MAGIC: Graph-based Cross Media Correlation Detection

Jia-Yu Pan  Hyungjeong Yang  Christos Faloutsos  Pinar Duygulu
Pittsburgh, U.S.A.  Gwangju, South Korea  Pittsburgh, U.S.A.  Ankara, Turkey

Abstract

Given a collection of multimedia objects consisting of items of different modalities, such as image, video, text, or audio, how can we find correlations between any two or more modalities. We propose a novel, cross-media correlation detection method called MAGIC. The main idea behind MAGIC is to treat the various media as nodes and the relationships between media as edges in a graph. The correlations across media are discovered by the proposed “random walk with restarts” method. MAGIC provides a general framework for discovering cross-media correlations in multimedia applications. It has several desirable properties: (a) it is general and domain-independent; (b) it can spot correlations across any two modalities; (c) it is completely automatic (no human intervention, no parameter tuning); (d) it scales up well for large datasets, and (e) it enables novel multimedia applications (e.g., group captioning).

We evaluate MAGIC on two applications: news video summarization and automatic image captioning. For video summarization, MAGIC provides a uniform framework to deal with the variety of materials available in the video. On image captioning, where ground truth is available, MAGIC outperforms previous methods with respect to captioning accuracy, achieving up to 58% relative improvement over one of the best published methods.

1 Introduction

Advances in digital technology make multimedia information easy to generate and archive. The huge amount of multimedia information creates opportunities to analyze and exploit information from mixed media objects.

A mixed media object has multimedia attributes of different modalities. For example, a video clip can be characterized by its visual attributes, as well as the auditory attributes or the closed caption text which is associated with the clip. Another example of a mixed media object is a captioned image such as news photograph on the web which is associated with a description, or a stock photograph annotated with a few keywords. In this paper, we would use the terms medium (plural form media) and modality interchangeably.
It is common to see correlations among attributes of different modalities of a mixed media object. For instance, a news clip usually has images of more static scenes accompanied with human speech, while a commercial has more dynamic scenes and has loud background music [46]. In image archives, caption keywords are chosen such that they describe objects in the images. Similarly, in digital video libraries and entertainment industry, motion picture directors edit sound effects to match the scenes in video frames. Detecting cross-media correlation provides helpful hints on exploiting information from different modalities for tasks such as segmentation [21] and indexing [8]. Also, establishing associations between low-level features and attributes having semantic meanings may shed light on media understanding.

The question that we are interested in is “*Given a collection of mixed media objects, how do we represent the information of each modality? How do we find the correlations between media? Attributes of mixed media objects tend to be set-valued.* For example, a captioned image (Figure 1(a)) has 2 set-valued attributes: the set of regions in the image and the set of caption words describing the image. Since set-valued attributes include the scalar-valued or vector-valued attributes as special cases, in this study, we developed a graph-based framework to represent set-valued attributes and detect correlations among them.

Moreover, we want to find correlations among all attributes, rather than just between specific attributes. For example, we want to find not just the image-term correlation between an image and caption terms, but also term-term and image-image correlations, using the same framework. *Any-to-any medium correlation* provides a greater picture of how attributes are correlated, e.g., “which word is usually used for images with blue top,” “what words have related semantics,” and “what objects appear often together in an image.”

In this study, we evaluate the proposed MAGIC method on two applications: *automatic image captioning* and *news video summarization*. For image captioning, the correlations between image and text are used to predict caption words for an uncaptioned image. For news event summarization, the correlations between news events and other media (video clips, keyframes, and transcripts) are used to select multimedia materials relevant to a news event. We can formulate these applications as follows:

**Application 1 (Automatic image captioning)** Given a set $\mathcal{I}_{\text{core}}$ of color images, each with caption words; and given an uncaptioned image $\mathcal{I}_{\text{new}}$, find the best $q$ (say, $q=5$) caption words to assign to it.

**Application 2 (Summarizing events in broadcast news)** Given a collection of news video clips, each has one or more (set-valued) multimedia attributes (keyframes, transcripts, event symbols, etc.), identify relevant materials of all media which are pertinent to a user-specified news event.
The proposed method can also be easily extended for various related applications such as captioning images in groups, or retrieving relevant video shots and transcript words.

The main contributions of this paper are as follow: We proposed a novel, domain-independent framework MAGIC which uses a graph to represent the set-valued attributes of a mixed media data set and is capable of detecting cross-media correlations. MAGIC is capable of detecting not just correlations between one particular medium and other media, but any-to-any medium correlations. MAGIC has the following advantages:

- It is domain independent and applicable to mixed media objects which have attributes of various modalities;
- It can spot any-to-any medium correlations;
- It is completely automatic (its few parameters can be automatically preset);
- It can scale up for large collections of objects.

The paper is organized as follows. In Section 2, we discuss previous attempts on multimedia correlation discovery. Section 3 describes our proposed method. Applications in which MAGIC successfully discovers correlations are presented in Sections 4 and 5. In Section 6 we discuss some
system issues and propose a method to further speedup MAGIC by precomputation. Section 7 gives the conclusion remarks.

2 Related work

Multimedia knowledge representation and application have attracted much research attention recently. Mixed media objects provide opportunities for finding correlations between low-level and concept-level features [32, 4, 43], and multi-modal correlations had been shown useful for applications such as retrieval, segmentation, classification, and pattern discovery [8]. In this section, we survey previous work on cross-modal correlation modeling, as well as image captioning and news event summarization, which are our application domains that we evaluate our proposed model.

Multimedia correlation Combining information about multimedia correlations in applications leverages all available information, and has led to improved performances in segmentation [21], classification [34, 10], retrieval [65, 71, 63], and topic detection [67, 14]. One crucial step of fusing multi-modal correlations into applications is to detect, extract and model the correlations from data.

Previous approaches on multimedia correlation modeling employ various techniques such as linear model [32, 59], graphical model [4, 43], statistical model [67, 21, 10], meta-classifier [65, 34], graph partitioning [70, 14], and link analysis [63]. While some of these works proposed general models multimedia correlation modeling, and evaluated the quality of the models by applying to some application domains [32, 4, 43], most of these works designed specific approaches for particular applications (e.g., [67, 65, 70]) which allow them to combine multi-modal information, as well as leverage domain knowledge to boost performance. In this paper, we introduce a general multimedia correlation framework, which is applied to image captioning and video summarization.

Previous work on cross-media correlation modeling attempts to discover correlations between low-level features [32] or mid-level concepts [4, 43]. In [32], a linear model is designed to represent correlations between raw features of different modalities. In contrast, in [4, 43], the multi-modal information is first classified into concepts (e.g., “human” or “explosion”), and then the interaction between concepts are modeled using graphical frameworks [4, 43]. These approaches assume that certain domain knowledge is given, including the specification of the concepts of interest, the basic relations among concepts, etc.

To achieve the best performance, most of the previous studies require domain knowledge on select appropriate parameter values, and may involve a training phase on a labeled training set. For example, one may need to specify in details of concepts (what is a “person”), or fine-tune the mid-level features (which clustering algorithm to use). For classifier-based approaches, decisions about the type and parameters (kernel, etc.) of the classifier have to be made.
Ideas in link analysis have been explored for modeling cross-media correlations. In [63], a similarity function for web images and another one for text blocks are defined initially. The relation between web images and the surrounding text blocks (cross-media links) are then utilized to adjust the similarity distance between images for better performance on image retrieval. The initial similarity functions between objects (web images, text blocks), which may be complicated and difficult to obtain.

Statistical modeling of cross-modal correlation usually requires a training phase on a labeled training set. Preparing a training set where cross-modal associations are fully labeled is not easy. Moreover, the statistical model may be complex (with many parameters to be trained) and be computationally costly to train.

Our proposed framework, MAGIC, does not need a training phase, and has fewer parameters to tune. In fact, as we show later, the results are insensitive to parameter values in our experiments (Section 6.1). MAGIC uses a graph to represent the relations between objects and low-level attribute values. By relating multimedia objects via the constituent single-modal domain tokens, MAGIC avoid detailed specifications of concepts or complicate similarity functions.

**Image captioning** Although a picture is worth a thousand words, extracting the abundant information from an image is not an easy task. Computational techniques are able to derive low-to-mid level features (e.g., texture and shape) from pixel information, however, the gap still exists between mid-level features to concepts used by human reasoning [55, 72, 71]. One consequence of this semantic gap in image retrieval is that the user’s need is not properly matched by the retrieved images, and may be part of the reason that practical image retrieval is yet to be popular.

Automatic image captioning, where the goal is to predict caption words to describe image content, is one research direction to bridge the gap between concepts and low-level features. Previous work on image captioning employs various approaches such as linear models [49, 42], classifiers [39], language models [61, 13, 25], graphical models [3, 6], statistical models [33, 26, 17], and a framework with user involvement [64].

Most previous approaches derive features from image regions (regular grids or blobs [13]), and construct a model between images and words based on a reference captioned image set. Images in the reference set are captioned by human experts, however, there is no information of the associations between individual regions and words. Some approaches attempt to explicitly infer the correlations between regions and words [13], with enhancements that take into consideration interactions between neighboring regions in an image [33]. Alternatively, there are methods which model the collective correlations between regions and words of an image [50, 51].

Comparing the performance of different approaches is not easy. Several benchmark data sets are available, however, not all previous work reports results on the same subset of images. On the other hand, various metrics such as accuracy, term precision and recall, and mean average precision
have been used to measure the performance. Since the perception of an image is subjective, some work also reports user evaluation of the captioning result.

In Section 4, our proposed method, MAGIC, is applied to automatic image captioning. The correlations between words and images are detected and applied to predict caption words of a previously unseen image. To better evaluate our approach, we conduct experiments on the same data sets and report using the same performance metric which are also used in other previous works [61, 13, 3, 6].

**Broadcast news event summarization** Video summarization is in great demands to enable users to efficiently access massive video collections [62]. In this study, we apply our proposed method to summarize events in broadcast news videos, by detecting cross-modal correlations between video frames, shots and transcript words.

