Automatic Multimedia Cross-modal Correlation Discovery

Jia-Yu Pan  
*Carnegie Mellon University*

Hyung Jeong Yang  
*Carnegie Mellon University*

Pinar Duygulu  
*Carnegie Mellon University*

Christos Faloutsos  
*Carnegie Mellon University*

Follow this and additional works at: [http://repository.cmu.edu/compsci](http://repository.cmu.edu/compsci)

Published In  
Proceedings of the Tenth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, 653-658.
ABSTRACT
Given an image (or video clip, or audio song), how to automatically assign keywords to it? The general problem is to find correlations across the media in a collection of multimedia objects, like video clips, with colors, and/or motion, and/or audio, and/or text scripts. We propose a novel, graph-based approach, to discover such multi-modal correlations.

Our method requires no tuning, no clustering, no user-determined constants; it can be applied to *any* multimedia collection, as long as we have a similarity function for each medium; and it scales linearly with the database size. We report auto-captioning experiments on the 'standard' Corel image database of 680 MB, where it outperforms domain specific, fine-tuned methods by up to 10 percentage points in captioning accuracy (50% relative improvement).

Categories and Subject Descriptors
H.4 [Information Systems Applications]: Miscellaneous; D.2.8 [Software Engineering]: Metrics—complexity measures, performance measures

Keywords
multi-modal correlation, auto captioning, graph-based model

1. INTRODUCTION
Given a collection of multimedia objects, we want to find correlations across media. The driving application is auto-captioning, where the problem is defined as follows:

**Problem 1 (Auto-captioning).** Given a set $S$ of color images, each with caption words; and given one more, uncaptioned image $I$, find the best $t$ (say, $t=5$) caption words to assign to it.

However, the method we propose is general, and can be applied to video clips (with text scripts, audio, motion); on audio songs, with text lyrics, and so on. We will refer to the following additional scenarios:

**Scenario 1: Video Auto-captioning** For example, given a training set of captioned video clips, we want to explore the correlation between features of video clips and captions so that new unseen video clips can be captioned. (The goal of this paper is to seek associations between features and terms in captions.)

**Scenario 2: Multi-lingual text** Given a collection of documents, in two languages, say, English and Spanish, find correlations between English and Spanish terms.

Finally, just to illustrate the generality of our method, we also mention this scenario:

**Scenario 0: Text** Given a collection of text documents, and a term (say "clustering"), find terms that are related (like "classification", "mining", "learning").

The general problem is intuitively defined as follows. A more formal definition is provided later

**Problem 2 (Informal-General).** Given a multimedia objects, each consisting of $m$ attributes, (traditional, or multimedia; that is, text, video, audio, time-sequence, etc); Find correlations across the media (e.g., correlated keywords with image blobs/regions; video motion with audio features, etc.)

For example, we want to answer questions of the form “which keywords show up, for images with blue top”; or “which songs are usually in the background of fast-moving video clips”. We assume that domain experts have provided us with similarity functions for all the involved media.

How should we proceed? Even in the case of scenario 0 (text), we need to have domain specific methods (tf/idf, stop-list of common words). When images and audio are considered, it is a challenging to associate them with keywords, let alone with each other.

There are multiple research papers, attacking parts of the problem: for example, to associate words with images, people have proposed clustering and 'expectation maximization' (EM); for script words and movie faces there is the work in [23]. However, they all used carefully designed methods,
using a lot of domain knowledge.

We would like to find a unifying method, with the following specifications:

- it should be domain independent
- it should spot correlations in any of the above scenarios, with missing values, feature vectors, set-valued attributes, and all combinations thereof.
- it should scale up for large collections of objects, both with respect to training, as well as for responses.

In section 2, we briefly discuss the previous attempts on image captioning and provide background on random walk in a graph. Section 3 describes our proposed method and its algorithms. In section 4 we give experiments on real data. We discuss our observations in 5 Section 6 gives the conclusions.

2. RELATED WORK
We group the related work into areas, like image captioning, video mining, and matrix algebra.

Automatic image captioning
The area has attracted huge interest, and multiple data mining tools have been proposed, exactly because manual image annotation is tedious and subjective.

