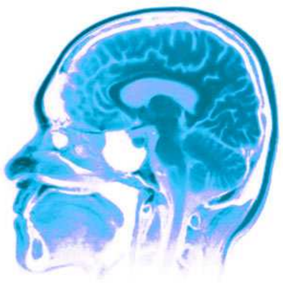
CPSC540



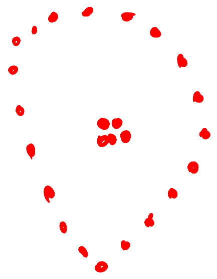
Clustering & Mixture Models
K-Means & the EM Algorithm

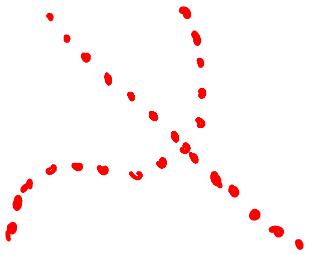
Nando de Freitas
November, 2011
University of British Columbia

Clustering

Gestalt

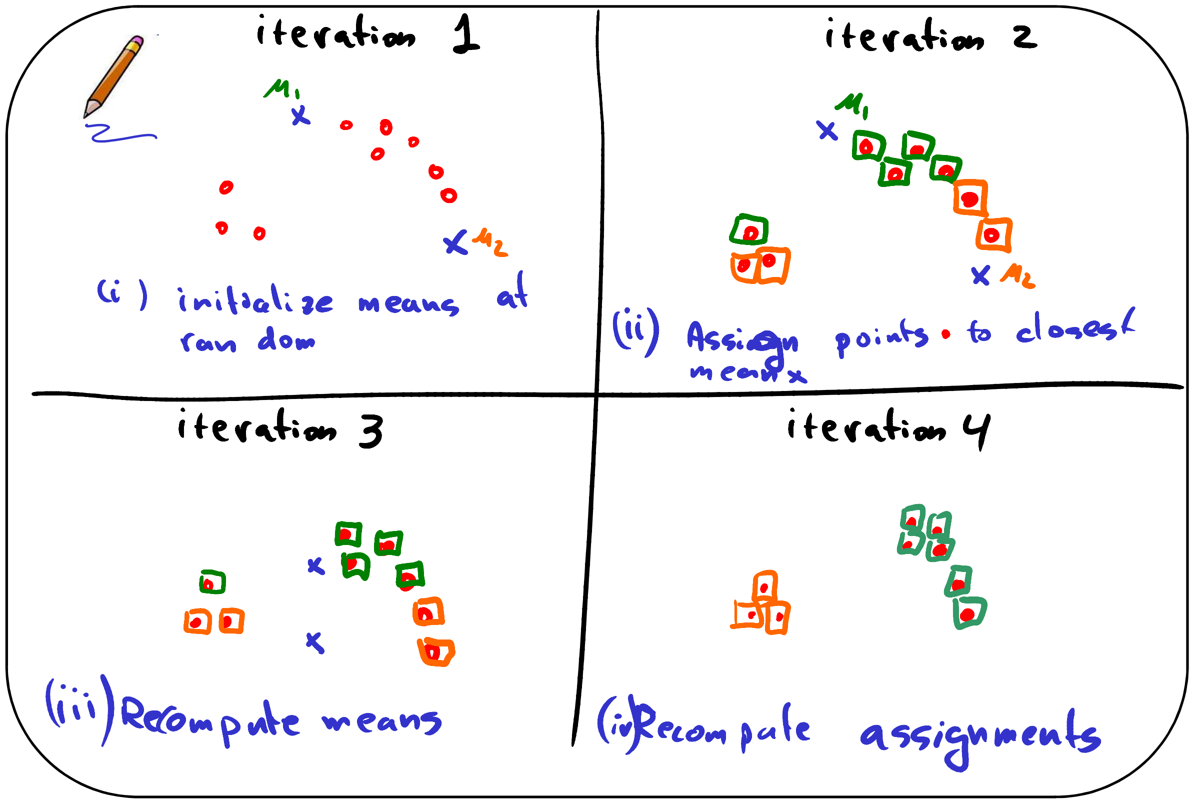
perceptual grouping





K-means

K = 2 clusters


iteration 1
μ_1
μ_2
(i) initialize means at random
iteration 2
μ_1
μ_2
(ii) Assign points • to closest mean x
iteration 3
(iii) Recompute means
iteration 4
(iv) Recompute assignments

