ABSTRACT
We propose a new approach to exploit the different discriminability of image features at different scales simultaneously. By modifying the Bag-of-words model, we represent an image as a matrix whose elements are the occurrences of a set of codewords within different scale ranges. In this way, we can represent an image collection using a 3rd-order tensor. Then a new classification method, tensor-pLSA, which is an extension of Probabilistic Latent Semantic Analysis (pLSA), is introduced to classify these images based on this tensor representation. Finally, we compare the tensor representation with the original matrix representation to show the effectiveness of our approach.

Categories and Subject Descriptors
I.4.8 [Scene Analysis]: Object Recognition

General Terms
Algorithms, Design

Keywords
Image classification, Tensor representation, Tensor-pLSA

1. INTRODUCTION
In the area of image classification or object categorization, there is a key issue, that is, how to represent an image appropriately. In the literature, several approaches have been developed. Dance et al.[2] model an image as a histogram of occurrence of codewords and represent an image collection as a matrix (Bag-of-words model). Yan et al.[10] model an image as a general tensor of 2nd or higher order using Gabor features. Fergus et al.[4] model an image as a flexible constellation of parts using a probabilistic representation for the shape, appearance, occlusion and relative scale of the object in the image. Epshtein et al.[5] model an image as a collection of hierarchical fragments. Berg et al.[1] model an image as a set of feature points in the framework of deformable shape matching.

It is difficult to say which approach above is the best because of their different advantages and disadvantages. However, we believe that all the useful information of an image should be exploited as much as possible.

From the observation of patches in an image, we find that within different scale ranges, the patches have different discriminability. However, in the Bag-of-words model, the scale information of patches is ignored. In order to make good use of the scale information, in this paper we model an image as a matrix instead of a vector used in the Bag-of-words model and employ a 3rd-order tensor to represent an image collection. We call this representation of an image collection TR for short. In [9], Sivic, et al. employ Probabilistic Latent Semantic Analysis (pLSA)[5] to classify the images based on the matrix representation of the image collection, which can learn the discriminability of features well. We call this representation of an image collection MR for short. Enlightened by [9], we extend pLSA, namely tensor-pLSA, to classify the images based on TR.

This paper is organized as below. In Section 2, TR is introduced in details. Section 3 explains how tensor-pLSA classifies the images based on TR. In Section 4, the comparison results of TR and MR are shown. Section 5 concludes the paper.

2. TENSOR REPRESENTATION
Mathematically, tensors are simple arrays of numbers or functions, that transform according to certain rules under a change of coordinates. Tensors can be described using indices along different coordinates. In this way, we can see that 1st-order, 2nd-order and 3rd-order tensors can be considered as vectors, matrices and cubes, respectively. In the remaining parts of this paper, all tensors are 3rd-order.

Fig. 1 illustrates how to construct a tensor for image classification. First, different scale ranges should be pre-defined. Then within each scale range, a patch detector is used to detect the interesting patches. Further these patches are described by descriptors, mapping these patches into a high dimensional vector space. Using clustering methods such as k-means, a codebook is created by the descriptors of the training patches, where the codewords are the centers of the clusters in the vector space. Replacing each descriptor with the nearest codeword in the feature space, each image can be represented as a histogram of occurrence of codewords. Ad-
Figure 1: An example to show how to construct a tensor to represent the image collection from different categories. The data shown here are artificial.

joining the histograms of an image created within different scale ranges, this image can be re-represented as a matrix.

Finally, the image collection can be represented as a tensor.

TR has 3 dimensions, denoted as “images”, “scale ranges” and “codewords”. See Fig. 2. Here a “codeword set” consists of the codewords at the same index position in the different codebooks corresponding to the different scale ranges.

Figure 2: A tensor to represent the image collection.

3. PLSA AND TENSOR-PLSA

3.1 pLSA

pLSA aims to introduce an aspect model, which builds an association between documents (images) and words (codewords) through the latent aspects by probability. By training a word set whose items are more likely to occur together and creating a context, it successfully handles the problem of polysemy. The graphic model of pLSA is illustrated in Fig. 3 (a).

Suppose \( D = \{d_1, \ldots, d_i\} \), \( W = \{w_1, \ldots, w_j\} \) and \( Z = \{z_1, \ldots, z_K\} \) denote a document set, a word set and a latent topic set, respectively. We would like to find the probability of documents and words as:

\[
P(d_i, w_j) = \sum_k P(w_j | z_k) P(z_k | d_i) P(d_i)
\]

pLSA tries to maximize the \( \mathcal{L} \) function below:

\[
\mathcal{L} = \sum_i \sum_j n(d_i, w_j) \log P(d_i, w_j)
\]

For learning and inference, pLSA employs Expectation Maximization (EM) algorithm to learn the distributions of \( P(w_j | z_k) \) and \( P(z_k | d_i) \) to maximize \( \mathcal{L} \). In E-step, the posterior probabilities for the latent variables \( P(z_k | d_i, w_j) \) are calculated, and in M-step, \( P(d_i, w_j) \) is updated. Here we restate the process of EM algorithm. Notice that these equations need normalization to make them probability distributions.