Maron et al. [18] used classifiers; Wenlin et al. [29] propose a semi-automatic strategy using user the feedback; Li and Wang [16] propose an automatic method, using a 2-D multiresolution Hidden Markov Model. Mori et al. [19] used co-occurrence statistics; Barnard et al. [5, 4, 3] propose a generative hierarchical model, inspired by Hofmann’s aspect model for text [13]; Blei and Jordan [6] propose the “correspondence Latent Dirichlet allocation” (CORR-LDA); Jeon et al. [14] propose a cross-media relevance model; Duygu et al.[10] use a machine translation approach, following Brown et al. [8].

Video and audio mining
The Infomedia project [28] has terabytes of video data; in general, video data bases spark efforts to associate text with images and faces [23]; and visual/auditory characteristics with video genres [21]. Similarly, there are successful efforts [27] to associate songs with their genre (like ‘jazz’, ‘classic’, etc).

In general, all the above methods report good result exploiting domain-specific knowledge, without providing a general method for finding correlations across modalities.

3. PROBLEM - PROPOSED METHOD
We proposed a novel approach for cross-media correlations, and we use image captioning as an illustration. Our main idea is to turn the problem into a graph problem. Next we describe (a) how to generate this graph (b) how to estimate cross-modal correlations and (c) how to do that efficiently.

Table 1 shows the terminology we used in the paper.

3.1 Problem definition
The problem is more formally defined as follows:

**Problem 3 (Formal).** Given a set \( S \) of \( n \) multimedia objects \( S = \{O_1, O_2, \ldots, O_n\} \), each with \( m \) multimedia attributes find patterns among them.

For example, if \( S \) is a collection of captioned images, how do we guess the (missing) caption terms of a new, uncaptioned image? In this paper, we study these questions and propose a framework for pattern discovery among multimedia objects, whose attributes could be either categorical or numerical.

We need to elaborate on the attributes: In traditional RDBMSs, attributes must be atomic (i.e., taking single values, like “ISBN”, or “video duration”). However, in our case, they can be set valued, like “caption”, or, even missing altogether.

We propose to gear our method towards set-valued attributes, because they include atomic attributes as a special case; and they also smoothly handle the case of missing values (null set). Thus, we only talk about set-valued attributes from now on.

**Definition 1.** The domain \( D_i \) of (set-valued) attribute \( i \) is the collection of atomic values that attribute \( i \) can choose from.

For example, for the attribute “caption”, its domain is the set of English words (e.g., the Oxford dictionary).

**Definition 2.** The values of domain \( D_i \) will be referred to as the domain tokens of \( D_i \).

**Assumption 1.** For each domain \( D_i \) \( (i = 1, \ldots, m) \), we are given a similarity function \( s_i(*,*) \) which assigns a score to each pair of domain tokens.

For example, for the attribute “caption”, the similarity function could be 1 if the two tokens are identical, and 0 otherwise.

A domain can consist of categorical values, numerical values, or numerical vectors. Perhaps surprisingly, we can encompass all the scenarios of the introduction with the above setting. For example, for Scenario 1 (image auto-captioning), we have objects of \( m=2 \) attributes, the first, “caption” has as domain a set of categorical values (English terms); the second, “image”, is a set of \( p \)-dimensional feature vectors (\( p=30 \), as we describe next).

Let’s elaborate on image captioning, before we present the main idea.

3.2 Case study: automatic image captioning
In this case, the objects of interest are images; each of them has numerical attributes (the feature vectors of its regions);
Table 1: Summary of symbols used in the paper

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O$</td>
<td>the $i$-th training object, $O_q$: the query object</td>
</tr>
<tr>
<td>$A_i$</td>
<td>the $i$-th attribute of an object</td>
</tr>
<tr>
<td>$V(O_i)$</td>
<td>the vertex of $G_{MMG}$ that corresponds to object $O_i$.</td>
</tr>
<tr>
<td>$I_i$</td>
<td>the $i$-th training image, $I_q$: the query image</td>
</tr>
<tr>
<td>$N$</td>
<td>the number of training objects</td>
</tr>
<tr>
<td>$N(A_i)$</td>
<td>the number of distinct values in the domain of attribute $A_i$</td>
</tr>
<tr>
<td>$m$</td>
<td>number of attributes per object</td>
</tr>
<tr>
<td>$N_{node}$</td>
<td>$N_{node} = N + \sum_{i=1}^{m} N(A_i)$, the number of nodes in $G_{MMG}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Matrix/vector</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>the (column-normalized) adjacency matrix</td>
</tr>
<tr>
<td>$v_q^*$</td>
<td>the restart vector of the query object (all zeros, except a single '1')</td>
</tr>
<tr>
<td>$\mathbf{u}_q$</td>
<td>the steady state probability vector for the $v_q^*$ restart vector</td>
</tr>
</tbody>
</table>