K-means algorithm

1. **Initialisation:** Choose $k = 2$ means $\mu_{1:2}$ at random.
2. **Compute distances:** For $c = 1, \ldots, k$ and $i = 1, \ldots, n$ compute the distance $\|x_i - \mu_c\|^2$.
3. **Assign data to nearest mean:** To keep track of assignments, introduce the indicator variable z_i, such that

$$\mathbb{I}_c(z_i) = \begin{cases} 1 & \text{if } c = \arg\min_{c'} \|x_i - \mu_{c'}\|^2 \\ 0 & \text{otherwise} \end{cases}$$

That is, $\mathbb{I}_2(z_i) = 1$ if observation x_i is closer to cluster 2. $\mathbb{I}_c(z_i)$ end up being the entries of an $n \times k$ matrix with only one 1 per row and many zeros.

$$\mathbb{I}_c(z) = \begin{array}{cccc} c_1 & c_2 & c_3 & \\ 1 & 0 & 0 & x_1 \\ 1 & 0 & 0 & x_2 \\ 0 & 1 & 0 & x_3 \\ 0 & 1 & 0 & x_4 \\ 0 & 0 & 1 & x_5 \\ 1 & 0 & 0 & x_6 \end{array}$$

$z_1 = 1$

$z_3 = 2$

$z_5 = 3$

$\vdots$

$z_i \in \{1, 2, 3\}$

K-means algorithm (continued)

$\sum_i \mathbb{I}_c(z_i) = $ # points in cluster c

4. **Update means:**

$$\mu_c = \frac{\sum_{i=1}^{n} \mathbb{I}_c(z_i) x_i}{\sum_{i=1}^{n} \mathbb{I}_c(z_i)}$$

5. **Repeat:** Go back to step 2. until the means and assignments stop changing.

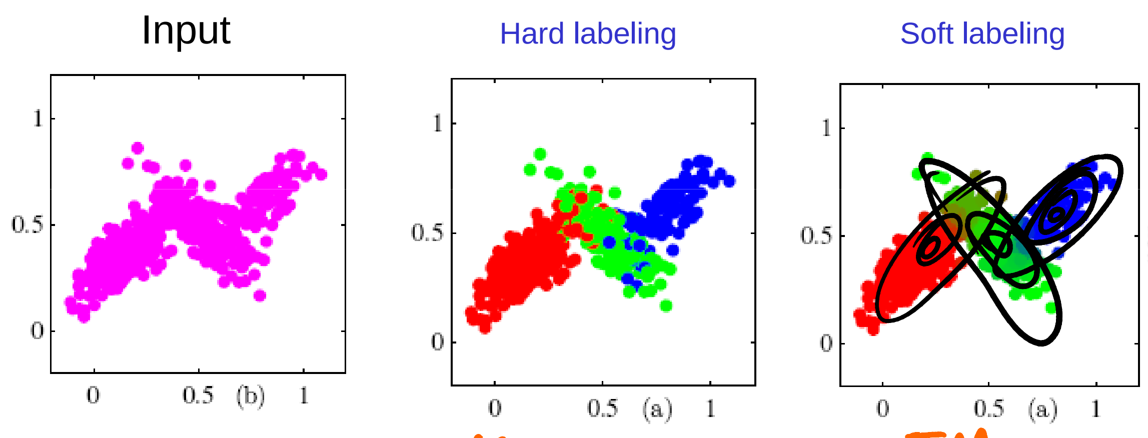
Hard Vs Soft assignments

The problem with this algorithm is that the assignments are hard. Something is either this or that. Sometimes, however, we would like to say that something is this with probability 0.7 or that with probability 0.3.

