Computer Vision and Image Understanding xxx (2010) xxx–xxx

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Towards a more discriminative and semantic visual vocabulary

R.J. López-Sastre [a,*], T. Tuytelaars [b], F.J. Acevedo-Rodríguez [a], S. Maldonado-Bascón [a]

[a] University of Alcalá, Department of Signal Theory and Communications, GRAM, 28805 Alcalá de Henares, Spain
[b] Catholic University of Leuven, ESAT/PSI-VISICS/IBBT, Kasteelpark Arenberg 10, B-3001 Heverlee, Belgium

ARTICLE INFO

Article history:
Received 28 February 2010
Accepted 1 October 2010
Available online xxxx

Keywords:
Category-level object recognition
Visual vocabulary
Correlation clustering
Cluster precision
Reciprocal Nearest Neighbours
Bag-of-Words

ABSTRACT

We present a novel method for constructing a visual vocabulary that takes into account the class labels of images, thus resulting in better recognition performance and more efficient learning. Our method consists of two stages: Cluster Precision Maximisation (CPM) and Adaptive Refinement. In the first stage, a Reciprocal Nearest Neighbours (RNN) clustering algorithm is guided towards class representative visual words by maximising a new cluster precision criterion. As we are able to optimise the vocabulary without the need for expensive cross-validation, the overall training time is significantly reduced without a negative impact on the results. Next, an adaptive threshold refinement scheme is proposed with the aim of increasing vocabulary compactness while at the same time improving the recognition rate and further increasing the representativeness of the visual words for category-level object recognition. This is a correlation clustering based approach, which works as a meta-clustering and optimises the cut-off threshold for each cluster separately. In the experiments we analyse the recognition rate of different vocabularies for a subset of the Caltech 101 dataset, showing how RNN in combination with CPM selects the optimal codebooks, and how the clustering refinement step succeeds in further increasing the recognition rate.

1. Introduction

A popular strategy for representing images within the context of category-level object recognition is the *Bag-of-Words* (BoW) approach [1,2]. The standard pipeline for this type of systems is shown in Fig. 1. It starts with the extraction of local features either at interest points or densely sampled followed by robust description of the features, e.g. using SIFT [3] (see steps 1 and 2 in Fig. 1). The key idea is to vector quantise the high-dimensional space of local image descriptors, to obtain a codebook of so-called visual words, sometimes also referred to as a visual vocabulary (Fig. 1, step 3). A BoW is then built as a histogram of visual word occurrences (Fig. 1, step 4). This image representation has been shown to characterise the images and objects within it in a robust yet descriptive manner, in spite of the fact that it ignores the spatial configuration between visual words.

These BoW systems have shown impressive results lately [4,5]. Variations on the BoW scheme won the recent PASCAL Visual Object Classes Challenge on object classification [6]. Moreover, also other methods for category-level object recognition often start with the construction of a visual vocabulary (e.g. [7,8]).

In the literature (e.g. [9]), visual words are sometimes justified on the basis of their ability to group semantically meaningful object parts such as wheels or eyes, hence narrowing the semantic gap. In practise though, this only holds for a limited number of visual words, and only if the dataset is sufficiently coherent (e.g. only images of one particular object class). When applied to a more diverse set of images, synonyms and polysemies are, unfortunately, the norm rather than the exception [10].

Using a fixed feature detector and descriptor scheme, different vocabularies of varying quality can be obtained for the same data set, depending on several parameters: the number of visual words, the chosen distance function to measure the similarity between descriptors, and the clustering algorithm.

These parameters are normally determined empirically, i.e. by applying a grid search over different parameter combinations, training the entire system, and selecting the optimal parameter setting on a validation set. Given that the normally used classifiers (e.g. SVM) also have their own set of parameters that require tuning, this results in a computationally heavy process, usually spread over multiple cores or days of processing.

Our aim in this paper is to study how to adapt the vector quantisation process so as to yield class representative visual words, i.e. how to exploit the class labels information during the process of vocabulary construction. This allows us to construct an optimal visual vocabulary without the need for cross-validation in the outer system-loop.

The contribution this paper makes is twofold. First we propose a novel cluster precision criterion. This evaluates the clusters representativeness. High representativeness is assigned to visual words

* Corresponding author. Fax: +34 91 885 6699.
E-mail address: robertoj.lopez@uah.es (R.J. López-Sastre).

1077-3142/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2010.10.009

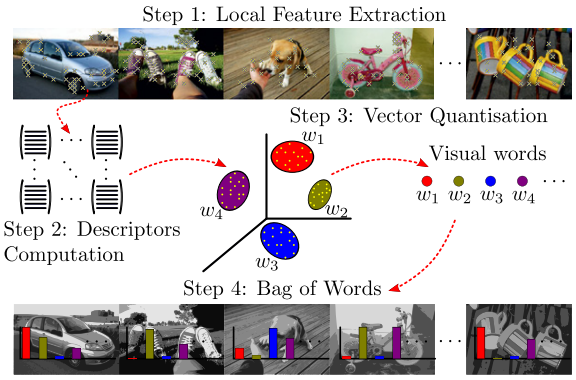

Fig. 1. BoW system overview. It starts with the extraction of local features either at interest points or densely sampled followed by robust description of the features, e.g. using SIFT [3] (steps 1 and 2). Step 3 consists in vector quantising the high-dimensional space of local image descriptors to obtain a visual vocabulary. A BoW is then built as a histogram of visual word occurrences (step 4).

that generalise well over the within-class variability, yet are discriminative with respect to the object class (between-class variability). We determine the optimal clustering by maximising this measure in an agglomerative clustering approach. We refer to this as Cluster Precision Maximisation (CPM).

Second, a new adaptive cluster threshold refinement scheme is proposed. It is based on a correlation clustering scheme [11], and its objective is to increase the recognition rate and representativeness of the codebook while at the same time reducing its size. The correlation clustering of an n vertex weighted graph is the partition of vertices which minimises the sum of positive weights that are cut minus the negative weights that are uncut. The number of clusters is not fixed by the user, but determined as part of the clustering process. The key idea behind our refinement procedure is to exploit the object class label of each feature in a traditional correlation clustering algorithm, i.e. we consider two clusters to be similar if their features come from the same object classes. A meta-clustering step such as this might also be useful in other scenarios (as long as cluster similarity is well-defined). To the best of our knowledge, this is the first paper to describe a correlation clustering approach within this context.

Both contributions pursue the same objective: to increase the representativeness of the visual words. The basic assumption behind the whole paper is that the better this representativeness, the better the classification rate (and this will indeed be validated experimentally as well).

Fig. 2 shows two examples of visual words. The first (upper row) represents a bad cluster with image patches that clearly come from different object classes and from single object instances, not generalising well within the class. The second example (lower row) on the other hand, shows a class representative visual word found almost exclusively on objects of a single class and including different instances of this class. Our goal is to have more clusters of the latter type than what one normally gets when using e.g. standard K-means clustering.

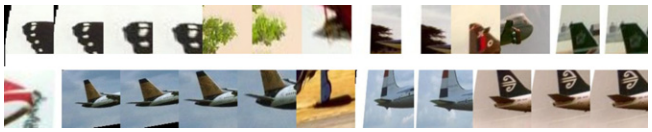

Fig. 2. First row: image patches that have been clustered together. Clearly all come from different object categories and from single object instances. Second row: cluster with image patches of the same object category and from different instances of the class, i.e. it contains class representative visual words.

1.1. Overview

The rest of the paper is organised as follows. Section 2 gives an overview of the related literature. Section 3 first formalises the cluster precision computation problem, then gives a complete description of the CPM approach. The adaptive threshold refinement using correlation clustering is explained in Section 4. Section 5 shows the results obtained and Section 6 concludes the paper.

2. Related work

This section gives a brief survey of recent work on both class representative visual words and correlation clustering.

2.1. Class representative visual words

First, there are several works based on frequent itemset mining [12–16]. Typically, these methods look for frequent co-occurring groups of descriptors in a transaction database obtained from the training images. These often correspond to larger, higher level features that are semantically more meaningful than the composing elements/original features separately (e.g. a wheel rather than an edge or corner).

In a similar spirit, others have tried to add more local geometric information to their codebook generation algorithms. Lazebnik et al. [17] constructed a codebook with groups of nearby regions whose appearance and spatial configuration occur repeatedly in the training set. In [18] Leibe et al. presented how to learn semantic object parts for object categorisation. They use what they call co-location and co-activation to learn a visual vocabulary that generalises beyond the appearance of single objects, and often obtains semantic object parts.

