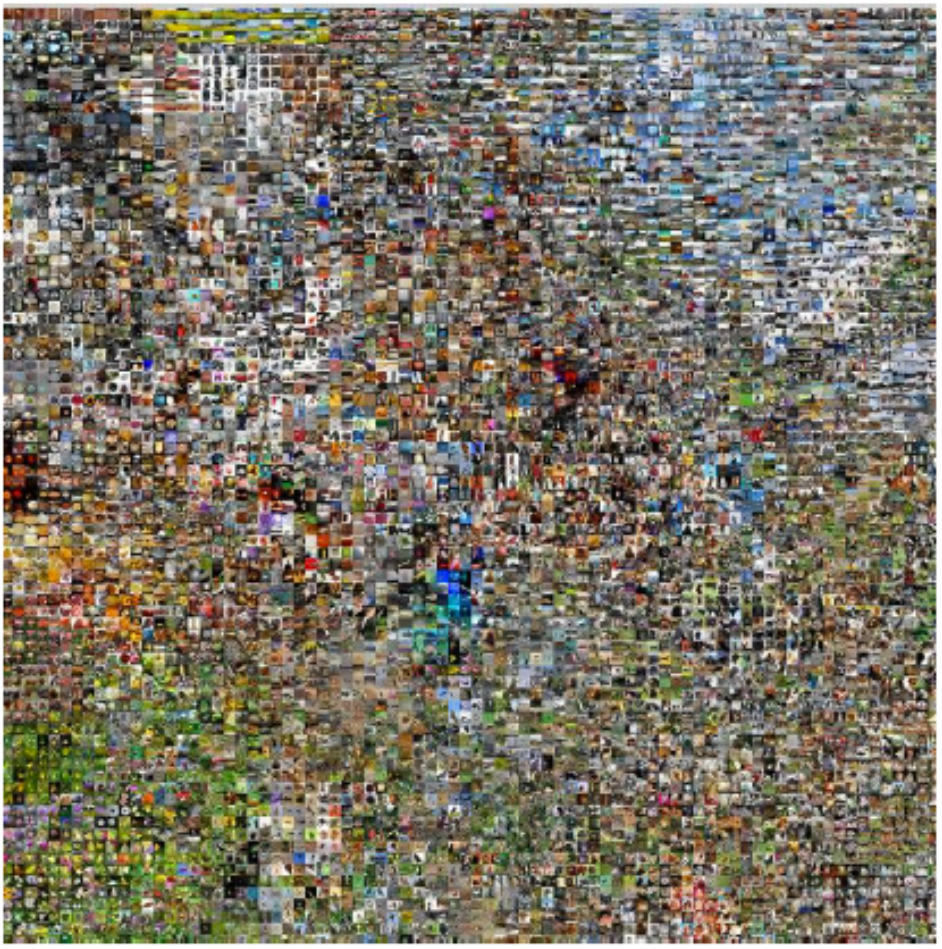
ImageNet Classification with Deep Convolutional Neural Networks
Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. NIPS 2012

How are we going to use this?

- Features from last laye



How to visualize or anal
feature space? —> **red
4096 to 2 dimensions**

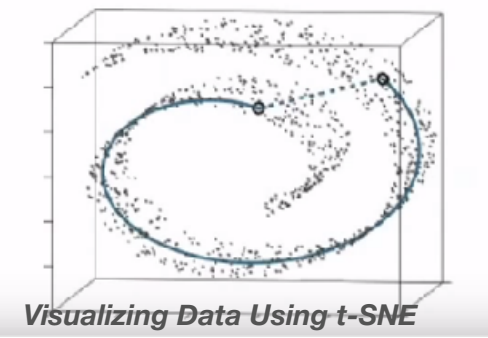


ImageNet Classification with Deep Convolutional Neural Networks
Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. NIPS 2012

http://cs.stanford.edu/people/karpathy/cnnembed/

Summary: dimensionality reduction

PCA and t-SNE



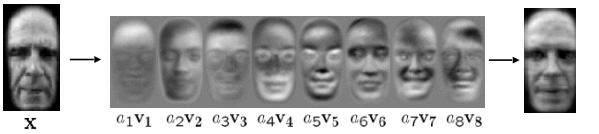
https://www.youtube.com/watch?v=RJVL80Gg3lA