We would like to find not only the means, but also the variances of each cluster and the probabilities of belonging to each cluster.

Clustering

Desired output



K-MEANS

EM & MIXTURES

K=3 is the number of clusters, here chosen by hand

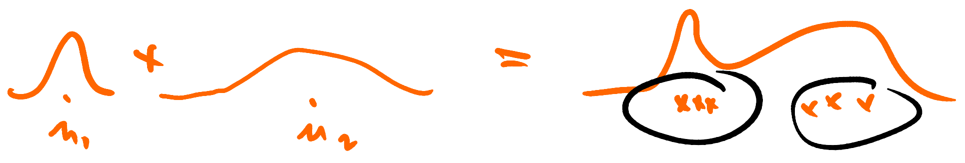
Probabilistic approach

For the 2 clusters, we approximate the probability of each data point with a weighted combination of Gaussians

mixture : cluster 1 cluster 2

$$p(x_i|\mu_{1:2},\sigma_{1:2}) = p(z_i=1)\mathcal{N}(x_i|\mu_1,\sigma_1^2)+p(z_i=2)\mathcal{N}(x_i|\mu_2,\sigma_2^2)$$

Here, the unknown parameters are $(\mu_{1:2},\sigma_{1:2}^2)$ and the cluster probabilities $p(z_i=1)$ and $p(z_i=2)$, which we rewrite as $p(1)$ and $p(2)$ for brevity. Note that $p(1)+p(2)=1$ to ensure that we still have a probability.



Probabilistic approach

In general, we have

$$p(x_i|\theta) = \sum_{c=1}^{k} p(c)\mathcal{N}(x_i|\mu_c, \sigma_c^2)$$

where $\theta = (\mu_{1:c}, \sigma^2_{1:c})$ summarises the model parameters and $p(c) = p(z_i = c)$. Clearly, $\sum_{c=1}^k p(c) = 1$.

The EM algorithm

In this section, we use intuition to introduce the expectation-maximisation (EM). If we know $\mathbb{I}_c(z_i)$, then it is easy to compute (μ_c, σ_c^2) by maximum likelihood. We repeat this for each cluster. The problem is that we have a chicken and egg situation. To know the cluster memberships, we need the parameters of the Gaussians. To know the parameters, we need the cluster memberships.

One solution is to approximate $\mathbb{I}_c(z_i)$ with our expectation of it given the data and our current estimate of the parameters θ. That is, we replace $\mathbb{I}_c(z_i)$ with

$$\xi_{ic} \triangleq \mathbb{E}\left[\mathbb{I}_c(z_i)|x_i, \theta\right]$$

$$P(x_i \mid z_i = c, \theta) = N(\mu_c, \sigma_c^2)$$

$$\xi_{ic} \triangleq \mathbb{E}\left[\mathbb{I}_c(z_i) \mid x_i, \theta\right] = P(z_i = c \mid x_i, \theta)$$

$$= \frac{P(z_i = c, x_i \mid \theta)}{P(x_i \mid \theta)}$$

$$= \frac{P(z_i = c)\, P(x_i \mid z_i = c, \theta)}{\boxed{\sum_{c'} P(z_i = c)\, P(x_i \mid z_i = c, \theta)}}$$

$p(x_i \mid \theta)$

The EM algorithm

Once we know ξ_{ic}, we can compute the Gaussian mixture parameters:

$$\begin{aligned}
\mu_c &= \frac{\sum_{i=1}^{n} \xi_{ic} x_i}{\sum_{i=1}^{n} \xi_{ic}} \\
\Sigma_c &= \frac{\sum_{i=1}^{n} \xi_{ic} (x_i - \mu_c)(x_i - \mu_c)'}{\sum_{i=1}^{n} \xi_{ic}} \\
p(c) &= \frac{1}{n} \sum_{i=1}^{n} \xi_{ic}
\end{aligned}$$

