Categorizing and Annotating Medical Images by Retrieving Terms Relevant to Visual Features

Desislava Petkova and Lisa Ballesteros *
Mount Holyoke College
dipetkov|lballest@mtholyoke.edu

Abstract

Images are difficult to classify and annotate but the availability of digital image databases creates a constant demand for tools that automatically analyze image content and describe it with either a category or a set of words. We develop two cluster-based cross-media relevance models that effectively categorize and annotate images by adapting a cross-lingual retrieval technique to choose the terms most likely associated with the visual features of an image. We also identify several important distinctions between assigning categories and assigning words.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing

Keywords

Cross-language Retrieval, Image Annotation, Categorization

1 Introduction

The exponential growth of multi-media information has created a compelling need for innovative tools for managing, retrieving, presenting, and analyzing image collections. Medical databases, for example, continue to grow as hospitals and research institutes produce thousands of medical images daily. The design and development of image retrieval systems will support a variety of tasks, including image retrieval, auto-illustration of text documents, medical diagnostics, organizing image collections such as digital photo albums, and browsing

Image retrieval techniques can be classified into two types, content based image retrieval (CBIR) and text-based image retrieval (TBIR). CBIR attempts to find images based on visual similarities such as shape or texture. TBIR techniques retrieve images based on semantic relationships rather than visual features and require that descriptive words or annotations have been previously assigned to each image. For collections of realistic size, it is impractical to rely exclusively on manual annotation because the process is both time-consuming and subjective. The task is even more challenging for special collections such as medical databases since they require expensively trained professionals to do the annotation. As a practical alternative, automatic annotation can either complement or substitute manual annotation.

The goal of automatic image annotation is to assign semantically descriptive words to unannotated images. As with most tasks involving natural language processing, we assume that a training collection of already annotated images is available, which the system can use to learn

*This work was funded in part by the Howard Hughes Medical Institute Cascade Mentoring Program (HHMI #52005134) and the Clare Boothe Luce Program of the Henry Luce Foundation.

what correlations exist between words and visual components or *visterms*. We specify the task further by considering annotation to be a cross-lingual retrieval problem: Two languages - textual and visual - are both used to describe images, and we want to infer the textual representation of an image given its visual representation. Therefore, we can think of words being the target language and visterms being the source language. Of course, the language of visterms is entirely synthetic but a CLIR system does not require specialized linguistic theory and knowledge.

2 Background

Other researchers have proposed methods for modeling the relationships between words and visual components. Mori *et al* divide images into regions and then use the co-occurrence of words and regions to make nonsmoothed maximum likelihood estimates [9]. Duygulu *et al* apply a segmentation algorithm to generate image blobs and then use a Machine Translation model to assign words as a form of multi-modal translation from blobs to words [3]. More recently, Jeon *et al* apply a Maximum Entropy model that treats annotation as a discrete stochastic process whose unknown parameters are word probabilities [6].

Our approach is a modification of the Cross-media Relevance Model (CMRM) developed by Jeon *et al* [5]. In this case, the visterms of an image to be annotated constitute a query and all candidate words are ranked in terms of their relevance to the visual representation. An annotation of any length can be created by selecting the n highest ranked words. More precisely, using a collection T of training images J, the joint probability of observing a word w and the set of visterms derived from an unannotated image $I = \{v_1, ..., v_m\}$ is computed as:

$$P(w, v_1, ..., v_m) = \sum_{J \in T} P(J) P(w|J) \prod_{i=1}^{m} P(v_i|J)$$

where $P(w|J)$ and $P(v|J)$ are maximum-likelihood estimates smoothed with collection frequencies.[1]

$$\begin{aligned} P(w|J) &= (1-\alpha)\frac{\#(w,J)}{|J|} + \alpha\frac{\#(w,T)}{|T|} \\ P(v|J) &= (1-\beta)\frac{\#(v,J)}{|J|} + \beta\frac{\#(v,T)}{|T|} \end{aligned}$$

Therefore, CMRM uses word-visterm co-occurrences across training images to estimate the probability of associating words and visterms together. But since this method computes the word and visterm distributions $P(\cdot|J)$ of each image separately, it does not take into account global similarity patterns, i.e. how individual images and their representations are related to each other. This shortcoming can be compensated by extracting and incorporating information from groups of similar images - *clusters* - created by examining the overall corpus structure.

