Advanced Techniques for Multimedia Search: Leveraging Cues from Content and Structure

Lyndon Kennedy

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

in the Graduate School of Arts and Sciences

Columbia University

2009

ABSTRACT

Advanced Techniques for Multimedia Search: Leveraging Cues from Content and Structure

Lyndon Kennedy

This thesis investigates a number of advanced directions and techniques in multimedia search with a focus on search over visual content and its associated multimedia information. This topic is of interest as the size and availability of multimedia databases are rapidly multiplying and users have increasing need for methods for indexing and accessing these collections in a variety of applications, including Web image search, personal photo collections, anc biomedical applications, among others.

Multimedia search refers to retrieval over databases containing multimedia documents. The design principle is to leverage the diverse cues contained in these data sets to index the semantic visual content of the documents in the database and make them accessible through simple query interfaces. The goal of this thesis is to develop a general framework for conducting these semantic visual searches and exploring new cues that can be leveraged for enhancing retrieval within this framework.

A promising aspect of multimedia retrieval is that multimedia documents contain a richness of relevant cues from a variety of sources. A problem emerges in deciding how to use each of these cues when executing a query. Some cues may be more powerful than others and these relative strengths may change from query to query. Recently, systems using classes of queries with similar optimal weightings have been proposed; however, the definition of the classes is left up to system designers and is subject to human error. We propose a framework for automatically discovering query-adaptive multimodal search methods. We develop and test this framework

using a set of search cues and propose a new machine learning-based model for adapting the usage of each of the available search cues depending upon the type of query provided by the user. We evaluate the method against a large standardized video search test set and find that automatically-discovered query classes can significantly out-perform hand-defined classes.

While multiple cues can give some insight to the content of an image, many of the existing search methods are subject to some serious flaws. Searching the text around an image or piece of video can be helpful, but it also may not reflect the visual content. Querying with image examples can be powerful, but users are not likely to adopt such a model of interaction. To address these problems, we examine the new direction of utilizing pre-defined, pre-trained visual concept detectors (such as "person" or "boat") to automatically describe the semantic content in images in the search set. Textual search queries are then mapped into this space of semantic visual concepts, essentially allowing the user to utilize a preferred method of interaction (typing in text keywords) to search against semantic visual content. We test this system against a standardized video search set. We find that larger concept lexicons logically improve retrieval performance, but there is a severely diminishing level of return. Also, we propose an approach for leveraging many visual concepts by mining the cooccurrence of these concepts in some initial search results and find that this process can significantly increase retrieval performance.

We observe that many traditional multimedia search systems are blind to structural cues in datasets authored by multiple contributors. Specifically, we find that many images in the news or on the Web are copied, manipulated, and reused. We propose that the most frequently copied images are inherently more "interesting" than others and that highly-manipulated images can be of particular interest, representing drifts in ideological perspective. We use these cues to improve search and

summarization. We develop a system for reranking image search results based on the number of times that images are reused within the initial search results and find that this reranking can significantly improve the accuracy of the returned list of images especially for queries of popular named entities. We further develop a system to characterize the types of edits present between two copies of an image and infer cues about the image's edit history. Across a plurality of images, these give rise to a sort of "family tree" for the image. We find that this method can find the most-original and most-manipulated images from within these sets, which may be useful for summarization.

The specific significant contributions of this thesis are as follows. (1) The first system to use a machine learning-based approach to discover classes of queries to be used for query-adaptive search, a process which we show to outperform humans in conducting the same task. (2) An in-depth investigation of using visual concept lexicons to rank visual media against textual keywords, a promising approach which provides a keyword-based interface to users but indexes media based solely on its visual content. (3) A system to utilize image reuse trends (specifically, duplication) behaviors of authors to enhance retrieval in web image retrieval. (4) The first system to attempt to recover the manipulation histories of images for the purposes of summarization and exploration.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this thesis, we propose several new directions in semantic visual search which aim to leverage many unique aspects of modern multimedia databases.

1.1 Background

From photographs of our families and friends to recordings of our favorite television shows and movies, images and videos affect our lives every day. In recent years, with the precipitously falling price of digital cameras and the rapid spread of broadband Internet access, the raw volume of media created every year has been exploding and the degree to which most of this media is freely available online has been staggering. To most effectively use this media, whether for recalling fond memories or researching current news events, we require tools for indexing these multimedia collections and enabling easy summarization, exploration, and search across the documents.

The closest example for the desired capabilities of such a multimedia search system is, perhaps, the text search engines, particularly Web search engines, with which many people are already familiar. Here the mode of interaction between the user and the system is highly intuitive: the user provides a few keywords and the system returns documents that are relevant to those query terms. Even the basic mechanism for such a system is easily imagined: the relevant documents will contain the query terms in the body of their text.

The management, indexing, and retrieval of multimedia documents is, of course, an entirely different animal when compared to text information sources. Whereas in text retrieval, the documents can be successfully represented as a collection of words; in multimedia retrieval, there is often no such logical semantic unit for representing the content of the documents. Instead, images and videos are really only composed of matrices of pixel colors and audio streams. There is clearly a disconnect between the information that is actually contained in the document (i.e., just the pixel intensities) and the higher level semantic meaning that a human observer, or search system user, would like to attach to the document. This is often called the "semantic gap."

On the other hand, many modern visual collections also present many opportunities for applications and methods that go beyond the feasible approaches of text-based collections. Many natural collections of visual data contain a richness of information from a variety of different sources that open up many possibilities for advanced retrieval techniques. For example, searching over images gathered from the Web opens up the opportunity to associate text with the images via the filenames of the images or the text around the image on the webpage. Broadcast news video collections can be reasonably searched using text-based methods over the closed captioning or speech recognition transcripts, but the video stream is also a rich source of information which can be leveraged for a variety of enhancements, such as extracting the story structure of the news broadcast or linking related stories via detecting duplicated visual scenes across diverse news sources. So, while extracting high-level semantics directly from visual data remains a challenging area of research, *the use of visual information to augment rough semantics gathered from related text and other sources is a robust and promising application area*, which can have an impact in a variety of forms, from retrieving relevant documents, improving the relative ordering of results, or creating useful visualizations and summarizations of data.

Current challenges in semantic visual search of multimedia collections, therefore, lie in discovering various useful information cues from within the documents and collections and then designing ways to effectively use a variety of information sources in unison to adequately respond to queries.

To date, there have been a variety of systems proposed for search over multimedia collections. Many of the earliest image and video retrieval systems recognized the diversity of possible information sources available in visual databases and incorporated a variety of tools for enabling search over visual content [7, 8, 68, 19, 67, 5]. One such tool is the query-by-example search method, whereby users could provide external examples, or use images from within the database to find other images that were similar in various low-level features, such as color distributions, textures, and spatial layout. Other systems included query-by-sketch search methods, where users could draw an approximation of the images that they are seeking and find images with similar spatially distributed color schemes. Other systems, still, made use of any textual information that was available to be associated with images (such as filenames and webpage body text) and enabled search via text keywords. (In fact this text-based approach is currently available in several commercial image search engines across billions of images on the Web.) A commonality across many of these early systems, though, is the seeming recognition on the part of the designers that no single combination of all the available search methods would be appropriate for every query seen by the system. So, despite the ability of many of the search engines to fuse across several different search methods, the selection of tools to use and the degree of fusion across those tools was often left to the users, who would be expected to gain some level of expertise in using the search system.

In recent years, much of the research in image and video retrieval has been driven in large part by the NIST TRECVID video retrieval evaluations [66], which are open benchmarks, wherein research teams can establish comparisons between a variety of image and video search techniques via evaluations on commonly shared data on identical tasks. In the years of 2003 to 2006, the TRECVID benchmark data was largely focused on broadcast news videos. In each year, NIST would provide a common set of broadcast news videos, typically on the order of 50-100 hours in size. The videos were accompanied with a common segmentation of the videos into shot units along with speech recognition transcripts [21] and machine translation of the speech recognition into English in the case of foreign news sources. One of the key evaluation activities has been for the "search" task, wherein NIST provides a set of 24 standard query topics that each team participating in the evaluation must submit to their system for processing. The results from all participating teams are then evaluated and a pooled set of ground truth labels is determined by NIST. The query topics provided typically consist of a natural language textual description of the required results (like "Find shots of Condoleezza Rice," or "Find shots of helicopters in flight") along with five to ten example query images or video clips. The task of the search system, then, is to find shots from within the search data that match the needs expressed by the query.

Search across TRECVID collections is typically done using a few core tools: text search, allowing keyword-based queries against the speech recognition transcripts of the videos; image matching, allowing the user to provide example images and find other images in the search set with similar low-level features; and, in more recent systems, concept-based search, using a set of pretrained visual concept detectors which can be selected and fused based on text keyword queries. These methods are representative of the core tools used in state of the art systems, but, in principle, any set of tools can be used. In most successful systems, each tool is applied independently and the results are combined through a weighted summation of either scores or ranks, resulting in a fused multimodal ranking of documents in the search set.

In more recent history, there has been a proliferation of new types of multimedia collections resulting from the popularity of social media sites, where users post photos or videos and provide lightweight textual annotations. Users can also create lists of friends and infuse the system with some explicit notion of the social relationships between the users. Some even allow for annotating the geographic location at which the piece of media was acquired. Accordingly, there has been great interest in leveraging these new types of cues in multimodal retrieval tasks that are much in the same vein as the TRECVID tasks.

The problems and opportunities that arise within a variety of collections, such as those used in TRECVID or those springing up around the Web, are largely the same. In each collection there are a number of valuable cues that can be used directly to solve problems in retrieval. The problems, then, are discovering the most-powerful cues in each type of collection and designing a system to best use each of these cues and their combinations to answer each individual incoming query.

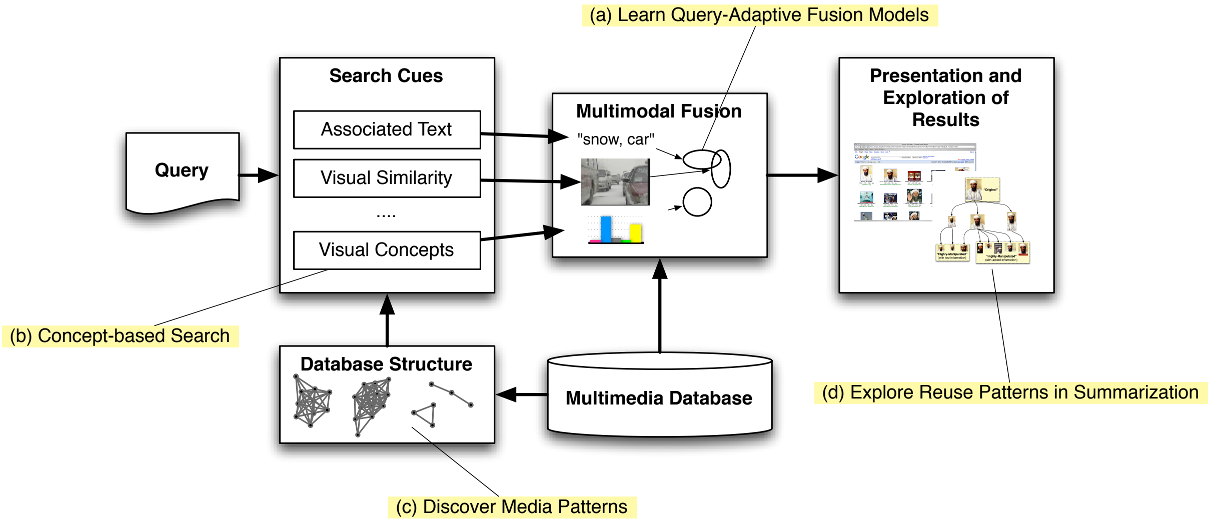


Figure 1.1: Architecture of a generic multimedia search engine with highlighted areas for improvement.

1.2 Motivations

Though a great deal of progress has been made in some of the core aspects of semantic visual search, there is still much more room for improvement. Many of the existing methods fall short by failing to adequately incorporate many of the important realities of many practical issues, ranging from the ways in which users will be willing (or able) to enter queries to important cues in newer types of databases, to the ways in which users would most like to interact with and explore the returned search results.

In Figure 1.1, we show a general schematic for visual search with each of the areas with opportunities for improvement highlighted. We will discuss each of these opportunities in more detail in the remainder of this section.

As we have discussed, many multimedia databases contain a rich supply of information from a variety of cues beyond the visual content itself, be they the text surrounding an image on a Web page or the nature of the audio stream in a video.

