The “Bag of words” model

- Each document is a bag of words, meaning: Assumes order of words has no significance (the term “home made” has the same probability as “made home”)

LSA

- Latent Semantic Analysis
- Goal: Given a corpus of K documents, comprising a dictionary of M words, find the “relations” of words and documents (usually cluster the documents).

The co-occurrence matrix

The element at (i,j) is the word count (or, frequency) of the i'th word in the j'th document.

$$
\begin{array}{c c}
 & \mathbf{d}_j \\
 & \downarrow \\
\mathbf{t}_i^T \rightarrow & \begin{bmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \dots & x_{m,n} \end{bmatrix}
\end{array}
$$

A row in the matrix is a vector of the term's occurrence in all documents:

$$\mathbf{t}_i^T = \begin{bmatrix} x_{i,1} & \dots & x_{i,n} \end{bmatrix}$$

While a column is a vector of the occurrence of all terms in a document.

$$\mathbf{d}_j = \begin{bmatrix} x_{1,j} \\ \vdots \\ x_{m,j} \end{bmatrix}$$

The dot product $\mathbf{t}_i^T \mathbf{t}_p$ gives the correlation between two terms over all documents

Likewise, the dot product $\mathbf{d}_j^T \mathbf{d}_q = \mathbf{d}_q^T \mathbf{d}_j$ gives the correlation between all the terms in two documents

By multiplying the correlation matrix (denoted X) by itself transposed, we get a matrix of the dot products between each two documents.

Likewise, multiplying the transposed matrix by itself gives us the dot products between all terms.

Using a SVD decomposition, we can decompose X into $X = U\Sigma V^T$, where U and V are orthonormal, and Σ is diagonal.

Now the correlations become:

$$
\begin{aligned}
XX^T &= (U\Sigma V^T)(U\Sigma V^T)^T = (U\Sigma V^T)(V^{T^T}\Sigma^T U^T) = U\Sigma V^T V\Sigma^T U^T = U\Sigma\Sigma^T U^T \\
X^T X &= (U\Sigma V^T)^T(U\Sigma V^T) = (V^{T^T}\Sigma^T U^T)(U\Sigma V^T) = V\Sigma U^T U\Sigma V^T = V\Sigma^T\Sigma V^T
\end{aligned}
$$

Select the k largest singular values from Σ, and their corresponding singular vectors from *U* and *V*.

Fact: this is the rank *k* approximation to the original matrix with the smallest error (using frobenius norm)

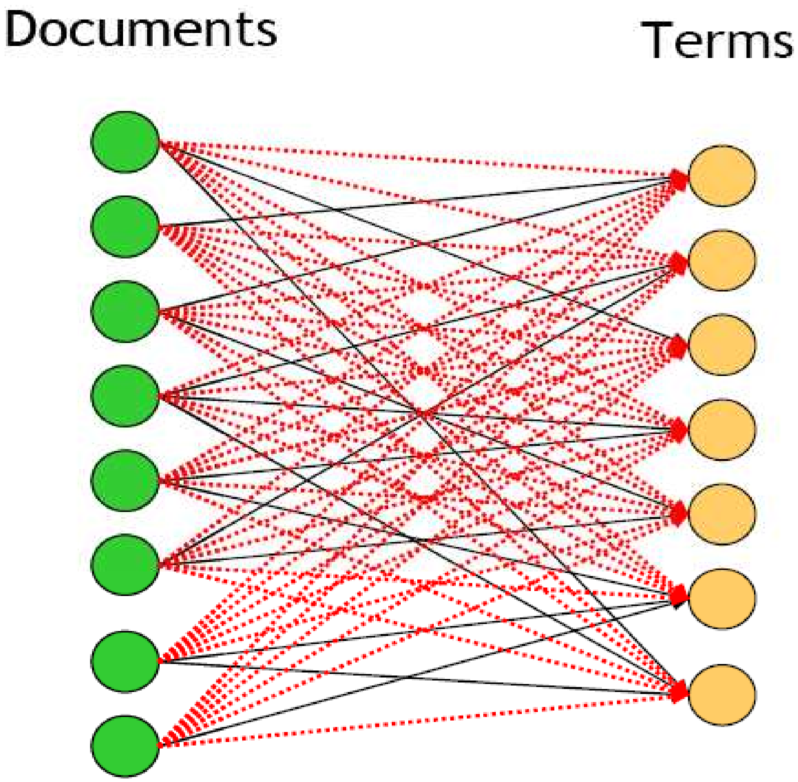
Moreover, each term vector in the k-approximation matrix has K entries, each correlating to a specific “topic”. The (j,m) entry shows how much the j’th term is related with the m’th topic.