Document clustering within the framework of full text retrieval has been investigated by Liu *et al* [8]. They define two cluster-based models: Cluster Query Likelihood (CQL) and Cluster-based Document Model (CBDM). Both explore across-document word co-occurrence patterns in addition to within-document occurrence patterns to improve the ranking of documents in response to user queries. CQL directly ranks clusters based on $P(Q|C)$, the probability of a cluster C generating the query Q, while CBDM ranks documents but smooths their language models with the models of respective clusters. Liu *et al* show that clustering improves retrieval performance indicating that clusters provide more representative statistics of word distributions because they combine multiple related documents.

We adapt these techniques to annotate and categorize images by extending the Cross-media Relevance Model to take advantage of cluster statistics in addition to image statistics.

[1] Throughout the paper I denotes an image to be annotated, and J - an already annotated training image.

$$P(w, v_1, ..., v_m) = \sum_{C \in T} P(C)P(w|C) \prod_{i=1}^{m} P(v_i|C) \qquad P(w, v_1, ..., v_m) = \sum_{J \in T} P(J)P(w|J) \prod_{i=1}^{m} P(v_i|J)$$

$$P(w|C) = (1-\gamma)\frac{\#(w,C)}{|C|} + \gamma\frac{\#(w,T)}{|T|} \qquad P(w|J) = (1-\alpha)\frac{\#(w,J)}{|J|} + \alpha\frac{\#(w,C_J)}{|C_J|}$$

$$P(v|C) = (1-\delta)\frac{\#(v,C)}{|C|} + \delta\frac{\#(v,T)}{|T|} \qquad P(v|J) = (1-\beta)\frac{\#(v,J)}{|J|} + \beta\frac{\#(v,C_J)}{|C_J|}$$

Mathematical definitions of CQL (left) and CBDM (right). Note that clusters C play two different roles - ranking in CQL and smoothing in CBDM.

The motivation is that by analyzing collection-wide co-occurrence patterns, a cluster-based approach to annotation can achieve a better estimation of word-visterm relationships. Clusters, viewed as large pseudo-images, have more words and visterms and therefore their language models $P(\cdot|C)$ can be approximated better than those of single images. In short, even if no prior knowledge about the collection is available, we can learn from its similarity structure by inferring word-visterm co-occurrences from similar images. For example, indirect relationships between words and visterms that do not occur together can be identified when there exist intermediate visterms with which they co-occur independently.

3 Methodology of categorization and annotation

3.1 From categories to concepts

Textual representations provided for ImageCLEFmed 2005 are clearly categories rather than annotations. Training images are divided into disjoint groups by being put into one of 57 folders, and each folder is given a short description (multi-axial code). Since we are interested in both categorizing and annotating images, we first need to create more realistic annotations. We achieve this by breaking up categorical records into sets of individual concepts.

We define a "concept" to be a comma-separated string, creating a restricted vocabulary of 46 distinct concepts. Some of these are literal dictionary words (e.g. 'x-ray" and 'spine'), others are sequences of words (e.g. 'plain radiography' and 'radio carpal joint'), and they all identify a single distinctive image property. For example, the third concept in a categorical description indicates body orientation - the choices are 'coronal', 'sagittal', 'axial' and 'other orientation'. Clearly, it does not make sense to have 'other' as a concept on its own.

x-ray
plain radiography
coronal
upper extremity arm
hand
musculoskeletal system

x-ray
plain radiography
coronal
upper extremity arm
radio carpal joint
musculoskeletal system

x-ray
plain radiography
coronal
upper extremity arm
hand forearm
musculoskeletal system

x-ray
plain radiography
coronal
upper extremity arm
elbow
musculoskeletal system

x-ray
plain radiography
coronal
upper extremity arm
upper arm
musculoskeletal system

x-ray
plain radiography
coronal
upper extremity arm
shoulder
musculoskeletal system

Figure 1: Images from six different categories whose annotations have all but one concept in common. Even though the annotations are very similar, the images themselves look quite different.

Thus we get two kinds of textual representations per image - a category and an annotation. We also recognize the first important difference between the two. Concepts do not point directly to objects in the images (there is one object per image anyway) but describe very high-level, specialized attributes which are not reflected directly by any visual feature. As a result, images that are apparently different can have very similar annotations, i.e. share many concepts (Figure

1). In contrast, all images classified in the same category are visually similar. In the rest of the paper, we refer to concepts and categories jointly as terms.