In the literature, there have been many methods proposed to use each of these cues individually (such as query-by-example image search or text search over associated text) and even to merge these methods together. By applying these approaches, what immediately becomes apparent is that different search methods work with varying degrees of success depending upon the query and the search set [38], so we need to be able to adapt our search strategies based on the intention of the user and the ability of each available method to address that need. How can we develop such models for query adaptation? Can they be learned automatically?

In recent years, some large collections of visual concept lexicons have been made publicly available [56, 73]. These lexicons typically consist of sets of objects, scenes, people, or locations, (such as "person," "animal," "boat," or "sky," among hundreds of others), which can be visually observed in an image or video. The key piece for these sets of concepts is that each concept has labeled examples of images or video shots in which it either appears or does not appear. Given enough of these labeled examples, we can learn models to predict their presence in an image, given only the low-level visual content of the image. These resulting models can be applied to a set to be searched over and effectively provide some soft labeling of the semantic visual content of the images and videos. The challenge, then, lies in finding intelligent ways to leverage this rich, yet often insufficiently reliable, semantic information to better answer textual queries provided by the user. Intuitively, it is straight-forward to simply leverage the "boat" concept detector for textual queries for "boats." Such an approach to visual search is enticing since it essentially allows users to semantically search over visual content using only textual keywords, effectively bridging the semantic gap, but there are still many issues. What if an incoming query has no directly-matching concept available in the database? Could we softly map the query for "boats" to other intuitive concepts, like "water" and "sky?" How do we handle the imperfect accuracies of such automatic detectors? Is an accurate “water” detector better than a modest “boat” detector?

In many modern multimedia databases, such as the world wide Web or social media sites, there is a wealth of knowledge that springs from the way in which they are created. These databases are created by contributions from a multitude of different authors, who consistently recapture, redistribute, or otherwise reuse various pieces of media. If we follow the intuition that the fact that a photographer chooses to photograph and share a particular scene or that a website author chooses to post a particular existing photo indicates that each of these individuals feel that there is a certain degree of importance or relevance embedded in the piece of media, then when many different photographers or Web authors start showing these behaviors around media, their cumulative actions begin to confer “authority” to highly reused pieces of media. This notion of authority, which comes from the reuse patterns surrounding a piece of media, is an important signal that can be incorporated into a multimodal visual search and exploration system. Indeed, these reuse patterns are exhibited across a variety of multi-author resources: broadcast news editors show footage from the day’s top stories, bloggers copy and reuse images, photographers capture popular landmarks from common angles, and concert-goers record interesting events using their camera phones. Can we capture these reuse patterns directly from the media itself and effectively leverage it to facilitate in search and summarization?

Finally, given that we can discover a sufficiently relevant set of images to answer a given query, how can we best display the results back to the searcher? Many existing systems, including many video or Web image search systems simply print the results out as a series of matrices of thumbnail images. We have noted above, however, that many practical databases exhibit significant internal structure, which might emerge from the media reuse patterns of the authors creating the data set.

Can we leverage this structure to enhance the ways in which users digest and explore the found relevant documents for a given search?

1.3 Overview of Approaches and Contributions

The primary contributions of this thesis are essentially as follows:

- **Automatic discovery of query-class-dependent models for multimodal search**. We develop a framework for the automatic discovery of query classes for query-class-dependent search models in multimodal retrieval [44]. The framework automatically discovers useful query classes by clustering queries in a training set according to the performance of various unimodal search methods, yielding classes of queries which have similar fusion strategies for the combination of unimodal components for multimodal search. We further combine these performance features with the semantic features of the queries during clustering in order to make discovered classes meaningful. We evaluate the system against the TRECVID 2004 automatic video search task and find that the automatically discovered query classes give an improvement of 18% in mean average precision (MAP) over hand-defined query classes used in previous works. We also compare the automatically-discovered classes against hand-defined ones from previous works to reveal the insights about where and how the conventional manually-built classes fail. The proposed framework is general and can be applied to any new domain without expert domain knowledge.

- **Rich concept lexicons and detectors for enhanced visual semantic search**. We propose to incorporate hundreds of pre-trained concept detectors to provide contextual information for improving the performance of multimodal video search [40]. We test simple concept-based search methods in the context of a query-adaptive multimodal search system (such as the one described above) and find that this approach is among the strongest tools available for visual search. We further evaluate the effects of increasing the number of available visual concepts and find significant gains in retrieval performance, though they are not nearly equal in magnitude to the relative increase in lexicon size. We then propose a new approach to concept-based search, which takes initial search results from established video search methods (which typically are conservative in usage of concept detectors) and mines these results to discover and leverage relationships between the results and hundreds of other concepts. We then use these relationships to refine and rerank the initial video search result. We test the method on TRECVID 2005 and 2006 automatic video search tasks and find improvements in MAP of 15%-30%. We also find that the method is adept at discovering contextual relationships that are unique to news stories occurring in the search set, which would be difficult or impossible to discover even if external training data were available.

- **Leveraging community reuse patterns for multimedia retrieval**. We present a novel approach to mining multimedia reuse patterns from multi-author databases and apply the patterns for search ranking and exploration. The approach takes a set of related photographs and finds patterns of reuse from within the set. This "reuse" can manifest itself as instances of the same image that have been copied from the Web, manipulated, and re-posted elsewhere. We evaluate the approach against Web image search queries by finding duplicated images across the results returned from a standard Web image search engine. We merge sets of duplicate images into duplicate clusters and rank the clusters according to their size. We find that more-frequently repeated images (those found in the largest clusters) are qualitatively more relevant to the queries than less-frequently repeated images. We also find that hiding images from within duplicate clusters can significantly decrease the amount of redundancy in the returned results.

- **Extracting image manipulation histories**. We propose a system for automatically detecting the ways in which images have been copied and edited or manipulated [41]. We draw upon these manipulation cues to construct probable parent-child relationships between pairs of images, where the child image was derived through a series of visual manipulations on the parent image. Through the detection of these relationships across a plurality of images, we can construct a history of the image, called the visual migration map (VMM), which traces the manipulations applied to the image through past generations. We propose to apply VMMs as part of a larger Internet image archaeology system (IIAS), which can process a given set of related images from the Internet and surface many interesting instances of images from within the set. In particular, the image closest to the "original" photograph might be among the images with the most descendants in the VMM. Or, the images that are most deeply descended from the original may exhibit unique differences and changes in the ideological perspective being conveyed by the author. We evaluate the system across a set of photographs crawled from the Web and find that many types of image manipulations can be automatically detected and used to construct plausible VMMs. These maps can then be successfully mined to find interesting instances of images and to suppress uninteresting or redundant ones, leading to a better understanding of how images are used over different times, sources, and contexts.

1.4 Organization

The remainder of the thesis is organized as follows. In Chapter 2, we discuss the problem of visual search through the lens of a multimodal search, wherein we leverage cues from a variety of different sources. We present a model for varying the exact approach applied to leverage all of these varying cues in order to provide more reliable fused search results.

Given this multimodal model of visual retrieval, where many cues can effectively be used for many queries in visual search, we move on to explore novel modalities and cues that can be leveraged within this framework. First, in Chapter 3, we explore the idea of *concept-based search*, wherein we leverage a large lexicon of visual concepts for search over a visual database.

In Chapter 4, we go on to explore the cues of importance and authority that can be extracted from image reuse patterns. We examine images on the Web and find that they are highly duplicated and reused. We find that image search results can be significantly improved if frequently-repeated images are pushed upwards in the ranked list.

In our explorations of how images are copied and reused on the Web, we see that, in most cases, the image content is changed through editing processes, by either cropping or scaling the image, or by adding overlayed content. In the most severe cases, the content can be significantly changed, so much so that it yields images with entirely different meanings. In Chapter 5, we propose a method for detecting and characterizing the types of edit operations that could plausibly have taken place between two images and check the consistency of the operations to determine if either one could plausibly have been derived directly from the other. Calculating these parent-child derivation relationships across a plurality of images yields a graph structure of the history of the image, which we can use to visualize and explore search results and discover interesting instances of images.

Finally, in Chapter 6, we offer some conclusions and propose directions forward for future research efforts.

Chapter 2

Query-class-dependent Models for Multimodal Search

2.1 Introduction

In multimodal search applications, we are dealing with large sets of documents which contain information and cues in a number of different modalities. On the Web, we can think of the documents as being individual webpages, which are composed of information from several different modalities, such as the text they contain, the structures (such as titles, sections, and metadata) associated with that text, as well as the count and quality of other pages linking to them. In personal image collections, we can think of the documents as the low-level visual content of the images as well as metadata (such as keywords, dates, and sizes) associated with them. And in video databases, we can think of shots or segments of video as the documents, which are composed of information from a number of different modalities, such as low-level visual content of the images in the video stream, the qualities of the audio stream, and the transcript of the words being spoken (captured via automatic speech recognition (ASR) or closed captions).

In each of these situations in multimodal search, we are tasked with finding the

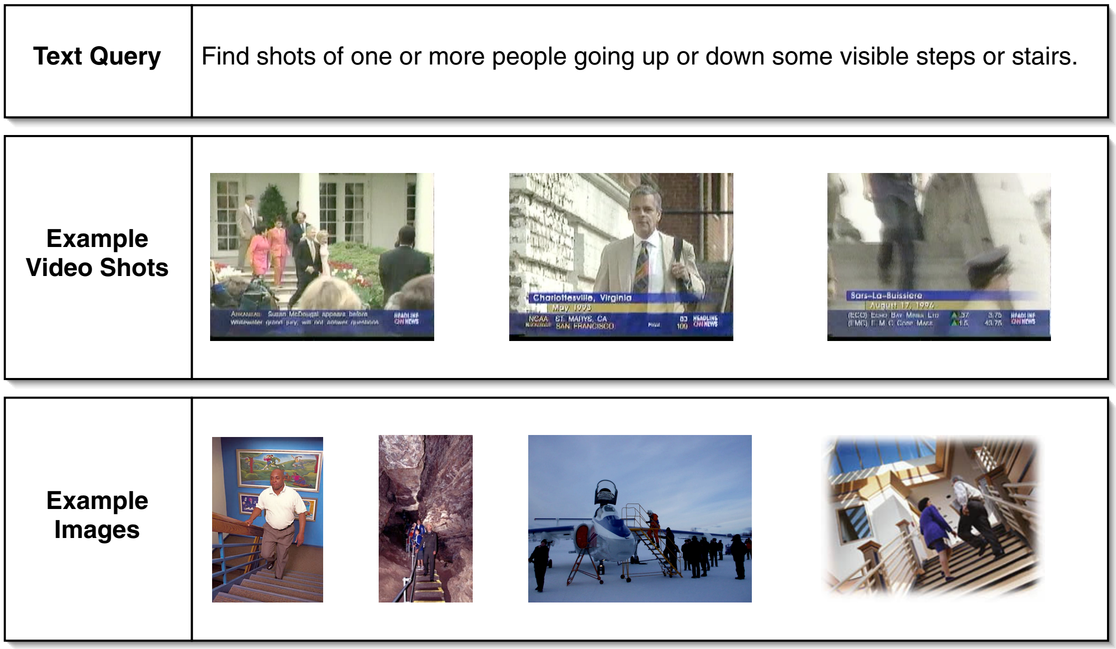

Text Query
Find shots of one or more people going up or down some visible steps or stairs.
Example Video Shots
Charlottesville, Virginia
Sars-La-Buissiere
Example Images

Figure 2.1: An example multimodal query for a video database, including text, example video shots, and images.

most relevant documents in the index according to some query which may itself be multimodal. In the case of multimodal search over a video database, we may issue a query with a set of keywords and a visual example (such as a shot or an image). An example of such a query is shown in Figure 2.1. A straight-forward strategy for finding relevant shots for this document might be to issue two unimodal queries. The first would be a text search over the ASR transcript of the videos and the second would be a content-based, query-by-example image search over the images contained in the video stream. We would then need to come up with a fusion strategy for merging the results from unimodal searches into a multimodal ranking of relevance. We would quickly discover that there are limitations to fusing the unimodal searches using a *query-independent* model, where the same fusion strategy is used for every query. The best fusion strategy for each given query is not necessarily the best fusion strategy for other queries and in certain cases the text search may be dominant while the image search only degrades the results (or vice versa). So, we would then benefit from developing a model wherein we choose the fusion strategy individually for each query: a *query-dependent* model. Of course, it's not feasible to know the proper fusion strategy for every possible query in advance. So, to address this need, recent work has led to the development of *query-class-dependent* models, which classify queries into one of several classes. Search strategies are then tuned to optimize average performance for queries within the same class. So, each query within the same class uses the same strategy, but different classes of queries may have different strategies.