Perronnin et al. [19] combine a universal vocabulary with class-specific vocabularies to improve the performance of the recognition system, in spite of the increased cost of histogram computations. To obtain more compact vocabularies Winn et al. [20] proposed an approach based on the bottleneck principle, while Moosmann et al. [21] organised the vocabulary using Extremely Randomised Clustering Forests. Finally, Perronnin et al. [22] have proposed the use of Fisher Kernels. Their gradient representation has much higher dimensionality than a histogram representation, resulting in very compact vocabularies yet highly informative representations.

However, in the previous works, the derived vocabularies are not universal. When one needs to add new categories, the whole system needs to be fully retrained. Furthermore, it has been observed, even on databases containing a restricted number of categories (e.g. $\leqslant$10), that using more visual words first results in better performance [22]. But increasing the number of visual words to certain levels finally saturates the performance of vision tasks. To find the optimal codebook size is a complex procedure that should be as effective as possible.

In order to overcome these limitations we present an efficient method for obtaining class representative visual words. It automatically determines the codebook size that allows the best recognition rates by maximising the precision of clusters.

Closer to our approach are the works of Mikolajczyk et al. [23] and Stark et al. [24]. In [23] the performances of local detectors and descriptors are compared within the context of object class recognition, and a new evaluation criterion based on the clusters' precision is proposed. However, following this approach, clusters with features from only one object instance get high precision

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009

scores. Stark et al. [24] give higher scores to feature descriptors that generalise across multiple instances of an object class, and propose a new cluster precision definition, but their approach obtains the best score for the degenerate case when each cluster contains only one vector. Nevertheless, our methodology is able to identify a maximum for the clustering precision, as well as to predict how the classifier will perform.

2.2. Background: correlation clustering

There is an extensive literature on the problem of correlation clustering, first defined by Bansal et al. [11]. They consider the problem of having a complete graph of n vertices, where each edge (u, v) is labelled either +1 or −1 depending on whether u and v have been deemed to be similar or different. The objective is to partition the nodes in order to minimise the number of positive edges that are cut, and the number of negative edges that are uncut. Cutting an edge, means that the nodes are not merged. The best known approximation algorithm for this problem is by Charikar et al. [25] who give an LP-based algorithm that achieves an approximation factor of 4. When the edge weights are arbitrary, the problem is equivalent to the multicut problem shown in Demaine et al. [26], and there is a $O(\log n)$-approximation bound. The approach implemented in this paper is based on the correlation clustering technique described by Gionis et al. [27,28], where a graph with edge weights W_{uv} satisfying the triangle inequality must be given. These weights represent how dissimilar vertices linked by the edge are.

Spectral clustering methods also build a graph, and then try to find the optimal cut (e.g. normalised cuts [29]). They rely on the eigen-decomposition of a modified similarity matrix to project data prior to clustering. Choosing a good similarity graph is not trivial, and spectral clustering techniques can be quite unstable under different choices of the parameters for the neighbourhood graphs. Correlation clustering techniques use a fully-connected graph. Moreover, the number of clusters is determined as part of the optimisation process, i.e. one does not need to fix the number of clusters as a separate parameter.

3. Obtaining class representative visual words via Cluster Precision Maximisation

In this section we present a novel methodology for obtaining more discriminative visual vocabularies. We define a new measure for cluster precision and show how this measure can be maximised using a Reciprocal Nearest Neighbours (RNN) clustering algorithm, resulting in class representative visual words.

Whether a particular clustering (or codebook) is better than the baseline or not can be evaluated within the context of the entire system, i.e. by evaluating its effect on the accuracy of the resulting classifier on a validation set. The main problem with this approach is clear: to validate just the clustering it is necessary to go through the system's entire pipeline. Moreover, this validation usually in turn involves cross-validation for tuning the parameters of the classifier. The combinatorics involved results in an explosion of the number of experiments needed. Moreover the result also becomes dependent on the classifier chosen.

Our aim is to directly build a codebook whose clusters have a high precision and representativeness. The Cluster Precision Maximisation (CPM) method searches directly for class representative visual words, i.e. representative clusters. The key idea behind this approach is that, the more images with different object instances of a particular class contain a particular visual word, the more representative the word is for that class and the better the classification based on a codebook that includes it will be.

3.1. Cluster precision

Let us assume a database $\mathcal{DB}$ containing N images of M different object classes, $\mathcal{DB} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_N\}$. For each image in the database we first extract local features f, so an image $\mathcal{I}_i$ can be represented with a set of features $\mathcal{I}_i = \{f_{i_1}, f_{i_2}, \ldots\}$. Note that this will not be a BoW representation until the vector quantisation is done. After the clustering, a codebook $\mathcal{W} = \{w_1, w_2, \ldots, w_K\}$ with K words is obtained, and thus the images can be represented with it. See Fig. 1 for a graphical overview of a traditional BoW approach.

3.1.1. Cluster precision following Mikolajczyk et al. [23]

A codebook $\mathcal{W}$ can be evaluated by computing the average cluster precision for all the object classes. Suppose there are K clusters in the vocabulary, but only K_m for which class m dominates, i.e. clusters where there are more vectors belonging to class m than any other. Mikolajczyk et al. [23] define the average precision for a given object class m as

$$P_m^{([23])} = \frac{1}{K_m} \sum_{j=1}^{K_m} p_{j_m}, \tag{1}$$

where p_{j_m} is the number of features of class m in cluster j divided by the total number of features in cluster j.

Eq. (1) was used in [23] for comparing the performance of several local detectors and descriptors within the context of object recognition. The main problem with Eq. (1) is that clusters for which none of the classes dominate do not have any impact on the score at all. For instance, in Fig. 3 (left), the clustering $\{C_1, C_2, C_3, C_4\}$ obtains cluster precisions of 1 for all classes, that is the same score as for a nearly perfect clustering as $\{C'_1, C'_2, C'_3, C'_4\}$. Moreover, Eq. (1) can obtain high scores for those clusters that contain features from only a single object instance, as shown in Fig. 3 (right).

3.1.2. Cluster precision following Stark et al. [24]

Stark et al. [24] discount such clusters by summing over the fraction of *objects* of a class m in cluster j (p'_{j_m}) instead of individual features, and weight these fractions by cluster sizes, obtaining a new expression,

$$P_m^{([24])} = \left(\sum_{j=1}^{K'_m} s_j \right)^{-1} \sum_{j=1}^{K'_m} s_j p'_{j_m}, \tag{2}$$

where j now ranges over all K'_m clusters in which objects of class m dominate and s_j is the total number of features in cluster j. This new cluster precision definition gives higher scores to clusters that generalise across multiple instances of an object class. However, it casts the maximum score when each cluster contains only one vector again.

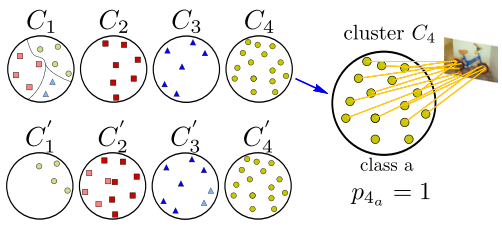

Fig. 3. Left – Following Eq. (1), clustering $\{C_1, C_2, C_3, C_4\}$ obtains cluster precisions of 1 for all classes, that is the same score for a nearly perfect clustering as $\{C'_1, C'_2, C'_3, C'_4\}$. Right – Can the Eq. (1) determine whether or not cluster C_4 is representative? As it is shown, cluster C_4 would get a high score for P_m, but it is not a class representative cluster. It brings together a lot of features, from class a, but only one image is represented by it.

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009

3.1.3. Our cluster precision formulation

Because neither of the two previous cluster precision definitions seem to meet our goal of selecting class representative visual words without artefacts, we propose a new cluster precision, this time summing over the number of features of class m times the number of objects of class m in each cluster,

$$P_m = \frac{K}{M}\sum_{j=1}^{K} s_{j_m} n_{j_m}, \tag{3}$$

where s_{j_m} is the number of features found in images of object class m in cluster j, n_{j_m} is the number of different objects of class m represented in cluster j, K is the number of clusters and M is the number of classes. This is illustrated in Fig. 4. This new cluster precision definition varies from $S \times S_m/M$ (when there is a cluster per feature), to $S_m \times N_m/M$ (when one cluster brings together all the features), S_m being the number of features extracted from images of the object class m, S the total number of extracted features and N_m the number of different images of class m in the database.