We also observe that concepts have an unusual distribution where the six most frequent ones account for more than 75% of the total number of occurrences (Figure 3). In fact, one concept - 'x-ray' - appears in every single image. Both CQL and CBDM would likely be biased in favor of these very frequent concepts, tending to select them rather than rare ones. Since we set the models to generate fixed-length annotations of six concepts (this is the maximum length of training annotations), we would expect the same set of concepts to be assigned over and over.

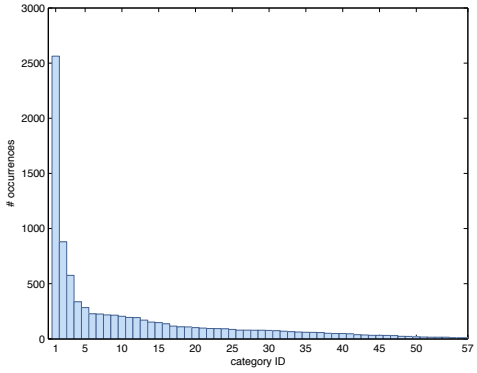

Figure 2: Category distribution in ImageCLEFmed.

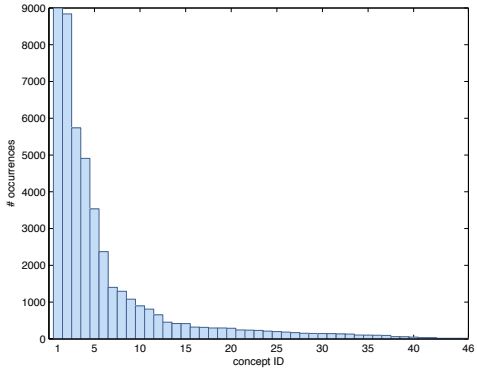

Figure 3: Concept distribution in ImageCLEFmed.

3.2 Describing image content

Given the fact that we make a distinction between categories and concepts, we need to redefine the task. We are interested both in selecting the category to which an image belongs and in choosing concepts to describe it. (And of course the appropriate concepts and category are themselves related.) Therefore, we are going to evaluate each model on two tasks simultaneously - annotation and categorization.

Recall that both CQL and CBDM compute a set of probabilities $P(w_i|I), i = 1...|V|$, based on the visterms of an image I. These probabilities are used to rank terms w according to their suitability to describe the content of I. The only restriction on the vocabulary V is that it is a finite set of discrete elements. Both categories and individual concepts satisfy this requirement therefore we can use the same implementation to assign either categories or concepts by only changing the input to the system.

3.2.1 Assigning categories

We consider each category to be an annotation of length 1. By learning relationships between categories and visterms we can categorize new images directly by assigning the term with the highest probability.

3.2.2 Assigning concepts

We divide categories into concepts and work with annotations of various lengths. By learning relationships between concepts and visterms we can annotate new images directly by assigning several of the highest probability concepts. Alternatively, we can categorize new images indirectly by representing categories as combinations of concepts:

$$P(\texttt{category}) = P(\texttt{concept}_1, ..., \texttt{concept}_k) = \sum_{i=1}^{k} P(\texttt{concept}_i)$$

4 Data processing and experimental setup

Preliminary image processing involves extracting visual features and obtaining an image vocabulary of visterms. Briefly, our representations are generated in the following way. Each image is grid partitioned into regions and the complete set of image regions is partitioned into disjoint groups based on corresponding feature vectors. All regions in a group are given the same unique identifier or *visterm*. Once image processing is complete, our approach relies on a model of the correspondences between terms and visterms, inferred from a set of training images that have been previously annotated.

4.1 Feature extraction and visterm generation

The dataset consists of 10000 images, divided into a training set of 9000 and a test set of 1000. First, each of these images is scaled to 256×256 pixels (regardless of the original aspect ratio) and divided into a 5×5 square grid. This produces 250,000 regions to be discretized into visterms. Regions are clustered on the basis of visual similarities and each cluster is assigned a unique identifier. Since the ImageCLEFmed collection consists entirely of black-and-white images, we only consider visual features that analyze texture. More specifically, we apply two texture analysis features - Gabor and Tamura.