Query-class-dependent modeling can give significant improvement over query-independent modeling, but there is still the unanswered fundamental question of how, exactly, to define the classes of queries. In previous work [83, 13], the classes used have been human-defined, typically by system designers examining some set of queries and identifying trends of semantic similarity that seem to be present. The classes discovered by these methods range from "Named Persons" to "General Objects" to "Financial" to "Sports." The results have confirmed the general benefits of query-class-dependent modeling over query-independent modeling, but they have not shown whether the hand-defined query classes are truly meaningful or optimal.

If the end goal of query-class-dependent models is to find query-classes wherein the search fusion strategies for the queries within the class is consistent, then it seems that the definitions of query classes should have the constraint that each of the queries within a class has similar performances in the various unimodal search methods. To address this constraint, we propose a data-driven method for the discovery of query classes, in which we rely on a set of example queries with labeled ground truth relevance. We discover query classes by clustering in a "Performance Space," which is defined by the performance of the query in various unimodal searches, as well as a "Semantic Space," which is defined by the semantic content of the query. Once query classes are discovered, fusion strategies for each class are found and unseen queries can be mapped via the semantic space to the class with the best fusion strategy.

We implement the model for the TRECVID 2004 video search task (an annual standardized evaluation of video retrieval systems administered by NIST) and find that we can increase performance from a MAP of 6.05% (using hand-defined query classes) to a MAP of 7.11% (using automatically discovered performance-based query classes). The results confirm the usefulness of query-class-dependent models and show that performance-based query classes outperform hand-defined query classes. The results also suggest that some hand-defined classes, such as "Named Person" and "Sports," do have consistency in performance, while other hand-defined classes, such as "Named Object" and "General" should either be split apart into subclasses or removed altogether and replaced with some other class, such as "Named Location," "Vehicles," or "Animals."

The unique contribution of this work is a framework for automatically discovering query classes which have consistent performance across various unimodal search methods and, thus, consistent search method fusion strategies. The method is the first approach to use a performance-based strategy to automatically discover query classes. By including the semantic space, the framework can handle unseen queries, with no known performance features, and map them to performance-based classes. The framework can be deployed over any new task with little expert knowledge and can alleviate the errors in class selection caused by human factors. We have uncovered query classes which have not been considered in previous work and can motivate future research in video search.

In Section 2.2, we will discuss previous and our proposed framework. Our experiments will be discussed in Section 2.3. In Sections 2.4 and 2.6, we will discuss and summarize the results of the experiments. We discuss related work in Section 2.5

2.2 Query Class Discovery

Having seen the limitations of query-independent fusion in multimodal search, some recent efforts in the TRECVID video search task, have implemented query-class-dependent models and have seen significant improvements.

A group from Carnegie Mellon (CMU) [83] defined five classes by hand: "Named Persons" "Named Object," "General Object," "Scene," and "Sports."

Another group from the National University of Singapore (NUS) [13] also developed a query-class-dependent model. Their model had six hand-defined classes: "Person," "Sports," "Finance," "Weather," "Disaster," and "General."

In both cases, fusion strategies are determined for each class (either by learning over a training set, or hand-tuning). The test queries could be automatically mapped to classes using some light natural language processing on the text of the query (since the classes of queries tend to have similar linguistic properties). The findings both confirm that the use of these query classes in a query-class-dependent model provides significant improvement over a query-independent approach. The work from both groups gives good reason to believe that there exists some way to classify queries where intra-class queries have very similar fusion strategies, so much so that the query-class-dependent model can approximate the performance of a query-dependent model.

There is an open question, however, of how to optimally choose these query classes. Inspection of the query classes provided by CMU and NUS shows that there is a significant human factor to the classes discovered. Only two classes, “Named Person” and “Sports,” are shared between the two sets. Some classes, such as “Finance” and “Weather” from NUS, are not representative of the types of queries typically used: there are virtually no such queries in the TRECVID search task. And some classes, such as the “General” class from NUS, are too broad: approximately half of all TRECVID queries fall into this category. And finally, the classes given by NUS and CMU seem to be chosen mostly by the semantic similarity between the queries they contain. The end goal should be the discovery of classes that contain queries that are best served by similar unimodal fusion strategies.

2.2.1 Performance Space

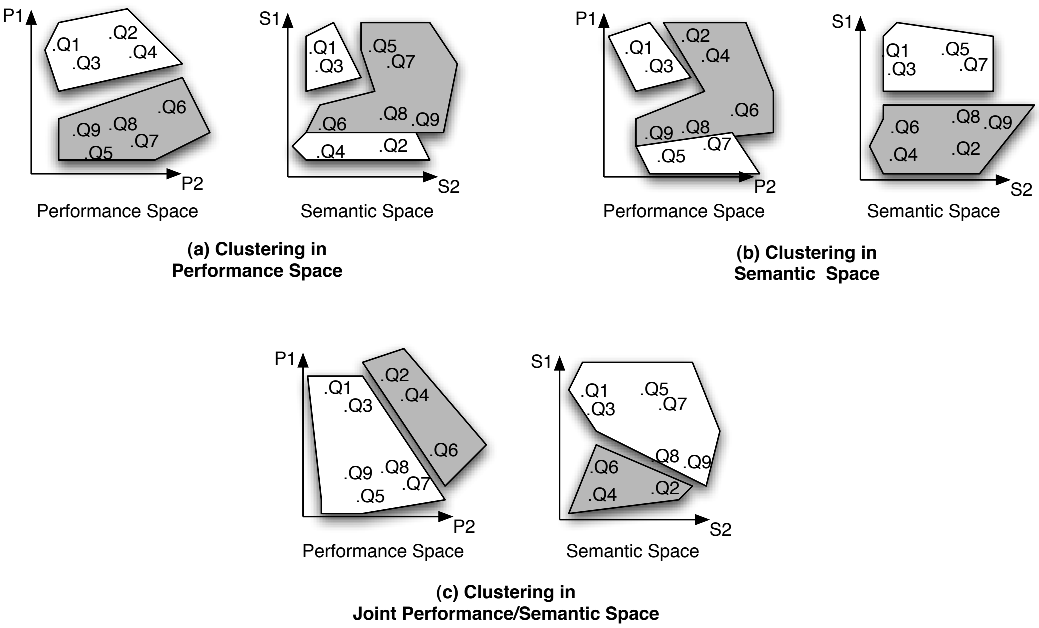
In our work, we propose a framework for automatically discovering classes of queries which have consistent performance in various unimodal searches.To achieve this class-consistency in performance, we discover the query classes by forming clusters of queries in *Performance Space*, which is defined for each query by the performance of various unimodal search methods on that query. Namely, each query, Q, is represented by the performance of all unimodal search methods, $P = \{P_1, ..., P_N\}$, where P_i is the performance measure (such as average precision, which is defined in Section 2.4) of the search method i in answering query Q against some training corpus with ground truth. Queries near each other in performance space will have similar performances in the various unimodal search methods, while queries far away from each other will have very different performances. For example, “Named Person” queries tend to have high performance in text searches and low performance in content-based image searches and could be discovered as a useful query class using this method. On the other hand, “General” queries have highly variant performance across all unimodal search methods and would most likely not be discovered as a useful query class using this method.

2.2.2 Joint Performance/Semantic Space

In later experiments, we will find that the amount of improvement due to query-class-dependency is limited by our ability to select the correct query class for each new query. New queries coming into a system do not have any known ground truth relevance information and it is impossible to predict the location of the query in performance space. We therefore need to introduce a space to describe incoming queries.

Typically, the only information available at query time (in the TRECVID task in particular) is a natural language textual statement of the query and some example images. We can therefore describe queries in terms of a *Semantic Space*, which can be composed of the characteristics of the query such as the occurrence of named entities, parts of speech, and senses of words in the text. Namely, each query, Q is represented by a semantic description vector, $S = \{S_1, ..., S_N\}$, where S_j are the linguistic features extracted from Q. In fact, we can think of the hand-defined queries from previous work as being defined in this semantic space, without knowledge of the performance space.

The semantic information about the queries can be leveraged in a number of ways. A first attempt might be to discover query classes by clustering in performance space and then attempt to learn a mapping between the performance space clusters and the semantic space. A more useful solution might be to temper the performance space clustering with constraints for semantic similarity as well as performance similarity during clustering. This can be achieved by forming a *Joint Performance/Semantic Space.* The joint performance/semantic space is simply composed of the combination of the unimodal search method performances



(a) Clustering in Performance Space

(b) Clustering in Semantic Space

(c) Clustering in Joint Performance/Semantic Space

Figure 2.2: Conceptual view of the mapping of queries and clusters into performance, semantic, and joint performance/semantic spaces. Performance space (P1,P2) is defined in terms of the performance of two example unimodal search methods. Semantic space (S1,S2) is defined in terms of some features which can be extracted from queries in the absence of performance information. The Queries (Q) are shown mapped to their respective locations in each space. Clusters found in the various spaces are shown in white and gray boxes.

(the performance space) with the query term features (the semantic space). The distances in the performance and semantic spaces can be measured using different metrics and can be combined through weighted summation to arrive at the joint performance/semantic space.

Figure 2.2 conceptually shows the positioning of hypothetical queries and discovered clusters in semantic, performance, and joint peformance/semantic spaces. We see that queries are mapped differently in performance and semantic spaces. Discovering query classes through clustering in performance space alone can cause the classes to be ill-formed in semantic space, making it difficult to map new queries to the correct class. Defining query classes through semantic space alone (like in hand-

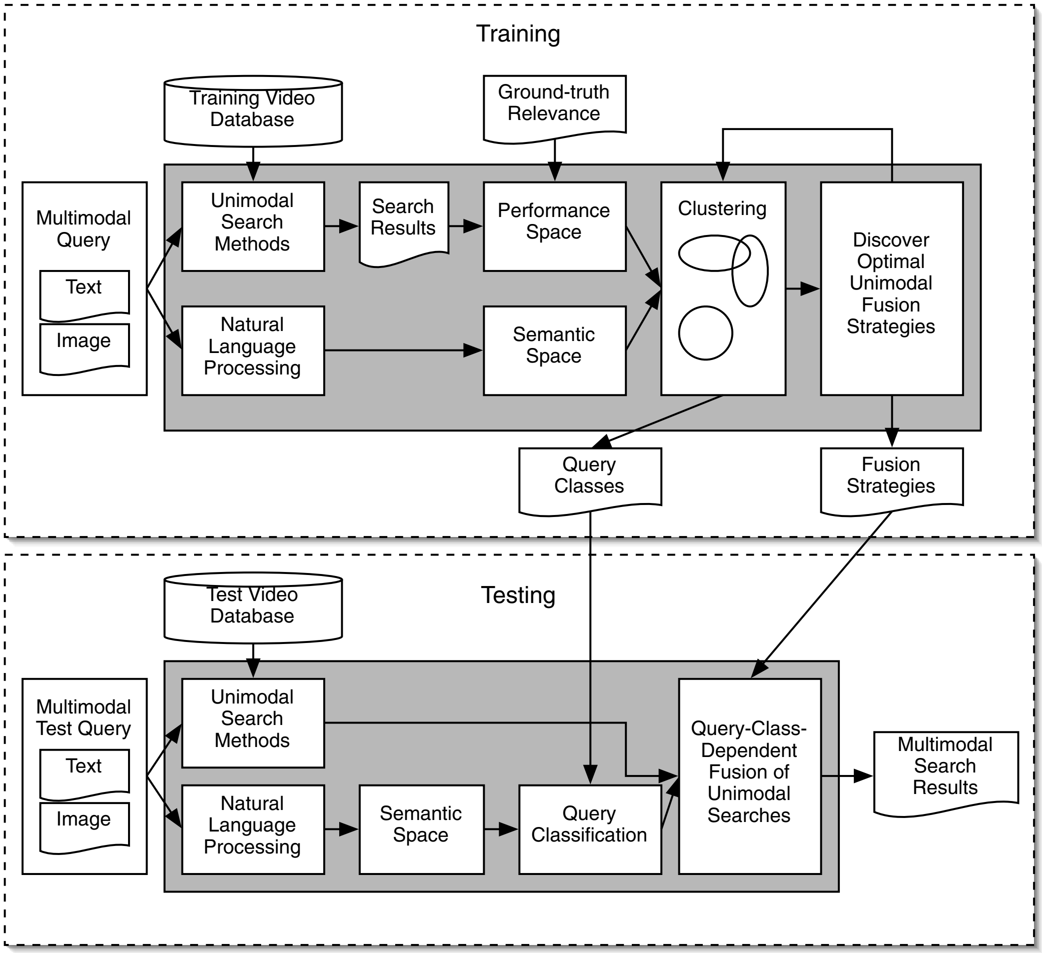


Figure 2.3: Overview of framework for query class discovery by clustering in joint performance/semantic space (in the "Training" phase) and the application of the discovered clusters to a query-class-dependent model (in the "Testing" phase).

defined schemes) can lead to clusters which are ill-formed in performance space, causing difficulties when trying to choose the best fusion strategy for each class. Discovering the query classes through clustering in joint performance/semantic space can lead to classes with consistency in both performance and semantic space, allowing for clear choice of fusion strategy and easy mapping of new queries.