Project to **other** space

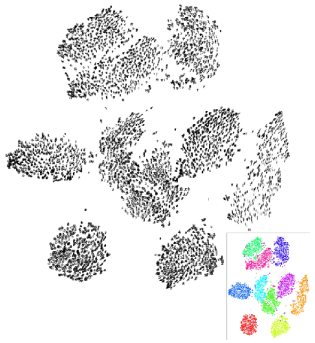
(is **not** just *I keep part of my original descriptor*)

- **PCA** - find ***mapping*** function

$$\mathbf{x} \approx \bar{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K}$$



- **t-SNE** - ***visualize*** given dataset

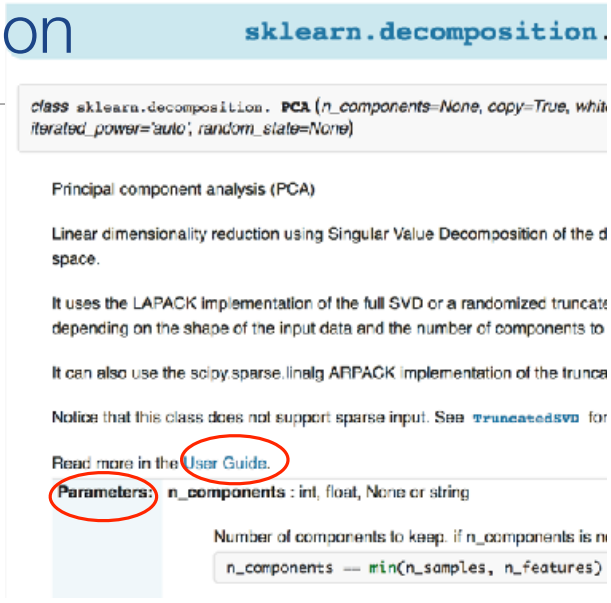


Summary: dimensionality reduction

PCA and t-SNE

Some issues to consider …

- *Enough* data (with too much some algorithms don't converge to a good solution, too little data is useless)
- *Play* with params
- PCA (how many dims? variation? iterations?)
- t-SNE (in and out dims? *perplexity*?)
- Combinations (PCA before t-SNE)



http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html#sklearn.decomposition.PCA

```
def tsne(X = Math.array([]), no_dims = 2, initial_dims = 50, perplexity = 30.0):
    """Runs t-SNE on the dataset in the NxD array X to reduce its dimensionality to no_dims dimensions.
    The syntaxis of the function is Y = tsne.tsne(X, no_dims, perplexity), where X is an NxD NumPy array."""

    # Check inputs
    if isinstance(no_dims, float):
        print "Error: array X should have type float.";
        return -1;
    if round(no_dims) != no_dims:
        print "Error: number of dimensions should be an integer.";
        return -1;

    # Initialize variables
    X = pca(X, initial_dims).real;
    (n, d) = X.shape;
```

https://lvdmaaten.github.io/tsne/

Unsupervised: clustering

Find groups with certain similarity across its members

Why? When?

Unsupervised: clustering

Find groups with certain similarity across its members

Why? When?

Organizing, understanding - discovery, pre-process, …




Unsupervised: clustering

- Objective based: **K-Means**

How does k-means work?

slides from M-F Balcan, CMU.