3.2. Cluster Precision Maximisation

Eq. (3) is the core of the CPM approach. With this new formulation, it is possible to evaluate the cluster's precision and representativeness after each iteration i of the agglomerative hierarchical clustering. Let $RNN(t_i)$ be the execution of the RNN clustering algorithm with threshold t_i in the iteration i, then the overall cluster precision is

$$P^{(i)} = \sum_{m=1}^{M} P_m^{(i)}, \tag{4}$$

where superscript i indicates the iteration number and we sum over all object classes. The aim of CPM is finding the value of the threshold t_i that maximises Eq. (4)

$$t_{opt} = \text{argmax}_{t_i \in [t_{min}:t_{max}]}\, P^{(i)}. \tag{5}$$

These threshold limits t_{min} and t_{max} must be chosen depending on the distance function used and the normalisation carried out with the data. Fig. 5 shows how $P^{(i)}$ evolves while the threshold varies in each iteration. For the first iteration, e.g. $t_0 = 0$, RNN casts only singletons and $P^{(i=0)} = S^2/M$. As the threshold increases, $P^{(i)}$ should also grow until a maximum is reached ($t = t_{opt}$). Then it quickly drops until RNN casts only one cluster bringing together all vectors ($t = t_{max}$) and

$$P^{(i=max)} = \sum_{m=1}^{M} \frac{S_m \times N_m}{M}. \tag{6}$$

Normally $S_m \ggg N_m$, so $P^{(i=0)} \ggg P^{(i=max)}$. The experiments must confirm what is shown in Fig. 5, i.e. whether the CPM approach is able to find a maximum for (4). In Appendix A we analyse that $P^{(i)}$ cannot

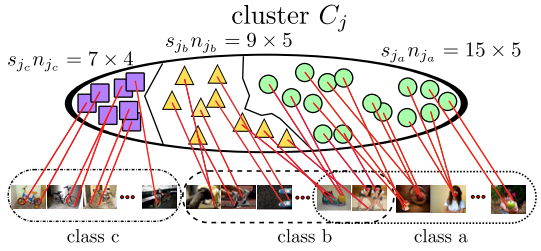

Fig. 4. Cluster C_j contains features from three different classes $\{a, b, c\}$. Eq. (3) takes into account the number of objects instances represented by the cluster C_j, i.e. the number of different images from which the local features are extracted.

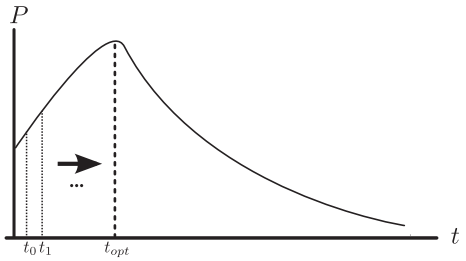

Fig. 5. $P(t)$ evolution in a CPM execution.

stabilise if any clusters are merged, i.e. the precision does not plateau at a maximum.

Algorithm 1 describes both the RNN clustering algorithm and its integration in a Cluster Precision Maximisation (CPM) approach. For the RNN clustering algorithm, we have chosen the efficient implementation described in [8], which has $O(N^2 d)$ time and $O(N)$ space complexity.

Algorithm 1. RNN clustering algorithm integrated in a CPM approach

```
P_max = 0
for thres = min to max do
  C = ∅; last ← 0; lastsim[0] ← 0; C contains a list of clusters
  L[last] ← v ∈ V; R ← V\v; Start chain L with a random vector v
  while R ≠ ∅ do
    (s,sim) ← getNearestNeighbor(L[last],R);
    if sim > lastsim[last] then
      last ← last + 1; L[last] ← s; R ← R\{s}
    else
      if lastsim[last] > thres then
        s ← agglomerate(L[last],L[last − 1]); R ← R ∪ {s};
  last ← last − 2
      else
        C ← C ∪ L; last ← −1; L = ∅;
      end if
    end if
    if last < 0 then
      last ← last + 1; L[last] ← v ∈ R; R ← R\s
    end if
  end while
  P ← getP(C); //Evaluate P
  if P > P_max then
    P_max ← P; C_optimum ← C;
  end if
end for
```

4. Correlation clustering based refinement step

In this section, an adaptive threshold refinement step for compacting the codebook while further increasing the average recognition rate in categorisation is described.

Just like the K-means algorithm, the agglomerative clustering algorithms in general, and the RNN clustering algorithm in particular, have several known deficiencies. The RNN algorithm overcomes those related to both the time and space complexity which are often significantly higher for agglomerative methods. It is clear that it is not necessary to fix the number of clusters beforehand, but a cut-off threshold must be provided instead, which divides the high dimensional vector space into clusters with a compactness that is always below the threshold. Because this threshold is the same for all clusters, it may occur that some real clusters are still split into several clusters.

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009

Moreover, in our experiments we found that after an RNN clustering algorithm execution, e.g. with the threshold obtained by the CPM approach, there is often lots of singletons, i.e. clusters with only one vector. This can be explained by the fact that during the clustering process chains of NN are sometimes discarded. This increases the size of the codebook and has a negative impact on the computation time during the on-line classification.

One possible solution is to consider these points as outliers, so they are not used to define any cluster centre. They can then simply be assigned to the nearest cluster centre after the clustering is finished. But following this approach we neither guarantee the resulting clusters are class representative, nor increase the discriminativity of the codebook.

Here, we explore whether it is not possible to follow a different policy for decreasing the number of clusters while pursuing this threefold objective: (1) to integrate the smaller clusters in bigger clusters to obtain more representative visual words, (2) to obtain compact codebooks and (3) to increase the recognition rate of the overall system. In this section we describe a new correlation clustering based refinement step so as to obtain an adaptive threshold scheme where the clusters are merged in accordance with the qualitative information we have.

The idea is simple. It is illustrated in Fig. 6. Once the clustering has been done, e.g. following the CPM approach with an RNN clustering algorithm, it is possible to construct a complete graph with K vertices, one per cluster, where each edge (C_u, C_v) is labelled either 0 or 1 depending on whether the clusters C_u and C_v have been deemed to be similar or different, respectively.

In our approach, how similar two clusters are depends on: the distance between the cluster centres, their sizes, and on the class labels of the features they bring together. One of the objectives is to reduce the number of small clusters, but not to merge clusters that have nothing in common. That is why the class labels are used. The label of each feature in a cluster is known, i.e. which class it belongs to. Only those clusters with classes in common may be merged by this refinement step.

Once such a graph is built the goal is to find a partition of the vertices of the graph that minimises the sum of 0 weight edges that are cut minus the 1 weight edges that are uncut. This is achieved by the correlation clustering algorithm proposed by Bansal et al. [11]. An example of correlation clustering on a graph with 5 vertices is shown in Fig. 7.

4.1. Clustering refinement algorithm

The clustering refinement algorithm first builds a complete graph G_0. Let $C = \{C_1, C_2, \ldots, C_k\}$ be the complete set of clusters. S is the set of *small* clusters in C, and the rest of clusters belong to $\overline{S}$, so $C = S \cup \overline{S}$. We consider a cluster *small* if the number of vectors it contains is below a fixed threshold. All the edges in G_0 joining two clusters in $\overline{S}$ are labelled with 1. Furthermore, all the edges linking clusters that have no features with the same class labels are labelled with 1 (recall Fig. 6). The remaining edges of G_0 are labelled with the distance between centroids. For the experiments we report results using the Euclidean distance between centroids.

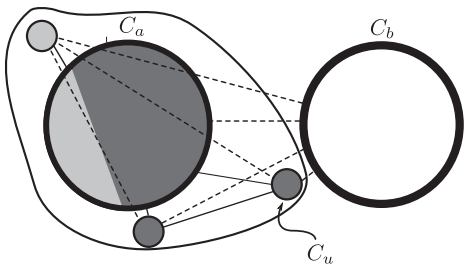

Fig. 6. Correlation clustering based refinement. Solid edges indicate weight 0, and dashed edges indicate weight 1. Observe singleton C_u, which is clustered with the one of the same class C_a, even though it is nearer to C_b but they do not share features with the same class label.

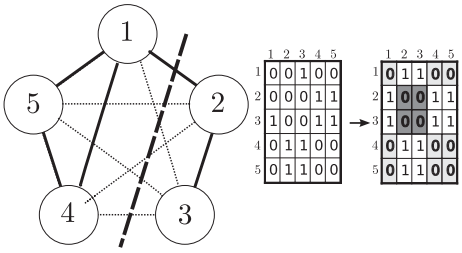

Fig. 7. Example of correlation clustering. Solid edges indicate weight 0, and dashed edges indicate weight 1. Observe the algorithm cuts edge (C_1, C_2) with weight 0, while it does not fail to cut edges with weight 1. The result is depicted on the right. Two clusters, one containing $\{C_2, C_3\}$ and the other $\{C_1, C_4, C_5\}$.