Analyzing *events* in video has attracted much attention recently. Unlike creating a condensed version of a video clip for browsing [58, 44], the goal of event summarization is to identify and gather pieces of video shots which are related to one event. Some work focuses on using only the visual information to detect events such as simple motions (“walking” and “waving” [69, 31]) or repeating/periodic scenes [35]. On the other hand, when incorporated with information from multiple media, it is possible to detect events at a higher semantic level [40, 9, 22]. Similar research directions have also been pursued in textual domain, where the focus is to summarize multiple documents of the same event [18, 5].

To visualize the summary of a video collection, the video collage method [9] arranges frames as a storyboard [68, 60] and shows them alongside the corresponding transcripts or other meta-data. A similar presentation is adopted in our experiment to present our result on summarizing broadcast news events (Section 5).

3 Proposed method: graph-based correlation detection model

Our proposed mixed media correlation discovery method employs a graph-based representation for objects with set-valued attributes, and a technique for finding any-to-any medium correlation, which is based on random walks. In this section, we explain how to generate the graph representation and how to detect cross-media correlations.

3.1 MAGIC graph ($G_{MAGIC}$)

In relational database management systems, a multimedia object is usually represented as a vector of $m$ features/attributes [16]. The attributes must be atomic (i.e., taking single values) like “size” or “the amount of red color” of an image. However, for mixed media data sets, the attributes can be set-valued, such as the caption of an image (a set of words) or the image regions.
Finding correlations among set-valued attributes is not easy: Elements in a set-valued attribute could be noisy or missing altogether: regions in an image are not perfectly identified (noisy regions); the image caption may be incomplete, leaving out some aspects of the content. Set-valued attributes of an object may have different numbers of elements, and there is no given alignment between set elements. For instance, an image may have unequal numbers of caption words and regions, where a word may describe multiple regions and a region may be described by zero or more than one word.

We assume that the elements of a set-valued attribute are tokens drawn from a domain. We propose to gear our method toward set-valued attributes, because they include atomic attributes as a special case; and they also smoothly handle the case of missing values (null set).

**Definition 1 (Domain and domain token)** The domain $D_i$ of (set-valued) attribute $i$ is a collection of atomic values, which we called domain tokens, which are the values that attribute $i$ can take.

A domain can consist of categorical values, numerical values, or numerical vectors. For example, for automatic image captioning, we have objects with $m=2$ attributes. The first attribute, “caption”, has a set of categorical values (English terms) as its domain; the second attribute, “regions”, is a set of image regions, each of which is represented by a $p$-dimensional vector of $p$ features derived from the region (e.g., color histogram with $p$ colors). As described later in Section 4, we extract $p=30$ features from each region. To establish the relation between domain tokens, we assume that we have a similarity function for each domain. Domain tokens are usually simpler than mixed media objects, and therefore, it is easier to define similarity functions on domain tokens than on mixed media objects.

**Assumption 1** For each domain $D_i$ ($i = 1, \ldots, m$), we are given a similarity function $\text{Sim}_i(\ast, \ast)$ which assigns a score to a pair of domain tokens.

For example, for the attribute “caption”, the similarity function could be 1 if the two tokens are identical, and 0 if they are not.

Perhaps surprisingly, with Definition 1 and Assumption 1, we can encompass all the applications mentioned in Section 1. The main idea is to represent all objects and their attributes (domain tokens) as nodes of a graph. For multimedia objects with $m$ attributes, we obtain a $(m + 1)$-layer graph. There are $m$ types of nodes (one for each attribute), and one more type of nodes for the objects. We call this graph a MAGIC graph ($G_{\text{MAGIC}}$). We put an edge between every object-node and its corresponding attribute-value nodes. We call these edges object-attribute-value links (OAV-links).

Furthermore, we consider that two objects are similar if they have similar attribute values. For example, two images are similar if they contain similar regions. To incorporate such information
Figure 2: MAGIC graph \((G_{MAGIC})\) corresponds to the 3 images in Figure 1. Solid edges: OAV-links; dash edges: NN-links.

into the graph, our approach is to add edges to connect pairs of domain tokens (attribute values) that are similar, according to the given similarity function (Assumption 1). We call edges that connect nodes of similar domain tokens nearest-neighbor links (NN-links).

We need to decide on a threshold for "closeness" when adding NN-links. There are many ways to do this, but we decide to make the threshold adaptive: each domain token is connected to its \(k\) nearest neighbors. We discuss the choice of \(k\) in Section 6.1, as well as the sensitivity of our results to \(k\). Computing nearest neighbors can be done efficiently, because we already have the similarity function \(Sim_i(\ast, \ast)\) for any domain \(D_i\) (Assumption 1).

We illustrate the construction of \(G_{MAGIC}\) graph by the following example.

**Example 1** For the images \(\{I_1, I_2, I_3\}\) in Figure 1, the MAGIC graph \((G_{MAGIC})\) corresponding to these images is shown in Figure 2. The graph has three types of nodes: one for the image objects \(I_j\)'s \((j = 1, 2, 3)\); one for the regions \(r_j\)'s \((j = 1, \ldots, 11)\), and one for the terms \(\{t_1, \ldots, t_8\} = \{\text{sea, sun, sky, waves, cat, forest, grass, tiger}\}\). Solid arcs are the object-attribute-value links (OAV-links), and dashed arcs are the nearest-neighbor links (NN-links).

In Example 1, we consider only \(k=1\) nearest neighbor, to avoid cluttering the diagram. Because the nearest neighbor relationship is not symmetric and because we treat the NN-links as un-directional, some nodes are attached to more than one link. For example, node \(r_1\) has two NN-links attached: \(r_2\)'s nearest neighbor is \(r_1\), but \(r_1\)'s nearest neighbor is \(r_6\). There is no NN-link between term-nodes, due to the definition of its similarity function: 1, if the two terms are the same; or 0 otherwise. Figure 3 shows the algorithm for constructing a MAGIC graph.

We use image captioning only as an illustration: the same framework can be generally used for other problems. To solve the automatic image captioning problem, we also need to develop a method to find good caption words - words that correlate with an image, using the \(G_{MAGIC}\) graph. This means that, for example, for image \(I_3\), we need to estimate the affinity of each term (nodes \(t_1, \ldots, t_8\)) to node \(I_3\). The terms with the highest affinity to image \(I_3\) will be predicted as its caption.
Input:
1. $O$: a set of $n$ objects (objects are numbered from 1 to $n$).
2. $D_1, \ldots, D_m$: the domains of the $m$ attributes of the objects in $O$.
3. $Sim_1(\ast, \ast), \ldots, Sim_m(\ast, \ast)$: the similarity functions of domains $D_1, \ldots, D_m$, respectively.
4. $k$: the number of neighbors a domain token connects to.

Output:
$G_{MAGIC}$: a MAGIC graph with a $(m+1)$-layer structure.

Steps:
1. Create $n$ nodes (the object nodes), one for each object. These nodes form the layer 1.
2. For each domain $D_i$, for $i = 1, \ldots, m$.
   (2.1) Let $n_i$ be the number of tokens in the domain $D_i$.
   (2.2) Create $n_i$ nodes (the token nodes), one for each domain tokens in $D_i$. This is the $(i + 1)$-th layer.
   (2.3) Construct the OAV-links from the object nodes to these token nodes.
   (2.4) Construct the NN-links between the token nodes.
3. Output the final $(m+1)$-layer graph, with $N = n + \sum_{i=1}^{m} n_i$ nodes, and the OAV-links and NN-links.

Figure 3: Algorithm: $G_{MAGIC} = \text{buildgraph}(O, \{D_1, \ldots, D_m\}, \{Sim_1(\ast, \ast), \ldots, Sim_m(\ast, \ast)\}, k)$

words.

3.2 Correlation detection with random walks

Our main contribution is to turn the cross-media correlation discovery problem into a graph problem. The previous section describes the first step of our proposed method: representing set-valued mixed media objects in a graph $G_{MAGIC}$. Given such a graph with mixed media information, how do we detect the cross-modal correlations in the graph?

We define that a node $A$ of $G_{MAGIC}$ is correlated to another node $B$ if $A$ has an “affinity” for $B$. There are many approaches for ranking all nodes in a graph by their “affinity” for a reference node. We can tap the sizable literature of graph algorithms and use off-the-shelf methods for assigning importance to vertices in a graph. These include the electricity based approaches [45, 12], random walks (PageRank, topic-sensitive PageRank) [7, 20], hubs and authorities [30], elastic springs [38] and so on. Among them, we propose to use random walk with restarts (RWR) for estimating the affinity of node $B$ with respect to node $A$. However, the specific choice of method is orthogonal to our framework.

The “random walk with restarts” operates as follows: To compute the affinity $u_A(B)$ of node
Input:
1. $G_{MAGIC}$: a MAGIC graph with $N$ nodes (nodes are numbered from 1 to $N$).
2. $\mathcal{R}$: a set of restart nodes. (Let $|\mathcal{R}|$ be the size of $\mathcal{R}$.)
3. $c$: the restart probability.

Output:
$\tilde{u}_\mathcal{R}$: the RWR scores of all nodes with respect to $\mathcal{R}$

Steps:
1. Let $A$ be the adjacency matrix of $G_{MAGIC}$. Normalize the columns of $A$ and make each column sum up to 1.
2. $\tilde{v}_R$ is the $N$-by-1 restart vector, whose $i$-th element $\tilde{v}_R(i)$ is $\frac{1}{|\mathcal{R}|}$, if node $i$ is in $\mathcal{R}$; otherwise, $\tilde{v}_R(i)=0$.
3. Initialize $\tilde{u}_\mathcal{R}=\tilde{v}_\mathcal{R}$.
4. while($\tilde{u}_\mathcal{R}$ has not converged)
   4.1 Update $\tilde{u}_\mathcal{R}$ by $\tilde{u}_\mathcal{R} = (1-c)A\tilde{u}_\mathcal{R} + c\tilde{v}_\mathcal{R}$
5. Return the converged $\tilde{u}_\mathcal{R}$.

