ABSTRACT

Despite considerable progress in recent years on Tag-based Social Image Retrieval (TaIR), state-of-the-art TaIR systems fail to provide a systematic framework for end users to ask why certain images are not in the result set of a given query and provide an explanation for such missing results. However, such why-not questions are natural when expected images are missing in the query results returned by a TaIR system. In this demonstration, we present a system called why (Why-not quEstIon aNswering Engine) which takes the first step to systematically answer the why-not questions posed by end-users on TaIR systems. It is based on three explanation models, namely result reordering, query relaxation, and query substitution, that enable us to explain a variety of why-not questions. Our answer not only involves the reason why desired images are missing in the results but also suggestion on how the search query can be altered so that the user can view these missing images in sufficient number.

Categories and Subject Descriptors
H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—Search Process

Keywords
Social Image; Flickr; Tag-based image search; Why-not questions; Explanation models

1. INTRODUCTION

Due to increasing popularity of social image sharing platforms (e.g., Flickr, Picasa), techniques to support Tag-based Social Image Retrieval (TaIR) for finding relevant high-quality images using keyword queries have recently generated tremendous research and commercial interests. In simple words, given a keyword query (or search query), a TaIR search engine returns a ranked list of images where the images annotated with the most relevant tags to the query are ranked higher. Most existing efforts in TaIR attempt to improve its search accuracy or diversify its search results so as to maximize the probability of satisfying users’ search intentions. Despite the recent progress towards this goal, it is often challenging to generate high quality search results for a search query which can satisfy search intentions of different users. Often, desired images may be unexpectedly missing in the results. However, state-of-the-art TaIR systems lack explanation capability for users to seek clarifications on the absence of expected images (i.e., missing images) in the result set. Consider the following set of user problems:

Example 1. Ann is planning a trip to Rome to visit its famous landmarks. She issues a search query "Rome" on a tag-based social image search engine1. Expectedly, many images of Rome’s famous landmarks appear as top result matches, such as the Spiral Stairs, the Gallery of Map, and the Sistine Chapel. However, surprisingly, there are no images related to the Colosseum, a famous landmark of Ancient Rome, in the top-100 results. So why is it not in the result set? Note that expanding the query by adding the keyword Colosseum to "Rome" changes the search intent from “famous landmarks of Rome including Colosseum" to "Colosseum in Rome". Consequently, such query expansion leads to loss of images of interesting landmarks in Rome other than Colosseum, depriving Ann to get a bird eye view of different attractions of Rome.

Bob has just returned from a trip to China. He specifically enjoyed the scenic Xi Hu lake in Hangzhou city of the Zhejiang province. However, Bob has forgotten its name. Hence, he posed the following query to retrieve images related to Xi Hu lake: "lake Hangzhou Zhejiang China". Surprisingly, no result is returned by the search engine. Why not? Note that simply searching for "lake" alone is ineffective as Bob primarily wants images of Xi Hu lake and not other lakes. In fact, the query "lake" returns more than 4000 images, many of these are irrelevant.

Carlos, a young archaeologist researching on Mesoamerican culture, hopes to find images related to their pyramids. He submits the query "pyramid" on the image search engine which returns mostly images related to Egyptian and Louvre pyramids (Figure 1(a)). So why are Mesoamerican pyramids not in the result set? Perplexed, Carlos expands the query by adding the keyword "Mesoamerica", hoping to retrieve relevant images. However, only four images are now returned and among them, only two are really relevant to Mesoamerican pyramids. Are there only two images of Mesoamerican pyramids in the image collection? Thinking that his modified query may be too strict, Carlos now removed the keyword "pyramid" from the query. However, only five additional results are returned now and none of these additional images are relevant to Mesoamerican pyramids. So why not more images related to Mesoamerican pyramids can be retrieved?

There is one common thread throughout these problems encountered above, despite the differences in search queries: the user

1All search results presented in our examples are obtained using the same TaIR system following the best performing configuration in [6] on NUS-WIDE data (http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm).

Copyright is held by the International World Wide Web Conference Committee (IW3C2). IW3C2 reserves the right to provide a hyperlink to the author’s site if the Material is used in electronic media.
ACM 978-1-4503-2745-9/14/04.
http://dx.doi.org/10.1145/2567948.2577028.
would like to know why certain images are missing in the top-$m$ result set of a given query or not there in sufficient number and suggestion on how his/her query can be altered effectively to view these missing images in sufficient number. In this paper we refer to this problem as the Why Not? problem in TadIR [2].