Figure 1: Three sample images, two of them annotated; their regions (d,e,f); and their “MMG” graph (g). (Figures look best in color.)

some of them also have a caption, with one or more words (categorical). See Figure 1 for the 3 sample images, their captions and their regions.

Following the domain experts and the sizable related literature, we can use any standard segmentation algorithms [25] for each image, break it into regions (see Figure 1(d,e,f)), and then map each region into a (say, 30-d) feature vector.

The mapping can be done with any of the published feature extraction functions (color-histograms, texture histograms, etc [11]). In this specific setting, we extracted $p=30$ features from each region ('blob'), like the mean and standard deviation of RGB values, average responses to various texture filters, its position in the entire image layout, and some shape descriptors (e.g., major orientation and the area ratio of bounding region to the real region). Note that the exact
feature extraction details are orthogonal to our approach - all our “MMG” method needs is a black box that will map each color image into a set of zero or more feature vectors.

Thus, we turned the original Problem 2 into the following: Given $n$ objects, each with $m$ set-valued attributes (categorical, numerical, or vector-valued), find correlations across modalities. E.g., which caption terms are more likely, when the image has a blue, sky-like blob.

The question is what to do next, to capture cross-media correlations. Should we use clustering on feature vectors, or should we use some classification method, as it has been suggested before? And, if yes, how many cluster centers should we shoot for? Or, if we choose classification, which classifier should we use? Next we show how to handle, and actually, bypass, all these issues, for any multimedia setting.

### 3.3 Main idea - Mixed Media Graph (“MMG”)

The main idea is to represent all the objects, as well as their attributes (domain tokens) as nodes in a graph. For multimedia objects with $m$ attributes, we obtain an $(m+1)$-partite graph. There are $m$ types of nodes (one for each attribute) and one more type of node for the objects.

#### Graph construction

See Figure 1 for an example. We will denote as $V(O)$ the vertex of object $O$, and as $V(a_i)$ to be the vertex of the attribute value $A = a_i$. Thus, we put an edge between every object-node and the corresponding attribute-value nodes.

There is only one subtle point: For numerical and vector attributes, we need a way to reflect the similarity between two vectors (e.g., the orange “tiger” region $r_0$ and the orange sky region $r_1$). Our approach is to add an edge if and only if the two feature vectors are close enough.

We need to decide on a threshold for the “closeness”. There are many ways, but we decided to make the threshold adaptive: for each feature-vector, choose its $k$ nearest neighbors, and add the corresponding edges. We discuss the choice of $k$ later, as well as the sensitivity of our results to $k$. Recall that computing the nearest neighbors is easy, because we already have the similarity function $s_i(·)$ for any domain $D_i$ (Assumption 1).

In summary, we have two types of links in our “MMG” graph: nearest neighbor, or $NN$-links, between the nodes of two similar feature vectors; and object-attribute-value ($OAV$-links), between an object node and an attribute value node.

Figure 1 illustrates our approach with an example:

**Example 1.** Consider the image set $S = \{I_1, I_2, I_3\}$ (Figure 1). The graph corresponds to this data set has three types of nodes: one for the image objects $i_3’s (j = 1, 2, 3)$; one for the regions $r_0’s (j = 1, \ldots, 11)$, and one for the terms $\{t_1, \ldots, t_8\} = \{\text{sea, sun, sky, waves, cat, forest, grass, tiger}\}$. Figure 1(g) shows the resulting “MMG” graph. Solid arcs indicate Object-Attribute-Value relationships; dashed arcs indicate nearest-neighbor (NN) relationships.

In this example, we consider only $k=1$ nearest neighbor, to avoid cluttering the diagram.