Figure 4: Algorithm: $\tilde{u}_\mathcal{R} = \text{RWR}(G_{MAGIC}, \mathcal{R}, c)$

B for node $A$, consider a random walker that starts from node $A$. The random walker chooses randomly among the available edges every time, except that, before he makes a choice, he goes back to node $A$ (restart) with probability $c$. 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 affinity of $B$ with respect to $A$. We also call $u_A(B)$ the RWR score of $B$ with respect to $A$. The algorithm of computing RWR scores of all nodes with respect to a subset of nodes $\mathcal{R}$ is given in Figure 4.

Definition 2 (RWR score) The RWR score, $u_A(B)$, of node $B$ with respect to node $A$ is the steady state probability of node $B$, when we do the random walk with restarts from $A$, as defined above.

The computation of RWR scores can be done efficiently by matrix multiplication. Let $A$ be the adjacency matrix of the given graph $G_{MAGIC}$, where columns of the matrix are normalized such that each sums up to 1. Let $\tilde{u}_q$ be a vector of RWR scores of all $N$ nodes, with respect to a restart node $q$. Let $\tilde{v}_q$ be the “restart vector”, which has all $N$ elements zero, except for the entry that corresponds to node $q$, which is set to 1. We can now formalize the definition of RWR scores (Definition 3).

Definition 3 (RWR scores computation) The $N$-by-1 steady state probability vector $\tilde{u}_q$, which
Step 1: Identify the objects $O$ and the $m$ attribute domains $D_i, i = 1, \ldots, m$.

Step 2: Identify the similarity functions $Sim_i(*, *)$ of each domain.

Step 3: Determine $k$: the neighborhood size of the domain tokens. (Default value $k = 3$.)

Step 4: Build the MAGIC graph,

$$G_{MAGIC} = \text{buildgraph}(O, \{D_1, \ldots, D_m\}, \{Sim_1(*, *), \ldots, Sim_m(*, *)\}, k).$$

Step 5: Given a query node $R = \{q\}$ ($q$ could be an object or a token),

(Step 5.1) Determine the restart probability $c$. (Default value $c = 0.65$.)

(Step 5.2) Compute the RWR scores:

$$\tilde{u}_R = \text{RWR}(G_{MAGIC}, R, c).$$

Step 6: Objects and attribute tokens with high RWR scores are correlated with $q$.

Figure 5: Instructions for detecting correlations using MAGIC. Functions “buildgraph()” and “RWR()” are given in Figures 3 and 4, respectively.

contains the RWR scores of all nodes with respect to node $q$, satisfies the equation:

$$\tilde{u}_q = (1 - c)A\tilde{u}_q + c\tilde{v}_q,$$

where $c$ is the restart probability of the RWR from node $q$.

The RWR scores specify the correlations across different media and could be useful in many multimedia applications. For example, to solve the image captioning problem for image $I_3$ in Figure 1, we can compute the RWR scores $\tilde{u}_{I_3}$ of all nodes and report the top few (say, 5) term-nodes as caption words for image $I_3$. Effectively, MAGIC exploits the correlations across images, regions and terms to caption a new image.

The RWR scores also provide a mean for detecting any-to-any medium correlation. In our running example of image captioning, an image is captioned with the term nodes of highest RWR scores. In addition, since all nodes have their RWR scores, other nodes, say image nodes, can also be ranked and sorted, for finding images that are most related to image $I_3$. Similarly, we can find the most relevant regions. In short, we can restart from any subset of nodes, say term nodes, and derive term-to-term, term-to-image, or term-to-any correlations.

MAGIC turns the correlation detection problem into a graph problem, and treats all media equally as nodes in a graph. By restarting the random walk from nodes of any modality, and assigning scores to all nodes of any modality, MAGIC enables any-to-any medium correlation detections. We discuss more on this in Section 4.3.

Figure 5 shows the overall procedure of using MAGIC for correlation detection. Table 1 summarizes the symbols we used in the paper.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>The number of objects in a mixed media data set.</td>
</tr>
<tr>
<td>$m$</td>
<td>The number of attributes (domains).</td>
</tr>
<tr>
<td>$N$</td>
<td>The number of nodes in $G_{MAGIC}$.</td>
</tr>
<tr>
<td>$E$</td>
<td>The number of edges in $G_{MAGIC}$.</td>
</tr>
<tr>
<td>$k$</td>
<td>Domain neighborhood size: the number of nearest neighbors that a domain token is connected to.</td>
</tr>
<tr>
<td>$c$</td>
<td>The restart probability of RWR (random walk with restarts, RWR).</td>
</tr>
<tr>
<td>$D_i$</td>
<td>The domain of the $i$-th attribute.</td>
</tr>
<tr>
<td>$\text{Sim}_i(*)$</td>
<td>The similarity function of the $i$-th domain.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sizes</th>
</tr>
</thead>
</table>

| $I_{core}$ | The given captioned image set (the core image set). |
| $I_{test}$ | The set of to-be-captioned (test) images. |
| $I_{new}$  | An image in $I_{test}$. |
| $G_{core}$ | The subgraph of $G_{MAGIC}$ containing all images in $I_{core}$ (Section 4). |
| $G_{aug}$  | The augmentation to $G_{core}$ containing information of an image $I_{new}$ (Section 4). |
| $\mathcal{G}_W$ | The gateway nodes, nodes in $G_{core}$ that adjacent to $G_{aug}$ (Section 4). |

<table>
<thead>
<tr>
<th>Image captioning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
</tr>
<tr>
<td>$\vec{v}_R$</td>
</tr>
<tr>
<td>$\vec{u}_R$</td>
</tr>
<tr>
<td>$\vec{v}_q, \vec{u}_q$</td>
</tr>
<tr>
<td>$\vec{v}<em>{GW}, \vec{u}</em>{GW}$</td>
</tr>
</tbody>
</table>

Table 1: Summary of symbols used in the paper

4 Application: Automatic image captioning

Cross-media correlations are useful for many multimedia applications. In this section and the next, we present results of applying the proposed MAGIC method to two applications - image captioning [50, 51] and broadcast news events summarization [48, 47]. Intuitively, the cross-media correlations are used in the way that an image will be captioned with words correlated with the image content, and an video news event will be summarized by correlated multimedia materials (video shots, transcript terms, and other event symbols).

On captioning images, we evaluate the quality of the cross-modal correlations by MAGIC in terms of the captioning accuracy. We show experimental results to address the following questions:

- Quality: Does MAGIC predict the correct caption terms?
- Generality: Beside the image-to-term correlation for captioning, does MAGIC capture any-to-any medium correlations?
Our results show that MAGIC successfully exploits the image-to-term correlation to caption test images. Moreover, MAGIC is flexible and can caption multiple images as a group. We call this operation “group captioning” and present some qualitative results.

We also examine MAGIC’s performance on spotting other cross-media correlations. In particular, we show that MAGIC can capture same-media correlations such as the term-term correlations: E.g., “given a term such as ‘sky’, find other terms that are likely to correspond to it.” Potentially, MAGIC is also capable of spotting other correlations such as the reverse captioning problem: E.g., “given a term such as ‘sky’, find the regions that are likely to correspond to it.” In general, MAGIC can capture any-to-any medium correlations.

4.1 Data set and $G_{MAGIC}$ graph construction

Given a collection of captioned images $I_{core}$, how do we select caption words for an uncaptioned image $I_{new}$? For automatic image captioning, we propose to caption $I_{new}$ using the correlations between caption words and images in $I_{core}$.

In our experiments, we use the same 10 sets of images from Corel that are also used in previous work [13, 3], so that our results can be compared to the previous results. In the following, the 10 captioned image sets are referred to as the “001”, “002”, ..., “010” sets. Each of the 10 data sets has around 5,200 images, and each image has about 4 caption words. These images are also called the core images from which we try to detect the correlations. For evaluation, accompanying each data set, a non-overlapping test set $I_{test}$ of around 1,750 images is used for testing the captioning performance. Each test image has its ground truth caption.

Similar to previous work [13, 3], each image is represented by a set of image regions. Image regions are extracted using a standard segmentation tool [57], and each region is represented as a 30-D feature vector. The regional features include 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). The image content is represented as a set-valued attribute “regions”. In our experiments, an image has 10 regions on average. Figure 1(d,e,f) show some examples of image regions.

The exact region segmentation and feature extraction details are orthogonal to our approach - any published segmentation methods and feature extraction functions [16] will suffice. All our MAGIC method needs is a black box that will map each color image into a set of zero or more feature vectors.

We want to stress that there is no given information about which region is associated with which term in the core image set - all we know is that a set of regions co-occurs with a set of terms in an image. That is, no alignment information between individual regions and terms is available.

Therefore, a captioned image becomes an object with two set-valued attributes: “regions” and
“terms”. Since the regions and terms of an image are correlated, we propose to use MAGIC to detect this correlation and use it to predict the missing caption terms correlated with the uncaptioned test images.

The first step of MAGIC is to construct the MAGIC graph. Following the instructions for graph construction in Section 3.1, the graph for captioned images with attributes “regions” and “terms” will be a 3-layer graph with nodes for images, regions and terms. To form the NN-links, we define the distance function (Assumption 1) between two regions (tokens) as the $L_2$ norm between their feature vectors. Also, we define that two terms are similar if and only if they are identical, i.e., no term is any other’s neighbor. As a result, there is no NN-link between term nodes.

For results shown in this section, the number of nearest neighbors between attribute/domain tokens is $k=3$. However, as we will show later in Section 6.1, the captioning accuracy is insensitive to the choice of $k$. In total, each data set has about 50,000 different region tokens and 160 words, resulting in a graph $G_{MAGIC}$ with about 55,500 nodes and 180,000 edges. The graph based on the core image set $I_{core}$ captures the correlations between regions and terms. We call such graph the “core” graph.