2.2.3 System Architecture

Figure 2.3 shows an overview of the system architecture for our framework. The system consists of two major components, one for training and another for testing.

2.2.3.1 Training

During training, we rely on a collection of queries with ground truth relevance labels. We use any number of unimodal search methods, run them against the queries, and evaluate the results against the ground truth to determine the performance space for each query. We then run some light natural language processing analysis on the queries to extract their semantic space features. The joint performance/semantic space is then used in clustering to discover the query classes. Optimal unimodal fusion strategies are then learned for each class of queries. The discovered query classes (along with their semantic-space characteristics) and the optimal fusion strategy for each class are then passed along to the testing component.

2.2.3.2 Testing

During testing, we take new queries, which have unknown ground truth and apply the query classes and fusion strategies learned in training to score the relevance of video shots in the test database. Each new query is processed to extract its location in semantic space, which is then used to select the proper class. The unimodal search methods are run, the results are fused according to the rules for the given class, and the combined multimodal result is returned.

2.3 Components and Experiments

We conduct experiments to verify the utility and performance of our query class discovery framework using the NIST TRECVID 2003/2004 corpus, which consists of more than 150 hours of news video from a series of 30-minute broadcasts from ABC and CNN collected throughout 1998. The corpus is divided into three major sections: the development and test sets from the 2003 evaluation and the test set from the 2004 evaluation. A reference automatic shot segmentation (with over 100,000 shots in total) is given along with the output from an automatic speech recognition system [21].

We evaluate the system against the TRECVID 2004 automatic video search task. In this task, we are given a set of 23 multimodal queries containing textual natural language statements of information need as well as example video shots and static images showing samples of the desired type of shot. Figure 2.1 is a real example query from the TRECVID 2004 search task. In the automatic video search task, the system must parse this query without any interaction from a human and rank shots in order of their relevance to the query. To train the system we develop against the test data from TRECVID 2003 using the 25 multimodal queries defined from that year's video search task.

To bolster our set of example queries, we define an additional 143 queries with natural language text queries and example shots. We define these queries by studying a log of over 13,000 actual requests for footage from a video archive at the BBC in 1998. We filter through the queries by hand and choose only the queries which would be appropriate for an archive of news footage such as the TRECVID 2003 search set. We also make an effort to choose queries in the style of the TRECVID queries, which tend toward searches for visual concepts in the video stream rather than news topics specified by the speech of the reporter.

Once we have established the set of queries, we find visual examples for each query by browsing the TRECVID 2003 development set to find example video shots for each query. We then generate “pooled” ground truth relevance labels for each of the queries by running some search methods for each query against the set and evaluating only the top-ranked results. Shots which are human-evaluated to be relevant are counted as relevant shots. All other shots (human-evaluated to be irrelevant or not human-evaluated at all) are counted to be irrelevant. NIST uses this method to generate the ground truth when evaluating the search task. In total, we evaluate approximately 3000 shots for each query. We then drop queries which have no example images found in the development set or fewer than 5 total relevant shots in the search set. We are left with 89 new queries that we’ve defined with pooled ground truth. We merge this set with the 23 usable queries from TRECVID 2003, giving us a set of 112 total queries, which we use to discover query classes and train our query-class dependent model.

The validity of the learned classes is certainly sensitive to the types of queries found in the training set. To be sure that the training queries are representative of the testing queries, we use a very similar method to NIST for determining worthwhile queries (browsing the BBC log). The fact that the queries come from a real-world application also ensures that the classes learned are, indeed, useful for real-world video search. If, in other deployments of the framework, we were to discover that the query-class-dependent modeling was not outperforming the query-independent model, we could back off to the query-independent model, or re-tune the system to accommodate the types of queries that it is seeing.

2.3.1 Search Methods

The query class discovery framework relies on a set of unimodal search methods which we use to define the semantic space. In principle, these component search methods can be any reasonable ranking of document relevancy with respect to some aspect of the query. They need not be unimodal and could even be composed of various combinations of simpler search methods. In this work, we keep the search methods as fundamental as possible. We are limited in the number of methods that we can reasonably use, since we have only 112 queries to cluster and must keep the dimensionality of the performance space small.

We include six search methods in our performance space: four are variants on text search against the ASR transcript, one is a content-based image retrieval method comparing query images against keyframes of shots from within the search set, and the last is a specialized named person search method, which incorporates text search information as well as time distribution and face detection information. All have been shown to be useful for searching news video [83, 2, 13].

2.3.1.1 Text Retrieval

In text retrieval, we conduct text searches against the ASR transcript of the broadcasts using key words extracted from the natural language text queries. The text associated with each shot is the entire text of the story in which it is contained. A “story” is defined as a segment of the news broadcast with a coherent news focus and at least two independent, declarative clauses. It has been shown that stories are good segments for semantic associations between words and visual concepts [2]. For training, we use ground truth story boundaries as provided by the LDC. For testing, we use automatically-detected story boundaries [31].

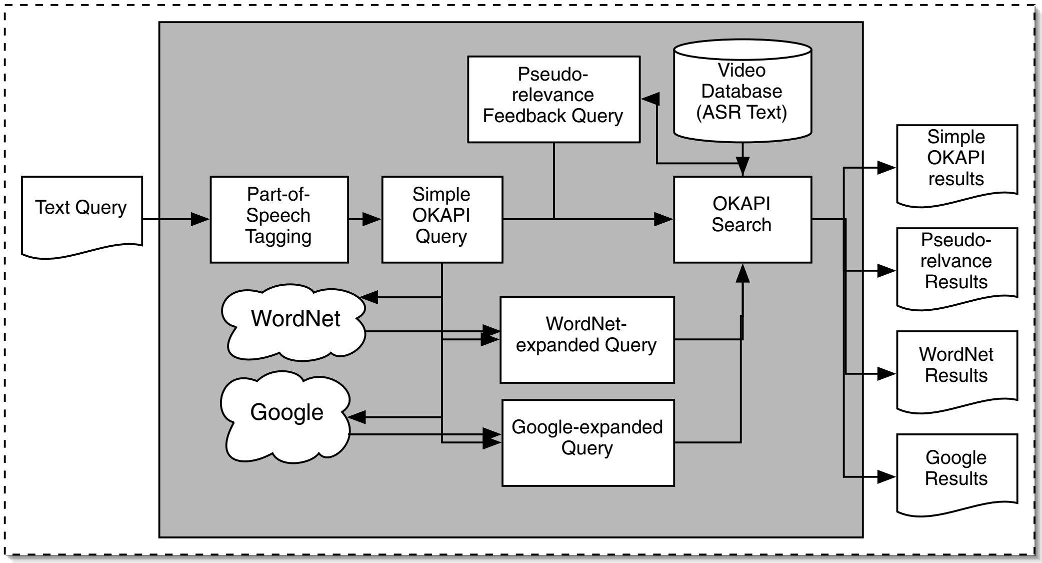


Figure 2.4: Overview of automatic text query processing. The natural language query is processed and expanded to give four unique sets of search results.

The text is then stemmed using Porter's algorithm [60] and stop words are removed. Retrieval is done using the OKAPI BM-25 formula [62] with key words automatically extracted from the natural language text queries. The manner in which the key terms are extracted from the text queries is the main point of difference between the four text search methods used. The query keyword extraction process is summarized below and in Figure 2.4. The keywords, extracted using the various methods for a few example queries, are shown in Table 2.1. We see that various queries have varying qualities of keywords found by each method, which has a direct impact on the performance of the various methods for search and indicates a need for a query-class-dependent model for properly choosing when to use each method.

Simple OKAPI In simple OKAPI retrieval, we just use keywords which we automatically extract from the text query to search against the ASR text associated with the videos. The text query is analyzed with a part-of-speech tagger [50]. Standard

Original Query	**Simple OKAPI**	**PRFB**	**WordNet**	**Google**
Find shots of the sun.	sun	sun computer microsystems	star sunlight sunshine light visible radiation	sun solaris software system storage
Find shots of Osama bin Laden.	osama bin laden	osama bin laden afgahnistan taliban	osama bin laden container bank	osama bin laden usama fbi wanted
Find shots of pills.	pills	pills viagra pfizer	pills lozenge tab dose	pills prescription drug
Find shots with a locomotive (and attached railroad cars if any) approaching the viewer.	locomotive railroad car viewer	locomotive railroad car viewer germany crash wreckage	locomotive railroad car viewer engine railway vehicle track machine spectator	locomotive railroad car viewer steam engine power place locomotion

Table 2.1: Keywords automatically extracted from various queries using part-of-speech tagging (simple OKAPI), pseudo-relevance feedback (PRFB), query expansion via WordNet and Google.

stop words are removed. Nouns are taken to be the query keywords. If there are no nouns in the query, then we back off to the main verb [13, 83]. The list of keywords is then issued as a query against the ASR index using the OKAPI formula. This simple keyword extraction scheme works well for queries which have well-pointed keywords already in the text query, such as queries for named entities.

Pseudo-Relevance Feedback In pseudo-relevance feedback, we operate under the assumption that the top-returned documents from a text search are mostly relevant and we can then just take the most frequent terms in those top documents and feed those terms back in to another search. This approach can be useful for identifying additional keywords for general object queries where some of the words in the original query can return some relevant documents, which contain clues on additional related words to search to give better results [13].

We begin by issuing the same query we used in simple OKAPI retrieval. We then analyze the top M returned documents to find the top N most frequent terms that appear in those documents. The discovered terms are then added to the query and search is repeated. M and N are chosen by an exhaustive search for the single combination that yields the best retrieval results on the training set in terms of mean average precision (MAP), which is defined in Section 2.4. In our experiments, we find an M of 30 documents and an N of 5 terms to perform best.

Query Expansion In query expansion, we look to add additional terms to the query which are related to the desired search term and give improved precision and recall. The strategy for query expansion involves using the simple OKAPI query against some knowledge resources to discover related terms. Query-expansion is a long-standing technique for enhancing information retrieval results which has been shown to be useful in video retrieval [13], particularly in cases where the exact terms contained in the query are lexically or semantically (but not exactly) related to the terms contained in relevant documents. We implement two query expansion systems using two knowledge sources: WordNet [54] and Google.

We use WordNet for query expansion by extracting all the hypernyms (general terms which also denote the specific term, for example, “vehicle” is a hypernym of “train”) and synonyms (specific terms which denote other specific terms, for example, “locomotive” is a synonym of “train”) for each of the terms from our Simple OKAPI query from the database. We add N terms to the query by constructing lists of related terms for each query term and iterating through each query term, adding the top related term to the query. N is set through experiments to maximize retrieval performance on the training set. In our tests, we find an N of 10 terms to perform best on the training data and retain this value for our experiments.

We use Google for query expansion by, again, issuing our simple OKAPI query to Google. Google then returns a list of ranked documents and we can examine the top M documents to discover the top N most frequent terms in those documents. The documents are part-of-speech tagged and only nouns are retained. Stop words are removed and only the remaining terms are analyzed for frequency. Those terms are then added to the expanded query. M and N are chosen to maximize performance on the training set. Here, we find an M of 10 documents and an N of 7 terms to perform best during training and we retain these values for our experiments.