Now we can cluster documents by comparing them (with cosine similarity).

pLSA

- **Probabilistic** Latent Semantic Analysis
- pLSA relies on the likelihood function of multinomial sampling and aims at an explicit maximization of the predictive power of the model

The naive approcach

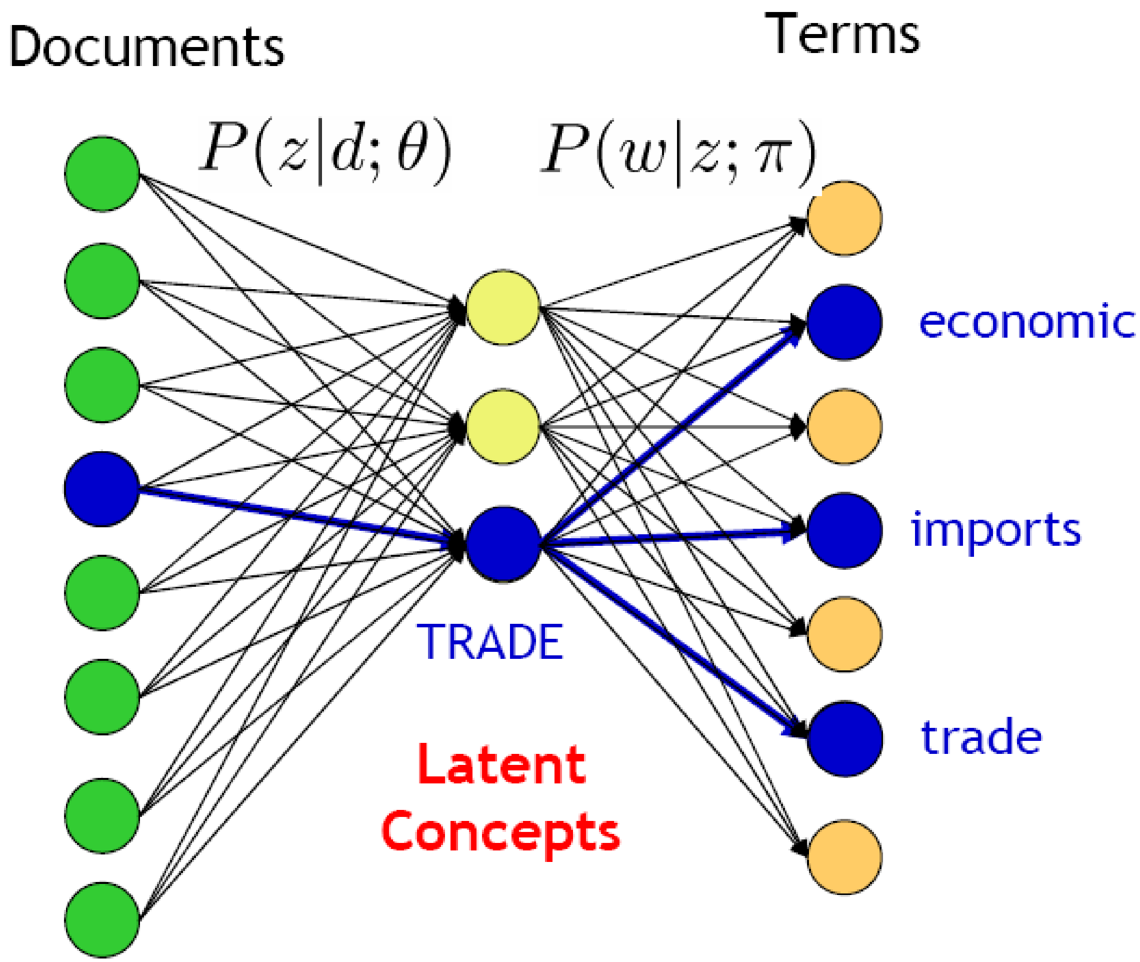


number of occurrences of term w in document d

$$\hat{P}_{\mathrm{ML}}(w|d) = \frac{n(d,w)}{\sum_{w'} n(d,w')}$$

Does not utilize the full corpus as it considers only one document at a time. Intuitively, One would assume that from a larger corpus you can infer more meaningful conclusions on the probabilities than from a small corpus.

PLSA - General idea



The latent concepts (or topics), denoted as Z, act as a bottleneck variable

Probabilistic latent semantic space

Reminder: The multinomial distribution represents the probability of conducting an experiment with K possible results, each one with it's own event probability, and getting each result a specific number of times (what are the odds of throwing two die 8 times, getting a sum of 6 on 3 occurrences and 12 on 5 occurrences)

Probabilistic latent semantic space

Let R be the M-1 dimensional simplex of all Possible multinomials of M components

Each “topic” z defines a point on the simplex R, by the multinomial distribution P(W|z). Thus, these K topics define K points which give us a K-1 dimensions simplex.