To solve the auto-captioning problem (Problem 1), we need to develop a method to find good caption words, eg., for image $I_3$. This means that we need to estimate the affinity of each term (nodes $t_1, \ldots, t_8$), to node $i_3$. We discuss it next.

**Correlation discovery by random walk**

This concludes the first step of our approach: We propose to turn the multimedia problem into a graph problem. Thus, we can tap the sizable literature of graph algorithms, and we can use off-the-shelf methods for assigning importance to vertices in a graph, as well as determining how related is an un-captioned image (represented by node, say “A” in the graph), to the term “tiger” (represented, say, by node “B” in the graph).

We have a choice of electricity based approaches [20] [9]: random walks (pageRank, topic-sensitive pageRank) [7, 12]; hubs and authorities [15]; elastic springs [17]. In this work, we propose to use random walk with restart (“RWR”) for estimating the importance/affinity of node “A” with respect to node “B”. But, again, the specific choice of method is orthogonal to our framework.

The “random walk with restarts” operates as follows: to compute the importance/affinity of node “B” for node “A”, consider a random walker that starts from node “A”, choosing randomly among the available edges every time. Except that, before he makes a choice, with probability $c$, he goes back to node “A”. Let $u_A(B)$ denote the steady state probability that our random walker will find himself at node “B”. Then, $u_A(B)$ is what we want, the importance of “B” with respect to “A”.

**Definition 3.** The importance of node $B$ with respect to node $A$ is the steady state probability $u_A(B)$ of random walk with restarts, as defined above.

For example, to solve the auto-captioning problem for image $I_3$ of Figure 1, we can estimate the steady-state probabilities $u_A(·)$ for all nodes of the “MMG”, we can keep only the nodes that correspond to terms, and we can report the top few (say, 5), as caption words.

### 3.4 Algorithms

In this section, we summarize the proposed MMG method for finding cross-modal correlations by presenting the pseudo code of the algorithm.

For the general problem, the algorithm is as follows: build the “MMG” graph; when the user asks for the importance of node “A” to node “B”, estimate the steady-state probability $u_A(B)$ defined above.

The computation of the steady-state probabilities is very interesting and important. We use matrix notation, for compactness. Let $Q_0$ be the query object (e.g., image $I_3$ of Figure 1) and suppose that we want to find the most related terms. We do an RWR from node $q$, and compute the
steady state probability vector \( \mathbf{u}_q = u_q(1), \ldots, u_q(N) \). Let \( N \) be the number of nodes in the "MMG" graph, and \( E \) the number of edges.

The estimation of vector \( \mathbf{u}_q \) can be implemented efficiently by matrix multiplication. Let \( A \) be the adjacency matrix of the "MMG" graph, and let it be column-normalized. Let \( \mathbf{v}_q \) be a column vector with all its \( N \) elements zero, except for the entry that corresponds to node \( q \); set this entry to 1. We call \( \mathbf{v}_q \) the "restart vector". Now we can formalize the definition of the 'importance' of a node (Definition 3).

**Definition 4. (Steady-state vector)** Let \( c \) be the probability of restarting the random walk from node \( q \). Then, the \( N \)-by-1 steady state probability vector, \( \mathbf{u}_q \), (or simply, steady-state vector) satisfies the equation:

\[
\mathbf{u}_q = (1 - c)A \mathbf{u}_q + c \mathbf{v}_q
\]

(1)

We can easily show that

\[
\mathbf{u}_q = c(I - (1 - c)A)^{-1} \mathbf{v}_q
\]

(2)

where \( I \) is the \( N \times N \) identity matrix. The pseudo code of finding cross-modal correlations is shown in Figure 2.

---

**Figure 2:** Algorithm-CCD: Cross-modal correlation discovery

### 3.5 Performance

Although fast already (linear on the database size), the proposed algorithm "Algorithm-CCD" can be even further accelerated. We can tap the old and recent literature of fast solutions to linear systems, to achieve fast approximations. We can do the matrix inversion and solve Eq. 2, or we can use a low-rank approximation [22, 1] or, for matrices that have block-diagonal shapes, we can use the methods by [26]. Given that this area is still under research, we only point out that our approach is modular, and it can trivially include whichever is the best module to do the fast matrix inversion.