At a first glance, it may seem that any large-scale social image search engine (e.g., Flickr) may facilitate answering these original queries more effectively simply because they have very large collection of social images compared to the sus-wine data collection used in the aforementioned examples. For instance, Bob’s query returns several images related to Xi Hu lake when posed directly on Flickr². Unfortunately, users’ expectations are just too diverse to eliminate the Why Not? problem in Flickr (detailed in [2]). For example, consider the query “pyramid” directly on Flickr. It only retrieves a single image related to Mesoamerican pyramid in its top-50 result set!

Our initial investigation shed some light on the possible reasons for this problem. First, the desired images may be ranked very low in the search results because the same keyword query may express very different search intentions for different users. The top-ranked images maybe considered relevant by some users but not by others. For instance, the reason Ann could not see the images related to Colosseum is because they are ranked too low. The first Colosseum image is ranked 217-th and Ann is unlikely to explore more than 100 images to search for Colosseum.

Second, the set of tags associated with images may be noisy and incomplete. Consequently, not all keywords mentioned in the search query may appear as tags in relevant images. For instance, a user may not annotate an image related to Xi Hu lake with the tag Zhejiang. In fact, none of the images related to Xi Hu lake are tagged with Zhejiang in the underlying image collection! However, it is unrealistic to expect a user to be aware of this fact.

Third, the query formulated by the user maybe too restrictive due to the user’s limited understanding of the data collection. That is, there may be a mismatch between the tags that the user expects to be associated with her desired images and the actual tags that annotate these images in the data collection. For instance, Carlos failed to retrieve sufficient number of images annotated with the tag Mesoamerica because it is rarely used in tagging images in the image collection. However, Carlos is unlikely to have this knowledge or possess the skill to alter the query to retrieve his desired images.

Clearly, it would be very helpful to Ann, Bob, and Carlos if they could simply pose a follow-up why-not question to the TadIR engine to seek an explanation for desired missing images and suggestions on how to retrieve them. In this demonstration, we present a novel system called WINE (Why-not questIon aNsWering Engine) [2] to address this problem. Wine automatically generates explanation for a why-not question (expressed using a why-not tag) and recommends refined query, if necessary, whose result may not only includes images related to the search query but also to the why-not question. To the best of our knowledge, this is the first system to address the Why Not? problem in TadIR.

Let us illustrate WINE with an example. Reconsider the query posed by Carlos. He may pose a follow-up why-not question using the why-not tag "mesoamerica" (See Panel 4 in Figure 1(b)). In Panel 5, a short explanation (i.e., the number of images related to mesoamerica is too small in the image collection) is automatically generated in response to the why-not question (an enlarged version is shown in Figure 3(b)). More importantly, three refined query suggestions (i.e., “pyramid maya”, “teotihuacan”, “aztec”) are also provided, each of which is likely to return more images related to Mesoamerican pyramids (Figure 3(b)). Suppose Carlos chooses "pyramid maya" as the refined query by clicking on it. The results are now shown in Figure 1(b). Observe that it offers more results related to Mesoamerican pyramids compared to the original query results in Figure 1(a).

2. RELATED SYSTEMS AND NOVELTY

It may seem that the Why Not? problem can be addressed by leveraging existing search techniques such as query expansion, query suggestion, and search result clustering. Unfortunately, this is not the case. For instance, as highlighted in Example 1, expanding the queries "rome" and "pyramid" with the why-not tags "colosseum" and "mesoamerica", respectively, do not address Ann’s and Carlos’ queries effectively. Notably a why-not question should not alter the original search intent. On the other hand, given the query "pyramid", the recommended tags by an existing query expansion technique would likely be "Egypt", "Louvre", "Giza", "Sphinx", etc., reflecting the commonly associated concepts to pyramid. Clearly, such suggestion not only modifies the search intent of Carlos, but also fails to address his why-not question. That is, without the explicit why-not tag "mesoamerica", state-of-the-art query suggestion models may fail to speculate that Carlos’ interest is in Mesoamerican pyramid.

More germane to this work are recent efforts in the database community to provide automatic explanation to a why-not question [1,3]. To answer why-not questions (i.e., why some expected data items are not shown in the result set) on relational databases, multiple answer models have been proposed. These models, however, are not applicable for TadIR environment because: (i) the data in TadIR is not represented using relational structure and (ii) these techniques typically exploit the relational query plan which is in-applicable in TadIR.