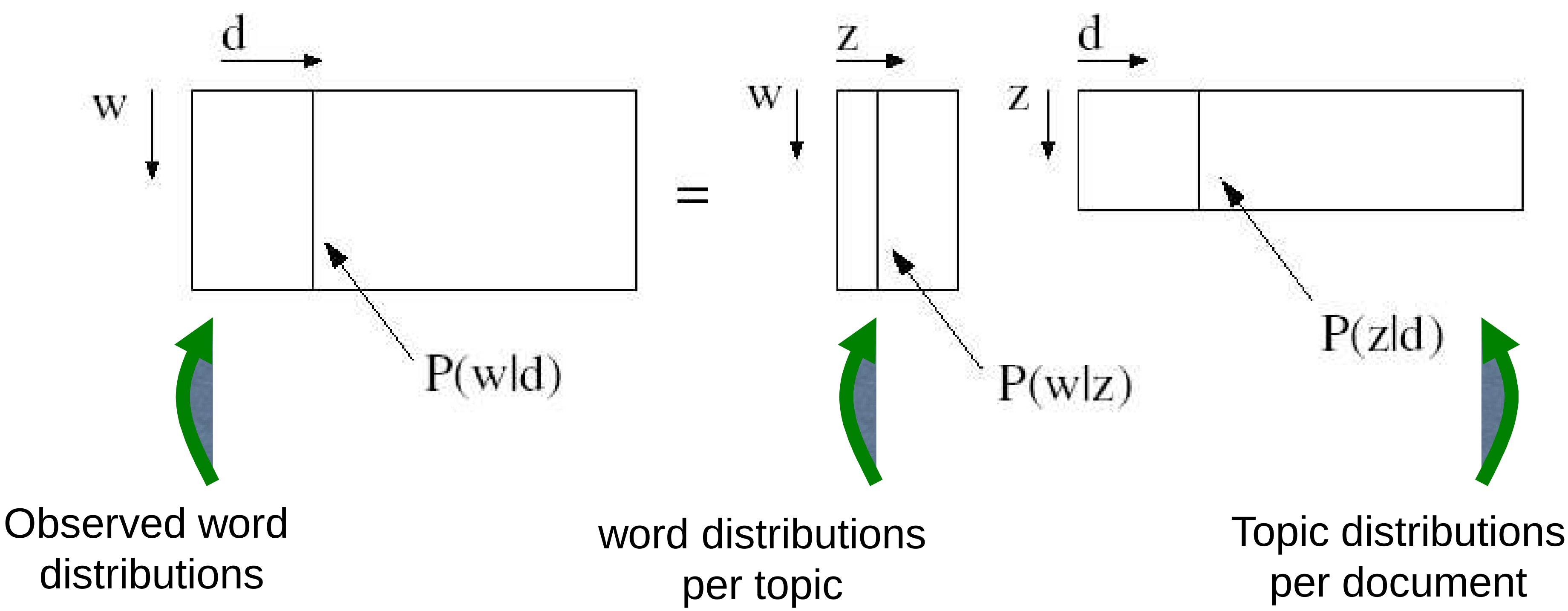
The modeling assumption is that P(w|d) can be created as a convex combinations (all factors non-negative) of P(w|z), where the factors are P(z|d).

Intuitively, this makes sense – the probability of a word appearing in a document is related to the probability of it appearing after each topic, and the probability of that topic being relevant to the document.

Thus giving us the formula:

$$p(w_i \mid d_j) = \sum_{k=1}^{K} p(w_i \mid z_k) p(z_k \mid d_j)$$

In matrix form:



Slide credit: Josef Sivic

$$\hat{P}_{\mathrm{LSA}}(w|d) = \sum_{z} P(w|z;\theta)P(z|d;\pi)$$

Document language model

Latent concepts or topics

Concept expression probabilities

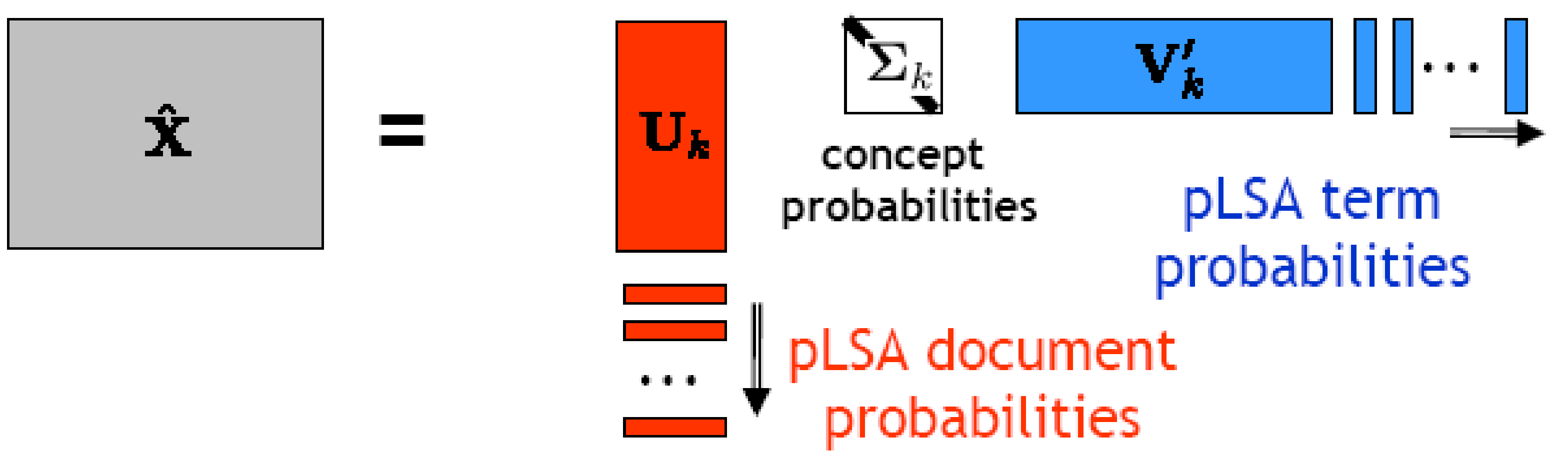
Document-specific mixture proportions

Model fitting

Similarity to LSA's SVD

$$\hat{P}_{\mathrm{LSA}}(d, w) = \sum_z P(d|z)\, P(z)\, P(w|z)$$

$$= P(d) \sum_z P(w|z) P(z|d)$$



Difference: sigma's values are normalized and non-negative, as they are probabilities

Learning the pLSA parameters

Observed counts of
word *i* in document *j*

$$L = \prod_{i=1}^{M} \prod_{j=1}^{N} P(w_i|d_j)^{n(w_i,d_j)}$$

$$\sum_{k=1}^{K} P(z_k|d_j) P(w_i|z_k)$$

Maximize likelihood of data using EM.

M … number of codewords

N … number of documents

Slide credit: Josef Sivic

EM for pLSA (training on a corpus)

- E-step: compute posterior probabilities for the latent variables

$$P(z_k \mid d_i, w_j) = \frac{P(w_j \mid z_k)P(z_k \mid d_i)}{\sum_{l=1}^{K} P(w_j \mid z_l)P(z_l \mid d_i).}$$

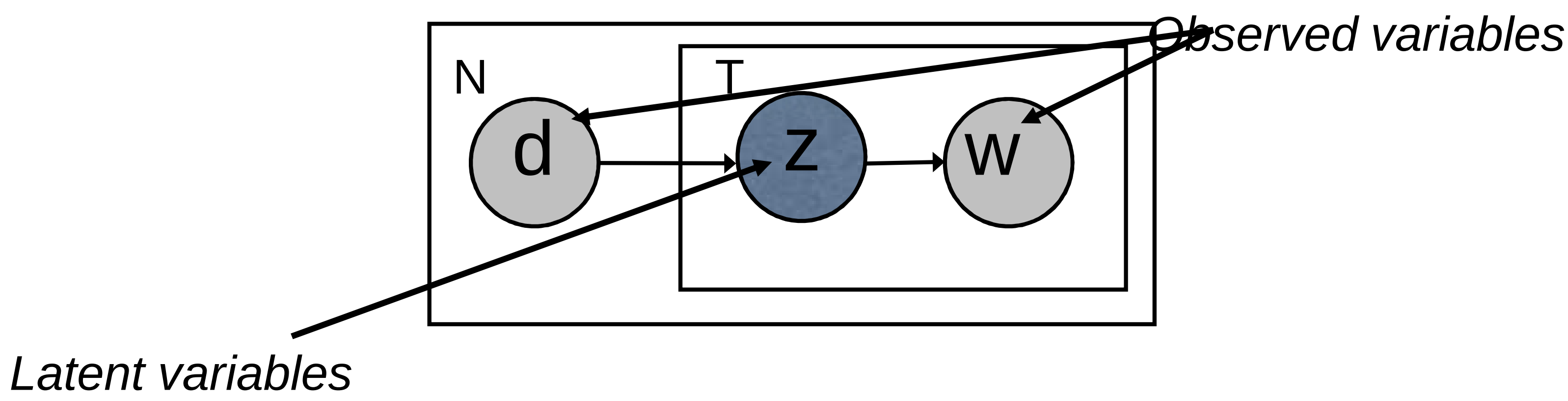
- M-step: maximize the expected complete data log-likelihood

$$P(w_j \mid z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j) P(z_k \mid d_i, w_j)}{\sum_{m=1}^{M} \sum_{i=1}^{N} n(d_i, w_m) P(z_k \mid d_i, w_m)},$$

$$P(z_k \mid d_i) = \frac{\sum_{j=1}^{M} n(d_i, w_j) P(z_k \mid d_i, w_j)}{n(d_i)}.$$