How do we caption a new image, using the information in a MAGIC graph? Similar to the core images, an uncaptioned image $I_{new}$ is also an object with set-valued attributes: “regions” and “caption”, where attribute “caption” has null value. To find caption words correlated with image $I_{new}$, we propose to look at regions in the core image set that are similar to the regions of $I_{new}$, and find the words that are correlated with these core image regions. Therefore, our algorithm has two main steps: finding similar regions in the core image set (augmentation) and identifying caption words (RWR). Next, we define “core graph”, “augmentation” and “gateway nodes”, to facilitate the description of our algorithm.

**Definition 4 (Core graph, augmentation and gateway nodes)** For automatic image captioning, we define the core of the $G_{MAGIC}$, $G_{core}$, be the subgraph that constitutes information in the given captioned images $I_{core}$. The graph $G_{MAGIC}$ for captioning a test image $I_{new}$ is an augmented graph, which is the core $G_{core}$ augmented with the region-nodes and image-node of $I_{new}$. The augmentation subgraph is denoted as $G_{aug}$, and hence the overall $G_{MAGIC}=G_{core} \cup G_{aug}$. The nodes in the core subgraph $G_{core}$ that are adjacent to the augmentation are called the gateway nodes, $GW$.

As an illustration, Figure 2 shows the graph $G_{MAGIC}$ for two core (captioned) images $I_{core}=$\{I$_1$, I$_2$\} and one test (to-be-captioned) image $I_{test}=$\{I$_3$\}, with the parameter for NN-links $k=1$. The core subgraph $G_{core}$ contains region nodes \{r$_1$, $\ldots$, r$_7$\}, image nodes \{I$_1$, I$_2$\}, and all the term nodes \{t$_1$, $\ldots$, t$_8$\}. The augmentation $G_{aug}$ contains region nodes \{r$_8$, $\ldots$, r$_{11}$\} and the image node \{I$_3$\} of the test image. The gateway nodes are the region nodes $GW=$\{r$_5$, r$_6$, r$_7$\} that bridge the $G_{core}$ and $G_{aug}$. 

14
Input: 1. The core graph $G_{core}$, an image $I_{new}$ to be captioned, and
   2. $g$, the number of caption words we want to predict for $I_{new}$.

Output: Predicted caption words for $I_{new}$.

Steps:
1. Augment the image node and region nodes of $I_{new}$ to the core graph $G_{core}$.
2. Do RWR from the image node of $I_{new}$ on the augmented graph $G_{MAGIC}$ (Algorithm 4).
3. Rank all term nodes by their RWR scores.
4. The $g$ top-ranked terms will be the output - the predicted caption for $I_{new}$.

Figure 6: Steps to caption an image, using the proposed MAGIC framework.

Different test images have different augmented graphs and gateway nodes. However, since we will caption only one test image at a time, the symbols $G_{aug}$ and $GW$ represent for the augmented graph and gateway nodes of the test image in question.

The first step of our image captioning algorithm, augmentation, can be done by finding the gateway nodes - the collection of the $k$ nearest neighbors of each region of $I_{new}$. In the second step, we propose to use RWR, restarting from the test image-node, to identify the correlated words (term-nodes). A predicted caption of $g$ words for the image $I_{new}$ will correspond to the $g$ term-nodes with highest RWR scores. Figure 6 gives the details of our algorithm.

To sum up, for image captioning, the core of the $G_{MAGIC}$ is first constructed based on the given captioned images $I_{core}$. Then, each test image $I_{new}$ is captioned, one by one, in steps summarized in Figure 6.

4.2 Captioning accuracy

We measure captioning performance by the captioning accuracy, which is defined as the fraction of terms which are correctly predicted. Following the same evaluation procedure as that in previous work [13, 3], for a test image which has $g$ ground-truth caption terms, MAGIC will also predict $g$ terms. If $p$ of the predicted terms are correct, then the captioning accuracy $acc$ on this test image is defined as

$$acc = \frac{p}{g}.$$  

The average captioning accuracy $\overline{acc}$ on a set of $T$ test images is defined as

$$\overline{acc} = \frac{1}{T} \sum_{i=1}^{T} acc_i,$$

where $acc_i$ is the captioning accuracy on the $i$-th test image.
Figure 7: Comparing MAGIC to the EM method. The parameters for MAGIC are $c = 0.66$ and $k = 3$. The x-axis shows the 10 data sets, and the y-axis is the average captioning accuracy over all test images in a set.

Figure 7 shows the average captioning accuracy on the 10 image sets. We compare our results with those reported in [13]. The method in [13] is one of the most recent and sophisticated: it models the image captioning problem as a statistical translation modeling problem and solves it using expectation-maximization (EM). We refer to their method as the “EM” approach. The x-axis groups the performance numbers of MAGIC (white bars) and EM (black bars) on the 10 data sets. On average, MAGIC achieves captioning accuracy improvement of 12.9 percentage points over the EM approach, which corresponds to a relative improvement of 58%.

We also compare the captioning accuracy with even more recent machine vision methods [3], on the same data sets: the Hierarchical Aspect Models method (“HAM”), and the Latent Dirichlet Allocation model (“LDA”). Figure 8 compares MAGIC with LDA and HAM, in terms of the mean and variance of the average captioning accuracy over the 10 data sets. Although both HAM and LDA improve on the EM method, they both lose to our generic MAGIC approach (35%, versus 29% and 25%). It is also interesting that MAGIC gives significantly lower variance, by roughly an order of magnitude: 0.002 versus 0.02 and 0.03. A lower variance indicates that the proposed MAGIC method is more robust to variations among different data sets.

Figure 9 shows some examples of the captions given by MAGIC. For the test image $I_3$ in Figure 1, MAGIC captions it correctly (Figure 9(a)). In Figure 9(b), MAGIC surprisingly gets the word “mane” correctly; however, it mixes up “buildings” with “tree” (Figure 9(c)).
Figure 8: Comparing MAGIC with LDA and HAM. The mean and variance of the average accuracy over the 10 Corel data sets are shown at the y-axis - LDA: $(\mu, \sigma^2) = (0.24, 0.002)$; HAM: $(\mu, \sigma^2) = (0.298, 0.003)$; MAGIC: $(\mu, \sigma^2) = (0.3503, 0.0002)$. $\mu$: mean average accuracy. $\sigma^2$: variance of average accuracy. The length of the error bars at the top of each bar is $2\sigma$.

![Image](image_url)

<table>
<thead>
<tr>
<th></th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truth</td>
<td>cat, grass, mane, cat, sun, water, tiger, water</td>
<td>lion, grass tree, sky</td>
<td></td>
</tr>
<tr>
<td>MAGIC</td>
<td>grass, cat, tiger, water</td>
<td>lion, grass tree, water, cat, mane</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>buildings, sky</td>
</tr>
</tbody>
</table>

Figure 9: Terms are ordered by their given importance. Figures look best in color.

### 4.3 Generalization

MAGIC treats information from all media uniformly as nodes in a graph. Since all nodes are basically the same, we can do RWR and restart from any subset of nodes of any medium, to detect any-to-any medium correlations. The flexibility of our graph-based framework also enables novel applications, such as captioning images in groups (group captioning). In this subsection, we show results on (a) spotting the term-to-term correlation in image captioning data sets, and (b) group captioning.
Beyond image-to-term correlation  MAGIC successfully exploits the image-to-term correlation for captioning images. However, the MAGIC graph $G_{MAGIC}$ contains correlations between all media (image, region, and term). To show how well MAGIC works on objects of any medium, we design an experiment to identify correlated captioning terms, using the term-to-term correlation in the graph $G_{MAGIC}$.

We use the same 3-layer MAGIC core graph $G_{core}$ that was constructed in the previous subsection for automatic image captioning. Given a query term $t$, we use RWR to find other terms correlated with it. Specifically, we perform RWR, restarting from the query term(-node). The terms deemed correlated with the query term are term(-node)s that receive high RWR scores.

Table 2 shows the top 5 terms with the highest RWR scores for some query terms. In the table, each row shows the query term at the first column, followed by the top 5 correlated terms selected by MAGIC (sorted by their RWR scores). The selected terms make a lot of sense, and have meanings related with the query term. For example, the term “branch”, when used in image captions, is strongly related to forest- or bird- related concepts. MAGIC shows exactly this, correlating “branch” with terms such as “birds”, “owl” and “nest”.

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” are more frequent than the others (like the terms “the” and “a” in normal English text). Yet, these frequent terms do not show up too often in Table 2, as a correlated term of a query term. It is surprising, given that we did nothing special when using MAGIC: no tf/idf weighting, no normalization, and no other domain-specific analysis. We just treated these frequent terms as nodes in our MAGIC graph, like any other nodes.

Group captioning  The proposed MAGIC method can be easily extended to caption a group of images, considering all of them at once. This flexibility is due to the graph-based framework of
MAGIC, which allows augmentation of multiple nodes and doing RWR from any subset of nodes. To the best of our knowledge, MAGIC is the first method that is capable of doing group captioning.

**Problem 1 (Group captioning)** Given a set $I_{\text{core}}$ of color images, each with caption words; and given a (query) group of uncaptioned images $\{I_1, \ldots, I_t\}$, find the best $g$ (say, $g=5$) caption words to assign to the group.

Possible applications for group captioning include video segment captioning, where a video segment is captioned according to the group of keyframes associated with the segment. Since keyframes in a segment are related, captioning them as a whole can take into account the inter-keyframe correlations, which are missed if each keyframe is captioned separately. Accurate captions for video segments may improve performances on tasks such as video retrieval and classification.

The steps to caption a group of images are similar to those for the single-image captioning outlined in Figure 6. A core MAGIC graph is still used to capture the mixed media information of the given collection of images. The differences for group captioning are, instead of augmenting the single-image to the core and restarting from it, now we augment all $t$ images in the query group $\{I_1, \ldots, I_t\}$ to the core, and restarts randomly from one of the images in the group (i.e., each with probability $\frac{1}{t}$ to be the restart node).