Unsupervised: clustering

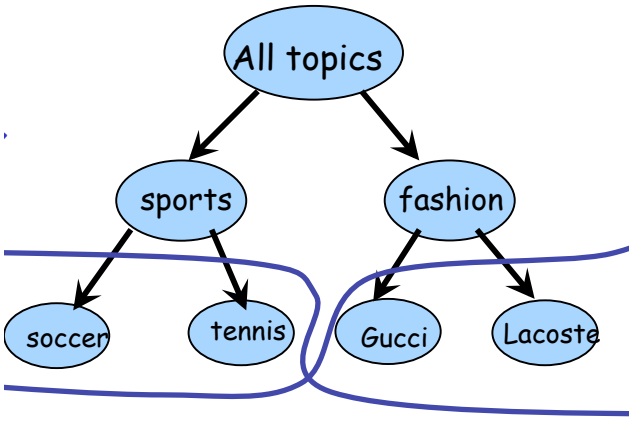
- Objective based: **K-Means**
  - initialization of K and centers
  - iteratively assign points to clusters
    - minimize different distances
  - and re-compute centers

How does k-means work?

slides from M-F Balcan, CMU.

Unsupervised: clustering

- **Hierarchical**:
  - top-down:
    - start all together
    - Iteratively split into two.
  - bottom-up (agglomerative):
    - start each point is a cluster
    - Iteratively merge closest clusters.



Unsupervised: clustering

- Latent Dirichlet allocation (LDA*)

  - NLP, ***topic discovery*** *(not exactly "clustering")*

  - *Document*: mix of various topics. *Topic*: frequency and co-ocurence of *words*

*** NOT = Linear Discriminant Analysis**

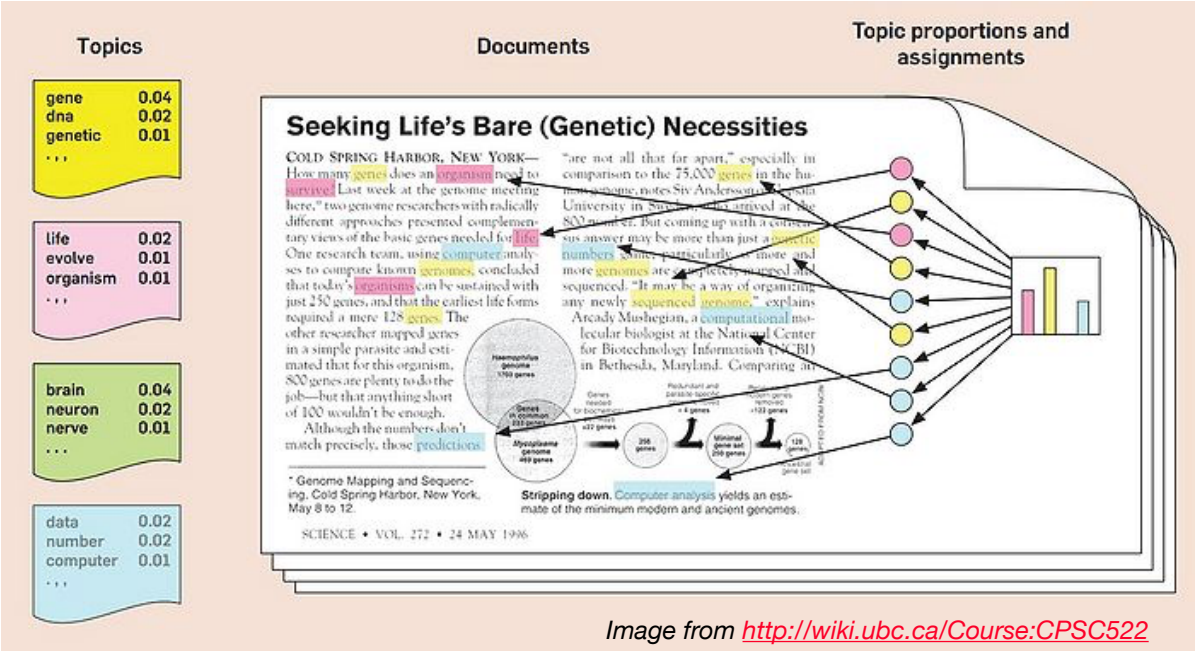


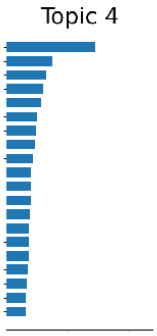
Image from http://wiki.ubc.ca/Course:CPSC522