#### 3.5.1 Scalability for auto-captioning

Let \( R \) be the number of regions extracted from the images, \( E \) be the number of edges in the graph \( G_{MMG} \), and \( \text{cost}\,NN(R) \) be the cost of performing a nearest-neighbor search in a collection of \( R \) feature vectors.

At the training phase, the proposed method MMG constructs the graph \( G_{MMG} \). To determine the nearest-neighbor links (NN-links), we perform a \( k \) nearest-neighbor (k-NN) search on each region. These searches can be accelerated using an index structure, like an R+tree [24]. The total training time is linear on the number of edges \( E \), and super-linear on the number of regions \( R \):

\[
\text{train time} = R \times \text{cost}\,NN(R) + E \times O(1)
\]

(3)

In the testing (i.e., captioning) phase, we need to estimate the steady state probability vector \( \mathbf{u}_q \) for a test image \( I_q \) iteratively, until the estimate stabilizes. In our experiments, the number of iterations (maxIter) is typically small (e.g., 50), or it can be set to have a upper bound, and is of order \( O(1) \). For each iteration, a sparse matrix multiplication is performed and costs around \( 2 \times E = O(E) \) operations. The overall cost for captioning a test image is therefore linear on the number of edges.

\[
\text{test time} = \text{maxIter} \times O(E) = O(E)
\]

(4)

### 4. EXPERIMENTAL RESULTS

In this section, we show experimental results to address the following questions:

- **Quality**: How does the proposed MMG method perform on captioning test images?
- **Parameter defaults**: How to choose good default values for the \( k \) and \( c \) parameters?
- **Generality**: How well does MMG capture other cross-media correlations: for example, how well does it solve the reverse problem (given a term like "sky", find the regions that are likely to correspond to it). Similarly, how well does MMG capture same-media correlations (say, term-term, or region-region correlations)

In our experiment, we use 10 image data sets from Corel, which is commonly used in previous work [10]. Table 2 shows the details of the data sets.

<table>
<thead>
<tr>
<th>Set</th>
<th>( R )</th>
<th>( I )</th>
<th>( T )</th>
<th>( N )</th>
<th>( E )</th>
<th>Test I</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48859</td>
<td>5188</td>
<td>133</td>
<td>54200</td>
<td>177093</td>
<td>1744</td>
</tr>
<tr>
<td>2</td>
<td>40576</td>
<td>5241</td>
<td>164</td>
<td>54981</td>
<td>180119</td>
<td>1783</td>
</tr>
<tr>
<td>3</td>
<td>50193</td>
<td>5289</td>
<td>154</td>
<td>55636</td>
<td>181785</td>
<td>1717</td>
</tr>
<tr>
<td>4</td>
<td>50082</td>
<td>5287</td>
<td>162</td>
<td>55351</td>
<td>181609</td>
<td>1746</td>
</tr>
<tr>
<td>5</td>
<td>50256</td>
<td>5273</td>
<td>160</td>
<td>55689</td>
<td>182451</td>
<td>1741</td>
</tr>
<tr>
<td>6</td>
<td>48962</td>
<td>5192</td>
<td>162</td>
<td>54316</td>
<td>177237</td>
<td>1737</td>
</tr>
<tr>
<td>7</td>
<td>50103</td>
<td>5266</td>
<td>174</td>
<td>55543</td>
<td>181533</td>
<td>1747</td>
</tr>
<tr>
<td>8</td>
<td>409064</td>
<td>5266</td>
<td>168</td>
<td>55398</td>
<td>181334</td>
<td>1724</td>
</tr>
<tr>
<td>9</td>
<td>49711</td>
<td>5239</td>
<td>173</td>
<td>55123</td>
<td>180261</td>
<td>1801</td>
</tr>
<tr>
<td>10</td>
<td>48908</td>
<td>5197</td>
<td>144</td>
<td>54249</td>
<td>177262</td>
<td>1731</td>
</tr>
</tbody>
</table>

**Table 2:** Details of the data sets: \( R \): training regions, \( I \): training images, \( T \): caption vocabulary size, \( N \): nodes \( = R + I + T \), \( E \): edges, Test I: test images

### 4.1 Quality

For each test image, we compute the captioning accuracy as the percentage of terms which are correctly predicted. For a test image which has \( m \) correct caption terms, MMG will predict also \( m \) terms. If \( p \) terms are correctly predicted,
then the captioning accuracy for this test image is defined as \( \frac{M}{m} \).