Graphical View of pLSA

- pLSA is a generative model



- Select a document d_i with prob $P(d_i)$
- Pick latent class z_k with prob $P(z_k|d_i)$
- Generate word w_j with prob $P(w_j|z_k)$

Problem: once calculated, there is no direct way to add new documents to the model without recalculating the probabilities again.

This is solved by the “fold in” heuristic, shown later on.

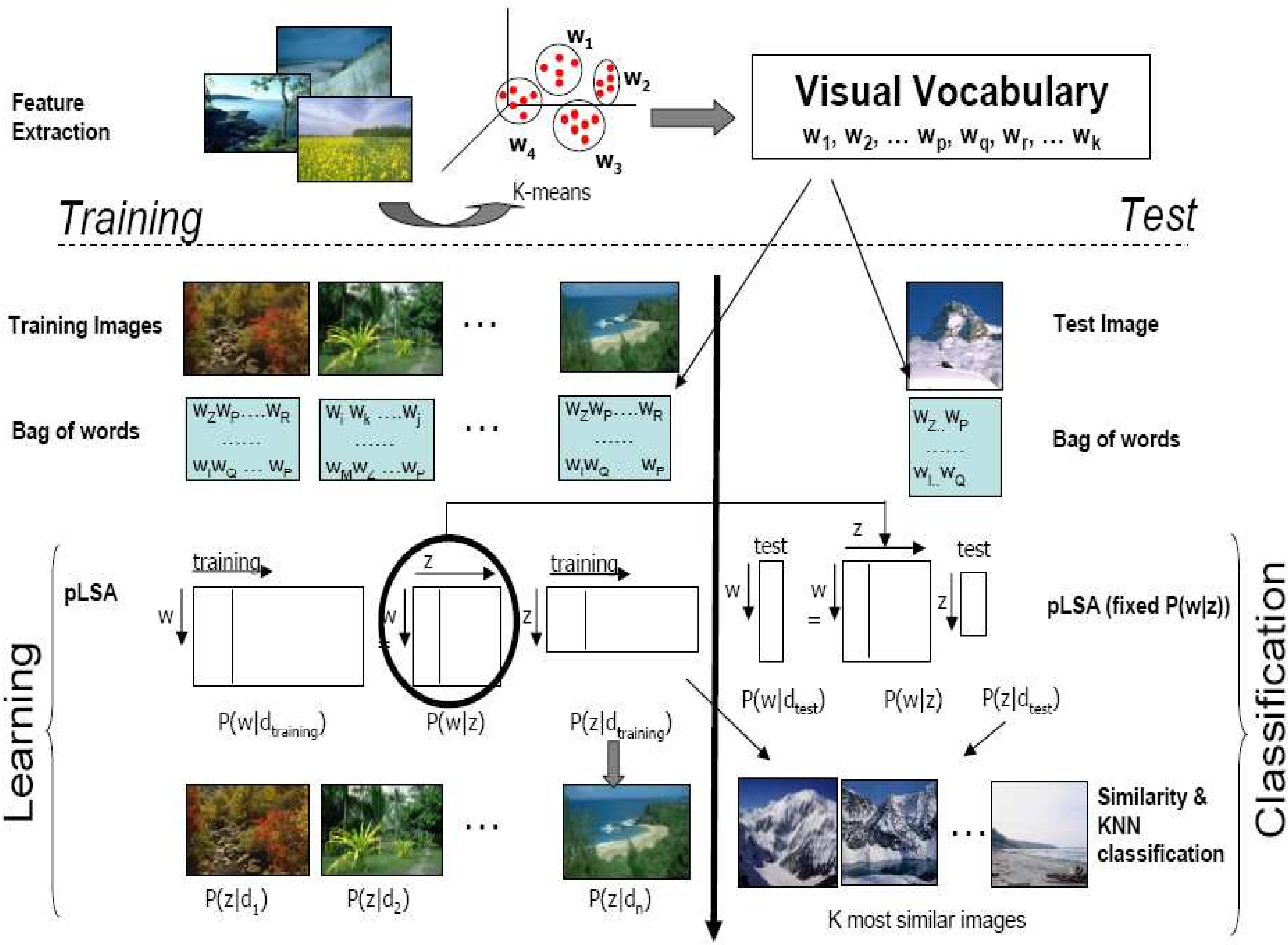
Scene classification

- Create visual words (denoted ‘w’).
- Learn the topic specific distribution P(w|z) from the training set by fitting the training set into the PLSA model.
- Each training image im is represented by a K vector of P(Z|im), where |Z|=K is the amount of topics.