The EM algorithm

The EM for Gaussians is as follows:

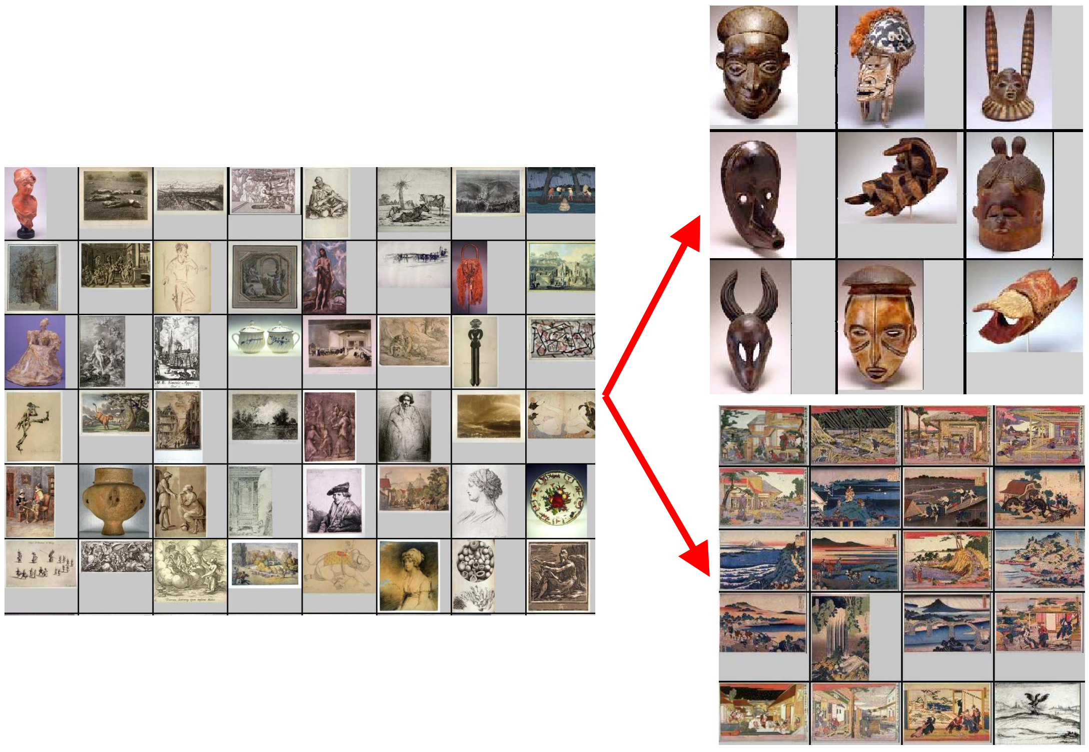
1. **Initialise.**
2. **E Step:** At iteration t, compute the expectation of the indicators for each i and c:

$$\xi_{ic}^{(t)} = \frac{p(c)^{(t)} \mathcal{N}(x_i | \mu_c^{(t)}, \Sigma_c^{(t)})}{\sum_{c'=1}^{k} p(c')^{(t)} \mathcal{N}(x_i | \mu_{c'}^{(t)}, \Sigma_{c'}^{(t)})}$$

   and normalise it (divide by sum over c).

3. **M Step:** Update the parameters $p(c)^{(t)}, \mu_c^{(t)}, \Sigma_c^{(t)}$.

Clustering images



EM for text data

"Large" text dataset:

- 1,000,000 words in 1967
- 1,000,000,000,000 words in 2006

Success stories:

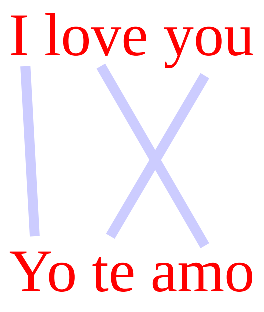
- Speech recognition
- Machine translation

What is the common thing that makes both of these work well?