4.1.1 Gabor energy

The Gabor Energy method measures the similarity between image neighborhoods and specially defined masks to detect spatially local patterns such as oriented lines, edges and blobs [4]. We use a MATLAB implementation courtesy of Shaolei Feng at the Center for Intelligent Information Retrieval, University of Massachusetts at Amherst. This feature computes a 12-bin histogram per image region.

4.1.2 Tamura texture

The Tamura features - Coarseness, Directionality and Contrast - are intended to reproduce human visual perception. They attempt to quantify intuitive information such as roughness, presence of orientation, and picture quality in terms of factors like sharpness of edges and period of repeating patterns. We use the FIRE Flexible Image Retrieval Engine to extract Tamura features [2]. Given an input image, FIRE creates three output partial images, one for each of the three features, which we convert into vectors. Each feature produces a 36-dimensional vector from every 6×6 partial image.

4.2 Combining visual features

Visual features describe distinctive image properties. Even if two features both analyze texture, they do so using different calculations and therefore might recognize different characteristics of the texture. On the other hand, we do not want to waste time and resources to extract correlated features, which are equivalent rather than complimentary sources of information. However, Deselaers *et al* show that Gabor filters and the individual Tamura features are not correlated [1]. Therefore, we try to combine the four of them for a more comprehensive texture analysis. We investigate two alternatives.

4.2.1 Combining features at visterm generation

First we join feature vectors produced by each feature in one compound vector, and then we cluster to quantize the vectors into visterms. For example, the length of Gabor energy is 12 (representing 12 histogram bins) and the length of Coarseness is 36 (representing 36 pixels of a 6x6 partial image). The result is a 250000×48 matrix of feature vectors, which is partitioned into 500 visterms. These theoretically reflect similarity of regions based both on Gabor energy and Coarseness.

4.2.2 Combining features at visterm representation

Rather than combining feature vectors prior to visterm assignment, we cluster the feature vectors produced by each feature separately. For example, we partition the regions into 500 visterms based on Gabor energy and then repartition them based on Coarseness. Use different cluster identifiers each time, e.g. integers from 1 to 500 for Gabor energy and integers from 501 to 1000 for Coarseness, and assign both types of visterms to individual images. So if an image is originally divided into 25 regions, it will end up with twice as many visterms. In this case, images can be similar in one respect, e.g. have some Gabor visterms in common, and dissimilar in another, e.g. share no Coarseness visterms. Also, their visual representations are longer, therefore probability estimates could be closer to the true underlying distribution.

The two approaches have different resource requirements. Combining at generation needs more memory (to fit a bigger matrix) while combining at representation needs more time (to group the regions separately for each feature). This fact should be taken into consideration, especially when working with large collections.

Our experiments show that combining features at generation is not very effective while two features combined at representation work better than either feature alone. Figure 4 graphs the performance of CQL according to error rate, as the number of clusters increases. Figure 5 graphs the same results for CBDM. It is likely that the combining features at generation fails because the weaker feature Coarseness is three times as long as the better feature Gabor energy. On the other hand, when combining at representation each feature accounts for 25 out of the 50 visterms per image, so in this respect the features are given equal weight.

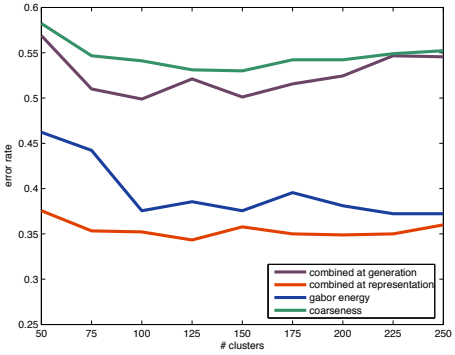

Figure 4: CQL performance with Gabor energy and Coarseness combined.

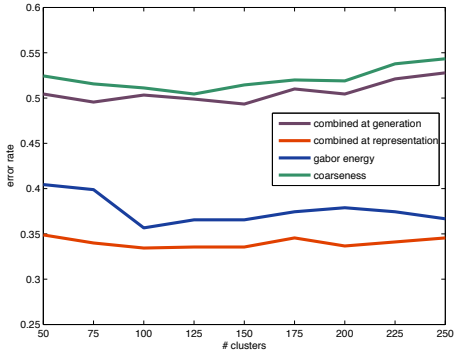

Figure 5: CBDM performance with Gabor energy and Coarseness combined.