Given a new image to classify, use the "fold in" heuristic – add the image to the corpus, and run the EM optimization again, only this time, keep the P(w|z) as they were, and only update P(Z|new) where new is the new image.

Now use a K nearest neighbors classifier to fine the K P(Z|im) vectors of the training set closest to P(Z|new)

Out of the |Z| topics, Find the topic that maximizes it's conditioned probability after each of the K neighbors.



Feature Extraction
w1
w2
w3
w4
K-means
Visual Vocabulary
w1, w2, … wp, wq, wr, … wk
Training
Test
Training Images
Bag of words
Test Image
Bag of words
Learning
pLSA
training
z
training
w
w
z
P(w|d_training)
P(w|z)
P(z|d_training)
test
z
test
w
w
z
P(w|d_test)
P(w|z)
P(z|d_test)
pLSA (fixed P(w|z))
Classification
P(z|d1)
P(z|d2)
P(z|dn)
K most similar images
Similarity & KNN classification

Visual words

Used four types of descriptors, and varied the parameters of each:

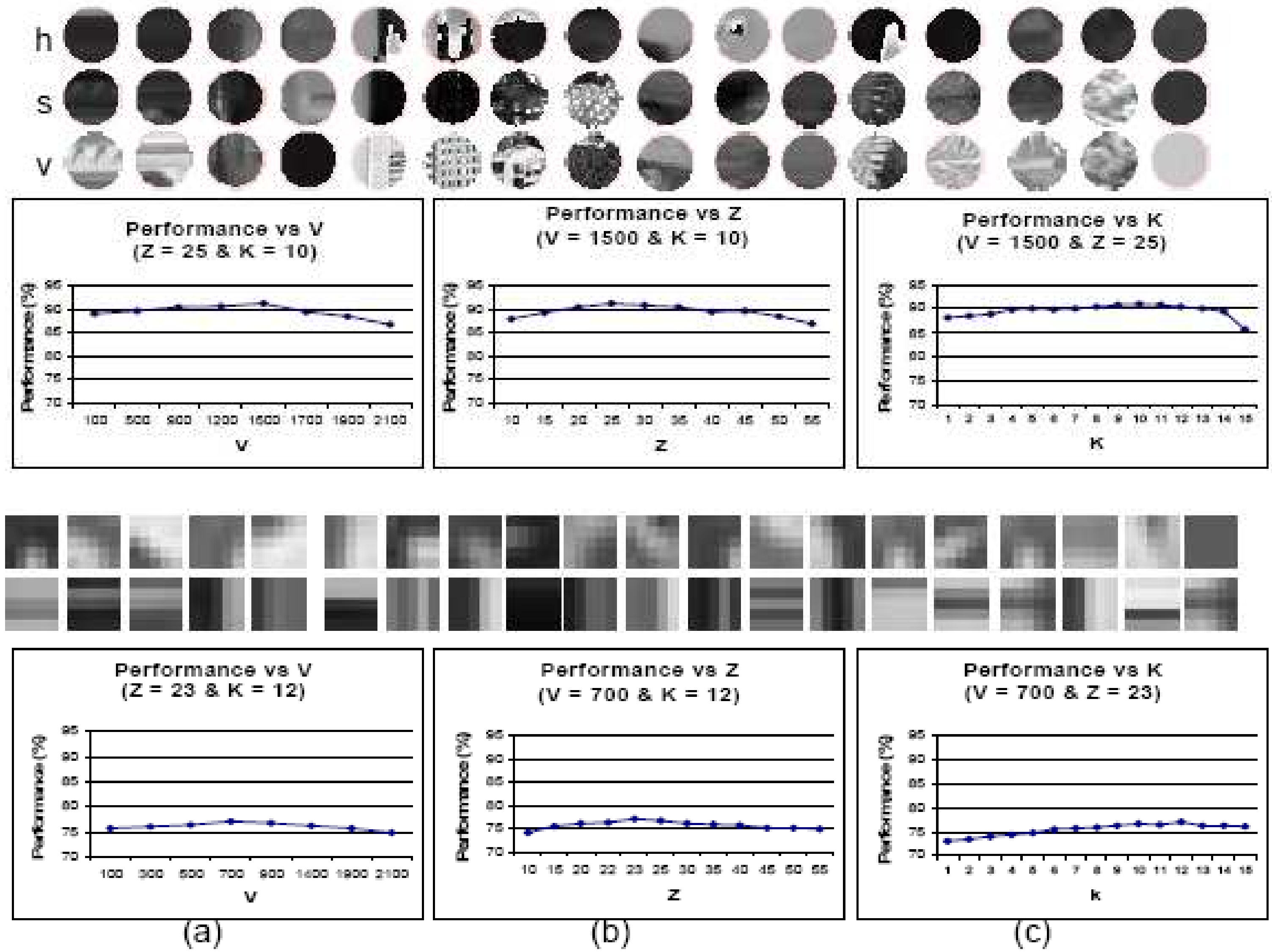
- Grey patches: represent a NXN area as a vector, using only grey values
- Color patches: same, with Color patches.
- Grey SIFT: computed at points with spacing M, each with radius R, with n dimensions
- Color SIFT: As above, only for the HSV components