- Lots of labeled data
- Memorization is a good policy

[Halevy, Norvig & Pereira, 2009]

Statistical machine translation



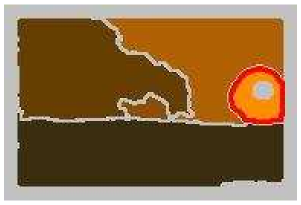
I love chocolate — Yo amo el chocolate

I am — Yo soy

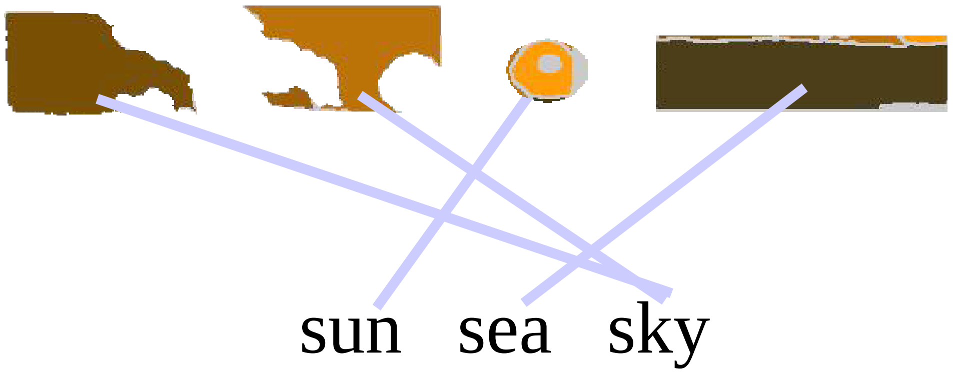
1. Get many sentence pairs – easy.
2. Compute correspondences
3. Compute translation table: P(*Spanish*|*English*)
4. Repeat steps 2 and 3 till convergence

Statistical machine translation

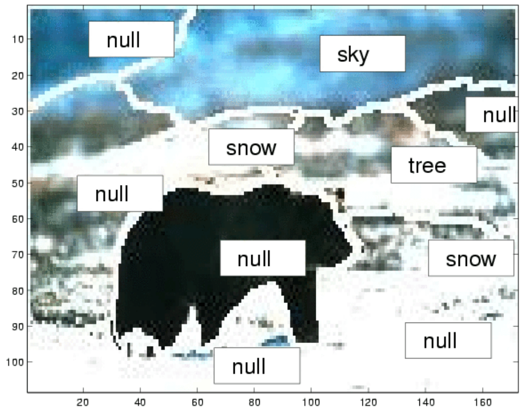
“Gorgeous red sea, sun and sky”