4.3 Clustering techniques

The theoretical framework of cluster-based relevance modeling does not depend on the specific implementation of clustering. We investigate three different clustering techniques that partition an image collection into groups of similar images: groups based on manually assigned categories, K-means clustering, or K-nearest neighbors (KNN) clustering.

4.3.1 Categories

Since ImageCLEFmed images are assigned to one particular category, we can assume that categories play the role of cluster labels. It becomes straightforward to partition the collection by putting all images of the same category in a separate cluster. The result is a set of 57 non-overlapping clusters of various lengths, depending on how many training examples from each category are provided.

In general annotations are longer than a single word and therefore clustering could not be that simple. Moreover, visterms tend to identify lower-level visual properties and terms - higher-level semantic ones, making it advantageous to consider both the visual and textual representations of images for cluster computations. By combining terms and visterms a clustering technique can generate clusters with both visual and semantic coherence.

4.3.2 K-means

This clustering algorithm takes K, the desired number of clusters, as input and returns a list of indices indicating to which cluster each point in the partitioned dataset belongs. The procedure starts by randomly selecting K elements from the set as initial cluster centroids. Each remaining element is added to the cluster to which it is most similar, then the centroids are reevaluated. The algorithm refines the partitioning iteratively by repeatedly reevaluating and reassigning until no element changes assignment and the clustering converges.

K-means is a *hard* clustering algorithm which produces mutually exclusive clusters. Performance depends on the starting condition - both the predetermined value of K and the initial choice of centroids. The appropriate number of clusters is determined by the dataset configuration which is usually unknown. And even if the value of K is close to the natural number of groupings, given the starting centroid positions K-means can still get trapped in a local maximum and fail to find a good solution. The method is also sensitive to extreme points which lie notably far away from most points or *outliers*. Because K-means computes centroids as within-cluster averages, an outlier can pull away a centroid away from its true position. We select the value for K experimentally. We test values that range from 50 to 250 in 25-step increments.

4.3.3 K-nearest neighbors

Kurland *et al* propose a clustering method that takes the K-1 nearest neighbors of each training image to form a cluster of size K [7]. In contrast to K-means, KNN is a *soft* clustering technique that can assign an element to more than one cluster. If an image is a top ranked neighbor to several others, then it belongs to each of the corresponding clusters. KNN generates as many clusters as there are training images, and all of them have exactly the same size since each includes an image and its K-1 nearest neighbors.

To find the nearest neighbors of a training image J_k, all images $J_m, m = 1...|T|, m \neq k$, are first ranked according to their similarity to J_k. In our work, language models are generated by smoothing image frequencies with collection frequencies. Then the similarity between J_k and J_m is estimated as $\texttt{sim}(J_k, J_m) = \prod_{i=1}^{|J_k|} P(t_i|J_m)$, where t_i are the terms and visterms of J_k. The ranking process is repeated $|T|$ times - once for each one of the training images in the collection T.

5 Experimental results

5.1 Parameter setting

The cluster-based models rely on several smoothing and clustering parameters. These include: α for smoothing terms in image models, β for visterms in image models, γ for terms in cluster models, δ for visterms in cluster models, K for the number of clusters with K-means, and K for the number of nearest neighbors with KNN clustering.

We apply 10-fold cross validation to set each parameter. We divide the 9000 training images into 10 subsets of equal size and optimize performance by minimizing the error rate. For each possible parameter value, we train the model 10 times using one of the folds for testing and the rest for training, and we average the accuracy of the 10 trails. This evaluation method is more reliable than the simpler holdout method because it uses every training image for validation exactly once.

We determine that CQL works best with $\gamma = 0.1$ and $\delta = 0.2$ while CBDM works best with $\alpha = 0.5, \beta = 0.8, \gamma = 0.5$ and $\delta = 0.3$. We use these values throughout the rest of the experiments. On the other hand, cluster quality is closely linked to the visual feature, more precisely to its effectiveness to produce visterms with discriminative power. Since the value of K is feature-dependent, we cross-validate it individually for each visual feature.

5.1.1 Feature effectiveness

To get a sense of the relative effectiveness of the extracted features, we compare Coarseness and Gabor energy. The former has highest performance at 100 clusters, the latter at 225, and Gabor energy is the more useful feature (Tables 1 and 2). However, we cannot improve accuracy by simply setting K to an ever higher value. Larger K does not automatically imply better performance - it is cluster quality that matters, not the number of clusters.