Comparing to previous results

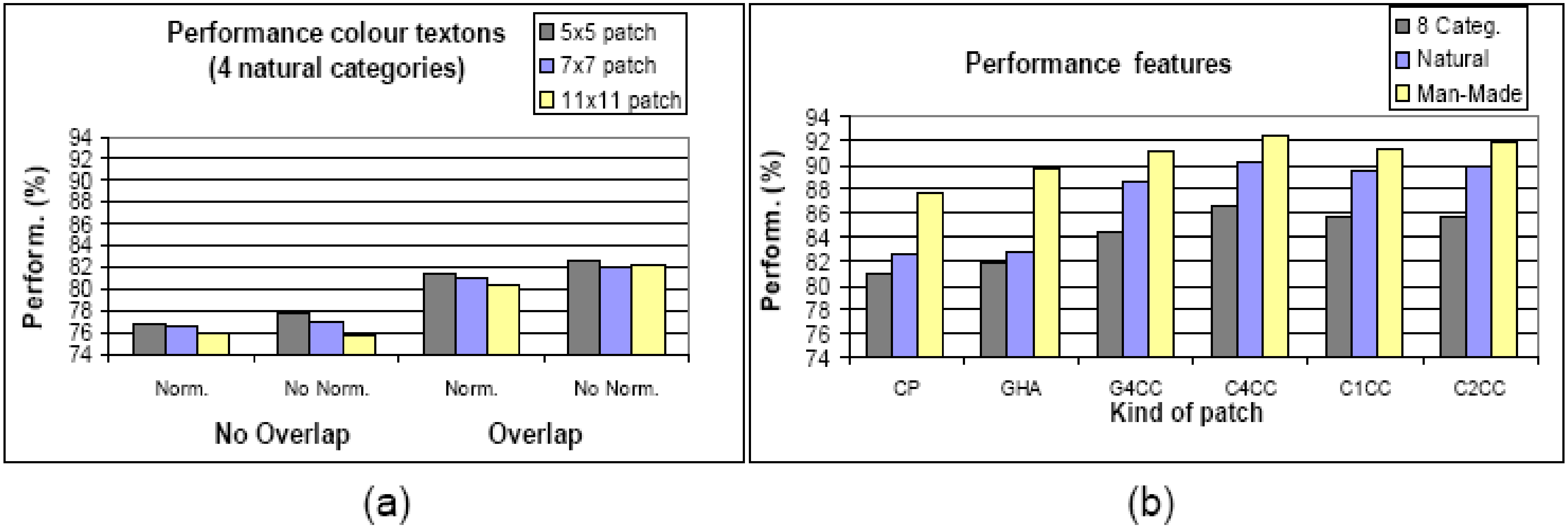
- Compare the performance of PLSA with these four dense descriptors to PLSA with a previously used sparse SIFT descriptor.
- Compare the results of the PLSA to simply using KNN on global HSV histograms and using KNN on the histogram of the gradient at each pixle
- Moreover – compare to simply using KNN on the bag of words (P(w|d))

Testing the algorithm

- The datasets are split in half, half for training and half for testing.
- results quality is tested by a confusion matrix. The more diagonal it is, the better the results.



Performance under variation in various parameters for the 8 category OT classification. Top: example visual words and performance for dense colour SIFT $M = 10$, $r = 4$, 8, 12 and 16 (each column shows the HSV components of the same word). Lower example visual words and performance for grey patches with $N = 5$ and $M = 3$. (a) Varying number of visual words, V, (b) Varying number of topics, Z, (c) Varying number k (KNN).