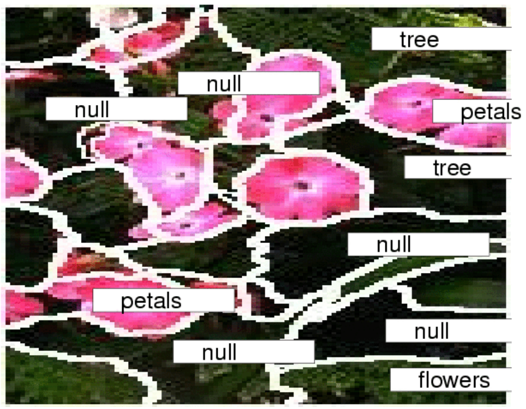


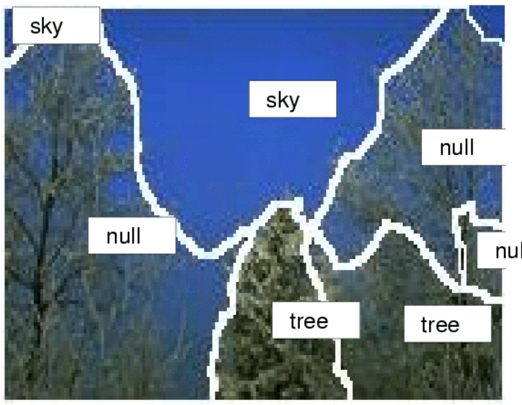
sun sea sky

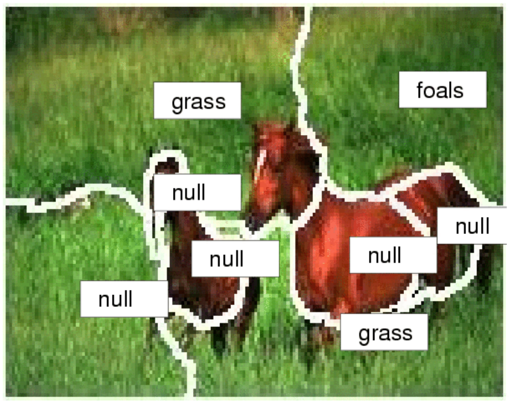


[Duygulu, Barnard, d F, Forsyth, 2001]









K-means for feature learning

1. Extract random patches from unlabeled training images.
2. Apply a pre-processing stage to the patches.
3. Learn a feature-mapping using an unsupervised learning algorithm.

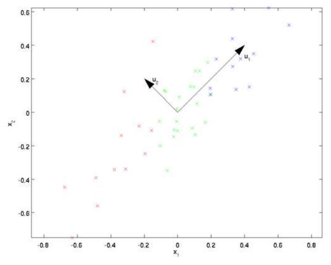
The above steps, for a particular choice of unsupervised learning algorithm, yield a function f that transforms an input patch $x \in \mathbb{R}^N$ to a new representation $y = f(x) \in \mathbb{R}^K$. Using this feature extractor, we now apply it to our (labeled) training images for classification.

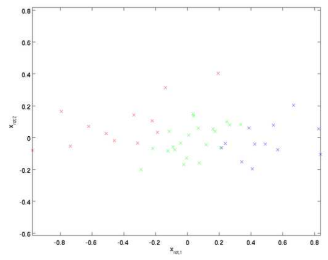
[Adam Coates, Honglak Lee & Andrew Ng 2009]

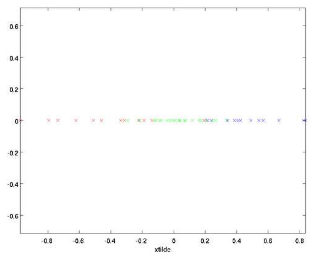
$XX^T = U\Lambda U^T$

Whitening

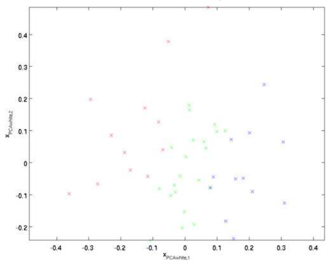
$$x_{\text{rot}} = U^T x = \begin{bmatrix} u_1^T x \\ u_2^T x \end{bmatrix}$$



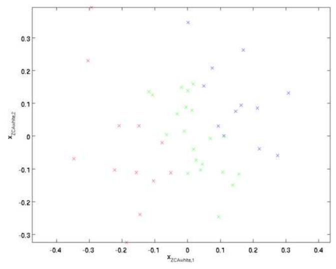


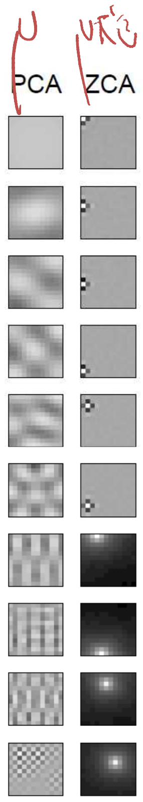


$$x_{\text{PCAwhite},i} = \frac{x_{\text{rot},i}}{\sqrt{\lambda_i}}.$$



$$x_{\text{ZCAwhite}} = U x_{\text{PCAwhite}}$$





[Adam Coates, Honglak Lee & Andrew Ng 2009]