E-step:

\[
P(z_k | d_i, w_j) \propto P(w_j | z_k) P(z_k | d_i)
\]

M-step:

\[
P(w_j | z_k) \propto \sum_i n(d_i, w_j) P(z_k | d_i, w_j)
\]

\[
P(z_k | d_i) \propto \sum_j n(d_i, w_j) P(z_k | d_i, w_j)
\]

In the learning stage, pLSA learns the distributions of \( P(z_k | d_i, w_j) \), \( P(w_j | z_k) \), \( P(z_k | d_i) \) and \( P(d_i) \) using EM algorithm. Then in the inference stage, keeping \( P(w_j | z_k) \) unchanged, pLSA re-learns the distributions of \( P(z_k | d_i, w_j) \), \( P(z_k | d_i) \) and \( P(d_i) \) using EM algorithm. Images are classified based on \( P(z_k | d_i) \).

3.2 Tensor-pLSA

pLSA classifies the images based on \( MR \). However, if we use \( TR \), we need to extend pLSA. Fig. 3 (b) shows the graphic model of tensor-pLSA, where \( d \) denotes images, \( z \) denotes topics, \( s \) and \( w \) denote the codeword sets and the codewords in the tensor, respectively.

Like pLSA, given an image, we would like to find its probability distribution among different topics, and then based
on these values classify it. Also, based on the graphical model of tensor-pLSA, we can model the joint probability of images, codeword sets and codeword as:

\[ P(d_i, s_r, w_j) = \sum_k P(w_j|s_r, z_k)P(s_r|z_k)P(z_k|d_i)P(d_i) \]  

(3)

In tensor-pLSA, \( P(s_r|z_k) \), \( P(z_k|d_i) \) and \( P(d_i) \) have similar meanings to those in pLSA, except the meaning of \( s_r \). And \( P(w_j|s_r, z_k) \) denotes the probability of codeword \( w_j \) occurring in codeword set \( s_r \) when the topic is \( z_k \). And tensor-pLSA tries to maximize the \( H \) function below:

\[ H = \sum_{i} \sum_{r} \sum_{j} n(d_i, s_r, w_j) \log P(d_i, s_r, w_j) \]  

(4)

Also, EM algorithm is employed to estimate the distributions of \( P(w_j|s_r, z_k) \), \( P(s_r|z_k) \) and \( P(z_k|d_i) \) so that the joint probability \( H \) can be maximized: In E-step, the posterior probabilities for the latent variables \( P(z_k|d_i, s_r, w_j) \) are calculated, and in M-step, \( P(d_i, s_r, w_j) \) is updated. Here we list the updating rules in the EM algorithm in tensor-pLSA.

E-step:

\[ P(z_k|d_i, s_r, w_j) \propto P(s_r|z_k)P(z_k|d_i)P(w_j|z_k, s_r) \]

M-step:

\[ P(s_r|z_k) \propto \sum_i \sum_j n(d_i, s_r, w_j)P(z_k|d_i, s_r, w_j) \]
\[ P(z_k|d_i) \propto \sum_r \sum_j n(d_i, s_r, w_j)P(z_k|d_i, s_r, w_j) \]
\[ P(w_j|z_k, s_r) \propto \sum_i \sum_j n(d_i, s_r, w_j)P(z_k|d_i, s_r, w_j) \]

Similar to pLSA, the distributions of \( P(w_j|s_r, z_k) \) and \( P(s_r|z_k) \) are learned in the learning stage, and they are kept the same in the inference stage to estimate other distributions. The images are classified according to the distribution of \( P(z_k|d_i) \). In the experiments, the average distribution of \( P(z_k|d_i) \) for each category is learned using the training images. Subsequently a test image is classified into the category whose average distribution on the topics is closest to that of the test image measured by the norm-1 distance.

4. EXPERIMENTS

The purpose of this paper is to show the benefits of TR compared with MR, integrated with there corresponding classification methods. Based on this consideration, we will compare their results in the following experiments.

4.1 Dataset

We compare TR with MR using two datasets: the Caltech image dataset [9] and COIL-20 [8]. See Fig. 4. Four categories are selected from Caltech dataset: motorbikes, faces, airplanes and cars (rear). COIL-20 comprises 20 objects which are placed on a motorized turntable against a black background. Images of the objects are taken at pose intervals of 5 degrees, resulting in 72 images per object. These images have the same size (128*128 pixels) and are gray-scale.

4.2 Implementation

4.3 Image classification

Here, we use Caltech images to compare the classification performance between TR and MR. First, we will show the usefulness of TR, then we will show the comparison results.