Now the adaptive threshold merging of clusters begins. In each iteration the algorithm increases the threshold t_i. Let t_{min} and t_{max} be the minimum and maximum threshold respectively, and t_i the threshold to be used in the iteration i. At the beginning of each iteration $G_i = G_0$. Edges in G_i with distance over t_i are automatically labelled with 1. Suppose N is the number of nodes in G_i that might be merged, i.e. vertices whose common edges have a distance $d \leqslant t_i$. These edges are labelled with 0 in G_i, which becomes a binary graph. Then the correlation clustering approach is run over the graph, and a new codebook for each iteration is obtained. Until t_{max} is reached the procedure is repeated. Since we can no longer rely on the cluster precision at this stage (giving rise to too many singletons), we use cross-validation in this final step, i.e. for every new codebook a classifier is trained and the average class recognition rate is measured. At the end of this refinement process the codebook allowing the best recognition results is selected.

This refinement procedure can be run iteratively. For instance, we first run the refinement for reducing the singletons. With the resulting codebook, we run again the correlation clustering approach for reducing clusters up to size 2, and so on.

As described earlier, in each iteration a correlation clustering algorithm is run. This correlation clustering works as a meta-clustering refinement merging clusters, reducing the codebook size and increasing the recognition rate. It is possible to apply any correlation clustering algorithm described in [11]. However, we propose a correlation clustering following the formulation detailed in [28]. This algorithm takes a complete graph G_i as input. The algorithm goes through the nodes in G_i, i.e. the centroids, and it considers merging them with a different centroid or allowing them to continue being alone. A node is merged with the cluster that yields the minimum cost. The process iterates until there is no move that can improve the cost or a maximum number of iterations has been exceeded. The cost function used in this approach is similar to the one proposed in [28] for correlation clustering. Given a complete graph G_i, each edge (C_u, C_v) has weight $W_{uv} \in \{0,1\}$. The cost $c(C_u, C_i')$ of assigning node C_u to a new cluster C_i' is computed as follows

$$c(C_u, C_i') = \sum_{C_v \in C_i'} W_{uv} + \sum_{C_v \in \overline{C_i'}} (1 - W_{uv}). \quad (7)$$

The first term is the cost of merging C_u in C_i', while the second is the cost of not merging node C_u with nodes not in C_i'. The experiments show that the algorithm is quite effective, improving the recognition rate from a clustering partition obtained with an RNN algorithm while the size of the codebook is dramatically reduced.

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009

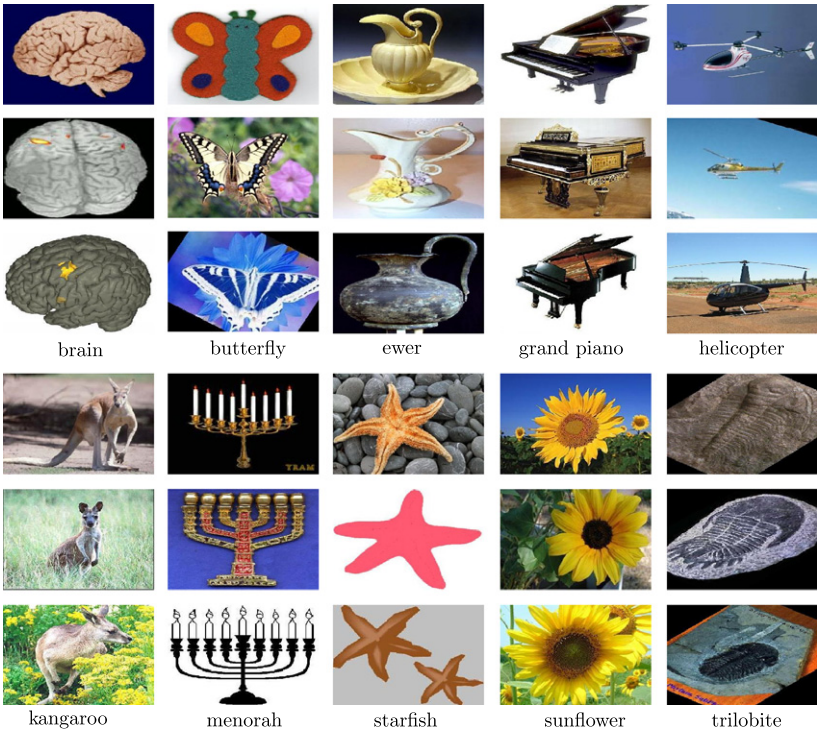

Fig. 8. Images examples from the subset of the Caltech 101 database used in the experiments. Note that some images have a partially black background due to artificial image rotations (e.g. corners of the third butterfly image).

5. Experiments

In the experiments we show how the CPM succeeds in finding the threshold which obtains a maximum for the cluster precision. It is shown that the average classification rate drastically drops with codebooks obtained with thresholds over this threshold; furthermore the best results in classification are obtained by codebooks with high values of cluster precision (CP). We also show how the correlation clustering refinement algorithm improves the average recognition rate while reducing the size of the codebooks.

5.1. Experimental setup

We use a subset of the Caltech 101 database [30]. This dataset contains 101 object categories with 40–800 images per class. Most of the images in the database have little or no clutter. Furthermore, the objects tend to lie in the centre of the images and appear in similar poses. Some images have a partially black background due to artificial image rotations. For our experiments we have randomly chosen the following 10 categories: ewer, sunflower, kangaroo, starfish, trilobite, menorah, helicopter, butterfly, brain and grand piano. In the experiments we randomly select 50% of the images for training the system. Some of the images are shown in Fig. 8. More details are summarised in Table 1.

We show results using a traditional BoW approach, which is based on regions of interest and not on dense sampling, although the latter has been shown to outperform interest point detector based methods in image classification [19]. The objective of this paper is to demonstrate that the CPM procedure is able to identify the thresholds for a RNN agglomerative clustering that cast the best results in recognition of object classes.

There are many different techniques for detecting and describing local image regions [31,32]. Here we use the Hessian-Laplace detector [32] and SIFT [3] as descriptor.[1]

For classification we use SVM with the kernels most widely used within the context of category-level object recognition in images: Histogram Intersection Kernel [33] (HIK) and the extended Gaussian kernel with χ^2 distance [5] (i.e. the χ^2-Kernel). We also present, for comparison, results obtained with general radial basis functions using Euclidean distances between the histograms.

Specifically, we use libSVM [34] and the built in one-versus-one approach for multi-class classification. A 10-fold cross-validation on the train set to tune SVM parameters is conducted. When the RBF kernel is used two parameters need to be tuned: C and σ.

The HIK applied to two feature vectors (histograms of visual words) H_x and H_y of dimension D is defined as

$$k(H_x, H_y) = \sum_{i=1}^{D} \min(H_x(i), H_y(i)). \tag{8}$$

For this particular case, only the C parameter needs to be tuned. When the χ^2-Kernel is used, we first compute the χ^2 distance between feature vectors H_x and H_y as

$$d_{\chi^2}(H_x, H_y) = \frac{1}{2} \sum_{i=1}^{D} \frac{(H_x(i) - H_y(i))^2}{H_x(i) + H_y(i)}. \tag{9}$$

The kernel function based on the χ^2 distance is then defined as

$$k(H_x, H_y) = e^{-\frac{1}{\sigma} d_{\chi^2}(H_x, H_y)}, \tag{10}$$

where σ is a scalar which normalises the distances. It is possible to fix this σ parameter to the mean of all distances, or to tune it through the cross-validation approach. In our experiments the latter has not shown better results, so to reduce training time we opted for the former.

5.2. CPM performance with RNN clustering

In these initial experiments the first objective is to analyse the performance of the CPM approach. Fig. 9a confirms what was predicted in Section 3.2: the CP reaches a maximum for a specific

[1] The binaries have been taken from http://www.robots.ox.ac.uk/~vgg/research/affine/.

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009

Table 1
Outline of the number of images and features (for training and testing) obtained from the subset of the Caltech 101 database.