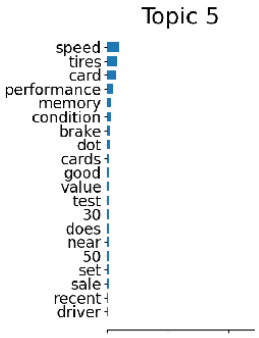
DEMO

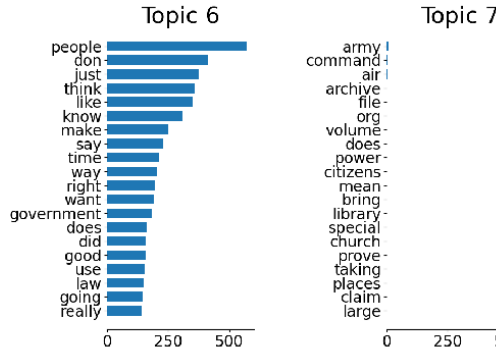
http://scikit-learn.org

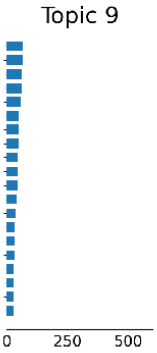
```
Topic #9: key chip clipper keys encryption gover
nton used legal standard

Fitting the NMF model (generalized Kullback-Leib
done in 2.233s.
```

```
ne nsa communications law encrypted security cli


_samples=2000 and n_features=1000...
```

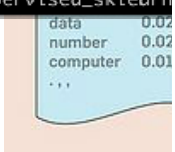









```
(tf_env) Anas-MacBook-Pro-2:unsupervised_sklearn
```



DEMO

http://scikit-learn.org

mage from http://wiki.ubc.ca/Course:CPSC522

Summary: clustering

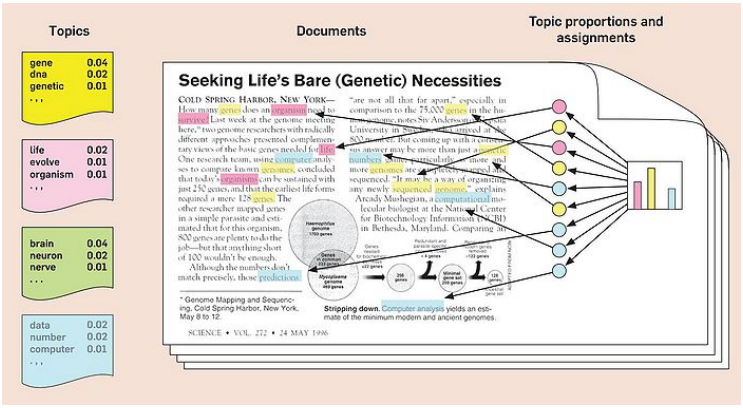
Clustering

- *What? How?*

Find groups with similarities —> **which ones?**

- K-means - based on distance
- LDA - based on *counts*

slides from M-F Balcan, CMU.



Summary: clustering

```
print("Fitting LDA models with tf features, "
      "n_samples=%d and n_features=%d..."
      % (n_samples, n_features))
lda = LatentDirichletAllocation(n_components=n_components, max_iter=5,
                                learning_method='online',
                                learning_offset=50.,
                                random_state=0)
t0 = time()
lda.fit(tf)
print("done in %0.3fs." % (time() - t0))
```

Clustering

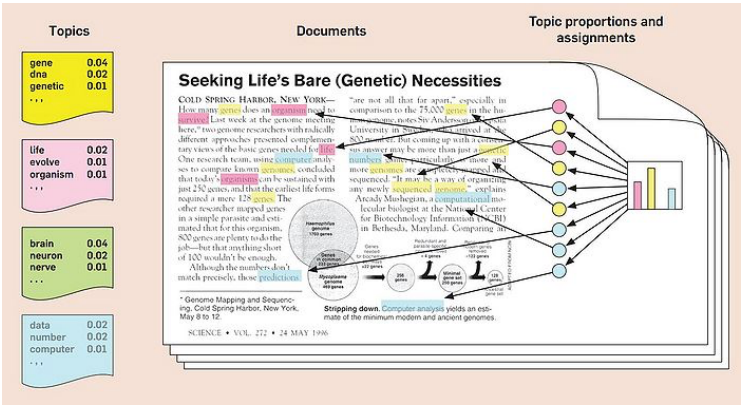
- *What? How?*

Problems:

- Careful with dimensions of descriptor, amount of data (samples vs num clusters) …
- *Play* with params
- k-means (k? iterations? distance?)
- LDA (topics?)

http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

slides from M-F Balcan, CMU.



Today

- Other supervision strategies:
  - **Unsupervised (*self-supervised*)**
  - **… with Deep Learning**

More Deep Nets architectures

Less human-supervision?

It's common to *fine-tune/adapt* pre-trained models (e.g. in ImageNet)

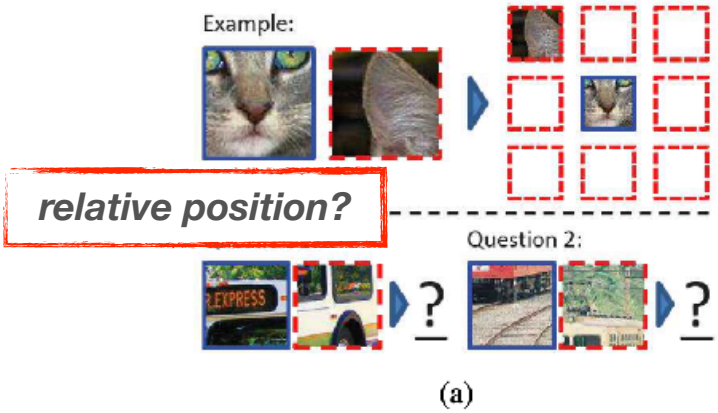
*Can we use **large amounts of un-labeled data to pretrain** base models?*

Self-supervised Deep Learning: *proxy tasks*

Several strategies, most well known:

- Using a ***pretext / proxy* task** for which we can automatically obtain the labels: *recover relative position of patches, colorization, rotations, inpainting missing patches, …*

Self-supervised Deep Learning: *proxy tasks*



Examples of self-supervised learning tasks: (a) guessing the relative positions of image patches—can you guess the answers to Q1 and Q2? (Doersch, Gupta, and Efros 2015) © 2015 IEEE; (b) solving a nine-tile jigsaw puzzle (Noroozi and Favaro 2016) © 2016 Springer; (c) image colorization (Zhang, Isola, and Efros 2016) © 2016 Springer; (d) video color transfer for tracking (Vondrick, Shrivastava et al. 2018) © 2016 Springer.

Self-supervised Deep Learning: *contrastive*

Several strategies, most well known:

- Using a ***pretext / proxy* task**
  - we can automatically obtain the labels for the proxy: *recover relative position of patches, colorization, rotations, inpainting missing patches, …*
  - then trained "supervised"
- Using ***contrastive* learning**
  - Semantically similar inputs to produce similar representations …
  - … and keep distant from *negative* samples