Figure 3 shows the average captioning accuracy for the 10 data sets. We compare our results with the results reported in [10]. The method in [10] is one of the most recent and sophisticated: it models the image captioning problem as a statistical translation problem and solves it using an probabilistic model using expectation-maximization (EM). We refer to their method as the “EM” approach. On the average, MMG achieves captioning accuracy improvement of 12.9 percentage points, which corresponds to a relative improvement of 58%.

![Figure 3: EM and MMG](image)

The parameters for MMG are \( c = 0.8 \), and \( K = 3 \).

We also compare the captioning accuracy with even more recent machine vision methods [8], on the same data set: the Hierarchical Aspect Models method (“HAM”), and the Latent Dirichlet Allocation model (“LDA”). Figure 4 compares the best average captioning accuracy among the 10 data sets reported by the two methods (HAM and LDA) [8], along with the average captioning accuracy of the proposed MMG method. Although both HAM and LDA improve on the EM method, they both lose to our generic “MMG” approach (35%, versus 29% and 25%).

It is also interesting that “MMG” also gives significantly lower variance by roughly an order of magnitude: 0.002 versus 0.02 and 0.03.

### 4.2 Parameter defaults

We experiment to find out how would different values of the parameters \( c \) and \( K \) affect the captioning accuracy.

Figure 5 shows the captioning accuracy of MMG using different values of decay factor \( c \). The parameter \( K \) is fixed at 3. The accuracy reaches a plateau as \( c \) grows from 0.5 to 0.9, which indicates the proposed MMG method is insensitive to the choice of \( c \). We show only the result on one data set “006”; the results on other data sets are similar.

![Figure 5: Captioning accuracy of different c](image)

Figure 6 shows the captioning accuracy of MMG using different values of decay factor \( k \). The decay factor \( c \) is fixed at 0.9. Again, the proposed MMG method is insensitive to the choice of \( K \), where the captioning accuracy reaches a plateau as \( K \) varies from 3 to 10.

![Figure 6: Captioning accuracy versus number of neighbors K](image)

### 4.3 Generality

MMG works on objects of any types. We design a experiment of finding similar caption terms using MMG. We use the same “MMG” graph constructed for automatic image captioning. To find the similar terms of a caption term \( t \), the restart vector is set have zeros at all elements except the one correspond to the caption term \( t \). Table 3 shows the
similar terms found for some of the caption terms. In the
table, each row shows the caption term in question at the
first column, followed by the top 5 similar terms found by
MMG (sorted by similarity degree).

Notice that the retrieved terms make a lot of sense for
example, the string 'branch' in the caption is strongly re-
lated to forest- and bird-related concepts ("birds", "owlF",
"night"), and so on. Notice again that we did nothing spe-
cial: no tf/idf, no normalization, no other domain-specific
analysis - we just treated these terms as nodes in our "MMG",
like everything else.

A second, subtle observation, is that our method does not
seem to be biased by frequent words. In our collection, the
terms "water" and "sky" were very frequent (like the terms
"the" and "a" are in normal English text). Yet, the frequent
terms do not show up too often in Table 3. We suspect that
the choice of large c help control the effects of popular nodes
such as "water" and "sky".

5. DISCUSSION

We are shooting for a method that requires no parameters.
Thus, here we discuss how to choose defaults for the decay
c parameter.

For web graphs, the recommended value for c is typically
c=0.15 [26]. Surprisingly, our experiments show that this
decision does not give good captioning performance. Instead,
good quality is achieved for c=0.8 or 0.9. Why is this dis-
crepancy?

We conjecture that what determines a good value for the
"decay factor" is the diameter of the graph. Ideally, we want
our "random walker" to have a non-trivial chance to reach the
outsiders of the whole graph. Thus, if the diameter of the
graph is d, the probability that he will reach a point on the
"periphery" is probably proportional to d^{1-c}.