²All results related to Flickr are last accessed on July 14th, 2013.
The aforementioned tag features are useful for generating query results but are not sufficient to answer all types of why-not questions. For instance, reconsider the why-not tag *mesoamerica* posed by Carlos to search for Mesoamerican pyramids. The shortage of images annotated by this why-not tag poses two intertwining challenges. Firstly, it cannot be leveraged directly for generating explanation to the why-not question as it is unlikely that the user wishes to see a very small number of result images (if any) associated with this tag. Secondly, while the desired images are likely to be annotated by some *closely related* tag(s) (e.g., *maya*) to the why-not tag, it is difficult to find these related tags using aforementioned measures as they require sufficiently large number of matching images to be effective. Hence, we exploit an external source (Wikipedia) to address this issue. Specifically, the *Keyphrase Extractor* submodule exploits the *keyphrase* data (title or an anchor text) of a Wikipedia article to measure the strength of relationship between tags. It extracts the keyphrases for each article and the relationship (similarity) between each pair of keyphrases (e.g., *maya* and *mesoamerica*) is measured by adopting the hyperlink-based *Wikipedia Link Measure* (wlm) [5]. This similarity value is then used as the similarity score between a pair of tags (keyphrases) which we shall be exploiting later to guide the why-not question answering process. For further details, please refer to [2].

The *Image Search Module*. This module encapsulates a standard TagIR engine interface. Given a keyword query $Q$, it leverages the *Tag Index* to retrieve the top-$k$ images that best match $Q$ where $k$ is the user-specified number of desired images. Note that the image retrieval algorithm is orthogonal to wine and any superior social image retrieval techniques can be adopted for *wine*. In our implementation, we adopt the framework in [6] for multi-tag queries. Furthermore, to display the *significant tags* in Panel 3, we compute the *relative tag frequency* of all tags in the top-$k$ results. For each tag $t_i$, it is computed as the difference between its frequency among the top-$k$ results and its frequency in the whole collection $D$ (which is pre-computed). Both frequencies are also weighted by $t_i$'s relatedness to each image. The tags with high related frequency are considered *significant* and displayed in Panel 3.

The *Why-Not Answer Module*. This module is the core component of *wine* and consists of the following three submodules.

**Why-Not Question Analyzer.** Given a query $Q$, result set $R(Q)$, and a follow-up why-not question $t_n$, this module analyzes the tag $t_n$ and classifies it to any one of the four types: (a) **Type 1**: $t_n$ is *incomprehensible* if it has no match in the image collection $D$ as well as in the *Keyphrase Index* (Wikipedia). (b) **Type 2**: Images related to $t_n$ are in $R(Q)$ but they are too lowly-ranked (e.g., the why-not tag *Colosseum* as follow-up to Ann’s query). (c) **Type 3**: There are too few images related to $t_n$ in $R(Q)$. However, there are sufficiently large number of images related to $t_n$ in $D$ (e.g., the why-not tag *Lake* as follow-up to Bob’s query). (d) **Type 4**: There are too few images annotated with $t_n$ in $D$ (e.g., the why-not tag *mesoamerica* as follow-up to Carlos’ query).

If $t_n$ is a Type 1 tag, then we notify the user that her question is incomprehensible. For Types 2-4 tags, we invoke the result reordering, query relaxation, and query substitution explanation models, respectively (discussed below), to respond to the user.

**Result Generator.** This component improves the original results $R(Q)$ by retrieving more images related to the why-not tag $t_n$ but maintaining the semantics of the original query $Q$. It realizes the following three explanation models, namely result reordering, query relaxation, and query substitution, that are designed to address three different scenarios of the Why Not? problem highlighted in Example 1 (details related to these models are given in [2]).

Figure 2: System architecture of wine.
Result Reordering Model. Intuitively, in this explanation model we reorder the search results so that images related to the why-not tag in the results appear in the top-k results. It is useful when the relevant images exist in the query results but are lowly ranked (e.g., images related to the Colosseum in Example 1). Given the query \( Q \), each result image \( d \in R(Q) \) is assigned a new score by combining the relevance score \( rel(d, t_w) \) of the why-not tag \( t_w \) and the original score \( rel(d, Q) \) through linear combination as follows.

\[
rel_{av}(d, Q, t_w) = (1 - \alpha) \times rel(d, Q) + \alpha \times rel(d, t_w)
\]

(1)

Note that we assume \( rel(d, t_w) = 0 \) when \( d \notin R(t_w) \). The tunable parameter \( 0 \leq \alpha \leq 1 \) indicates the importance of \( rel(d, t_w) \) compared to \( rel(d, Q) \). In other words, \( \alpha \) indicates the user’s level of dissatisfaction to the current result list \( R(Q) \). Note that the slider in Panel 6 allows us to vary this parameter.