Self-supervised Deep Learning: *contrastive*

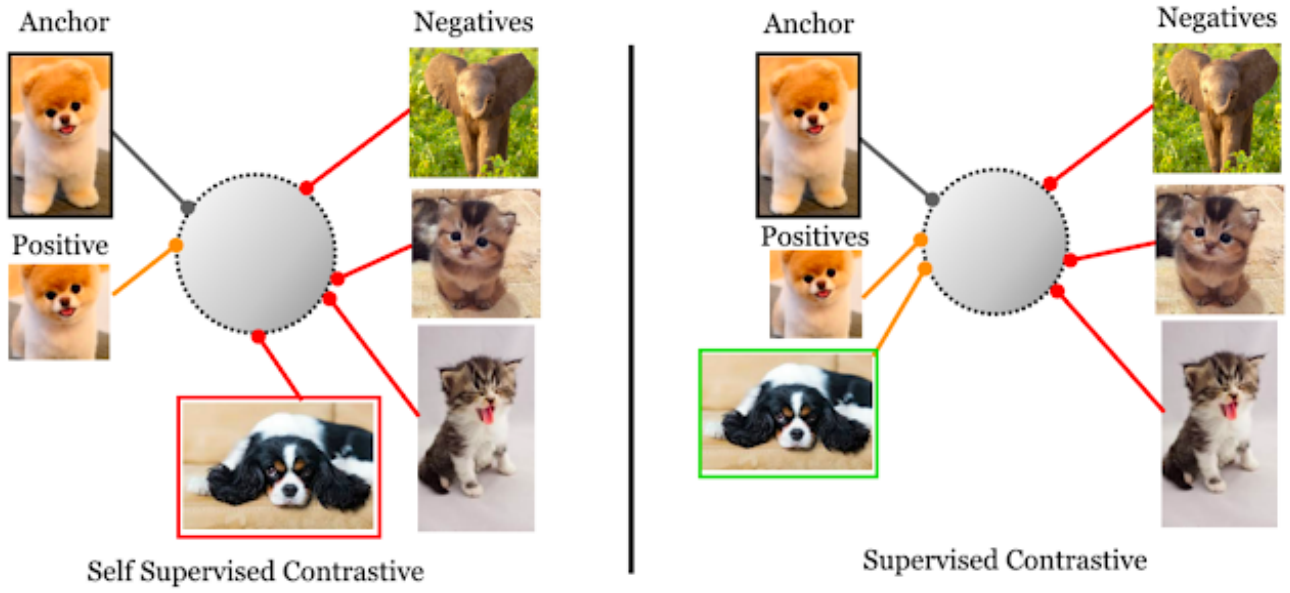


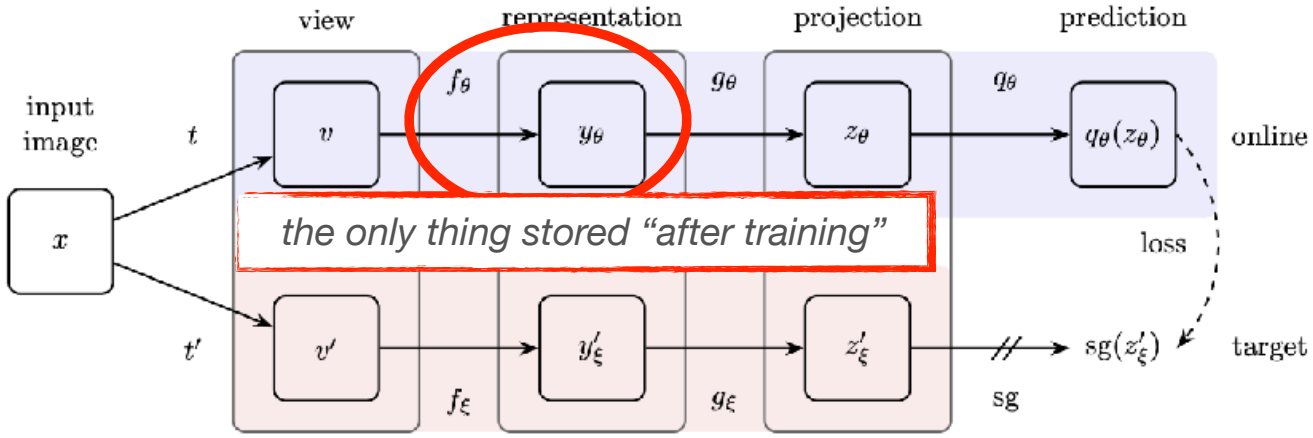
Figure from https://ai.googleblog.com/2021/06/extending-contrastive-learning-to.html

Many variations:

- positives from same and/or different samples
- many negatives (triplet loss, just 1 positive, 1 negative) - *or no negatives!*

Self-supervised Deep Learning: examples

- **BYOL** *(Boostrap your own latent: A New Approach to Self-Supervised Learning)*
  - **only positive** pairs “contrast”. **Self-supervised** learning.
  - **online and target networks**: interact and learn from each other.
    Augmented view of an image —> train online network to predict the target network representation of the same image under a different augmented view



Grill, Jean-Bastien, et al. "Bootstrap your own latent: A new approach to self-supervised learning." NeurIPS 2020.