2.3.1.2 Content-Based Image Retrieval

In content-based image retrieval (CBIR), we use the example shots and images from the query to find shots from within the search set which have similar visual characteristics in some low-level feature space. CBIR is a common approach for finding visually similar images in image databases [19]. It works particularly well when the images being sought in the database have consistent visual features, compared with each other and the query image, which are distinct from other images in the database and can be very useful for addressing some of the queries in video search tasks. Shots in the search set and the query are represented by single keyframes from the center of the shot. Each still image is then represented in LAB color space and segmented into a 5x5 grid. The features representing each image are the first three moments of the distribution in each color channel in each grid. Each query shot is then matched against each shot in the search set using the Euclidean distance. The score for each shot in the search set is then just the distance between the shot and the query image. If there are multiple query images, the minimal distance

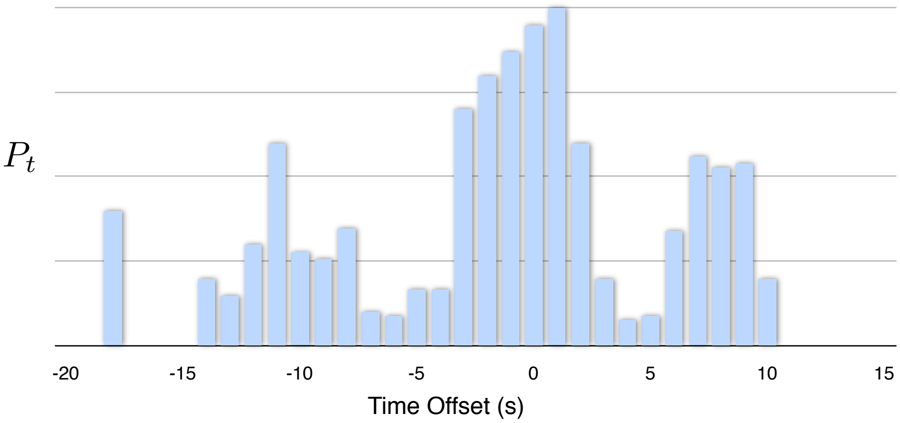


Figure 2.5: The discovered distribution for P_t, the likelihood of a person's face appearing a particular temporal distance from a mention of person's name.

between the shot and any of the query images is used. The CBIR system deployed is described in more depth in [4].

2.3.1.3 Person-X Retrieval

In previous works in the TRECVID search task, it has been shown that the retrieval of named persons has been the most important source of performance for a system, and so it has been common practice to build search methods specifically geared toward retrieval of named persons. Person-X search leverages text searches for the person's name along with a prior model of the distribution of the time delay between the appearance of a person's name in the ASR and the actual appearance of their face in the video stream. Detection of anchors and faces are also incorporated. The score for Person-X retrieval is then given as

$$P_X = \lambda P_T + (1 - \lambda)(\alpha P_t + \beta P_f + (1 - \alpha - \beta) P_{\overline{a}}) \qquad (2.1)$$

where P_f is the probability of a face detected in the shot (using the open-source OpenCV toolkit [32]), $P_{\overline{a}}$ is the probability of the absence of an anchor detected in

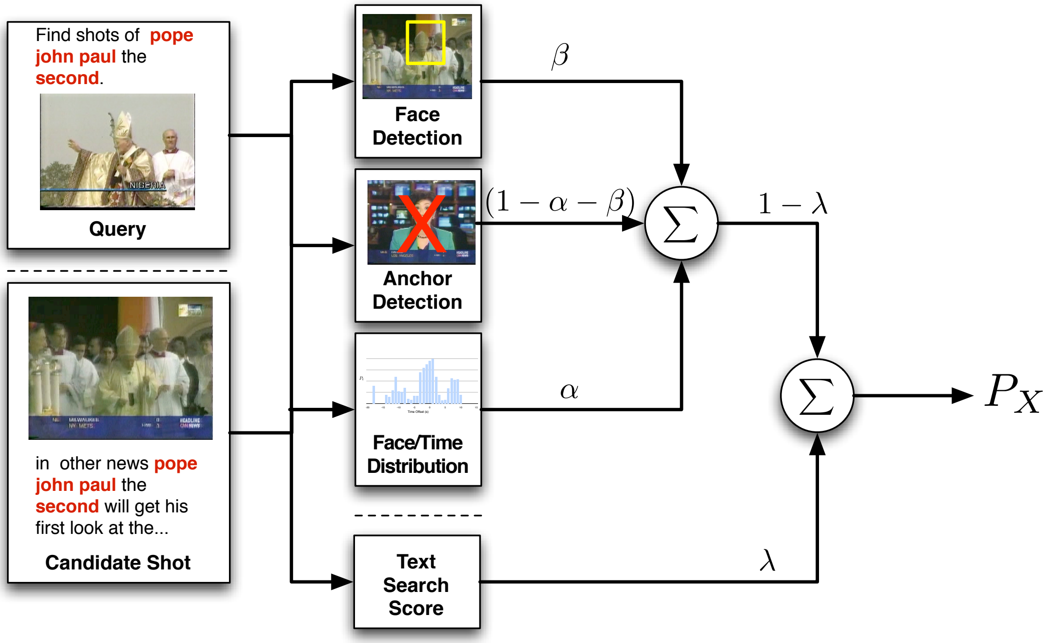


Figure 2.6: Architecture of the Person-X retrieval system, which applies a weighted layered fusion of text search scores with other facial and temporal cues.

the shot (using the approach described in [85]), P_T is the probability of a text match existing between the person's name and the text in the ASR transcript, and P_t is the probability of the shot being relevant, given its time distance from the nearest appearance of the person's name in the ASR text. This value is calculated for each shot within a story containing matched terms. λ, α, and β are weighting factors, where λ and $(\alpha+\beta)$ are constrained to be between 0 and 1.

The probability distribution for P_t is calculated over the training set by using manual annotations of relevant shots for named person queries and observing the temporal distances between these shots and the nearest occurrence of the subject's name. We group these distances into 1-second bins and show the resulting distribution in Figure 2.5. We see that there is, intuitively a large peak in the neighborhood of almost zero temporal offset between the name and the face. There are also further smaller peaks at 10 seconds either before or after the mention of the name. Beyond these distances, the likelihood of a face/name match drops off precipitously.

The weighting factors, λ, α, and β, are learned over a training set. By exhaustively trying each possible value for each weight, in increments of tenths (0, 0.1, 0.2, ..., 1) and choosing the setting that retains the highest AP over this set. This method is adapted from a similar model proposed in [26]. Our primary adaptations are to add a factor for face detection and to accord more weight to text search by separating it from all other factors using the weight, λ. This is made apparent in the system flow shown in Figure 2.6: the weights α and β affect the temporal and facial cues used by the system, but λ provides a final smoothing between all of these cues and simple text search. In the end, we find that a high weight for text search (a λ of 0.8 is best, which indicates that text is the most powerful cue here. The other features are roughly equally weighted, with α and β values found to be 0.3 and 0.4, respectively.

Previous work has also included specific-person face detection, where example shots of faces are provided by a user or mined from the Web and eigenfaces is employed to rate the appearance of a specific person's face. At this time we have not deployed a specific face detection component, but one could be incorporated into the framework.

2.3.2 Query Clustering

With the pool of training queries and the set of search methods, we set out to automatically discover meaningful query classes. The approach we take is to compute pairwise distances between each of the training queries, measured in performance and semantic spaces, and apply a clustering algorithm to discover the classes.

2.3.2.1 Performance Space

The performance space for each query is represented as a vector of performances in the six search methods defined in Section 2.3.1. Each of the search methods is run for each of the queries. The performance is measured in terms of non-interpolated average precision (AP) at 1000 returned shots, a metric used by NIST in the TRECVID evaluations, which approximates the area underneath the Precision-Recall curve. AP is defined mathematically as follows:

$$AP = \frac{\sum_{r=1}^{D} P(r)\mathrm{rel}(r)}{N} \tag{2.2}$$

where D is the number of documents retrieved, r is a particular position in a ranked list of results, $P(r)$ is the precision (the percentage of shots ranked above r that are relevant) at that particular position r, $\mathrm{rel}(r)$ is a binary value equal to 1 if the document at rank r is relevant and 0 if it is not. N is the total number of relevant documents in the collection, regardless of whether or not they are in the returned ranked list or not. The various search methods used are predisposed to having different performances: text search is usually high and content-based image retrieval is usually low. To avoid having one performance dimension dominate the clustering process, we normalize the results from the various search engines to have similar dynamic ranges by subtracting the mean from each dimension and dividing by the variance. The performance vector for each query is then L1-normalized: each value in the vector is divided by the sum of the values in the vector. This makes the performance metric in each dimension a measure of the relative usefulness of each search method for each query, rather than a measure of absolute performance. Experimentation shows that these normalization steps lead to better performance. Euclidean distances in performance space are calculated pairwise between all queries.

2.3.2.2 Semantic Space

We have developed two methods for measuring distances in the semantic space. The first method, *Query Term Features*, calculates features based on a light natural language processing of the text query. The second method, *WordNet Semantic Distance*, uses WordNet to estimate a measure of semantic similarity between the terms in two queries.

With query term features, we calculate a 5-dimensional representation of each query based on counts of nouns, verbs, and named entities appearing within the text query. We run a part-of-speech tagger as well as a named entity tagger [1] against each text query and count the noun phrases, verb phrases, named persons, named locations, and named organizations contained within each query, leaving out query stop words, such as “find” and “shot.” Differences in dynamic range between dimensions are normalized to have unit variance (without shifting the distribution mean). This normalization step is shown to empirically improve results in the query-class-dependent model. Distances between query term features are calculated pairwise between all queries using cosine distance, which is simply 1 - the cosine similarity:

$$\text{cosine distance} = (1 - \cos\theta) = (1 - \frac{\mathbf{q} \cdot \mathbf{q}'}{\|\mathbf{q}\|\|\mathbf{q}'\|}) \tag{2.3}$$

where $\mathbf{q}$ and $\mathbf{q}'$ are the vectors of two queries in query term feature space and cosine distance is the pairwise distance between them. $\mathbf{q}$ and $\mathbf{q}'$ have zero cosine distance if the proportional distributions among dimensions are equal.

For WordNet semantic distance, we use the semantic similarity metric defined by Resnick [61], which fuses semantic similarity in WordNet with probabilistic corpus-based semantic similarity. The similarity between two terms can be determined via corpus-based techniques by counting cooccurrences for the pairs of words in some corpus. This approach leads to problems since incredibly large corpora are needed to gather statistically meaningful counts for all possible word pairs. Resnick's technique overcomes this challenge by counting occurrences of hypernyms of words. So, in the case where pairwise counts for two terms are unknown, we can back off up the WordNet hierarchy and examine occurrences of hypernyms of terms. Specifically, the Resnick similarity between two terms is the information content of the lowest super-ordinate (lso, or lowest hypernym for both terms) which can be counted from the corpus (in this case, the Wall Street Journal corpus):

$$\mathrm{sim}_{\mathrm{Resnick}}(t_1, t_2) = -\log P(\mathrm{lso}(t_1, t_2)) \tag{2.4}$$

where t_i are the terms and $P(\mathrm{lso}(t_1, t_2))$ is the probability of observing the lowest super-ordinate of the two terms in the corpus. The intuition is that the information content of a term captures its specificity: less frequent terms are more specific and yield more information. If two given terms fall under a high-information hypernym, then they are likely to be highly related.

Since Resnick's measure is a similarity, we use its inverse to measure the WordNet semantic distance. Given two queries, each with multiple terms, we measure the pairwise WordNet semantic distance for each individual term in the first query against all other terms in the second query. By averaging all of these word-level similarities, we can arrive at a single similarity value between the two queries.

2.3.2.3 Joint Performance/Semantic Space

To leverage the power of both the performance and semantic spaces, we combine them into a joint space. The pairwise distances between queries in joint performance/semantic space can easily be calculated as a weighted sum of the pairwise distances in each of the performance and semantic spaces:

$$D = \alpha D_P + (1 - \alpha)(\lambda D_{QT} + (1 - \lambda) D_{WN}) \tag{2.5}$$

where D is the joint performance/semantic distance matrix, and D_P, D_{QT}, and D_{WN} are the distance matrices in performance space, query term space, and WordNet, respectively. α and λ are weights which are either set to 0, 0.5, or 1. In other words, we generate a number of different joint performance/semantic distance matrices in which particular dimensions are either entirely on, entirely off, or evenly averaged. Any weighting can be used, but we only explore these few cases, since they give us a rough measure of the value of using each space alone or in combination with others. In Section 2.4, we discuss the performances in each of these rough settings. The general finding is that using weights of 0 or 1 (essentially omitting certain components) degrades the overall performance. Setting all the weights to 0.5 (roughly averaging the distances in various spaces) yields better retrieval performance.

2.3.2.4 Clustering

Given our pairwise distance matrices between queries, we can cluster the queries using any number of clustering algorithms (from k-means [25], to normalized cuts [64], to hierarchical clustering). We avoid normalized cuts since the clusters discovered by normalized cuts are not necessarily of the form that we would like to find. Namely, the clusters can have highly irregular shapes, and since we are looking for compact clusters with consistent performance across various search methods, the normalized cut method may hurt more than it helps. K-means and hierarchical clustering are both well-suited for finding the sorts of tight clusters that we're looking for. We choose hierarchical clustering over k-means, since hierarchical clustering gives us a better method for choosing the number of clusters. Varying the number of clusters in k-means can lead to wildly different clusters, while varying the number of clusters in hierarchical clustering leads only to splitting or merging various clusters. The hierarchical clustering is performed by forming a linkage tree by iteratively joining queries with their nearest neighboring queries (or clusters of queries). We measure distances between clusters using complete link, meaning the distance between two clusters of queries is measured by the further distance between any of the pairs of queries between the two. This yields clusters that are more tightly formed, when compared to other alternatives, such as single link or average link. We can form clusters from the linkage tree by taking a horizontal cut across the linkage tree, adjusting the height of the cut in order to produce a particular number of subtrees. Each subtree forms a cluster consisting of all the leaf nodes which are children of that subtree. We choose the number of clusters by trying all numbers of clusters and picking the number which maximizes retrieval performance (in terms of mean average precision) on a held-out validation set. Here, we find a total of about 8 clusters to be best for the query-class-dependent model, though, this is most likely limited by the total number of training queries that we are clustering. Most likely, more training queries would yield more classes (an, perhaps, finer-grained classes).