Dataset	#Images (training|test)	#Features (training|test)
Caltech 101 subset	446|444	55,679|56,590
Total	890	112,269

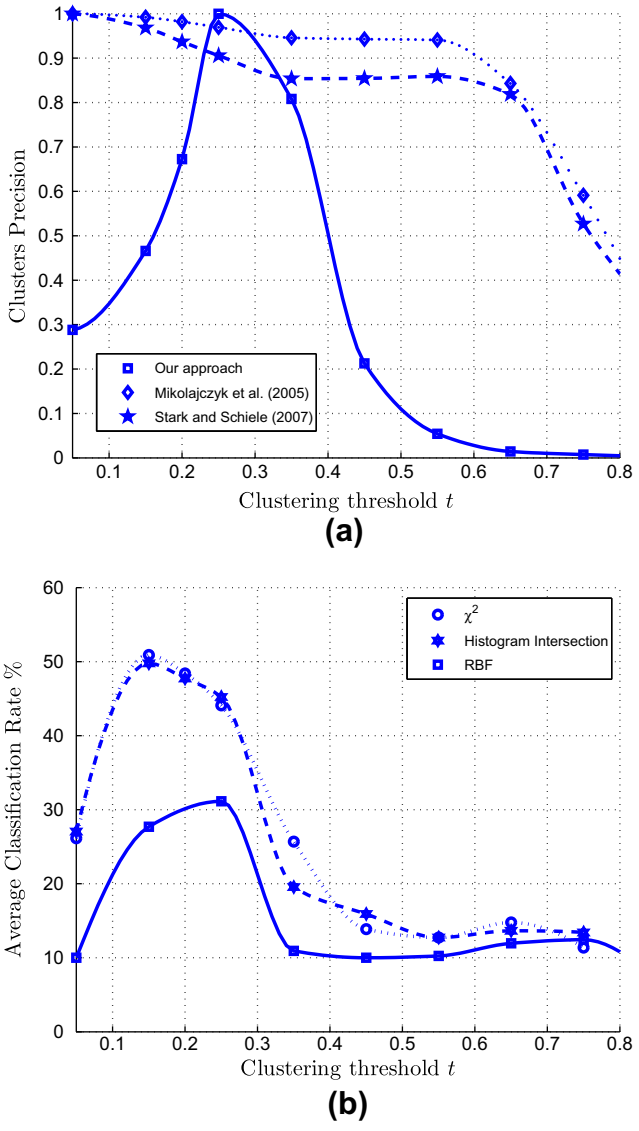

Fig. 9. (a) CPM performance. The threshold resulting in the maximum CP is $t_{opt} = 0.25$ for the Caltech 101 subset. A comparison with the cluster precision measures of Mikolajczyk et al. [23] and Stark et al. [24] is shown too. (b) Average classification rate recognition versus Clustering threshold. For the RBF kernel, the codebook with the highest CP obtains the highest classification rate. For the HIK and the χ^2-Kernel the maximum of the CP does not coincide with the maximum of the classification rate. Nonetheless, the HIK and the χ^2-Kernel obtain the best results for thresholds near t_{opt}. Furthermore, it can also be seen that for thresholds over t_{opt} the classification rate drastically drops.

threshold ($t_{opt} = 0.25$) for the subset of Caltech 101 database, and then it quickly drops to suboptimal values. The experiments confirm that there are no local maxima either. Moreover, Fig. 9a shows a comparison of our cluster precision formulation in Eq. (3) with the formulations in Eqs. (1) and (2), proposed by Mikolajczyk et al. [23] and Stark et al. [24] respectively. While our cluster precision obtains a maximum for a specific threshold, the other formulations are monotonically decreasing functions. Both precision measures (1) and (2) start with a maximum (when all the clusters are singletons), and drop until all the features are assigned to only one cluster.

The experiments also confirm that the CPM is able to identify for a RNN clustering the thresholds that allow the best recognition rates. In each iteration of the CPM, we fix a threshold and obtain a visual vocabulary using the RNN clustering algorithm. With this codebook, we train and test the classifier using the kernels described in Section 5.1. The average recognition rate that the classifier obtains on the test images is shown in Fig. 9b. For the RBF kernel, the codebook with the highest CP obtains the highest classification rate. For the HIK and the χ^2-Kernel the maximum of the CP does not coincide with the maximum of the classification rate. Nonetheless, the HIK and the χ^2-Kernel obtain the best results for thresholds near t_{opt}. Furthermore, it can also be seen that for thresholds over t_{opt} the classification rate drastically drops. That is, the codebooks obtained with thresholds over the upper bound t_{opt}, defined by the maximum of CP, result in poor recognition rates for the three different types of kernels. As we are able to avoid expensive cross-validation through the optimisation of vocabulary, the CPM method significantly reduces the overall training time without a negative impact on the results. Finally, Fig. 9 shows that among the three precision measures, ours is the only one that allows to predict how the classifier is going to perform.

5.3. Correlation clustering based refinement

The results obtained with a traditional BoW approach, or building the codebook with an RNN clustering algorithm and the CPM methodology, can be improved further by the meta-clustering algorithm described in Section 4.

We first run the correlation clustering based refinement for reducing the number of singletons. The algorithm is initialised with the codebook obtained with an RNN with threshold 0.25. The threshold is increased from 0.25, in steps of 0.05, until a maximum of 0.9 is reached. After each iteration the average recognition rate is measured. For this experiment we use the χ^2-Kernel which reported the best results in [4,5]. Testing with the Caltech 101 subset of images, this refinement step finds that the best average recognition rate is obtained when the cut-off threshold varies from 0.25 to 0.9. This means that some clusters have been merged by the correlation clustering algorithm, and that the maximum threshold for some of them is now 0.9. The refined codebook is now made up of clusters with different cut-off thresholds, that is, these thresholds have been separately optimised for each cluster. The average classification rate after this first iteration has been increased from 43.4% to 52.5%, while the codebook size has been drastically reduced from 17,575 to 6614, i.e. by 63%. This huge reduction is due to the high number of initial singletons the codebook had. Table 2 presents the confusion matrix for the classifier trained with the codebook before and after this first adaptive threshold refinement. It can be observed how the misclassifications decrease. So the refinement step obtains more discriminative codebooks reducing their size, both of which are desirable aspects.

After this first refinement process the resulting codebook has 6614 clusters. It consist of 8 singletons, 1571 clusters of size 2, 1223 clusters of size 3, and the rest of clusters. We again run the correlation clustering based refinement process, but this time with the aim of merging the clusters of size up to 3. The algorithm is initialised with the codebook obtained in the previous iteration. The threshold is increased from 0.5, in steps of 0.1, until a maximum of 0.9 is reached. The best average recognition rate is obtained in the third iteration, when the cut-off threshold varies from the minimum until 0.7. The resulting codebook has 3818 clusters, and it allows an average recognition rate of 58.33%. That is, the average

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009

Table 2
Confusion matrix before the adaptive clustering refinement process (a) and after (b). The entry in the ith row and jth column is the number of images from class i that were classified as class j.

(a)

	Ewer	Sunflower	Kangaroo	Starfish	Trilobite	Menorah	Helicopter	Butterfly	Brain	Gran piano
Ewer	**5**	3	1	4	0	0	1	2	1	1
Sunflower	4	**12**	0	2	2	0	0	1	2	2
Kangaroo	4	4	**16**	10	2	1	3	6	0	0
Starfish	0	0	0	**11**	1	1	0	1	0	0
Trilobite	5	3	2	3	**24**	2	1	8	3	0
Menorah	9	2	7	4	5	**31**	6	4	4	11
Helicopter	8	8	11	4	6	4	**28**	11	3	1
Butterfly	3	7	2	1	0	0	2	**4**	0	3
Brain	4	3	4	2	1	2	3	8	**36**	0
Gran piano	0	0	0	1	0	2	0	0	0	**31**

(b)

	Ewer	Sunflower	Kangaroo	Starfish	Trilobite	Menorah	Helicopter	Butterfly	Brain	Gran piano
Ewer	**9**	2	6	1	0	2	3	2	0	1
Sunflower	5	**22**	2	2	2	0	0	5	4	2
Kangaroo	3	4	**22**	9	4	3	7	8	3	0
Starfish	4	3	1	**18**	3	1	1	1	1	0
Trilobite	2	2	3	4	**24**	0	0	6	1	0
Menorah	5	0	0	3	3	**28**	5	2	1	5
Helicopter	3	1	2	0	0	3	**16**	2	1	0
Butterfly	10	8	7	3	2	3	7	**17**	1	0
Brain	1	0	0	1	2	1	3	2	**37**	0
Gran piano	0	0	0	1	1	2	2	0	0	**41**

classification rate has been increased from 52.5% to 58.33%, while the codebook size has been reduced from 6614 to 3818.

Fig. 10 shows the class recognition rates before and after these two iterations of the correlation clustering based refinement. It is important to note that not all the class rates vary in the same way. In the experiments, in 80% of the classes the recognition rate is increased.

5.4. A comparison with K-means codebooks

So far we have shown how the CPM algorithm and the correlation clustering based refinement perform with RNN codebooks. However, the most common clustering algorithm used in BoW approaches is *K*-means. So, an experimental comparison with vocabularies obtained with *K*-means clustering has also been carried out. Our objective it to evaluate: how the CPM performs in combination with each clustering algorithm; what recognition performance the codebook allows; and the computational complexity.