Self-supervised Deep Learning: examples

- **BYOL.** *Boostrap your own latent: A New Approach to Self-Supervised Learning*
  - **only positive** pairs “contrast”. **Self-supervised** learning.
  - **online and target networks**: interact and learn from each other.
    Augmented view of an image —> train online network to predict the target network representation of the same image under a different augmented view
  - **representation** learned is useful to improve **many downstream tasks**

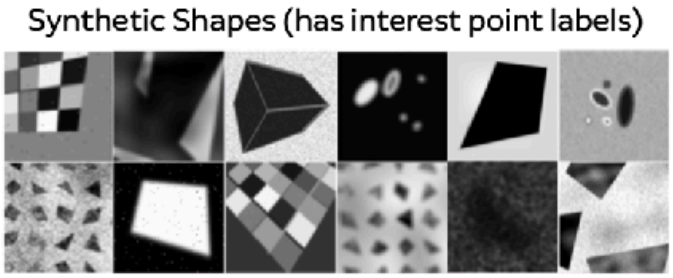
Method	Food101	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
BYOL (ours)	**75.3**	91.3	**78.4**	**57.2**	**62.2**	**67.8**	60.6	82.5	75.5	90.4	94.2	**96.1**
SimCLR (repro)	72.8	90.5	74.4	42.4	60.6	49.3	49.8	81.4	**75.7**	84.6	89.3	92.6
SimCLR [8]	68.4	90.6	71.6	37.4	58.8	50.3	50.3	80.5	74.5	83.6	90.3	91.2
Supervised-IN [8]	72.3	**93.6**	78.3	53.7	61.9	66.7	**61.0**	**82.8**	74.9	**91.5**	**94.5**	94.7
Fine-tuned:												
BYOL (ours)	**88.5**	**97.8**	86.1	**76.3**	63.7	91.6	**88.1**	**85.4**	**76.2**	91.7	**93.8**	97.0
SimCLR (repro)	87.5	97.4	85.3	75.0	63.9	91.4	87.6	84.5	75.4	89.4	91.7	96.6
SimCLR [8]	88.2	97.7	85.9	75.9	63.5	91.3	88.1	84.1	73.2	89.2	92.1	97.0
Supervised-IN [8]	88.3	97.5	**86.4**	75.8	**64.3**	**92.1**	86.0	85.0	74.6	**92.1**	93.3	**97.6**
Random init [8]	86.9	95.9	80.2	76.1	53.6	91.4	85.9	67.3	64.8	81.5	72.6	92.0

Table 3: Transfer learning results from ImageNet (IN) with the standard ResNet-50 architecture.

Grill, Jean-Bastien, et al. "Bootstrap your own latent: A new approach to self-supervised learning." NeurIPS 2020.

Self-supervised Deep Learning: examples

- **Superpoint***: Self-Supervised Interest Point Detection and Description*
  - opportunities for “adapted” features to specific domains, e.g. medical