4.3.1 Learning ability of tensor-pLSA

In our experiments, all the images from the Caltech dataset are gray-scale and resized to 300-pixel wide, while those from COIL-20 are kept unchanged. For the Caltech dataset, we randomly select 400 images with 100 from each category. For the COIL-20 dataset, we use all the images, 1440 in total. Each dataset is split randomly into two separate sets with equal size: training set and test set.

For TR, considering the number of patches that can be detected, the scale ranges of patches for Caltech images are pre-defined as [10 20], [20 30], [30 40], [40 50] and [50 60], while for COIL-20 images they are [5 10], [10 15], [15 20], [20 25] and [25 30], because the sizes of the Caltech images are larger than those of the COIL-20 collection. Within each scale range, we extract the local image patches using the saliency region detector[6], which tends to give a more manageable number of informative features per image as compared to other detectors such as multiscale Harris. Each local patch is resized to 16*16 pixels, and further divided into 4*4 smaller patches, each with 4*4 pixels. Each smaller patch is then represented by an 8-dimension gradient bins similar to SIFT descriptor[7]. Concatenating these 16 smaller patch descriptors together and normalizing the vector, we obtain a normalized 128-dimension vector for each local patch. Then using these descriptors within different scale ranges, a tensor can be created for the image collection. Next, tensor-pLSA is employed to classify the images. For the EM in tensor-pLSA and pLSA, the maximal number of iterations is 500, and the minimal allowable likelihood change is 1. The number of topics should not be too small or too large (relative to the number of categories), considering the discriminability of the distributions on the topics and the computational time. In our experiments, the number of topics is fixed at 8 in Caltech experiments and 40 in COIL-20 experiments. For MR, the process is similar. See the details in [9].

We use average test error rate (ATER) of 10 runs to compare the results. Test error rate is calculated using the number of misclassified images in all the categories divided by the number of all the test images in all the categories. And the performance improvement (PI) is calculated over ATER.
Figure 5: Comparison in classification.

Fig. 5(a) illustrates the relationship between \( ATER \) and the number of scale ranges within which features are used to create the tensor. In this experiment, the scale ranges are chosen randomly from the pre-defined scale ranges.

From Fig. 5(a), we can see clearly that regardless of the different sizes of the codebooks, \( ATER \) always decreases when the number of scale ranges increases. Comparing the performance of using 1 scale-range features with that of using 5 scale-range features based on different codebook sizes, \( PI \)s are 59.8%, 66.1%, 59.7%, 67.1% and 72.9%, respectively. It demonstrates that tensor-\( p \)LSA can learn the discriminability of all the features properly and use this information to improve the classification performance. Increasing the number of scale ranges will need more storage and computational time. So in the following experiments, we use 5 scale-range features to create the tensor.

4.3.2 Comparison

Fig. 5(b) shows the comparison results between tensor-\( p \)LSA and \( p \)LSA using different sizes of the codebooks. Here the matrix of the image collection based on which \( p \)LSA classifies the test images is created by all the features from different scale ranges. Similar to the process for creating \( TR \), for \( p \)LSA all the features of training images are clustered to form a codebook where the codewords are the centers of the clusters, then each image can be represented as a histogram of occurrence of codewords, and using these histograms \( MR \) can be created. In other words, the process for extracting the information in both methods is the same, but our method, \( TR \), makes full use of the scale information of each feature, whereas \( MR \) does not distinguish features from different scales.

From Fig. 5(b), we can see that in most cases \( ATER \) using tensor-\( p \)LSA is smaller than that using \( p \)LSA for different codebook sizes. For tensor-\( p \)LSA, the mean and the standard deviation of these 8 \( ATER \)s are 0.080 and 0.017, whereas for \( p \)LSA, the corresponding values are 0.094 and 0.023. Thus the classification performance based on \( TR \) is better and more stable than that based on \( MR \).

4.4 Object recognition

We use COIL-20 images to compare the ability of object recognition between \( TR \) and \( MR \). Fig. 6 shows the confusion matrices using \( TR \) and \( MR \) with the size of the codebooks fixed at 200. The mean \( ATER \)s are 0.13 and 0.18, respectively, with \( PI \) of 27.8%. Moreover, there are only 21 misclassified positions in the \( TR \) confusion matrix compared to 36 in the \( MR \) confusion matrix. This means that \( TR \) can discriminate one object from others much better than \( MR \).

Based on these observations, we can infer that for object recognition, \( TR \) is better and more discriminative than \( MR \).

5. CONCLUSION

In this paper, we propose a tensor representation and a new classification method, tensor-\( p \)LSA, for image classification. Experiments show that the tensor representation can capture more useful information than the original matrix representation based on the Bag-of-words model, and when combined with tensor-\( p \)LSA, a more discriminative model for each category can be learned to improve the classification performance.

6. REFERENCES