K-means for feature learning

K-means clustering: We apply K-means clustering to learn K centroids $c^{(k)}$ from the input data. Given the learned centroids $c^{(k)}$, we consider two choices for the feature mapping f. The first is the standard 1-of-K, hard-assignment coding scheme:

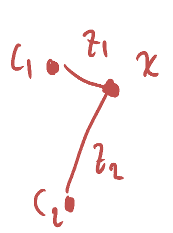
$$f_k(x) = \begin{cases} 1 & \text{if } k = \arg\min_j ||c^{(j)} - x||_2^2 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The second is a non-linear mapping that attempts to be "softer" than the above encoding, but also yield sparse outputs through simple competition:

$$f_k(x) = \max\{0, \mu(z) - z_k\} \tag{3}$$

$z_2 > z_1$

where $z_k = ||x - c^{(k)}||_2$ and $\mu(z)$ is the mean of the elements of z.

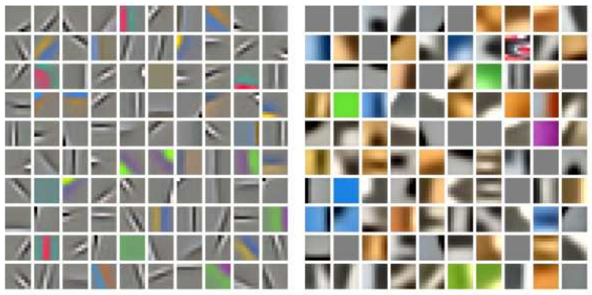


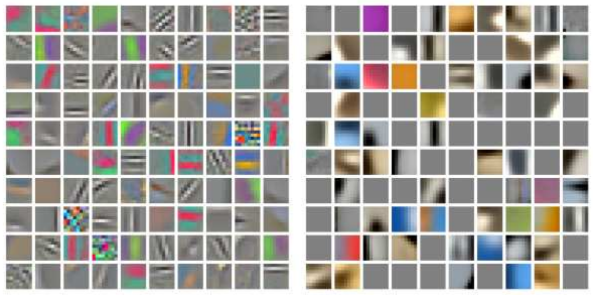
$$f_1 = \max\left(0, \frac{z_1+z_2}{2} - z_1\right) = \frac{z_2 - z_1}{2}$$

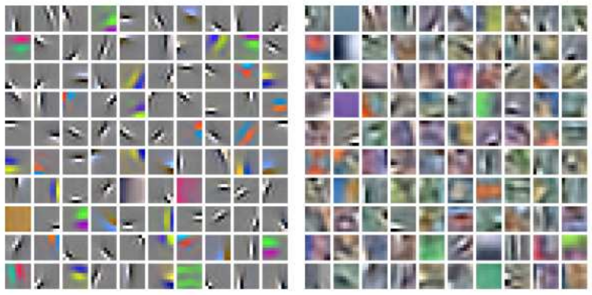
$$f_2 = 0$$

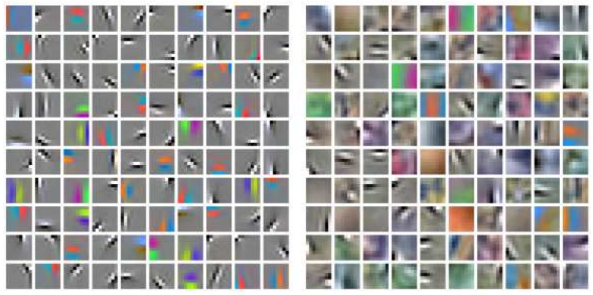
[Adam Coates, Honglak Lee & Andrew Ng 2009]

Learned bases (centroids)


(a) K-means (with and without whitening)


(b) GMM (with and without whitening)


(c) Sparse Autoencoder (with and without whitening)


(d) Sparse RBM (with and without whitening)