2.3.3 Search Method Fusion

After query classes are discovered, we need to find optimal fusion strategies for combining the various search methods in each class. For this initial study, we use linear weighting of the scores from each of the unimodal search methods:

$$\text{Score}(c)_M = \sum_{i \in \text{search engine}} \lambda_i(c)\text{Score}_i \tag{2.6}$$

where Score_i is the score of each individual search engine, λ_i is the weight for the individual search engine, and $\text{Score}(c)_M$ is the combined multimodal score. The weights, λ_i, are conditioned on c, the class of the query.

The weights, λ_i are found through a full grid search of all possible combinations, where each weight is quantized to 11 values between 0 and 1 (0, 0.1, 0.2, etc) and it is constrained that all weights sum to one. The combination of weights for each query class which gives the best performance for that class in the training set is retained and used in the test application. The scores, Score_i, can be estimated from the raw scores, Score_{ri} (the outputs from each unimodal search method) using rank-based normalization.

For rank-based normalization, we essentially ignore the raw scores for each shot given by each of the search methods and only evaluate based on the position of each shot in the ranked list of scores [83]. This eliminates the need to scale disparate scoring metrics and helps smooth large quantizations that happen when using story-based grouping of shots, like we do with our text search. (With story text search, all of the shots in a story get the same raw score and the difference in scores between stories may be large, making it difficult to gain anything from content-based image search. Rank does not allow shots to have the same score and smoothes out the large gaps between stories). This, of course, discards a great deal of information, but it has been shown that rank-based fusion is powerful for cross-modal fusion applications [53]. The rank-normalized score is calculated as $\text{Score}_{ij} = 1 - R_{ij}/N$, where R_{ij} is the location of the shot j in the ranked list returned from search method i. N is the total number of shots.

2.3.4 Query Classification

At query time in an application situation, we need to properly choose the best class for each incoming query and employ the optimal fusion strategy for that query. With these incoming queries, we have no advance knowledge of search engine performance and we only have the semantic space coordinates of each query to aide us in our decision. In this work, we evaluate two methods to make this decision: shortest distance in semantic space and a support vector machine (SVM) using the semantic space kernel.

2.3.4.1 Shortest Distance

Using shortest distance in semantic space is straight-forward and quite powerful in our case, since the classes we find are constrained to form reasonable clusters in semantic space. To measure the distance between an incoming query and a class of queries, we take the average distance in semantic space between the query and each of the queries within the class. We then choose the class of the query to be the class with the shortest distance in semantic space.

2.3.4.2 SVM

A potentially powerful method for classification is the support vector machine [77]. SVMs learn optimal hyperplanes for binary decisions in some high-dimensional space given only a kernel matrix for that space. Traditionally, kernels such as linear, polynomial, and radial basis functions have been used as SVM kernels. Recent research, however, has shown that any kernel matrix can be used in SVMs, as long as the matrix is positive semi-definite and represents a plausible distance between examples [45, 46]. We can use our semantic space distance matrix, given by the

	Method	**Baseline**	**Classification**	
		O	**SD**	**SVM**
Query-Independent	Text	5.95	-	-
	Multimodal	5.95	-	-
Hand-defined Classes	CMU	6.05	-	-
	NUS	5.98	-	-
Automatically Discovered Classes	P+Q+WN	7.48	6.73	3.59
	P+Q	7.49	**7.11**	4.78
	P+WN	7.45	6.45	4.21
	P	7.69	1.72	3.20
	Q+WN	6.09	5.98	4.65
	Q	6.15	6.02	5.33
	WN	5.99	5.90	5.44

Table 2.2: Summary of performance (in Mean Average Precision as percentages) on the TRECVID 2004 search test set using various query-class-dependent models. Baselines are query-independent models using text-only and multimodal searches. Automatically discovered classes are shown with performance in three classification scenarios: Oracle (another baseline: the best class is known ahead of time, denoted by "O"), Shortest Distance ("SD"), and Support Vector Machine ("SVM"). Classes are automatically discovered in variants of joint performance/semantic space, in the presence or absence of the three subspaces: P (performance space), Q (query term space), and WN (WordNet space).

combination of the distances in query terms space and WordNet, and simply plug it into an SVM. Since SVMs are binary discriminators, and our application requires a multiclass classifier, we need to extend the SVM to a multiclass tool [63], which can be done using error-correcting output codes [15]. In our analysis, we find that the SVM does not perform as well as simply using the shortest distance. This is most likely due to the small size of our training set (only 112 examples).

2.4 Analysis of Results

We perform the query class discovery over our training set of queries and apply the learned clusters and fusion strategies to the TRECVID 2004 search task. The results, expressed as Mean Average Precisions ("MAP," or the mean of the average precisions for each of the 23 queries in the evaluation set), are summarized in Table 2.2. The results confirm that query-class-dependent models improve upon query-independent models and that classes discovered by clustering in joint performance/semantic space can improve upon classes defined by hand. The results also show that the choice of space for query clustering has significant impact on our ability to map to the correct class for test queries. The performance is discussed in more detail in Section 2.4.1.

Analysis of the classes discovered by clustering in joint performance/semantic space confirms that some of the query classes that have been defined by hand in previous efforts, such as "Named Person" and "Sports" are, indeed, useful classes, while other classes, such as "Named Object," can be split into subclasses and other classes, which have not been used before may be helpful. The qualities of the classes discovered are discussed in more detail in Section 2.4.2.

We have experimented with number of clusters. In these analyses, we only discuss the best case, which uses eight.

2.4.1 Performance

We use two query-independent fusion strategies as baselines: text-only (using only simple OKAPI search) and query- independent multimodal fusion ("Text" (MAP: 5.95%) and "Multimodal" (MAP: 5.95%) in Table 2.2, respectively). We find that query-independent multimodal fusion cannot outperform text-only queries, which confirms the need for query-class-dependent models to fully leverage the power of content-based image searches and textual query expansion.

We then test our fusion strategies against the hand-defined classes from the prior work of CMU and NUS (marked "CMU" (MAP: 6.05%) and "NUS" (MAP:

6.05%)) in Table 2.2. We learn optimal fusion strategies for each of query class in our training set and apply them in the test case. Both groups present rule-based algorithms, which can classify the incoming queries nearly perfectly, so we assume errorless classification. We see that these query-class-dependent models give some improvement over query-independent models.

We then apply our query class discovery framework over the training set to discover query classes. We experiment with several variations on the joint performance/semantic space for clustering queries and also with several methods for classifying incoming queries. The joint performance/semantic space is formed using all possible combinations of the performance space, "P," the query term space, "Q," and the WordNet space, "WN." For each space, we also experiment with different classification methods for assigning classes to incoming queries. The "O" classification method is the "Oracle" method, which is not a real classification method at all, but a baseline method in which we explore the limits of the discovered classes by assuming that we can automatically pick the best query class for each incoming query. The "SD" (Shortest Distance) and "SVM" (Support Vector Machine) classification methods are real classifiers, which were discussed in Section 2.3.4

We see that using the performance space to conduct query class discovery provides the best potential for overall performance, as evidenced by the "P" method with Oracle classification (MAP: 0.769). We see, however, that it is impossible to use real classifiers to recover the proper class for incoming queries, unless we incorporate the semantic space into our query class discovery framework: the classes discovered in joint performance/semantic space have only slightly degraded potential for performance and can be better recovered at query time, with a maximum MAP of 7.11%. Clustering in semantic-space alone gives performance similar to the hand-defined clusters found by CMU and NUS, with MAPs ranging from 5.90%

to 6.02%, which makes sense since those clusters were essentially found by human inspection of queries, looking for common semantic groupings.

The quality of the classification schemes is difficult to evaluate empirically: it's unclear what the "true" class membership of each incoming query is. Various classes can still have similar fusion strategies, so it's unclear what penalties a "misclassification" of an incoming query will actually have on the end performance. We can, however, look at the best-case performance from classification using the Oracle baseline. We see that in joint performance/semantic clusters, like "P+Q," we are able to map to classes which are close to the best-performing class by using the shortest distance classification method. In performance-only clustering, "P," we are unable to map to useful classes in any meaningful way at query time. This is due to the lack of cohesion in semantic space found in the pure performance clusters.

In almost all cases, the SVM classification fails to map to useful classes, which is most likely due to the small number of training examples available. Interestingly, the SVM classification provides significant improvement over shortest distance in performance-only clustering (MAP: 3.20% vs. MAP: 1.72%). This is probably since the divisions between classes in semantic space in this clustering approach is highly non-linear (since there is no constraint on semantic space similarity in performance space clustering) and SVMs are particularly adept at learning such complicated decision boundaries. This also gives hope that, given enough training examples, we may be able to exploit the power of SVMs for classification, uncovering some nonlinear mapping between the performance space and the semantic space, which would allow us to discover clusters in pure performance space, leaving the decision on semantic mappings up to the classifier.

Figure 2.7 shows the mean average precision of each of the methods discussed on the TRECVID 2004 video search task using shortest distance classification (in

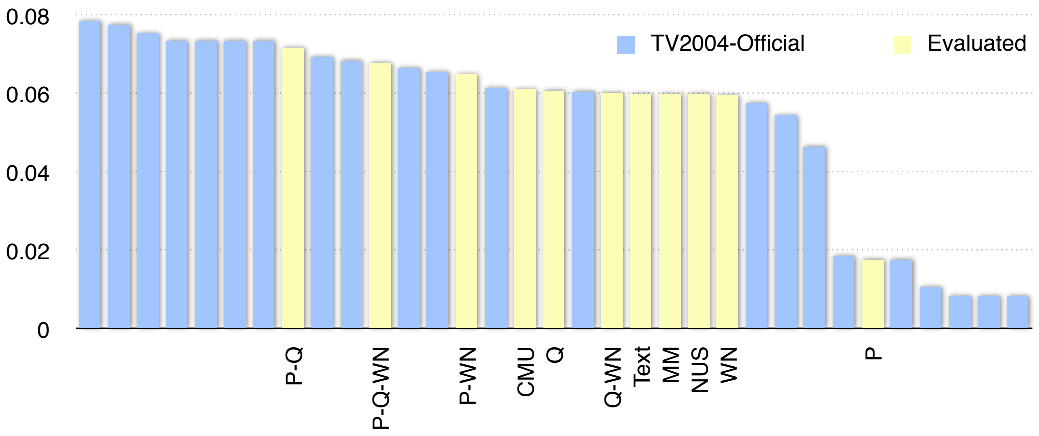


Figure 2.7: Performance (MAP) of our methods (yellow) and all official TRECVID 2004 automatic search task submissions (blue). The labels correspond to those in Table 2.2 and are described in Section 2.4.1.

yellow) against all official TRECVID 2004 runs (as scored by NIST). We see that while we are able to show improvement by automatically selecting the query classes, we are still inhibited by the quality and number of our component search methods as well as our search method fusion strategy. In this work, we focus on the evaluation of the effects of automatic query class discovery, and we leave the tasks of exploring better search methods as well as better fusion strategies to future work.