For the web graph, the diameter is approximately d=19 [2]
which implies that the probability P_{periphery} for the random
walker to reach a node in the periphery is roughly

\[ P_{periphery} = (1 - c)^{19} = 0.045 \tag{5} \]

In the case of auto-captioning, with a tri-partite graph, the
diameter is roughly d=3. If we demand the same P_{periphery}
for our case, then we have

\[ (1 - 0.15)^{19} = (1 - c)^{3} \]
\[ \Rightarrow c = 0.65 \tag{6} \]

which is much closer to our empirical observations. Of course,
the problem requires more careful analysis - but we are the
first to show that c=0.15 is not always optimal for random
walks with restarts.

6. CONCLUSIONS

We started from the image auto-captioning problem, and we
developed "MMG", a general method that can spot corre-
lations across media. The proposed graph based model can
be applied to diverse multimedia data to find multi-modal
correlations. The method has the following desirable char-

- It is domain independent - the s_t() similarity functions
  completely isolate our "MMG" method from the do-

- It requires no user-defined parameters, nor any other
tuning (in contrast to linear/polynomial/kernel SVMs,
K-means clustering, etc) For its only two parameters,
k and c, we give good default values, and we show empirically that the performance is not too sensitive
to them, anyway.

- Specifically applied for image auto-captioning, it pro-
vides excellent results, outperforming highly fine-tuned,
domain-specific methods.

- It is fast, scaling up well with the database size; it can
be made even faster, with clever, off-the-shelf matrix
algebra methods.

Moreover, to the best of our knowledge, this is the first
effort in multimedia databases, that proposes such a graph-
based approach to find patterns and correlations across me-
dia. We were pleasantly surprised that such a domain-
dependent method, with practically no parameters to tune,
managed to outperform some of the most recent and most
carefully tuned methods for image auto-captioning.

Future work could further exploit the promising connection
between multimedia databases and graph algorithms, that
we propose here: Imputation of missing values, outlier de-
tection and any other data mining task that require the
discovery of correlations as its first step.

7. REFERENCES

[1] D. Achlioptas and F. McSherry. Fast computation of
Low-rank approximations. In STOC '01, pages 611–618,


D. B. lei, and M. Jordan. Matching words and
pictures. Journal of Machine Learning Research,

Clustering art. In IEEE Conf. on Computer Vision
and Pattern Recognition, volume 2, pages 434–441,

of words and pictures. In Int. Conf. on Computer

26th Annual International ACM SIGIR Conference,
July 28-August 1, 2003, Toronto, Canada.
<table>
<thead>
<tr>
<th>Term</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>branch</td>
<td>birds</td>
<td>night</td>
<td>owl</td>
<td>nest</td>
<td>hawk</td>
</tr>
<tr>
<td>bridge</td>
<td>water</td>
<td>arch</td>
<td>sky</td>
<td>stone</td>
<td>boats</td>
</tr>
<tr>
<td>cactus</td>
<td>saguaro</td>
<td>desert</td>
<td>sky</td>
<td>grass</td>
<td>sunset</td>
</tr>
<tr>
<td>car</td>
<td>tracks</td>
<td>street</td>
<td>buildings</td>
<td>turn</td>
<td>prototype</td>
</tr>
<tr>
<td>F15</td>
<td>plane</td>
<td>jet</td>
<td>sky</td>
<td>runway</td>
<td>water</td>
</tr>
<tr>
<td>market</td>
<td>people</td>
<td>street</td>
<td>food</td>
<td>closeup</td>
<td>buildings</td>
</tr>
<tr>
<td>mushrooms</td>
<td>fungus</td>
<td>ground</td>
<td>tree</td>
<td>plants</td>
<td>coral</td>
</tr>
<tr>
<td>pillars</td>
<td>stone</td>
<td>temple</td>
<td>people</td>
<td>sculpture</td>
<td>ruins</td>
</tr>
<tr>
<td>reefs</td>
<td>fish</td>
<td>water</td>
<td>ocean</td>
<td>coral</td>
<td>sea</td>
</tr>
<tr>
<td>texture</td>
<td>pattern</td>
<td>background</td>
<td>texture</td>
<td>designs</td>
<td>close-up</td>
</tr>
</tbody>
</table>

Table 3: Semantically similar terms for selected caption terms