Figure 10 shows the result of using MAGIC for captioning a group of three images. MAGIC found reasonable terms for the entire group of images: “sky”, “water”, “tree”, and “sun”. Captioning multiple images as a group takes into consideration the correlations between different images in the group, and in this example, this helps reduce the scores of irrelevant terms such as “people”.

In contrast, when we caption these images individually, MAGIC selects “people” as caption words for images in Figure 10(a) and (b), which do not contain people-related objects.

5 Application: Broadcast news event summarization

As more and more digital video libraries [62] become available, video summarization is in great demands for accessing these video collections efficiently. A video clip is essentially a mixed media object, containing video and audio streams, transcripts, and objects extracted from keyframes such as people’s face or overlaid text. In order to facilitate browsing and utilizing video clips in a large digital library that contains thousands of hours of video clips, it is desirable to have summaries of video clips.

In particular for broadcast news video, summaries for individual news events could help users gain an overall view picture of an event: “how did it start?” and “what were the following developments?” [40, 9, 22]. Usually, a news event is covered by news stories (or shots) scattered in daily broadcasts during a period of time. A summary of a news event, say “winter olympics”, may consist of scene shots of various sport events, the names of relevant locations and players, as well as relevant
topics such as “the tourism industry of the hosting country”. Identifying event-relevant materials from different media in the video and collecting them together as a summary of an event would benefit users and find applications in areas such as education, decision making, and entertainment, etc.

Correlations across media in video provide strong clues for collecting summarization materials. News channels compile news shots by showing footages (images) correlated with the story the anchorperson is reporting (audio/text). There are also correlations among video shots of a continuing event - same scenes or symbols are repeatedly used in these correlated shots. Information about such correlations would facilitate gathering of information relevant to an event.

MAGIC provides a graph-based framework to find correlations among video shots, transcripts, and news events. In this section, we apply MAGIC to the problem of video summarization, and design experiments to examine two issues: (a) what is the quality of summaries by MAGIC? And, (b) what advantages does MAGIC provide over traditional retrieval techniques?

5.1 Event (logo) identification and problem formulation

A news video contains information in different media such as shots and transcripts. Besides shots and transcripts, in order to do event summarization, we need to identify the specific event of each shot, which is not a easy task. Particular difficulties include the definition of events and the identification of video shots of the same event.

Daily news reports usually consist of shots about different events, where these shots have various durations and are covered in different orders from day to day. Broadcast news programs usually

<table>
<thead>
<tr>
<th></th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truth</td>
<td>sun, water, tree, sky</td>
<td>sun, clouds, sky, horizon</td>
<td>sun, water</td>
</tr>
<tr>
<td>MAGIC</td>
<td>tree, people, sky, water</td>
<td>water, tree, people, sky</td>
<td>sky, sun</td>
</tr>
<tr>
<td>Group</td>
<td>sky, water, tree, sun</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 10: Group captioning: Captioning terms with highest RWR scores are listed first.
Figure 11: Three logo events, their keyframes (containing logos) and nouns in the transcripts.

In this study, the logos are used as identifiers of news events. The events that have corresponding logos are called logo events, and these are the events that will be considered in this work. Therefore, the formulation of our problem becomes: Given the shots, transcripts and (event) logos from video clips, construct a summary for a (query) logo event.

5.2 Data set and $G_{MAGIC}$ graph construction

We conduct our experiments on the TRECVID 2003 data set, which contains 115 news clips (46 giga bytes) from a U.S. news source. We process video clips to extract the shots, transcripts and logos: Each news clip is segmented into individual video shots using an off-the-shelf algorithm [53]. A keyframe is extracted from each shot, and the logo is extracted from the keyframe if there exists one. We call a shot a logo shot, if its keyframe contains a logo (we call such keyframes logo frames). The transcript words mentioned in a shot are also collected, but we use only nouns in our experiments. Since there is a one-to-one correspondence between a shot and its keyframe, in the
following discussions, we will use a shot’ keyframe to visualize (in figures) the content of the shot, and will use the words “shot” and “frame” interchangeably.

The data set becomes a collection of shots (objects) that each associates with zero or one logo, and a set of transcript words (nouns). In other words, a shot is an object, with two attributes: “logo” and “transcript”. The “logo” attribute has categorical values, however, many shots have this value missing. The “transcript” attribute is a set-valued attribute.

Although there are missing values and set-valued attributes, our proposed MAGIC method has no problem in representing these video shots by using our $G_{MAGIC}$ graph. Following the instructions in Section 3.1, the graph for this video data set contains three types of nodes: shot-nodes ($s_i$) for the shot objects, while logo-nodes ($l_i$) and term-nodes ($t_i$) are for the domain tokens. Figure 12 illustrates how the graph looks like on a set of 5 shots, 2 logos and 10 transcript nouns. The full graph of our data set contains 27,809 nodes (17,528 shot-nodes, 80 logo-nodes and 10,201 term-nodes) and 69,710 edges.

We also need to define the similarity functions between domain tokens (Assumption 1). In this study, we consider every logo or transcript noun to be unique. In other words, the similarity function between two “logo” tokens are defined as: 1 if the logos are identical, and 0 otherwise. The similarity function between two transcript nouns is defined similarly. Consequently, on the graph $G_{MAGIC}$, neither a “logo” token, nor a “transcript” token has neighbors - the graph only has OAV-links, but no NN-links (Subsection 3.1).

5.3 Cross-media correlations for event summarization

Having identified the shots, transcripts and (event) logos from video clips, we can now construct a summary for a (query) logo event. The idea is to collect shots, transcripts, and logos that are correlated with the query logo event as the event’s summary.

We propose to use MAGIC and the graph $G_{MAGIC}$ that contains information of shots, tran-
scripts and logos (Figure 12) to approach this event summarization problem. To find the shots and terms correlated with the query logo event, we perform the random walk with restarts (RWR) from the corresponding query-logo-node. The shots and terms whose corresponding nodes have the highest RWR scores are selected for the summary of the query event.

We examine the contents of shots and terms selected by MAGIC, and evaluate if they are relevant to the query event. Figure 13 shows the top 20 shots selected by MAGIC for the logo event “Iraq”. We use the keyframe of a shot to present the content in a video shot. The top 7 shots correspond to logo shots which are directly connected to the query-logo-node. These are the logo shots identified by the logo extraction algorithm when we are building the graph $G_{MAGIC}$. Interestingly, MAGIC finds logo shots that are not detected by the logo extraction algorithm (e.g. the shot ranked 16). Informative shots, such as faces of major players involved in the query event, are selected by MAGIC. For example, Kofi Annan appears in the shots at rank 9, 11, 20, and William Cohen in shots at rank 13, 15. Other logos pertaining to the query logo “Iraq” are also selected, for example, the “Yeltsin” logo at rank 14, which corresponds to the Yeltsin’s involvement...
Table 3: Selected terms by MAGIC as summary for a logo event. Terms are sorted (highest score first).

in the affairs with Iraq. The selection of shots for other query logos yields similar observations, but are not shown here due to the page limit.

The terms that MAGIC selected for the event “Iraq” are shown in Table 3 (first row). The selected terms are relevant, showing the names of key players (e.g. ‘Kofi Annan’ and ‘Albright’), the person’s position (‘secretary general’) and relevant locations (‘baghdad’, ‘sudan’) and activities (‘(air) strike’, ‘weapon talk’), etc. On other events like “Winter Olympics” and “Lewinsky”, MAGIC also successfully selected relevant terms that convey the content of the events (Table 3, second and third rows).

In general, MAGIC is capable of finding meaningful shots and terms for an event. The selection of shots and terms can be done simultaneously (using RWR), in spite of the different medium types. Readers are referred to our technical report [47] for more experimental results.

### 5.4 Generalization: any-to-any medium correlation in broadcast news

MAGIC finds more correlations other than the specific event-to-shot and event-to-term correlations used for event summarization in the previous section. Its capability of finding any-to-any medium correlations also gives us, for example, the term-to-shot and term-to-term correlations, and provides
the opportunity for other multimedia applications. In this section, we show results of using the term-to-shot and term-to-term correlations for another application: video retrieval with textual summary.

**Problem 2 (Video retrieval with textual summary)** Given a query of one or more terms, return a list of video shots relevant to the query, and furthermore, give a textual summary of the retrieved shot list.

Query by text has always been an important mean for users to express their information needs, even when the targets are mixed media objects such as video clips. Having a list of shots relevant to a query is useful, if one is exploring the collection. However, sometimes it is desirable to summarize the retrieved shots, to provide a concise view of the retrieved query result, especially in situation where the user is not able to watch every video shot in the retrieved set. One possible solution is to summarize the retrieval result by a textual summary - a set of terms which are correlated with the retrieved shots.

Traditional document retrieval methods usually give a list of relevant documents, but do not provide a summary of the retrieved result. In contrast, MAGIC provides a convenient framework to implement a video shot retrieval mechanism, and is able to provide a textual summary of the retrieved shots. Intuitively, we could use the cross-modal correlations detected by MAGIC to achieve video retrieval with textual summary: the term-to-shot correlation for shot retrieval, and the term-to-term correlation for textual summary.

We conduct video retrieval experiments on the same TRECVID 2003 video data set that we used for event summarization (Section 5.2). There are two steps to apply MAGIC: (1) graph construction and (2) RWR. Since the same data set is used, the $G_{MAGIC}$ graph for video retrieval is the same as that for the event summarization - a graph with three types of nodes (terms, shots, logos) (Figure 12). Given a textual query of one or more terms, we perform RWR from the corresponding query term-nodes and assigns RWR scores to all shots, logos and terms. The RWR scores will be used to select shots and terms as the retrieval result. Example queries used in our experiments are: {"lewinsky"}, {"clinton"}, {"lewinsky", "clinton"}, {"white", "house", "scandal"}, {"annan"}, {"iraq"}, {"annan", "iraq"}, {"olympics"}, etc.