2.4.2 Discovered Classes

Figure 2.8 shows some sample queries from the classes automatically discovered by clustering in joint performance/ semantic space. The queries are segmented into clusters and each cluster shows only a few example queries. (There are 112 training queries in total). The blocks to the left of each query show the performance space mappings for the query. Each block gives the relative performance for the query in each independent search method. The bright blocks indicate high performance, while the dark blocks indicate low performance. The performances are normalized for each query, so the performances indicate which methods are most helpful and

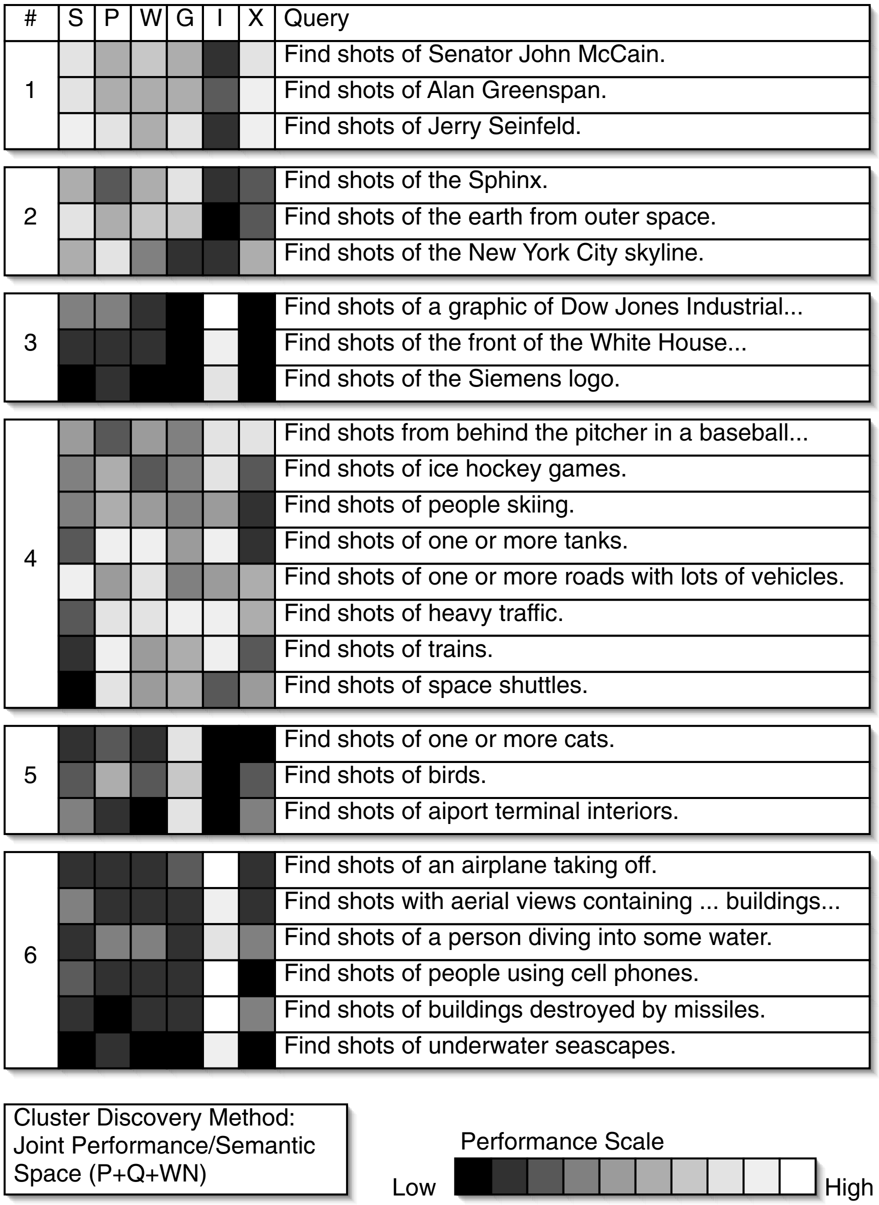


Figure 2.8: Sample queries from the clusters discovered in joint performance/semantic space. Class IDs are shown along the left side, with relative performance in each dimension in performance space to the right ("S" = Simple OKAPI, "P" = Pseudo-Relevance Feedback, "W" = Query Expansion via WordNet, "G" = Query expansion via Google, "I" = Content-based Image Retrieval, and "X" = Person-X), followed by the text of the query.

least helpful on a per-query basis. The figure shows the six classes (out of the total eight), which have interesting interpretations and consistency in performance space. The remaining two classes have no easy interpretation and are quite similar to the "General" class described by NUS.

The first cluster contains virtually all of the **Named Persons** in the training set. The queries all also seem to have consistently high performance across the various search methods, except for content-based image retrieval. Simple OKAPI and person-X searches are most helpful, pseudo-relevance feedback and query expansion seem to slightly degrade performance, and content-based image retrieval does not help at all. This cluster seems to confirm the usefulness of the "Named Person" classes defined both by CMU and NUS, since these classes are tightly-clustered in performance space.

The second and third clusters both contain many **Named Objects**. There is a distinction between the two classes, however, in performance space. In the second cluster, text-based searches seem to be the most useful, while content-based image retrieval doesn't help at all. In the third cluster, the case seems to be quite the opposite: content-based image retrieval is much more accurate than text searches. Examining the semantics of the queries seems to show that the queries in the third cluster are more graphical (such as the Dow Jones financial screen or the Siemens logo), while the queries in the second cluster seem to be more or less specific scenes or locations (such as the Sphinx or the New York City skyline). These two clusters seem to confirm the need for a "Named Object" class, as suggested by CMU, but they also seem to indicate that this class could be split into two subclasses (**Named Object** and **Named Location**) to give more cohesive classes in performance space. We may also refer to these graphical Named Object queries as Named Icon queries, since they are logos or single-view scenes, which are characterized by their repeated, iconic visual appearance.

The fourth cluster presents an interesting group of queries with various meanings. Most interestingly, many of the queries related to **Sports** and **Vehicles** end up in this class. In performance space, these queries all seem to perform similarly in all search methods, which leads to the conclusion that for these queries, all search methods (including content-based image retrieval) are equally important. This cluster seems to support the validity of the "Sports" classes identified by both CMU and NUS, but also indicates that this class could be augmented to include other types of queries, such as "vehicle" queries, since they have similar profiles in performance space and benefit from similar fusion strategies.

The fifth cluster contains mostly **Animals** (along with some other queries) and gets the most performance from query expansion via **Google**. No such animal cluster has been identified by previous work, but it seems that queries for animals have a compact class cluster in performance space and would be well-served by similar fusion strategies.

The final cluster contains a number of queries which benefit more from **content-based image retrieval** than any other search method. The queries contained within this cluster seem to be mostly scenes, but it is difficult to assign an understandable semantic label to this cluster. There is, however, some semantic relationship which can be recovered through semantic space classification, which demonstrates the power of clustering in joint performance/semantic space, particularly its ability to discover unique useful classes for query-class-dependent retrieval.

2.4.3 Computational Complexity

Our proposed query-class-discovery system is light-weight and runs relatively quickly. Each of the six described unimodal search methods runs in about 1 second on an

Intel Xeon 3.0 GHz machine against a search set of 65 hours of news video (represented as 33,000 shot keyframe images), with the method implemented in a mixture of C and Matlab code. Beyond that, fusing and sorting the results takes a trivial amount of time. So, ultimately, the system's complexity grows with the number of unimodal search methods. Training the system takes somewhat longer. Each of the six unimodal search methods must be run for each of the training queries (112 of them, in this case). Beyond that, clustering the results in performance and semantic space takes seconds. A final expensive step is exhaustively searching for optimal fusion weights for each of the discovered classes, a step which can take several hours and dominates the overall cost of training the system. This might be overcome by using machine learning approaches to learn the fusion models, instead. Beyond the computational complexity, there is also a human labor cost in terms of generating a list of ground truth annotations for a set of training queries, which might be measured in terms of days.

2.5 Related Work

This work is, of course, related to the previous works on query-class-dependent models for visual search [83, 27], which we reviewed in Section 2.2. The primary differentiation between these prior works and the work presented in this chapter is that we propose to discover query class models automatically, whereas prior efforts required human designers to create these models.

In the time since these experiments were conducted, there have been some additional relevant related works which automatically discover query-class-dependent models. In a similar work [82], Yan and Hauptmann also propose a framework for automatically determining class-dependent search models. The framework proposes a "probabilistic latent query analysis" (pLQA) model, wherein the query classes are found as a latent variable under the assumptions that queries within the same class share the same optimal weighting of individual search methods and that the plain-text description of the query has enough information to classify the query into a class. The proposed approach offers many benefits beyond the framework proposed in our prior work. Whereas our prior work uses the "performance space" as a proxy for estimating the optimal combination weights for a given query during class discovery, the pLQA model uses the combination weights directly during class discovery. And while our prior model discovers query classes first and then fusion models second in a two-stage process, the pLQA model optimizes class membership and fusion models in a single joint process. Further advantages come from the inclusion of a principled approach for determining the number of query classes and the ability to map a single query into a mixture of several different query classes. Evaluation of the pLQA model on a broad range of TRECVID queries shows significant performance gains over both hand-defined query classes, which are greater in magnitude than the improvements shown in our work. The method yields latent classes with some similarities to our discovered classes. Namely, "named person" and "sports" queries are both found. The pLQA method finds a "vehicle" class, while our method conflates this class with the "sports" class. And, indeed, both exhibit classes that are difficult to explain semantically and are likely dominated by the relative strengths of the available search methods.

In [79], Xie et al. propose a system that dynamically creates query classes as incoming queries are received (as opposed to the cases above, where query classes are learned once from the training data and kept static across all queries). To achieve this, the system measures the semantic distance between the incoming query and the pool of training queries. This semantic distance is measured with various textual query features, such as named entities and concepts, as provided by a question-answering system. The top k closest queries from the training pool are then pulled together as a dynamic query class. An adaptive fusion model is then learned on the fly, by finding the optimal fusion weights for each of the available retrieval methods such that it achieves the best average retrieval performance over this subset of training queries, and the model is applied to the query. In the end, the system only shows slight improvements over static query class methods similar to the above-described approaches; however, this dynamic query class approach is unique and shows promise. One possible shortcoming is its reliance entirely on semantic similarity between queries (and not the performance-based similarity discussed above). Further extending the work to incorporate such performance cues may lead to greater performance increases.

2.6 Summary

We have developed a framework for automatically discovering query-class-dependent models for multimodal search by defining query classes through a clustering process according to search method performance and semantic features. We apply this framework to the TRECVID 2004 video search task. We confirm that query-class-dependent models can outperform query-independent models and find that automatically discovered performance-based classes can outperform the hand-defined query classes from previous works. The query classes that we discover indicate that some hand-defined classes from previous works are useful for query-class-dependent modeling, while others should be split into subclasses or replaced with different classes.

The unique contribution of this work is a system which can automatically discover classes of queries having consistent performance in various search methods, and therefore, similar optimal strategies for fusing those search methods. The discovered classes can also be constrained to have consistent semantic characteristics, allowing us to map incoming queries, with no performance information, into appropriate performance-based classes. The process can be used to develop a query-class-dependent model for any new domain without requiring expert knowledge, while also alleviating errors induced by human factors in query class definition.

While we have shown that automatic discovery of query classes improves over hand-defined query classes, we have also seen that much improvement can be gained through choosing better component search methods and employing better search fusion strategies. In future work, we will explore ways to discover the best search methods and fusion strategies. In this work, we have seen that the *performance* of search methods for a query is as important (if not more important) that the queries semantic similarity to other queries when it comes to predicting the right fusion method. In future works, we expect similar performance-based metrics to be crucial for query adaptation.

Chapter 3

Leveraging Concept Lexicons and Detectors for Semantic Visual Search

3.1 Introduction

In the previous chapter, we have seen an introduction to a number of available tools in multimedia retrieval, particularly methods based on low-level similarity to query images or text extracted from documents surrounding a piece of media. While these methods are powerful in certain cases, they can ultimately be unsatisfying because they are essentially stop-gap solutions that skirt the difficult issue of actually indexing the content of visual databases and making it semantically accessible. Ultimately, the low-level color distributions of query images or the text keywords scraped from filenames or speech transcripts do not provide an expressive enough representation of the rich space of possible queries or relevant images.

In recent years, a promising new approach has emerged, which is based on using semantic visual concepts as a middle ground between the desired text-based search interaction and the low-level content in images and videos. In this scenario a large ontology of visual concepts is selected and visual models are learned to automatically detect the presence of these concepts. The predicted scores of all the concepts in the ontology provide a rather rich semantic representation of the visual content of the document. The textual keywords that the users provide, then, can also be mapped into this ontology of visual concepts. Ideally, a query for “ship” could be mapped to concepts for “boat,” “water,” and “sky,” for example. Then, images or videos containing these visual concepts could be returned to the user. Essentially, a rich enough set of detectable visual concepts could serve as intermediate representation between high-level textual queries and low-level visual content and provide a powerful tool for multimedia retrieval.

In reality, the mapping from a query for “ship” to a concept for “boat” might be feasible, but discovering other peripherally-related concepts (like “water” and “sky”) can be difficult. Early approaches to utilizing peripheral concepts for retrieval have included engineering intuitive filtering approaches, such as removing shots with an “anchor person” present or positively weighting the “face” concept in searches for named persons, giving small improvements [12, 27]. More recent work has included the angle of directly matching query keywords with concepts in the lexicon (like in our “ships” query / “boat” concept example) [11, 9]. However, deeper relationships to peripheral concepts are difficult to uncover, particularly in search, where the details of the query and possible relative concepts are unknown to the system [12, 9, 71].