5.4.1. CPM performance

Using the same Caltech 101 subset described above, *K*-means codebooks of similar sizes to those obtained with the RNN clustering have been obtained. For each vocabulary we first measure its cluster precision. Fig. 11 shows a comparison of the CP obtained for different *K*-means and RNN codebooks. Note the CP is similar at the beginning when the number of clusters is near the number of features. Once the number of clusters decreases, the CP for *K*-means codebooks decreases too. Note that there is no maximum for the CP when *K*-means codebooks are used, the CP monotonically decreases. This is due to the fact that the RNN produces visually compact clusters, with a high proportion of singletons, which makes the CP increase when they merge during the first iterations. With *K*-means there is neither any guarantee that the clusters are visually compact nor such a large number of singletons. Because of the fixed value of *K*, some cluster centres may lie in-between several *real* clusters. That is, *K*-means has the defect that, in high-dimensional spaces, it tends to focus on the high-density area, resulting in clusters that extend quite far into the less dense areas, so not compact at all. However, with an agglomerative clustering algorithm, such as RNN, where the clusters aggregation process is heavily dominated by the similarity of the vectors they contain, the CPM is able to identify the threshold limit above which the average classification rate drastically decreases. This capacity is important, especially in high-dimensional spaces where the distances (similarities) tend to concentrate.

5.4.2. Classification performance

We compare the average classification obtained with *K*-means, RNN and refined RNN codebooks. The most important results in classification are shown in Fig. 12. We have used the HIK and the χ^2-Kernel for this comparison. It is clear that *K*-means codebooks perform better than RNN codebooks, but after the correlation clustering based refinement, the refined RNN codebooks obtain better results than *K*-means codebooks for both types of kernels.

5.4.3. Computational complexity

Consider a classical BoW approach using *K*-means as clustering algorithm. To find the number of K clusters with which the codebook allows the best classification rate implies empirically determining the parameters of the whole system. We define the run-time of an iteration i for a classical BoW as $t^{(i)}_{BoW} = t^{(i)}_{K-means} + t^{(i)}_{train} + t^{(i)}_{val}$, where $t^{(i)}_{K-means}$ is the run-time of *K*-means clustering, and $t^{(i)}_{train}$ and $t^{(i)}_{val}$ are respectively the time spent training the classifier and testing with a validation set. Let M be the number of iterations until the desired classification rate is achieved, we define $t_{BoW} = \sum_{i=1}^{M} t^{(i)}_{BoW}$.

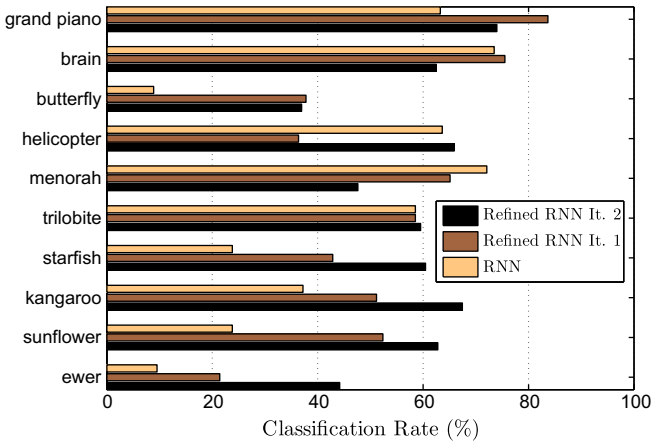

Fig. 10. The Figure shows how the correlation clustering refinement iterations get to increase the average recognition rate. We show the class rate allowed by the original RNN vocabulary and the refined RNN codebooks after the first and second iteration.

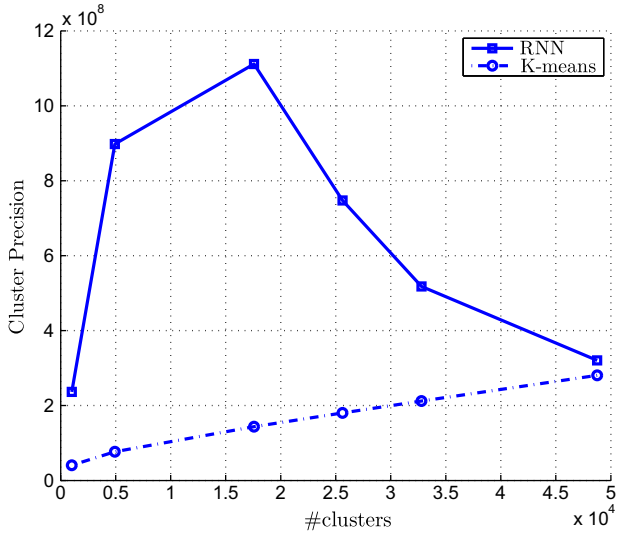

Fig. 11. Comparison of the CP obtained with RNN and *K*-means codebooks. Note that there is no maximum for the CP when using *K*-means codebooks.

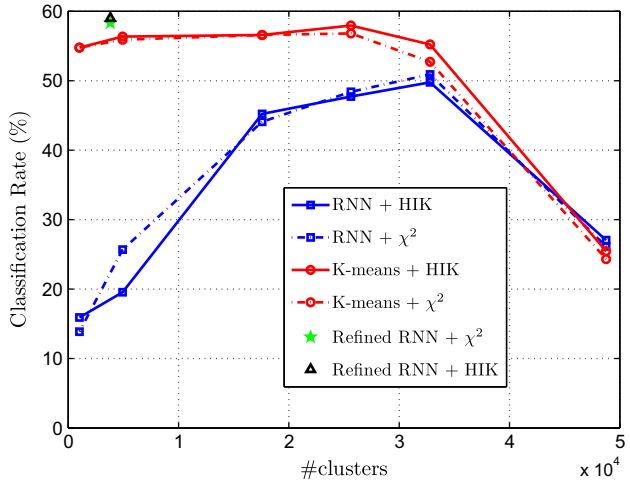

Fig. 12. Average classification rate for different clustering algorithms (*K*-means, RNN refined RNN), different vocabulary sizes and different kernels.

On the other hand we have the CPM with RNN clustering scenario. A CPM approach requires the clustering algorithm to be run with varying parameters, e.g. the threshold for the RNN clustering algorithm. It is possible to perform a full RNN clustering (i.e. with the maximum threshold) so as to save both the indices of clusters merged in every step and the similarities between them. With this information it is possible to rebuild a visual vocabulary for a different threshold at almost no computational cost. The run-time for a CPM approach with RNN can then be written as $t_{CPM+RNN} = t_{RNN} + Mt_{CP} + (M-1)\, t_{RNNnew}$, where t_{CP} is the time used to measure the cluster precision in each CPM iteration, M is the number of iterations, t_{RNN} is the run-time of a full RNN clustering algorithm, and t_{RNNnew} is the time taken by the algorithm to compute the new clusters when the threshold changes using the saved indices and similarities. Furthermore, $t_{RNN} \ggg (Mt_{CP} + (M-1)t_{RNNnew})$, so $t_{CPM+RNN} \approx t_{RNN}$.

The RNN clustering algorithm has $O(N^2 d)$ time complexity, which is high but almost independent of the number of clusters. The complexity of *K*-means is $O(NKdl)$. Recall l is the number of iterations until the algorithm converges. Within the context of category-level object recognition, normally the number of clusters K is high (sometimes it is in almost the same order as the number of features N, e.g. [8]). This implies that the run-time for *K*-means might exceed the one for RNN when K is large, e.g. [35]. In that case, $t_{BoW} > t_{CPM+RNN}$. However, it is possible to use efficient implementations of *K*-means, such as [36,37], in order to reduce t_{BoW}. In our experiments, we have run the *K*-means algorithm with the same number of clusters obtained by the RNN algorithm during the CPM. Specifically, we have used the efficient *K*-means implementation described in [36], setting the maximum number of stages of the *K*-means algorithm to 10. For the first iteration of both algorithms (when K is in almost the same order as the number of features, see Fig. 11), $t_{RNN} < t_{K-means}$. This implies that, $t_{BoW} > t_{CPM+RNN}$.

In terms of run-time, the RNN+CPM procedure is more efficient than a traditional *K*-means based BoW approach. However, the RNN vocabularies do not obtain better classification results than *K*-means codebooks until the correlation clustering based refinement is processed. That is, we have to add to the time $t_{CPM+RNN}$, the time spent in the refinement, t_R. The run-time complexity of the correlation clustering approach used for clustering refinement is $O(K^2 I)$, where K is the number of clusters and I is the number of iterations the algorithm needs to obtain the final solution. In our experiments, we have used $I = 10^7$, and we have confirmed that $t_R + t_{CPM+RNN} < t_{BoW}$.

6. Conclusions and future work

In this paper we consider the problem of measuring and increasing the representativeness of visual words within the context of category-level object recognition. First, we have introduced an improved measure for cluster precision, as well as a procedure to optimise the parameters during codebook construction without cross-validation, by maximising this cluster precision measure.

The CPM method measures the cluster precision for each class and automatically finds the thresholds for an RNN clustering algorithm that cast the best average recognition rates. A complete description of the method has been given,[2] showing how to integrate an RNN clustering algorithm in it. CPM evaluates the intrinsic quality of the clusters for a classification task and as a result, allows us to compare the quality of clusters computed with different thresholds. Results confirm that the CPM is able to identify a

[2] The CPM method and the clustering algorithm are available at http://agamenon.tsc.uah.es/Personales/rlopez/data/cp/.