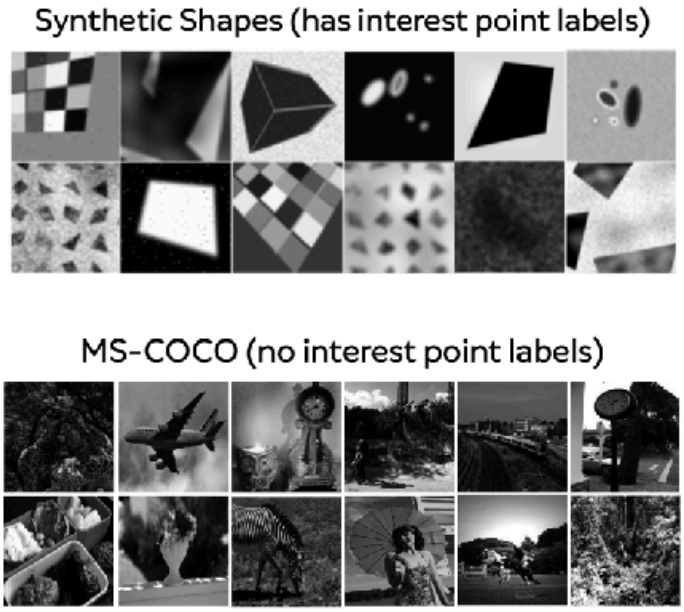


train on synthetic
(using homographies)
to get initial model

DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich. "Superpoint: Self-supervised interest point detection and description." Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2018.

Self-supervised Deep Learning: examples

- **Superpoint**: *Self-Supervised Interest Point Detection and Description*
  - opportunities for “adapted” features to specific domains, e.g. medical



train on synthetic
(using homographies)
to get initial model

… combined with *super-glue or light-glue* (supervised) for matching
https://psarlin.com/superglue/
https://github.com/cvg/LightGlue

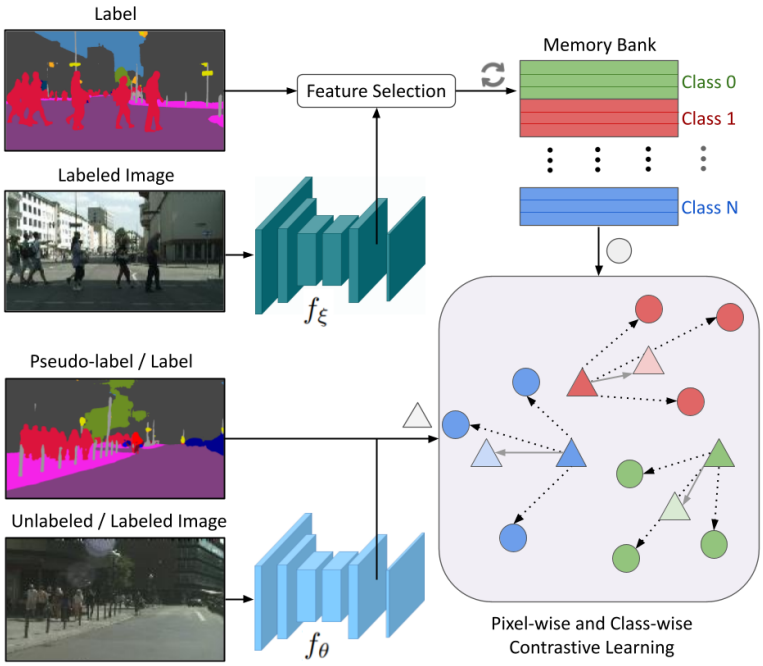
generate **pseudo-labels and keep training**

DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich. "Superpoint: Self-supervised interest point detection and description." Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2018.

Self-supervised Deep Learning: examples

Additional recent “ideas” with very interesting results:

- **Semi-supervised:** use a few labeled samples and lots of unlabeled data
- *student-teacher*: teacher gives training examples to the student
- *memory-bank*: representative or reliable samples
- contrastive *per pixel*



I. Alonso et al. Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank. ICCV 2021.

Self-supervised Deep Learning: examples

Plenty of very “famous” deep models are unsupervised (Generative, GPTs, …)

Next …

- **Generative models (Auto-encoder, GAN, Diffusion)**

Bibliography - Resources for some of the materials today

- Stanford classes on deep learning for Computer Vision (http://cs231n.stanford.edu) and Deep Learning (https://cs230.stanford.edu/)

- Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

- *Computer Vision: Algorithms and Applications*. 2nd Edition. Richard Szeliski. https://szeliski.org/Book/