Figure 14 shows the shots retrieved by MAGIC for the query {"lewinsky", "clinton"}. The shots are ranked in the order of relevance depicted by the RWR scores. Shots containing scenes of key persons related to the query are ranked at the top of the list, for example, prosecutor Kenneth Starr at rank 1 and Monica Lewinsky at rank 4. Relevant logo stories are also found, for example, “Clinton investigation” at rank 3 and “Jordan” (a Democrat lawyer) at rank 5. In the figure, we use the shot keyframe as the surrogate of a shot, for visualization purpose. The textual summary for this query is shown in Table 4 (first row). Words in the summary (e.g., ‘lawyer’, ‘whitewater’, ‘jones’, ‘affair’) give users a good picture about the retrieved set for the query {"lewinsky"},
Figure 14: Keyframes of the top 10 shots retrieved by MAGIC on the query {‘‘lewinsky’’, ‘‘clinton’’}. Frames are sorted (highest RWR score first).

<table>
<thead>
<tr>
<th>Method</th>
<th>Textual summary of the retrieval result</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAGIC</td>
<td>clinton lewinsky president monica attorney today house jury starr bill washington relationship story whitewater lawyer daughter jones counsel intern investigation immunity conversation headline minute ginsburg affair question judge mother office</td>
</tr>
<tr>
<td>OKAPI</td>
<td>clinton(16) lewinsky(11) monica(6) president(6) service(2) minute(2) report(2) blitzer(1) wolf(1) conversation(1) controversy(1) lewis(1) testimony(1) agent(1) nature(1) officer(1) claim(1) exchange(1) immunity(1) intern(1) house(1) affair(1) attorney(1) question(1) relationship(1) whitewater(1) headline(1) time(1) office(1)</td>
</tr>
<tr>
<td>LSI (using 50 singular vectors)</td>
<td>clinton(20) president(20) mandela(1) friend(1) congress(1) lady(1) visit(1) administration(1) relationship(1) washington(1)</td>
</tr>
</tbody>
</table>

Table 4: Summarizing retrieval result of query {‘‘lewinsky’’, ‘‘clinton’’}. Numbers in the parentheses are frequency counts of terms in the top 30 shots retrieved by OKAPI or LSI. Terms are sorted (highest RWR scores or frequency counts first).

‘‘clinton’’ which are stories about an judicial investigation involving president Bill Clinton and White House intern Monica Lewinsky on their relationship.
To evaluate our result, we compare MAGIC to two state-of-the-art document retrieval techniques: OKAPI [54] and the Latent Semantic Indexing (LSI) [11]. To apply document retrieval techniques to our domain, we consider each shot as a document of transcript words. The goal is to compare the quality of the retrieved shots.

Unlike MAGIC, which could do both shot retrieval and textual summary generation at the same time, OKAPI and LSI only do shot retrieval but not query result summarization. In fact, the topic of multi-document summarization is still an important, ongoing research area [18, 41]. Therefore, we design an approach to create a textual summary from the retrieval results by OKAPI or LSI: we collect the frequency histogram of all terms appear in the top $s$ retrieved documents, and make the textual summary as the list of these terms, ranked by the frequency. In our experiments, we choose $s=30$, as human users seldom check more than 30 documents in a query result.

Table 4 compares the textual summaries by the three methods: MAGIC, OKAPI and LSI, for the query {"lewinsky", "clinton"}. We found that MAGIC’s result is as good as that of OKAPI, and is more informative than that of LSI. Instead of focusing on details about "lewinsky" and "clinton", the textual summary by LSI has words about a broader concept of “politics in Washington”: ‘president’, ‘congress’, 'washington', etc. This is because that LSI was designed to focus on the semantic “topics” of documents, rather than the specifics about a query.

Moreover, OKAPI and LSI are limited to the terms which appear in the top $s=30$ retrieved shots. Relevant terms that are not in the retrieved shots can not be found by OKAPI or LSI. For example, relevant terms “starr” and “jones” are selected by MAGIC, but not by OKAPI or LSI, because these two terms are not mentioned in the top 30 shots retrieved. On the other hand, our MAGIC method does not have this limitation – the graph framework of MAGIC provides an elegant way to select terms effectively.

6 System Issues

MAGIC provides an intuitive framework for detecting cross-media correlations. The RWR computation in MAGIC is fast that it scales linearly with the graph size. For example, a straightforward implementation of RWR can caption an image in less than 5 seconds. In previous sections 4 and 5, we shown results where MAGIC is successfully applied to automatic image captioning and news event summarization.

In this section, we discuss system issues such as parameter configuration and fast computation. In particular, we present results showing

- MAGIC is insensitive to parameter settings;

- MAGIC is modular that we can easily employ the best module to date to speedup MAGIC; and
an image-captioning method based on MAGIC, which could caption a new image in constant time, using precomputed information from the core images $I_{core}$.

6.1 Optimization of parameters

There are several design decisions to be made when employing MAGIC for correlation detection: what should be the values for the two parameters: the number of neighbors $k$ of a domain token, and the restart probability $c$ of RWR? And, should we assign weights to edges, according to the types of their end points? In this subsection, we empirically show that the performance of MAGIC is insensitive to these settings, and provide suggestions on determining reasonable default values.

We use automatic image captioning as the application to measure the effect of these parameters. The experiments in this section are performed on the same 10 captioned image sets (“001”, ..., “010”) described in Section 4.1, and we measure how the values of these parameters affect the captioning accuracy.

**Number of Neighbors $k$** The parameter $k$ specifies the number of nearest domain tokens to which a domain token connects via the NN-links (Section 3.1). With these NN-links, objects having little difference in attribute values will be closer to each other in the graph, and therefore, are deemed more correlated by MAGIC. For $k=0$, all domain tokens are considered distinct; for larger $k$, our application is more tolerant to the difference in attribute values.

We examine the effect of various $k$ values on image captioning accuracy. Figure 15 shows the captioning accuracy on the data set “006”, with the restart probability $c=0.66$. The captioning accuracy increases as $k$ increases from $k=1$, and reaches a plateau between $k=3$ and 10. The plateau indicates that MAGIC is insensitive to the value of $k$. Results on other data sets are similar, showing a plateau between $k=3$ and 10.

In hindsight, with only $k=1$, the collection of regions (domain tokens) is barely connected, missing important connections and thus leading to poor performance on detecting correlations. At the other extreme, with a high value of $k$, everybody is directly connected to everybody else, and there is no clear distinction between really close neighbors or just neighbors. For a medium number of neighbors $k$, the NN-links apparently capture the correlations between the close neighbors, and avoid noise from remote neighbors. Small deviations from that value make little difference, which is probably because that the extra neighbors we add (when $k$ increases), or those we retained (when $k$ decreases), are at least as good as the previous ones.

**Restart probability $c$** The restart probability $c$ specifies the probability to jump back to the restarting node(s) of the random walk. Higher value of $c$ implies giving higher RWR scores to nodes closer in the neighborhood of the restart node(s).
Figure 15: The plateau in the plot shows that the captioning accuracy is insensitive to value of the number of nearest neighbors $k$. Y-axis: Average accuracy over all images of data set “006”. The restart probability is $c=0.66$.

Figure 16 shows the image captioning accuracy of MAGIC with different values of $c$. The data set is “006”, with the parameter $k=3$. The accuracy reaches a plateau between $c=0.5$ and 0.9, showing that the proposed MAGIC method is insensitive to the value of $c$. Results on other data sets are similar, showing a plateau between $c=0.5$ and 0.9.

For web graphs, the recommended value for $c$ is typically $c=0.15$ [19]. Surprisingly, our experiments show that this choice does not give good performance. Instead, good quality is achieved for $c=0.6 \sim 0.9$. Why is this discrepancy?

We conjecture that what determines a good value for the restart probability is the diameter of the graph. Ideally, we want our random walker to have a non-trivial chance to reach the outskirts of the whole graph. If the diameter of the graph is $d$, the probability that the random walker (with restarts) will reach a point on the periphery is proportional to $(1-c)^d$.

For the web graph, the diameter is estimated to be $d=19$ [1]. This implies that the probability $p_{\text{periphery}}$ for the random walker to reach a node in the periphery of the web graph is roughly

$$p_{\text{periphery}} = (1-c)^{19} = (1-0.15)^{19} = 0.045 .$$

(2)

In our image captioning and event summarization experiments, we use graphs that have three layers of nodes (Figures 2 and 12). The diameter of such graphs is roughly $d=3$. If we demand the same $p_{\text{periphery}}$ as equation (2), then the $c$ value for our 3-layer graph would be

$$(1-0.15)^{19} = (1-c)^3$$

$$(1-c)^3 = 0.65 ,$$

(3)

which is much closer to our empirical observations. Of course, the problem requires more careful
Figure 16: The plateau in the plot shows that the captioning accuracy is insensitive to value of the restart probability $c$. Y-axis: Average accuracy over all images of data set “006”. The number of nearest neighbors per domain token is $k=3$.

<table>
<thead>
<tr>
<th>$w_{term}$</th>
<th>$w_{region}$</th>
<th>0.1</th>
<th>1</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.370332</td>
<td>0.371963</td>
<td>0.370812</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.369900</td>
<td>0.370524</td>
<td>0.371963</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.368969</td>
<td>0.369181</td>
<td>0.369948</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Captioning accuracy is insensitive to various weight settings on OAV-links to the two media: region ($w_{region}$) and term ($w_{term}$).

analysis - but we are the first to show that $c=0.15$ is not always optimal for random walk with restarts.