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009

maximum for the clustering precision, as well as to determine an upper bound for the codebook size that guarantees the highest categorisation rates are obtained.

On the other hand, a meta-clustering refinement algorithm has been presented. To the best of our knowledge, this is the first correlation clustering based approach for improving the class representativeness of visual words. The proposed methodology increases the average recognition rate and also dramatically compacts the size of the codebook.

When large amounts of data are given a more efficient implementation of the RNN clustering algorithm is highly desirable. So as future work we plan to optimise the RNN clustering algorithm. Experimenting with other descriptors, detectors and datasets (e.g. PASCAL VOC Challenge datasets) will also be considered. Another line of research involves bringing in local information on the clustering process to discover more semantic and more class representative visual words.

Acknowledgments

We acknowledge support from the Spanish Projects MEC TEC2008-02077, CAM CCG07-UAH/TIC-1740 and TIN2010-20845-C03-03. We would like to thank Rafael Sendra for helpful discussions.

Appendix A. Evolution of cluster precision

Fig. 5 depicts the evolution of $P^{(i)}$ through iterations in the CPM approach. In this appendix we analytically analyse that P cannot stabilise, i.e. it does not plateau at a maximum, if any clusters are merged.

For each iteration i, let $\left\{P_m^{(i)}\right\}$ be the finite sequence of real numbers defined as

$$P_m^{(i)} = \frac{K^{(i)}}{M} \sum_{j_m=1}^{K^{(i)}} s_{j_m}^{(i)} n_{j_m}^{(i)}, \tag{A.1}$$

where $K^{(i)}$, M, $s_{j_m}^{(i)}$ y $n_{j_m}^{(i)} \in \mathbb{N}$.

Let $i+1$ be the iteration, where we consider that at least two clusters, say C_u and C_v, have been merged. This implies that the number of clusters, namely $K^{(i+1)}$, has decreased by one; i.e. $K^{(i+1)} = K^{(i)} - 1$. Therefore, one obtains

$$P_m^{(i+1)} = \frac{\left(K^{(i)} - 1\right)}{M} \sum_{j_m=1}^{K^{(i)}-1} s_{j_m}^{(i+1)} n_{j_m}^{(i+1)}. \tag{A.2}$$

It is possible to develop Eq. (A.2) as follows

$$P_m^{(i+1)} = \frac{\left(K^{(i)} - 1\right)}{M} \left\{ \left(\sum_{j_m=1}^{K^{(i)}-2} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) + \left(s_{u_m}^{(i)} + s_{v_m}^{(i)}\right) n_{u \cup v_m}^{(i+1)} \right\}, \tag{A.3}$$

where $n_{u\cup v_m}^{(i+1)}$ is the new number of class m object instances represented in the new cluster $C_{u\cup v}$. $n_{u\cup v_m}^{(i+1)}$ will depend on whether the clusters C_u and C_v have features which belong to the same object instances. So to analyse Eq. (A.3), we must consider two possible situations when merging clusters C_u and C_v: the clusters contain features which come from the same object instances or from different object instances.

A.1. The same object instances in C_u and C_v

Let us assume that no new instance is added to cluster C_u when merging with C_v, and vice versa. Then $n_{u\cup v_m}^{(i+1)} = n_{u_m}^{(i)} = n_{v_m}^{(i)}$, therefore

$$\begin{aligned} P_m^{(i+1)} &= \frac{\left(K^{(i)}-1\right)}{M} \left\{ \left(\sum_{j_m=1}^{K^{(i)}-2} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) + \left(s_{u_m}^{(i)} + s_{v_m}^{(i)}\right) n_{u_m}^{(i)} \right\} \\ &= \frac{\left(K^{(i)}-1\right)}{M} \left(\sum_{j_m=1}^{K^{(i)}} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) = P_m^{(i)} \left(1 - \tfrac{1}{K^{(i)}}\right). \end{aligned} \tag{A.4}$$

Due to $K^{(i)} > 1$, then $P_m^{(i+1)} < P_m^{(i)}$. So, if no new view is added when merging the clusters, the cluster precision for each class decreases.

A.2. Different object instances in C_u and C_v

Let $\delta_{u_m}^{(i+1)}$ and $\delta_{v_m}^{(i+1)}$ be the number of new class m object instances added to cluster C_u when merging with cluster C_v and vice versa, respectively. Then,

$$n_{u\cup v_m}^{(i+1)} = n_{u_m}^{(i)} + \delta_{u_m}^{(i+1)} = n_{v_m}^{(i)} + \delta_{v_m}^{(i+1)}. \tag{A.5}$$

Once this merging is done the cluster precision for class m can be expressed as follows

$$P_m^{(i+1)} = \frac{\left(K^{(i)} - 1\right)}{M} \left\{ \left(\sum_{j_m=1}^{K^{(i)}-2} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) + \left(s_{u_m}^{(i)} + s_{v_m}^{(i)}\right) \left(n_{u_m}^{(i)} + \delta_{u_m}^{(i+1)}\right) \right\}, \tag{A.6}$$

and by algebraic manipulation

$$P_m^{(i+1)} = P_m^{(i)} - \frac{P_m^{(i)}}{K^{(i)}} + \frac{\left(K^{(i)} - 1\right)}{M} \left(s_{u_m}^{(i)} \delta_{u_m}^{(i+1)} + s_{v_m}^{(i)} \delta_{v_m}^{(i+1)} \right). \tag{A.7}$$

It is straightforward to note that $P_m^{(i+1)} \gtrless P_m^{(i)}$ when

$$-\frac{P_m^{(i)}}{K^{(i)}} + \frac{\left(K^{(i)} - 1\right)}{M} \left(s_{u_m}^{(i)} \delta_{u_m}^{(i+1)} + s_{v_m}^{(i)} \delta_{v_m}^{(i+1)} \right) \gtrless 0, \tag{A.8}$$

which implies

$$\left(s_{v_m}^{(i)} \delta_{v_m}^{(i+1)} + s_{u_m}^{(i)} \delta_{u_m}^{(i+1)} \right) > \frac{M P_m^{(i)}}{K^{(i)}(K^{(i)} - 1)} \rightarrow P_m^{(i+1)} > P_m^{(i)}, \tag{A.9}$$

$$\left(s_{v_m}^{(i)} \delta_{v_m}^{(i+1)} + s_{u_m}^{(i)} \delta_{u_m}^{(i+1)} \right) < \frac{M P_m^{(i)}}{K^{(i)}(K^{(i)} - 1)} \rightarrow P_m^{(i+1)} < P_m^{(i)}. \tag{A.10}$$

So when there are different object instances in C_u and C_v, the cluster precision for each class can either increase or decrease, depending on Eqs. (A.9) and (A.10).

Iteration by iteration, while the threshold increases and the RNN clustering is executed, K decreases but clusters grow in terms of size. When clusters are big, the likelihood of having more different object instances represented in a cluster is greater. This implies that while the threshold increases cluster precision tends to decrease as situation described in Appendix A.1 is more probable than the situation in Appendix A.2.

On the other hand, during the first iterations, when the threshold t is small and the number of clusters K is high, situation in Appendix A.2 is more probable. When the threshold increases, s_{u_m} and s_{v_m} increase for sure. Moreover, δ_{u_m} and δ_{v_m} increase during the first iterations until a threshold after which no new object instances are added when clusters are merged. In the beginning we can also consider $K \sim S$. All this concludes in that condition (A.9) is more probable during the first iterations and condition (A.10) for the last ones.

Taken together, these considerations show what happens to P_m when clusters merge. As shown in Fig. 5 P reaches a maximum. Where this maximum is depends on given data. Nonetheless, it is possible to prove that P cannot stabilise, i.e. it does not plateau at a maximum, if any clusters are merged.

For any $\Delta t > 0$, however small, $t_{i+1} = t_i + \Delta t$. If any clusters are merged then $P^{(i+1)} \neq P^{(i)}$. Let us consider situation in Appendix

A.1, where no new views are added when clusters are merged. As proved in Eq. (A.4), $P_m^{(i+1)} < P_m^{(i)}$, so $P^{(i+1)} < P^{(i)}$.