[Adam Coates, Honglak Lee & Andrew Ng 2009]

Mapping image to feature vector

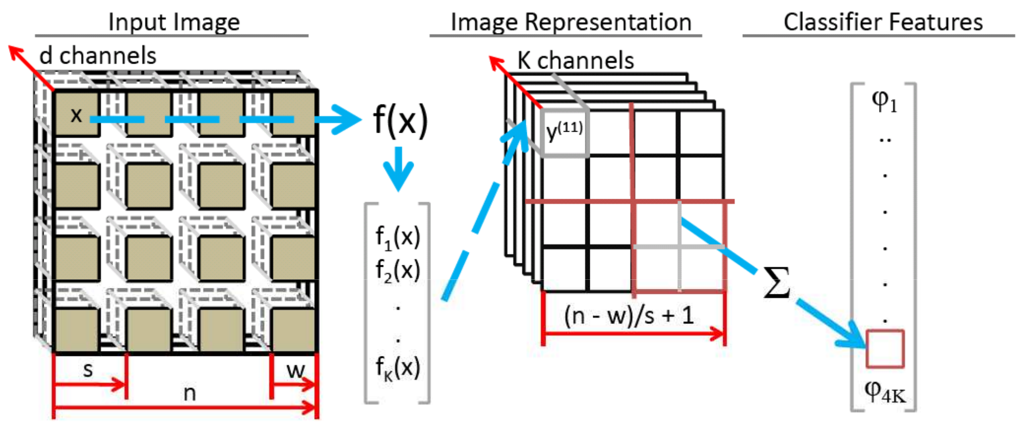
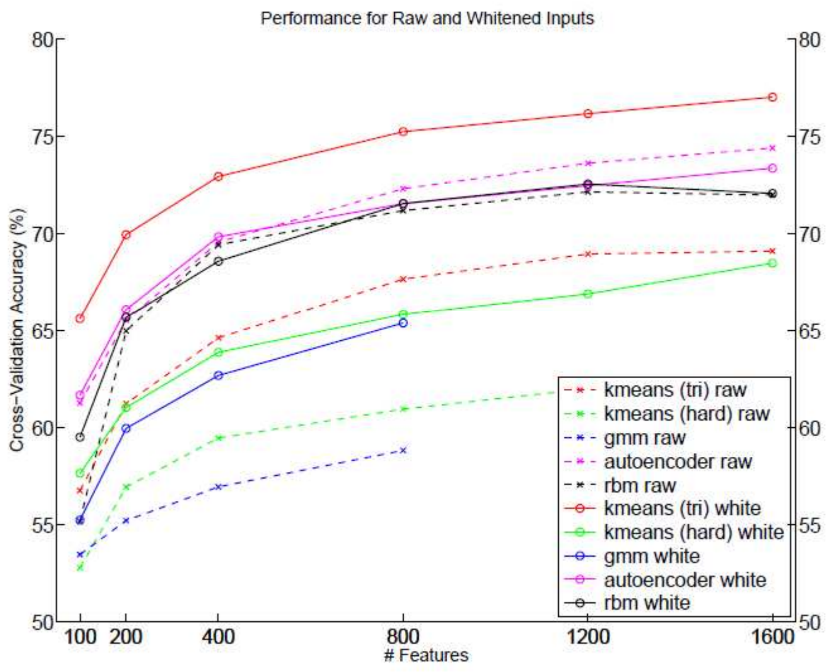


Figure 1: Illustration showing feature extraction using a w-by-w receptive field and stride s. We first extract w-by-w patches separated by s pixels each, then map them to K-dimensional feature vectors to form a new image representation. These vectors are then pooled over 4 quadrants of the image to form a feature vector for classification. (For clarity we have drawn the leftmost figure with a stride greater than w, but in practice the stride is almost always smaller than w.

[Adam Coates, Honglak Lee & Andrew Ng 2009]

K-means for feature learning



[Adam Coates, Honglak Lee & Andrew Ng 2009]