Since images represented with Coarseness visterms are clustered into fewer groups, it is likely that dissimilar images will occasionally be contained in the same cluster. Perhaps Coarseness captures less information about content, yielding poorer discrimination between the visual representations of images. This would be true if the images are naturally structured into more groups, but the clustering algorithm fails to distinguish between some groups based on the Coarseness representations. However, even though Coarseness extracts less information than Gabor energy (or rather, less useful information), its texture analysis does not overlap with that of Gabor energy. Since they identify different image properties, combining the two features proves to be an advantage (Section 4.2).

5.2 Evaluation measures

Possible evaluation measures do not necessarily suggest the same feature as most effective. Therefore, we need to decide which measure is most appropriate for either task. We compare four measures with respect to categorization using the CQL model: `error rate`, `precision at 0.0 recall`, `average F-measure`, and `mean average precision`. As discussed in Section 5.1, we set the smoothing parameters γ and δ to 0.1 and 0.2, respectively. The clustering parameter K is feature-dependent - we use 225 for Gabor energy, 100 for Coarseness, and 200 for Gabor energy combined with the three Tamura features (Coarseness, Directionality and Contrast). Results are reported in Tables 1 and 2.

	Ranking according to error rate		Ranking according to highest precision		Ranking according to F-measure		Ranking according to mAP	
I.	Gabor energy and Tamura	.3178	Gabor energy and Tamura	.6792	Gabor energy and Tamura	.4125	Gabor energy	.3800
II.	Gabor energy	.3722	Gabor energy	.6527	Gabor energy	.3724	Gabor energy and Tamura	.3195
III.	Coarseness	.5078	Coarseness	.5087	Coarseness	.2010	Coarseness	.2412

Table 1: Ranking visual features according to their categorization effectiveness (CQL performance).

The experiments show that Gabor energy is the best feature for assigning annotations. On the other hand, Gabor energy and Tamura combined is the optimal feature for assigning categories according to all but mean average precision, in which Gabor energy is best. This leads to the question of which evaluation measure should be used to optimize parameters.

	Ranking according to error rate		Ranking according to highest precision		Ranking according to F-measure		Ranking according to mAP	
I.	Gabor energy	.1513	Gabor energy	.8909	Gabor energy	.5560	Gabor energy	.5863
II.	Gabor energy and Tamura	.1516	Gabor energy and Tamura	.8338	Gabor energy and Tamura	.5530	Gabor energy and Tamura	.4137
III.	Coarseness	.2060	Coarseness	.7008	Coarseness	.3546	Coarseness	.3842

Table 2: Ranking visual features according to their annotation effectiveness (CQL performance).

Perhaps the most important difference between categories and annotations is that a category consists of a single term and an annotation is constructed from multiple terms (not necessarily but in most cases). We select the evaluation measure based on this important distinction.

When assigning categories, only the highest ranked category is selected, so we need not be concerned about the tradeoff between recall and precision. On the other hand, when we assign annotations we select several concepts. In this case, we are interested in both recall and precision. These properties of categorization and annotation help us choose the appropriate evaluation measure. For categorization, an evaluation measure that reflects the precision of assigning categories should be selected - either error rate or precision at 0% recall. For the annotation task an evaluation measure that combines recall and precision should be selected - either F-measure or mean average precision. In the remaining experiments, effectiveness of categorization and annotation are measured via error rate and F-measure, respectively.

5.3 Weighting concept probabilities to assign categories

The method of splitting annotations into concepts and then multiplying concept probabilities to rank categories is, not surprisingly, very ineffective (Table 3).

	Categories	Concepts not scaled	Concepts scaled
CQL Gabor energy	.372222	.588889	.389999
CQL Coarseness	.507778	.627778	.528889
CBDM Gabor energy	.374444	.596667	.377778
CBDM Coarseness	.468889	.624444	.483333

Table 3: Improving the categorization performance (measured in error rate) of concepts by scaling their probabilities.

Concepts by themselves have a very good annotation performance, so in theory they should contain enough information for categorization. By directly multiplying concept probabilities to estimate the probability of assigning a category, we treat all concepts as 'equal'. In general this is the right approach since we do not know in advance what the user would be interested in, so there are no 'significant' and 'insignificant' words. Therefore, we would want to assign both frequent (background) words and rare (foreground) words correctly. But in the very specific case of using concepts to categorize medical images, we can in fact make a distinction between concepts based on the number of times a concept appears in categorical definitions.