Suppose now that new views are added when clusters merge, i.e. situation described in Appendix A.2. Remember that $K, s_{u_m}, s_{v_m}, \delta_{u_m}, \delta_{v_m} \in \mathbb{N}$ and $P_m^{(i)} \in \mathbb{Q}$. Now, the following multivariate rational function is considered

$$f(x_1, \ldots, x_6) = -\frac{x_1}{x_2} + \frac{(x_2 - 1)}{M}(x_3 x_5 + x_4 x_6), \tag{A.11}$$

Then, equality (A.7) can be expressed as

$$P_m^{(i+1)} = P_m^{(i)} + f\left(P_m^{(i)}, K^{(i)}, s_{u_m}^{(i)}, s_{v_m}^{(i)}, \delta_{u_m}^{(i+1)}, \delta_{v_m}^{(i+1)}\right), \tag{A.12}$$

note that $K^{(i)} \neq 0$, and hence the above specialisation of f is well-defined. f can be expressed as $f = g/(x_2)$, where $g = -x_1 + x_2(x_2 - 1)(x_3 x_5 + x_4 x_6)$. In this situation, we observe that the iterative sequence may stabilise, i.e. $P_m^{(i+1)} = P_m^{(i)}$ for some i, if $g(x_1, \ldots, x_6) = 0$ has zeros over $\mathbb{N}$. In our case, since $P_m^{(i)} \in \mathbb{Q}$ and $K^{(i)}, s_{u_m}^{(i)}, \delta_{u_m}^{(i+1)}, s_{v_m}^{(i)}, \delta_{v_m} \in \mathbb{N}$, this phenomenon might happen. Nevertheless, because of the nature of the experiment and data, in practise this situation does not occur. In order to formally guarantee that the sequence never stabilises, one can proceed as follows. It is possible to introduce a small perturbation ϵ in the sequence so as to guarantee that P_m always varies when clusters merge. More precisely, instead of working with $P_m^{(i)}$, we take

$$\widetilde{P}_m^{(i)} = \frac{K^{(i)}}{M}\left(\sum_{j_m=1}^{K^{(i)}} s_{j_m}^{(i)} n_{j_m}^{(i)} + \epsilon\right), \tag{A.13}$$

where ϵ is taken as a real non-rational number; e.g. one might take $\epsilon = \sqrt{K^{(i)}/(K^{(i)} + 1)}$ or a similar expression in case it turns to be a rational number. Now, the important fact is that $\widetilde{P}_m^{(i)} \notin \mathbb{Q}$. Under these new conditions one can prove that $\{\widetilde{P}_m^{(i)}\}$ never stabilises. Indeed: if there exists i such that $\widetilde{P}_m^{(i)} = \widetilde{P}_m^{(i+1)}$, one has that $g\left(\widetilde{P}_m^{(i)}, K^{(i)}, s_{u_m}^{(i)}, s_{v_m}^{(i)}, \delta_{u_m}, \delta_{v_m}\right) = 0$. This implies that $\widetilde{P}_m^{(i)}$ can be expressed rationally in terms of $K^{(i)}, s_{u_m}^{(i)}, s_{v_m}^{(i)}, \delta_{u_m}, \delta_{v_m}$:

$$\widetilde{P}_m^{(i)} = \frac{K^{(i)}(K^{(i)} - 1)}{M}\left(s_{u_m}^{(i)} \delta_{u_m} + s_{v_m}^{(i)} \delta_{v_m}\right). \tag{A.14}$$

So, $\widetilde{P}_m^{(i)} \in \mathbb{Q}$ which is a contradiction. So to conclude, it has been proved that P_m will always vary if at least one pair of clusters is merged when the threshold changes, i.e. P_m does not stabilise.

References

[1] J. Sivic, A. Zisserman, Video google: a text retrieval approach to object matching in videos, in: ICCV, 2003, pp. 1470–1477.
[2] G. Csurka, C.R. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: ECCV, 2004.
[3] D. Lowe, Object recognition from local scale-invariant features, in: ICCV, 1999, pp. 1150–1157.
[4] K. van de Sande, T. Gevers, C. Snoek, Evaluation of color descriptors for object and scene recognition, in: CVPR, 2008, pp. 1–8.
[5] J. Zhang, M. Marszalek, S. Lazebnik, C. Schimd, Local features and kernels for classification of texture and object categories: a comprehensive study, International Journal of Computer Vision 73 (2) (2007) 213–238.
[6] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, A. Zisserman, The PASCAL visual object classes (VOC) challenge, International Journal of Computer Vision 88 (2) (2010) 303–338.
[7] K. Grauman, T. Darrell, The pyramid match kernel: discriminative classification with sets of image features, in: ICCV, 2005, pp. 1458–1465.
[8] B. Leibe, A. Leonardis, B. Schiele, Robust object detection with interleaved categorization and segmentation, International Journal of Computer Vision 77 (1–3) (2008) 259–289.
[9] J. Sivic, B.C. Russell, A.A. Efros, A. Zisserman, W.T. Freeman, Discovering object categories in image collections, in: ICCV, 2005, pp. 1–8.
[10] P. Quelhas, F. Monay, J.-M. Odobez, D. Gatica-Perez, T. Tuytelaars, A thousand words in a scene, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (9) (2007) 1575–1589.
[11] N. Bansal, A. Blum, S. Chawla, Correlation clustering, Machine Learning 56 (2004) 89–113.
[12] A. Gilbert, J. Illingworth, R. Bowden, Fast realistic multi-action recognition using mined dense spatio-temporal features, in: ICCV, 2009, pp. 925–931.
[13] J. Sivic, A. Zisserman, Video data mining using configurations of viewpoint invariant regions, in: CVPR, 2004, pp. 488–495.
[14] T. Quack, V. Ferrari, B. Leibe, L. Van Gool, Efficient mining of frequent and distinctive feature configurations, in: ICCV, 2007, pp. 1–8.
[15] J. Yuan, Y. Wu, M. Yang, Discovery of collocation patterns: from visual words to visual phrases, in: CVPR, 2007, pp. 1–8.
[16] J. Yuan, Y. Wu, Context-aware clustering, in: CVPR, 2008, pp. 1–8.
[17] S. Lazebnik, C. Schmid, J. Ponce, Semi-local affine parts for object recognition, in: BMVC, 2004, pp. 779–788.
[18] B. Leibe, A. Ettlin, B. Schiele, Learning semantic object parts for object categorization, Image and Vision Computing 26 (1) (2008) 15–26.
[19] P. Perronnin, C. Dance, G. Csurka, M. Bressan, Adapted vocabularies for generic visual categorization, in: ECCV, 2006, pp. 464–475.
[20] J. Winn, A. Criminisi, A. Minka, Object categorization by learned universal visual dictionary, in: ICCV, 2005, pp. 1800–1807.
[21] F. Moosmann, B. Triggs, F. Jurie, Fast discriminative visual codebooks using randomized clustering forests, in: Advances in Neural Information Processing Systems, 2006, pp. 985–992.
[22] F. Perronnin, C. Dance, Fisher kernels on visual vocabularies for image categorization, in: CVPR, 2007, pp. 1–8.
[23] K. Mikolajczyk, B. Leibe, B. Schiele, Local features for object class recognition, in: ICCV, 2005, pp. 1792–1799.
[24] M. Stark, B. Schiele, How good are local features for classes of geometric objects, in: ICCV, 2007, pp. 1–8.
[25] M. Charikar, V. Guruswami, A. Wirth, Clustering with qualitative information, in: IEEE Symposium on Foundations of Computer Science, 2003, pp. 524–533.
[26] E.D. Demaine, D. Emanuel, A. Fiat, N. Immorlica, Correlation clustering in general weighted graphs, Theoretical Computer Science 361 (2) (2006) 172–187.
[27] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, in: International Conference on Data Engineering, 2005, pp. 341–352.
[28] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, ACM Transactions on Knowledge Discovery from Data 1 (1) (2007) 1–30.
[29] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: Advances in Neural Information Processing Systems 14, 2001, pp. 849–856.
[30] L. Fei-Fei, R. Fergus, P. Perona, Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories, in: Workshop on Generative-Model Based Vision, IEEE Proc. CVPR, 2004.
[31] T. Tuytelaars, K. Mikolajczyk, Local invariant feature detectors: a survey, Foundations and Trends in Computer Graphics and Vision 3 (3) (2008) 177–280.
[32] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (10) (2005) 1615–1630.
[33] S. Maji, A.C. Berg, J. Malik, Classification using intersection kernel support vector machines is efficient, in: CVPR, 2008, pp. 1–8.
[34] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, 2001.
[35] B. Leibe, K. Mikolajczyk, B. Schiele, Efficient clustering and matching for object class recognition, in: BMVC, 2006.
[36] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, A.Y. Wu, An efficient *K*-means clustering algorithm: analysis and implementation, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 881–892.
[37] C. Elkan, Using the triangle inequality to accelerate *k*-means, in: Proceedings of the Twentieth International Conference on Machine Learning, 2003, pp. 147–153.

Please cite this article in press as: R.J. López-Sastre et al., Towards a more discriminative and semantic visual vocabulary, Comput. Vis. Image Understand. (2010), doi:10.1016/j.cviu.2010.10.009