**Link weights**  MAGIC uses a graph to encode the relationship between mixed media objects and their attributes of different media. The OAV-links in the graph connect objects to their domain tokens (Figure 2). To give more attention to an attribute domain $D$, we can increase the weights of OAV-links that connect to tokens of domain $D$. *Should we treat all media equally, or should we weight OAV-links according to their associated domains? How should we weight the OAV-links? Could we achieve better performance on weighted graphs?*

We investigate how the change on link weights influences image captioning accuracy. Table 5 shows the captioning accuracy on data set “006” when different weights are assigned on the OAV-links to regions (weight $w_{region}$) and those to terms ($w_{term}$). For all cases, the number of nearest neighbors is $k=3$ and the restart probability is $c=0.66$. The case where $(w_{region}, w_{term})=(1,1)$ is
that of the unweighted graph, and is the result we reported in Section 4. As link weights vary from 0.1, 1 to 10, the captioning accuracy is basically unaffected. The results on other data sets are similar - captioning accuracy is at the same level on a weighted graph as on the unweighted graph.

This experiment shows that an unweighted graph is appropriate for our image captioning application. We speculate that an appropriate weighting for an application depends on properties such as the number of attribute domains (i.e., the number of layers in the graph), the average size of a set-valued attribute of an object (such as, average number of regions per image), and so on. We plan to investigate more on this issue in our future work.

6.2 Speeding up graph construction by fast K-NN search

The proposed MAGIC method encodes a mixed media data set as a graph. The construction of the $G_{MAGIC}$ graph is intuitive and straightforward, where the only expensive component is the computation of the nearest neighbors of every domain token ($k$-NN search) to establish the NN-links. The number of domain tokens is usually large, and finding nearest neighbors of every of the tokens by sequentially computing all pairwise distances will take a lot of time.

For example, in our image captioning experiments, there are about 50,000 region tokens in each of the 10 Corel image sets, where each region token is a 30-dimensional feature vector (Section 4.1). To form the NN-links among region-nodes in the MAGIC graph, $k$-NN searches are performed 50,000 times (one for each region token) in the 30-dimensional feature space.

One common way is to speedup the nearest neighbor computation is to build a spatial index, such as R-tree [56], SR-tree [29], or more recently, IOC [37] and GORDER [66], which facilitates the search for nearest neighbors by efficiently pruning the search space. The modularity of the MAGIC framework allows us to use whichever method that best suits us. In our experiment, we use the approximate nearest neighbor method (ANN) [2], for its good performance in high dimensional search space.

Table 6 lists the average wall clock time to compute the top 10 neighbors of a region in the 10 Corel image sets. Each image set has about 50,000 regions. To compute the top $k=10$ nearest neighbors (among 50,000 regions) of a region token, ANN takes about 0.0038 seconds (totally, about 3 minutes for all 50,000 NN searches). Compared to the baseline method of computing all pairwise distances between regions sequentially (sequential search, SS), which takes around 0.046 seconds for one NN search, using the spatial index gives us more than 12-time speedup. The baseline method is implemented in C++, and is compiled with the code optimization ($g++$ -O3).

In MAGIC, the NN-links are proposed to capture the similarity relation among domain tokens. The goal is to associate tokens that are similar, and therefore, it might be suffice to have the NN-links connect to neighbors which are close enough, even if they are not the closest one. For many applications, such approximate nearest neighbor search is probably sufficient.
Table 6: Using a spatial index (ANN) speeds up the nearest neighbor computation on constructing graph $G_{MAGIC}$, compared to the baseline sequential search (SS). The first row (Elapse time) is the average wall clock time to compute the nearest neighbors of one region token. The second row (Speedup) is the ratio of elapse time, with respect to that of SS. The third and fourth rows (Error) show the percentage of mistakes made by approximation ($\epsilon=0.2, 0.8$) in the estimated $k$ nearest neighbors. $\epsilon=0$ indicates the exact k-NN computation. The symbol “-“ indicates zero error.

The spatial index “ANN” we used also allows approximate nearest neighbor search. The approximate search of ANN trades accuracy for speed, by estimating the distance between neighbors up to $(1+\epsilon)$ of the exact distance. The wall clock time for approximate nearest neighbor search is also reported in Table 6, for $\epsilon$ set at 0.2 and 0.8 ($\epsilon=0$ is the exact nearest neighbor search). Compared to the baseline sequential search method, the approximation could achieve up to a 51-time speedup ($\epsilon=0.8$).

The approximation yields just a tiny percentage of quality lost in the computed nearest neighbors. We are more interested in the possible errors at the top $k=3$ nearest neighbors, which are what we used in our experiments (Figures 7 and 8). No error is made in the $k=3$ nearest neighbors, when approximating at $\epsilon=0.2$. A faster approximation $\epsilon=0.8$ makes negligible mistakes - only 0.46% for $k=3$, in other words, one error in every 217 NN-links.

This small difference on NN-links does not change the characteristic of the MAGIC graph significantly, and has limited affect on the performance of image captioning. At $\epsilon=0.2$, no error is made on the NN-links in the MAGIC graph, and therefore the captioning accuracy is the same as exact computation. At $\epsilon=0.8$, the average captioning accuracy decreases by just 1.59 percentage point, averaged over the 10 Corel image sets (Figure 17).

Recently, there are other sub-linear time methods for approximate nearest neighbor search [36] being developed. Thanks to the modularity of MAGIC, we can easily leverage advances in these works to further improve the efficiency and performance of MAGIC.
Figure 17: Approximating the NN-links in the MAGIC graph does not affect the captioning accuracy much. The parameters for MAGIC are $c = 0.66$ and $k = 3$. The x-axis shows the 10 data sets, and the y-axis is the average captioning accuracy over all test images in a set. On average, a fast approximation at $\varepsilon=0.8$ reduces captioning accuracy by just 1.59%.

6.3 Precomputation for fast RWR computation

As outlined in Figure 4, our implementation of RWR is fast already (linear to the database size). Nevertheless, the computation of the RWR scores can be even further accelerated. In this subsection, we discuss approaches for fast RWR computation and introduce our approach for speeding up image captioning by precomputation.

**RWR on sparse graph** Our proposed MAGIC graph $G_{MAGIC}$, by construction, has a structure of layers (Figure 12). Edges exist only between certain nodes, making the graph $G_{MAGIC}$ a sparse graph. In fact, only a small fraction of all possible edges are in the graph. Figure 18 shows the adjacency matrix of the MAGIC graph for broadcast news event summarization in Section 5. The adjacency matrix is sparse and shows block structures corresponding to the entities involved: logos, shots, and terms.

From Equation 1 (Section 3.2), it is easy to show that

$$\tilde{u}_q = c (I - (1 - c)A)^{-1} \tilde{v}_q$$

(5)

where $I$ is the $N \times N$ identity matrix, $\tilde{v}_q$ is the restart vector, and $\tilde{u}_q$ is the vector of RWR scores on all $N$ nodes. To solve for the RWR scores $\tilde{u}_q$, we can take advantage of the sparse structure in the adjacency matrix $A$, and tap the old and recent literature of fast solutions to sparse linear systems [52], to efficiently solve the matrix inversion in equation (5).

Recently, there are also methods for fast random walk computation which exploit the blockstructure in a graph [19], or adapting the computation to the convergence behavior [27]. Given
Figure 18: The adjacency matrix of the MAGIC graph for broadcast news event summarization (Section 5). The graph has three types of nodes, for logos, shots, and terms. A dot at position \((i,j)\) indicates that there is an edge between the \(i\)-th and \(j\)-th node. Graph statistics: 27,809 nodes and 133,282 edges. The graph is sparse, only 0.17% of all possible edges are connected.

that this area is still under intensive research [24, 28], we only point out that our MAGIC approach for correlation discovery is modular, and it can trivially include whichever is the best module for fast RWR computation.

**Speeding up image captioning by precomputation**  In image captioning, MAGIC first builds a core graph (Definition 4) according to the given captioned images. This core graph is then used to caption thousands of new images, one at a time, using RWR on the augmented graph (Figure 6). As there are many images to be captioned in practice, fast RWR score computation is desirable.

Before introducing the proposed method to speedup image captioning, it is helpful to review how MAGIC captions an image. Let \(q\) be the node of the image which we want to caption. To caption image \(q\), we first take the core graph \(G_{\text{core}}\) and augment the image-node and region-nodes of image \(q\) to \(G_{\text{core}}\), via the gateway nodes \(GW\), to get the augmented MAGIC graph \(G_{\text{MAGIC}}\). Then, we compute the RWR scores \(\tilde{u}_q\), with the restart vector \(\tilde{v}_q\) - a vector of all zeros, except one 1 for node \(q\). (Figure 6 gives the detail algorithm of captioning an image.) The vector \(\tilde{u}_q\) satisfies equation (1), which is reproduced here in equation (6) for the reader’s convenience.

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

Captioning different test images will use different \(G_{\text{MAGIC}}\) graphs (and therefore, different column-normalized adjacency matrices \(A\)), due to the different augmentation subgraphs they have. However, since the specific augmentation for each test image is small (with only about 10 nodes)
compared to the core graph $G_{core}$, the augmented graph $G_{MAGIC}$ (matrix $A$) of each test image is similar to the core graph (the matrix $A_{core}$).

Whenever we caption a new image, we solve a version of equation (6) with a slightly different $A$. However, since all matrices $A$ share the same $A_{core}$, how could we make use of this observation? Can we precompute some information from $A_{core}$, to speedup the captioning of a new image?

We propose a method, PRECOM, which uses precomputation to speedup the captioning of thousands of images. Before introducing our proposed method (PRECOM), we need more definitions. For a test image $q$, suppose that the set of gateway nodes connecting $q$ and graph $G_{core}$ is $\mathcal{GW}$ and has size $z = |\mathcal{GW}|$. Let $\tilde{u}_{\mathcal{GW}}$ denote the vector of RWR scores, when restarting from the gateway nodes on the core graph $G_{core}$. That is, $\tilde{u}_{\mathcal{GW}}$ satisfies the following equation:

$$\tilde{u}_{\mathcal{GW}} = (1 - c)A_{core}\tilde{u}_{\mathcal{GW}} + c\tilde{v}_{\mathcal{GW}},$$

where $\tilde{v}_{\mathcal{GW}}$ is the restart vector, whose $i$-th element is $\frac{1}{z}$ if node $i \in \mathcal{GW}$, and is 0, otherwise.