(a) The performance when classifying the four natural categories using normalized and unnormalized images and with overlapping and non-overlapping patches. Colour patches are used. (b) Performance when classifying all categories, man-made and natural using different patches and features. (CP = Colour patches - dense; GHA = Grey Harris Affine - sparse; G4CC = Grey SIFT concentric circles - dense; C4CC = Colour SIFT 4 concentric circles - dense; C1CC = Colour SIFT 1 Circle - dense; C2CC = Colour SIFT 2 concentric circles - dense.

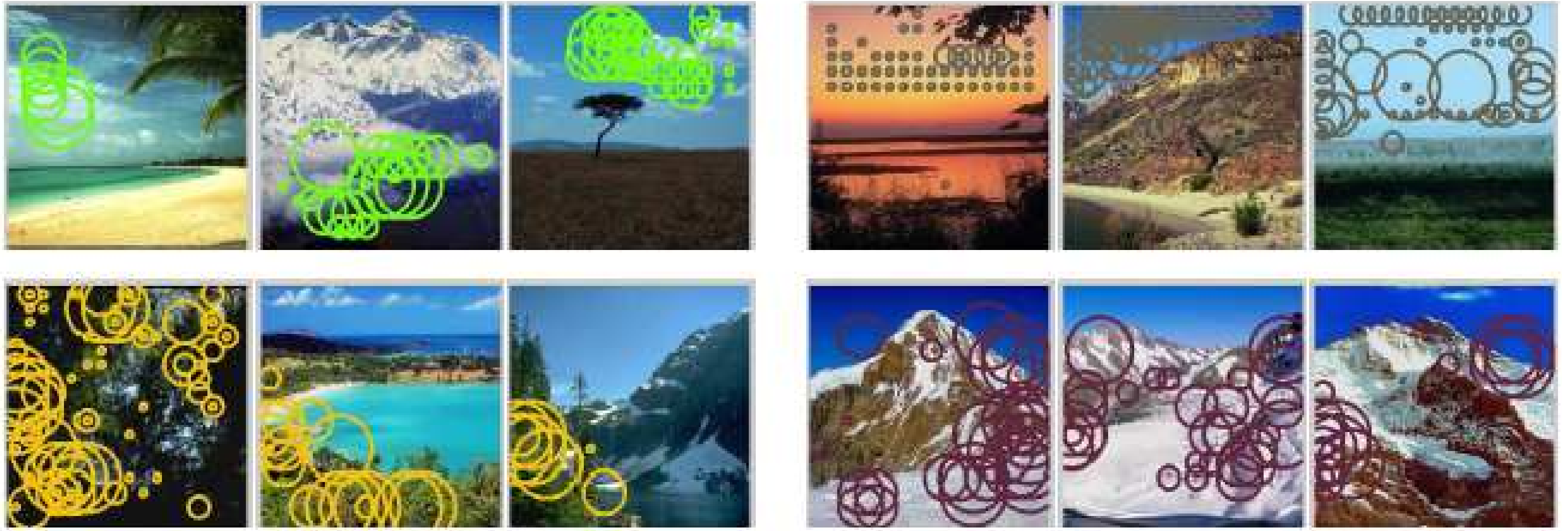
Best results with dense descriptors!

Note that nature scenes are more color dependant

Visual Vocabulary	GP	CP	G4CC	C4CC	PS	BOW	GlC	GlT
All categ.	71.51	77.05	84.39	86.65	82.6	82.53	55.12	62.21
Natural categ.	75.43	82.47	88.56	90.28	84.05	88.74	59.53	69.61
Man-made categ.	77.44	83.56	91.17	92.52	89.34	89.67	66.11	73.14

Rates obtained different features when using database OT: GP (Grey Patches), CP (Colour Patches), G4CC (Grey SIFT four Concentric Circles), C4CC (Colour SIFT four Concentric Circles), PS (Colour Patches and Colour SIFT), BOW (Bag-of-Words), GlC (Global colour), GlT (Global Texture).

- Baseline texture (GIT) perfroms rather well on man made scenes
- Man made is better classified than natural
- SIFT is the best, better than both patches and SIFT mixed with patches



Topics segmentation. Four topics (clouds – top left, sky – top right, vegetation – lower left, and snow/rocks in mountains – lower right) are shown. Only circular regions with a topic posterior $P(z|w,d)$ greater than 0.8 are shown.

# img. (nt)	2000	1600	1024	512	256	128	32
Perf. $P(z\|d)$	86.9	86.7	84.6	79.5	75.3	68.2	58.7
Perf. BOW	83.1	82.6	80.4	72.8	60.2	52.0	47.3

Comparison of $P(z|d)$ and BOW performance as the number of training images used in KNN is decreased. The classification task is into 8 categories from the OT dataset.

Comparison of P(z|d) to the simple BOW

Summary

- Best performance is achieved with dense color SIFT with overlapping regions.