Query Relaxation Model: This model aims to automatically identify the selectable tag set in the search query that prevents the user from retrieving desired images. It notifies the user to remove these selective tag(s) from the query so that desired images related to the why-not tag can be retrieved from \( D \). For example, consider Bob’s query in Example 1. The query relaxation model identifies that Zhejiang is a selective tag and advises Bob to remove it from the original query in order to view images related to Xi Hu lake. Note that this model is effective when there are few images related to the why-not tag in the result set (i.e., the Result Reordering model is ineffective) but there are a large number of such images in \( D \).

Intuitively, a set of tags \( T \) of a multi-tag query \( Q \) is selective when removing a single tag from \( T \) would generate significantly larger size of query results. We extend the Hypercube algorithm, a popular algorithm in parallel computing, to efficiently compute the selectivity of all query tag subsets by scanning the images annotated with \( t_w \) only once and rank them based on their selectivities.

Query Substitution Model: This explanation model is suitable when there are too few relevant images annotated by \( t_w \) in \( D \) (as in Carlos’ query in Example 1). The goal in this case is to suggest closely related keywords to the why-not tag, which are associated with many images in \( D \), as surrogates to the current query keyword(s). Specifically, we leverage the knowledge embedded in Wikipedia (KeyPhrase Index) to identify these closely related tags. For instance, the query “pyramid Mesoamerica” in Example 1 is modified to “pyramid Maya” after identifying maya to be the most closely related to mesoamerica using the KeyPhrase Index. This new query generated 27 query results many of which are images of Mesoamerican pyramid (Figure 1(b)).

Specifically, \( \text{wns} \) seeks for a tag \( t_c \) such that \( t_c \) annotates sufficiently large number of images in \( D \) and maximizes the tag relatedness score \( \Phi(t_c) \):

\[
\Phi(t_c) = (1 - \beta) \frac{\sum_{t \in Q} \text{sim}(t_c, t)}{|Q|} + \beta \times \text{sim}(t_c, t_w)
\]

(2)

The similarity function \( \text{sim}() \) of a tag pair computes the similarity between their corresponding keyphrases where the mapping between tags and keyphrases are achieved by string matching with minor syntactic modifications. The parameter \( \beta \) controls how much change in the original query the user can tolerate. We efficiently find the top-k closely related tags by casting the problem to the combining fuzzy grade problem which can be solved using Fagin et al.’s Threshold Algorithm.

Explanation Generator. This component generates answer to the user’s why-not question from the output of the Why-Not Question Analyzer and Result Generator submodules. The generated explanation consists of three parts, the explanation, the refinement method(s) (e.g., remove the tag Zhejiang, refine the query to “pyramid maya”), and some statistics of the new results (e.g., result size) if those refinement methods are followed. Figure 3 depicts examples of explanations provided by \( \text{wns} \) in response to the why-not tags “Lake” and “mesomerica”.

4. DEMONSTRATION OVERVIEW

Our demonstration will be loaded with the \( \text{wns} \)-wide dataset containing 269,648 images from Flickr. We aim to showcase the functionality and effectiveness of the \( \text{wns} \) system in answering why-not questions. Example queries (original as well as why-not questions) illustrating the three explanation models will be presented. Users can also write their own ad-hoc why-not questions through our \( \text{wns} \) website. A video of \( \text{wns} \) is available at http://youtube/be/A4212geQVk. Specifically, we will showcase the following.

Interactive experience of why-not question answering process. Through our \( \text{wns} \), the user will be able to formulate search queries (Panel 1), browse the top-k results (\( k \) can be specified by the user) and associated tags (Panels 2 and 3), and then follow-up with a why-not question (Panel 4). The Why-Not Answer module will then generate detailed answer (e.g., Figure 3) by exploiting the explanation models (we shall demonstrate all three models). Going a step further, the user may accept one of the suggested actions and visualize in real-time the new result set generated by \( \text{wns} \) as well as associated significant tags. Clicking on any tag will allow her to view immediately all images in the result set that are annotated by this tag. Additionally, by setting the slider in Panel 6 at different threshold values, she can view updates to the search results instantly. Lastly, the user will be able to compare the original and refined results by clicking on the tabs in Panel 3.

Superior performance of \( \text{wns} \). We shall demonstrate that all three explanation models in \( \text{wns} \) have superior accuracy and precision for different result size and parameters (e.g., \( \alpha, \beta \)). Also, we shall demonstrate that the execution of these models are very efficient (less than 100ms) for a wide variety of why-not questions.

5. REFERENCES