The idea of PRECOM is based on the resemblance between equations (6) and (7), as well as that between matrices $A$ and $A_{core}$. The precomputation is a one-time cost, and all subsequent captioning can be done very efficiently in constant $O(1)$ time. The basic ideas are:

- Approximate $\tilde{u}_q$ by $\tilde{u}_{\mathcal{GW}}$ (the PRECOM scores).
- $\tilde{u}_{\mathcal{GW}}$ can be computed in constant time, using precomputed information (explain next).

In the following, we first outline the algorithm of the PRECOM method, followed by discussions on how the PRECOM scores ($\tilde{u}_{\mathcal{GW}}$) can be computed in constant time using precomputed results (Lemma 1). And then, we empirically show that $\tilde{u}_{\mathcal{GW}}$ is approximately the same as $\tilde{u}_q$ (scores used by MAGIC on captioning).

Figure 19 gives the algorithm of PRECOM. PRECOM replaces the RWR computation at steps 1 and 2 of the original algorithm in Figure 6, with precomputation and an $O(1)$ time captioning step.

The information that PRECOM precomputes is the RWR score vector $\tilde{u}_i$ of every blob-node $i$ in the core graph $G_{core}$. Let $\tilde{v}_i$ be the restart vector with all elements 0, except the $i$-th element, which is 1. The RWR score vector $\tilde{u}_i$, restarting with respect to $\tilde{v}_i$ on the core graph, satisfies

$$\tilde{u}_i = (1 - c)A_{core}\tilde{u}_i + c\tilde{v}_i.$$  

Also, $\tilde{v}_{\mathcal{GW}}$ and $\tilde{v}_i$ are related as follows:

$$\tilde{v}_{\mathcal{GW}} = \frac{1}{z} \sum_{i \in \mathcal{GW}} \tilde{v}_i,$$

where $z = |\mathcal{GW}|$ is the number of the gateway nodes in set $\mathcal{GW}$.  

35
Input: 1. The core graph $G_{core}$, an image $I_{new}$ to be captioned, and
2. $g$, the number of caption words we want to predict for $I_{new}$.

Output: Predicted caption words for $I_{new}$.

Steps:
0. (One-time cost) Precompute the RWR score vector $\tilde{u}_i$ of each blob-node on $G_{core}$.
1. Identify the gateway nodes $GW$ of $I_{new}$.
2. Compute the PRECOM score vector $\tilde{u}_{GW}$ (using Lemma 1).
3. Rank all term nodes by their PRECOM scores.
4. The $g$ top-ranked terms will be the output - the predicted caption for $I_{new}$.

Figure 19: Steps to caption an image, using the proposed PRECOM method.

If we precompute the RWR score vector $\tilde{u}_i$ for every node $i$ in the core graph, Lemma 1 says that the $\tilde{u}_{GW}$ vector of a test image $q$ is a linear sum of the precomputed $\tilde{u}_i$’s, and can be computed in constant time. In the following, we will refer to $\tilde{u}_{GW}$ as the PRECOM score vector for a test image $q$. Figure 20 gives the proof of Lemma 1.

Lemma 1 Let $z = |GW|$ be the size of the set of gateway nodes ($GW$), then

$$\tilde{u}_{GW} = \frac{1}{z} \sum_{i \in GW} \tilde{u}_i.$$ (10)

We would empirically show that the proposed PRECOM predicts similar captioning terms as MAGIC does. Basically, we compare the orderings of the terms according to scores $\tilde{u}_{GW}$ by PRECOM and $\tilde{u}_q$ by MAGIC, and show that the two methods give similar term orderings.

We assign rank 1 to the term with the highest score, rank 2 to the next, and so on. Each term gets two ranks: one according to the scores $\tilde{u}_{GW}$ from PRECOM, and one based on $\tilde{u}_q$ (from MAGIC). In Figure 21(a), we plot the ranks of terms by PRECOM (Y-axis) against those by MAGIC (X-axis), for an image shown in Figure 9(b) (the image with two lions). Specifically, a point at location $(i, y_i)$ corresponds to the rank $i$-th term dictated by MAGIC, which has rank $y_i$ if according to PRECOM. A perfect diagonal pattern in the figure means no difference between PRECOM and MAGIC. We can see that the two methods give roughly the same ranking on terms - the data points are along the 45° diagonal line.

The two methods, PRECOM and MAGIC, agree on the rankings of the terms in general. Especially, they agree on the top-ranked words, which are our predicted caption words. The ranking of the top-ranked words are shown at Figure 21(b), which is an expanded view of the bottom-left corner of Figure 21(a). In fact, for this image, the captions from the two methods are the same.
Proof:
1. We want to show that equation (10) is consistent with equation (7).
   We do this by showing that equation (7) still holds, after substituting equation (10) into it.
2. After substitution, the left hand side of (7) is
   \[ \text{LHS} = \frac{1}{2} \sum_{i \in G_W} \tilde{u}_i, \]
   and the right hand side is
   \[ \text{RHS} = (1 - c) A_{\text{core}} \frac{1}{2} \sum_{i \in G_W} \tilde{u}_i + c \tilde{v}_i. \]
3. Substitute equation (9) into RHS, we have
   \[ \text{RHS} = (1 - c) A_{\text{core}} \frac{1}{2} \sum_{i \in G_W} \tilde{u}_i + c \frac{1}{2} \sum_{i \in G_W} \tilde{v}_i \]
   \[ = \frac{1}{2} \sum_{i \in G_W} ((1 - c) A_{\text{core}} \tilde{u}_i + c \tilde{v}_i) \]
   \[ = \frac{1}{2} \sum_{i \in G_W} \tilde{u}_i \quad \text{(By equation 8)} \]
   \[ = \text{LHS}. \]

Figure 20: Proof of Lemma 1

To summarize the overall captioning differences between PRECOM and MAGIC, we compute the sum of absolute rank difference in the top 5 terms which, intuitively, indicates the total amount of rank reshuffling between the two methods. On average, the top 5 terms are reshuffled by 2. In other words, a top term dictated by MAGIC remains a top ranked term predicted by PRECOM, with the rank differs by less than 2 for most of the query images. Therefore, in most cases, the two methods will give the same caption terms to an image.

Figure 22 shows the captioning accuracy of MAGIC and PRECOM on our 10 Corel image sets. The average difference on captioning accuracy over the 10 sets is just about 0.08%. Therefore, speeding up image captioning by precomputation does not compromise the captioning accuracy of PRECOM - PRECOM achieves the same captioning accuracy as MAGIC.

7 Conclusions

Mixed media objects such as captioned images or video clips have attributes of different media (image, text, or audio). Detecting correlations and patterns across different media is useful for tasks such as imputation of missing media values, and has many applications such as image captioning (predicting missing caption words of an image). In this paper, we developed MAGIC, a general method for detecting cross-media correlations in mixed media data set.

There are two challenges in detecting cross-media correlations, namely, representation of set-valued attributes and the detection of correlations between any medium and all media (the any-to-any medium correlation). MAGIC adopts a graph-based model which provides a natural way
Figure 21: Comparing the rankings of terms according to the RWR scores on term-nodes from PRECOM and MAGIC. A point corresponds to a term; the point’s location is defined by (x,y)=(rank according to MAGIC, rank according to PRECOM). (a): ranks of all terms; (b): ranks of the top 10 terms. The query image is the image of two lions shown at Figure 9(b). A 45° diagonal line pattern means PRECOM and MAGIC give similar rankings on terms.

Figure 22: The proposed method PRECOM achieves the same captioning accuracy as MAGIC.

to represent set-valued attributes. The proposed model accommodates problems such as missing values or the non-aligned, noisy set elements, with no extra efforts. In addition, the graph-based framework that MAGIC provides is capable of finding any-to-any medium correlation, using the technique of random walk with restarts (RWR).

MAGIC provides cross-media correlations which can be applied in many multimedia applications. In this study, we applied MAGIC on two applications: automatic image captioning and news
event summarization (Applications 1 and 2).

For image captioning, MAGIC spots the correlations between images and caption words, and applies the correlations to caption new images. On the same benchmark, MAGIC outperforms previous methods on captioning accuracy (up to 58% relative improvement) (Section 4). Moreover, MAGIC is able to spot correlations between video shots, logos, and transcript terms, and gives meaningful news event summarization (Section 5). Although some video shots do not have logo information, MAGIC deals with these missing values smoothly.

Furthermore, the graph framework of MAGIC is versatile, giving any-to-any medium correlations which enable other applications such as group captioning (Problem 1) and video shot retrieval (Problem 2). In fact, to the best of our knowledge, MAGIC is the first attempt for group captioning which is useful for applications such as video segment captioning.

Technically, MAGIC has the following desirable characteristics:

- It is domain independent: The $\text{Sim}_d(\ast,\ast)$ similarity functions (Assumption 1) completely isolate our MAGIC method from the specifics of an application domain, and make MAGIC applicable to detect correlations in all kinds of mixed media data sets.

- It requires no fine-tuning on parameters or link weights: The performance is not sensitive to the two parameters - the number of neighbors $k$ and the restart probability $c$, and it requires no special weighting scheme like tf/idf for link weights (Section 6.1).

- It is fast and scales up well with the database/graph size. We also proposed a constant-time method (PRECOM) for speeding up image captioning, using the idea of precomputation (Section 6.3).

- It is modular and can easily tap recent advances in related areas to improve performance (Section 6.2).

We are pleasantly surprised that such a domain-independent method, with no parameters to tune, managed to outperform some of the most recent and most carefully tuned methods for automatic image captioning, and can be applied to other problems like news event summarization. Future work could further exploit the promising connection between multimedia databases and graph algorithms, including outlier detection and any other data mining task that require the discovery of correlations as its first step.

References