As Figure 6 shows, some concepts are more important than others for defining a category - its a rare concept that defines a category best by distinguishing it from the rest. To take advantage of this information, we scale concepts using a TF×IDF weighting scheme. Concept probabilities are computed as:

$$P(c_i|J) = \frac{1}{log(\#(c_i, S))} P(c_i|J)$$

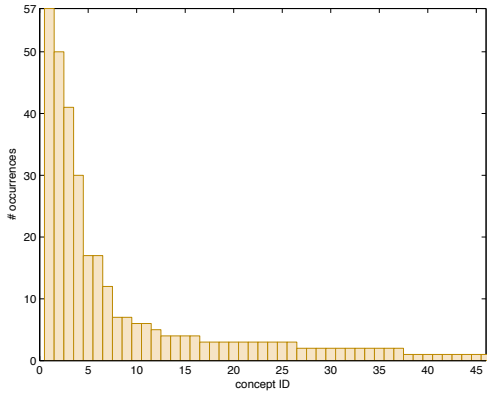

Figure 6: Concept distribution across the set of 57 categories. The most frequent concept appears is every single category.

where S is the set of categorical definitions.

Thus we emphasize concepts that appear in only a few categories and penalize concepts that appear in many categories since they are not very effective for differentiating between categories.

5.4 Clustering Effectiveness

Clustering is an important ingredient of our method and the choice of clustering technique can significantly affect performance. We look at three alternatives as described in Section 4.3.

5.4.1 Categories

Our baseline method is clustering according to manually assigned category (Section 4.3.1). In this case there are no parameters to optimize because all cluster information is explicitly specified in the data.

Even though category-clusters have satisfactory performance (Table 4), we should acknowledge that most collections are not processed, so this kind of 'naive' clustering would not always be an option. And even if there are categorized images available (for instance, if the user provides some manual examples to a browsing system which automatically organizes an image collection according to user preferences), these categories would be user-dependent and therefore not necessarily well-defined. They could be either too broad or too specific, i.e. generate clusters that are either too large and loose, or too small and with no real advantage over individual images.

	CQL		CBDM	
	error rate	nonzero categories	error rate	nonzero categories
Categories	.3010	37	.2570	40
K-means	.2650 (.0014)	36	.2630 (.4709)	39
KNN	.2440 (.0166)	40	.2310 (.0006)	46

Table 4: Categorization performance of cluster-based CMRM improves with unsupervised clustering (K-means or KNN). 95%-confidence p-values according to the Wilcoxon signed-rank test are reported in parenthesis.

Therefore, we would like to find an unsupervised clustering method which performs as well as or even better than manually labeled clusters. Then we could rely on the system to automatically

find appropriate values for clustering parameters, so that generated clusters approximate the collection's natural configuration.

5.4.2 K-means

K-means is a general clustering technique described in Section 4.3.2. It has one input parameter, the number of clusters K, which is optimized separately for each visual feature. K-means gives CQL a statistically significant improvement but slightly hurts CBDM (Table 4). The results indicate that the medical categories are relatively broad. For example, there might be a category which contains two visually different types of images, and the accuracy of CQL increases as a result of separating them into two different clusters. (We know that K-means breaks up some of the category clusters because the value of K is larger than 57 (Table 5). In this way, the system deals with the issue of some clusters not being compact enough. On the other hand, cluster compactness has less influence on their usefulness as background models for smoothing and this could explain why the performance of CBDM does not improve. (With CBDM emphasis is on generalization and therefore recall, and with CQL - on correctness and therefore precision.)

For other collections manual categories can be too narrowly defined. In such situations we would expect K-means to generate fewer clusters than categories, thus increasing recall, which would have a positive effect both on CQL and CBDM.

However, it is not necessary to use CQL and CBDM with the same set of clusters. In fact, CBDM shows a consistent tendency to perform best with fewer but larger clusters as compared to CQL:

	CQL	CBDM
Gabor energy	225	175
Coarseness	100	75
Dimensionality	200	150
Contrast	150	75
Gabor energy and Tamura	200	100

Table 5: K, the number of clusters K-means generates, is a feature-dependent parameter. However, CBDM consistently is set to use smaller K, and hence bigger clusters, than CQL.

CQL and CBDM apply clusters in two conceptually different roles - on one hand, as training examples which are somewhat more general than images, and on the other hand, as background collections which are somewhat more specific than the entire collection. Implicitly, bigger clusters are more useful for generalizing patterns observed in individual images - if the clusters are too small, they would fail to capture some aspects of member images and their content. Therefore, with CBDM we are less concerned about the compactness of the clusters, and can allow some relatively dissimilar elements to join the same cluster.

5.4.3 K-nearest neighbors

KNN is a soft clustering technique described in Section 4.3.3. We optimize K separately for the two cluster-based models and establish empirically that K=25 for CQL and K=50 for CBDM.

First, this corroborates our previous conclusion that CQL works well with very compact clusters and CBDM works well with more general clusters. We also observe that categorization performance improves with a statistically significant difference as compared to K-means clustering (Table 4). KNN clusters have more local coherence because they are defined with respect to particular image (i.e. locally). Since by generation a KNN cluster is specific to an image, it is better at describing its context. In addition, the KNN method does not reduce the number of training examples. It generates as many clusters as there are images. On the other hand, K-means creates considerably fewer clusters, which implies that there are fewer observations on which to base the model's probability estimations.

6 Conclusion

In this work, we analyzed a cluster-based cross-lingual retrieval approach to image annotation and categorization. We described two methods for incorporating cluster statistics into the general framework of cross-media relevance modeling and showed that both build effective probabilistic models of term-visterm relationships. We also discussed how different clustering techniques affect the quality and discriminative power of automatically generated clusters. Finally, we demonstrated an efficient method for combining visterms produced by several visual features.

We regard clustering as a kind of unsupervised classification that offers greater flexibility than manual classification. If the actual categories are too broad, then the system can break them into smaller clusters. If the actual categories are too specific, then it can redefine them by generating bigger clusters. If manually assigned categories are unavailable, the system can create them automatically. The only disadvantage is that automatic clusters do not have explicit textual descriptions, but the word distribution in clusters could be analyzed to build statistical language models.

In the future, we plan to investigate grouping by concept (similar to the method of grouping by category described here but based on annotation words) as an alternative version of soft clustering. We are also interested in analyzing the categorization performance of CQL and CBDM on a collection of true-color images to examine how visual properties influence accuracy.

References

[1] Thomas Deselaers, Daniel Keysers, and Hermann Ney. Features for Image Retrieval: A Quantitative Comparison. In *Proceedings of the 26th DAGM Pattern Recognition Symposium*, 2004.

[2] Thomas Deselaers, Daniel Keysers, and Hermann Ney. FIRE - Flexible Image Retrieval Engine: ImageCLEF 2004 Evaluation. In *Proceedings of the CLEF 2004 Workshop*, 2004.

[3] Pinar Duygulu, Kobus Barnard, Nando de Freitas, and David Forsyth. Object Recognition as Machine Translation: Leaning a Lexicon for a Fixed Image Vocabulary. In *Proceedings of 7th European Conference on Computer Vision*, 2002.

[4] I Fogel and Dov Sagi. Gabor Filters as Texture Discriminator. *Journal of Biological Cybernetics*, 61(102-113), 1989.

[5] Jiwoon Jeon, Victor Lavrenko, and R. Manmatha. Automatic Image Annotation and Retrieval using Cross-Media Relevance Models. In *Proceedings of the 26th International ACM SIGIR Conference*, 2003.

[6] Jiwoon Jeon and R. Manmatha. Using Maximum Entropy for Automatic Image Annotation. In *Proceedings of the 3rd International Conference on Image and Video Retrieval*, pages 24–32, 2004.

[7] Oren Kurland and Lillian Lee. Corpus Structure, Language Models, and Ad Hoc Information Retrieval. In *Proceedings of the 27th Internation ACM SIGIR Conference*, 2004.

[8] Xiaoyong Liu and W. Bruce Croft. Cluster-Based Retrieval using Language Models. In *Proceedings of the 27th International ACM SIGIR Conference*, 2004.

[9] Yasuhide Mori, Hironobu Takanashi, and Ryuichi Oka. Image-to-word Transformation based on Dividing and Vector Quantizing Images with Words. In *Proceedings of the 1st MISRM International Workshop*, 1999.