In this chapter, we evaluate the use of pre-trained visual concept detectors for visual search. We evaluate their utility in the context of a query-class-dependent framework, such as the one proposed in the previous chapter and find that they are the single largest-contributing search component in such a system. We then examine the effects that we observe by greatly increasing the size of the concept vocabulary and find that significant increases in search performance can be achieved by including a large number of concepts, but these are usually not proportional to the human and computational strain required by these large lexicon increases. Finally, we propose a framework for automatically discovering and leveraging peripherally related concepts. The approach is largely unsupervised and is shown to give improvements of 15%-30% in mean average precision (MAP) on video search tasks. Much of the improvements are drawn from queries for named persons or sports, while the impact on other classes, where simple concept-based search models were already the dominant search tool, is less: around 12%-15%.

The remainder of the chapter is organized as follows. We begin in the next section by providing an overview of the state of the art in concept-based search and propose a new method for exploiting peripheral concepts in a large visual vocabulary. In Section 3.3, we present our experiments and analyze the results. We review related work in Section 3.4 and present a summary and suggest future directions in Section 3.5.

3.2 Concept-based Search

In concept-based search, we are essentially using the space of several hundred concept detectors as an intermediate semantic representation for both visual content and the queries that users are issuing [12, 3, 72, 71]. The detection scores of each of the concept detectors can encapsulate the visual content of the images. Textual keyword queries can then be mapped to concepts via some set of pre-defined lexical relationships. Given the concept-space representation of both queries and images, textual searches can be conducted against visual content by ranking visual documents by their similarity to the query in concept space.

In Figure 3.1, we show an example of how this framework might work with a lexicon of five visual concepts (“anchor,” “snow,” “soccer,” “building,” and “outdoor,”

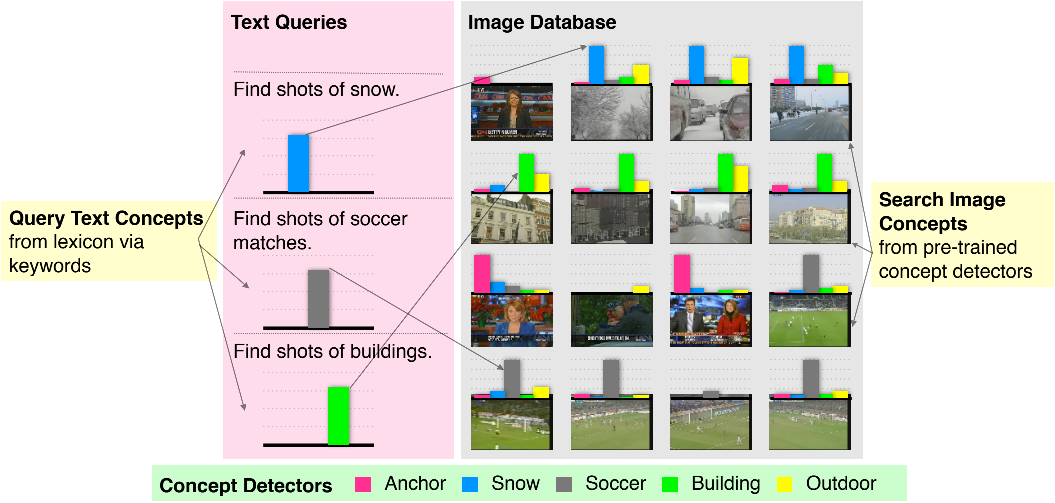


Figure 3.1: In concept-based search, both images and queries are represented in an intermediate concept space. Images are mapped into this space by their concept detection scores. Textual queries are mapped into this space based on any number of lexical or statistical methods, typically attempting to map the semantics of the query into a set of relevant concepts. Images can then be ranked based on their similarity to the query in this concept space.

shown at the bottom of the figure). Then, a set of images (shown on the right) can be represented by the detected presence of each of these concepts in the image (shown in the graphs above each image). Similarly, the incoming text queries (shown on the left) can be mapped into similar confidences of the relevance of each concept (shown in the graphs below each query). The search problem then is reduced to using the concept distributions given in the query and the concept detection scores in the images to rank the images. We generally formulate the relevance of a given image as:

$$S(I,Q) = \sum_{i=1}^{N} w(c_i,Q)P(c_i,I) \tag{3.1}$$

where $S(I,Q)$ is the score for a given image I against query Q, c is the bank of concepts, N is the number of concepts, $P(c_i,I)$ is the probability of the presence of a concept in the image, and $w(c_i,Q)$ is the weight given to the concept, depending upon the given query.

The remaining problems, then, are to develop methods for (1) determining $P(c_i, I)$, the indexing value assigned to each concept for each image, and (2) calculating $w(c_i, Q)$, the weight given to each concept given the query. We will discuss approaches to these problems in the remainder of this section.

3.2.1 Indexing Images for Concept-based Search

In our model for concept-based search, as given above in Equation 3.1, each image in the corpus is essentially indexed with a static vector, $P(\vec{c}, I)$, which gives the probability of the presence of each of the concepts in the lexicon in the image. The fundamental building block for determining these values is the result of a concept detection system. We describe our specific concept detection implementation in further detail in Section 3.3.2; however, for the purposes of the current discussion we can simply state that any standard concept detection system could be used and that the output of such a system would be a confidence score of the presence of each of the concepts in a given image. That confidence score could then be easily normalized to lie between 0 and 1. This is a good starting point, but these confidence values are not sufficient for representing the visual concepts in images. The problem lies in the fact that concept detection is noisy and not all detectors can be trusted equally, since some perform extremely well and others are completely unreliable. So, a confidence value of 1 from a very noisy detector might overpower a value of 0.5 from a very reliable detector. To mitigate this effect, we propose to scale the expected reliability of the concept detector. We measure this reliability using the estimate average precision (AP) of the concept by testing it over some held-out evaluation data. The probability of the presence of a concept in a given shot is then given as:

$$P(c_i, I) = con(c_i, I)(AP_i) + (1 - AP_i)(P(c_i)) \tag{3.2}$$

where $con(c_i, I)$ is the confidence of the presence of concept in the image provided by the concept detector, AP_i is the estimated average precision of the concept, and $P(c_i)$ is the prior probability of the presence of the concept in the training set. So, essentially, we are using the estimated performance of the concept detector to smooth between the raw score provided by the detector and the prior probability of the concept. Intuitively, in the case of a high-performing concept detector, we will rely more on the detector and in the case of low-performing detectors, we will rely more on the prior.[1]

3.2.2 Matching Queries Against Concepts

Given an incoming query into an concept search system, the primary problem is determining an appropriate method for mapping textual keywords to visual concepts. For example, a query for "Find shots of boats or ships" could directly utilize the "boat/ship" detector, but could also benefit from "sky," "water," and "outdoor" detectors.

The relationship between the query and the "boat/ship" detector is very direct and easily discovered, but the relationship with other peripheral concepts is difficult to discover. Similarly, for a query for "Find shots of Condoleezza Rice," there is no directly-matching "Condoleezza Rice" detector, so we would have to be able to discover peripheral concepts, like "government leader," "female," and "face." So,

[1] Alternately, we could use $P(c_i, I) = con(c_i, I)\frac{AP_i - P(c_i)}{1-P(c_i)} + \frac{1-AP_i}{1-P(c_i)}P(c_i)$ to formulate this smoothing function. The only difference here is that the interpretations of the AP scores as weights are altered. Random ranked lists have non-zero values for AP, which should be equal to the prior frequency of the concept. So APs do not range between 0 and 1, but rather between the prior and 1. Here, we have taken this into account to adjust the weights for this fact.

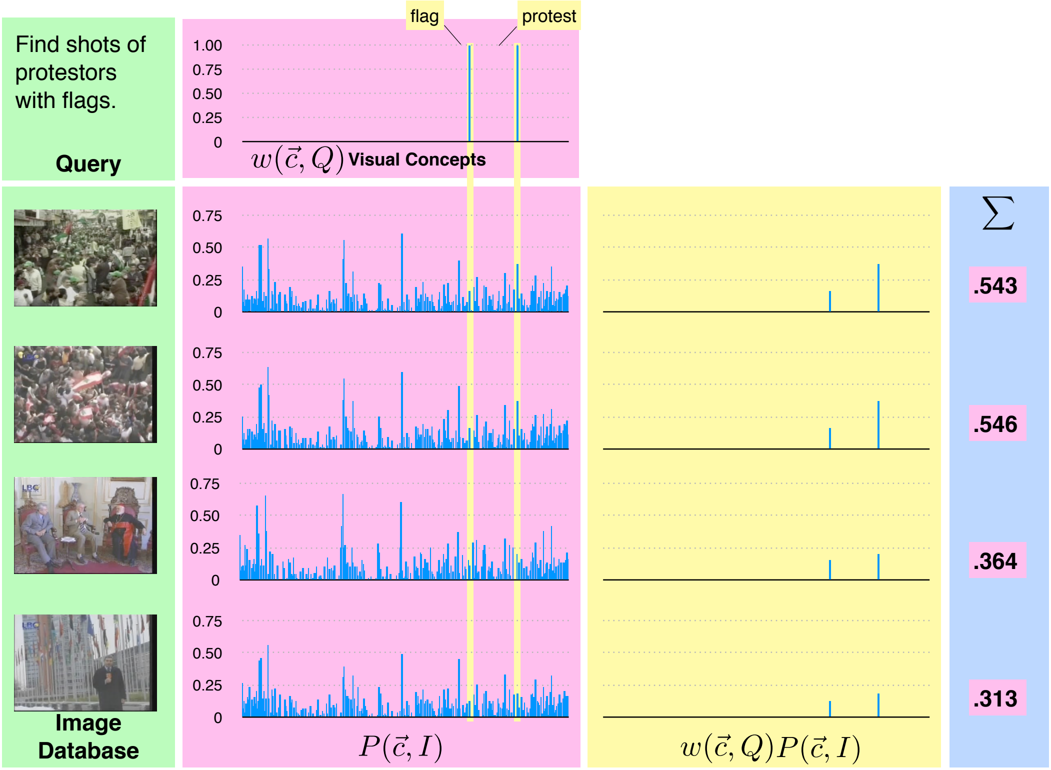


Figure 3.2: Visualization of the concept-space representations of an example query and set of images using a dictionary-based mapping. The method effectively focuses in on only two related keywords.

it's intuitive that many concepts can be leveraged to help search. But, discovering the right concepts can be difficult. Most current successful approaches are very conservative, typically only using the most direct relationships, while newer approaches aim to discover peripherally related concepts by either utilizing ontological or statistical relationships between the concepts. We will discuss the implications of all of these methods below.

3.2.2.1 Dictionary Mapping

The most obvious approach to mapping keywords to concepts is probably also the most immediately powerful [11]. Since the concept ontology already presumably includes names for each of the concepts, it is then straight-forward to map queries to concepts whose names contain keyword terms. This may be too restrictive, so a system designer may also choose to include a list of synonyms or other keywords for which the concept may be a powerful choice. For a concept named “boat,” we might also include a dictionary of terms such as “ship,” “vessel,” and “watercraft,” which would provide some additional coverage for the concept and enable its use without requiring the user to guess its exact name.

Regardless of the exact construction of these sets of terms, dictionary-based approaches rely on having a sets of terms, $T_i = \{t_1, t_2, ...t_N\}$, associated with each concept, c_i. Textual queries are also sets of keyword terms, $Q = \{q_1, q_2, ...q_M\}$. In a straight-forward implementation of a dictionary-based method, the weight $w(c_i, Q)$ associated with a concept given the query (as defined in the concept-based search model in Equation 3.1) can be found via the intersection between T_i and Q. If $|T \cap Q| > 0$ (the number of terms associated with c_i also contained in the query terms is greater than 0), then we set $w(c_i, Q)$ (the weight for concept c_i) equal to 1. Otherwise, the weight will be zero. Other methods might allow for non-binary weightings, perhaps if there are multiple matching terms or if different values are assigned to the terms by the system designer.

In Figure 3.2, we see a visualization of how these $w(c_i, Q)$ and $P(c_i)$ terms can represent queries and images in an intermediate concept space in this dictionary-based mapping approach. Here, the weights in $w(c_i, Q)$ have effectively selected two concepts to represent the query and have zeroed out the effects of all other concepts. Summing the probability of the appearance of these two concepts in the example images gives a nice separation between relevant and irrelevant images.

Dictionary-based mapping approaches are very simple (in the good sense), but are limited in their coverage for queries and the lists of possible key words are highly

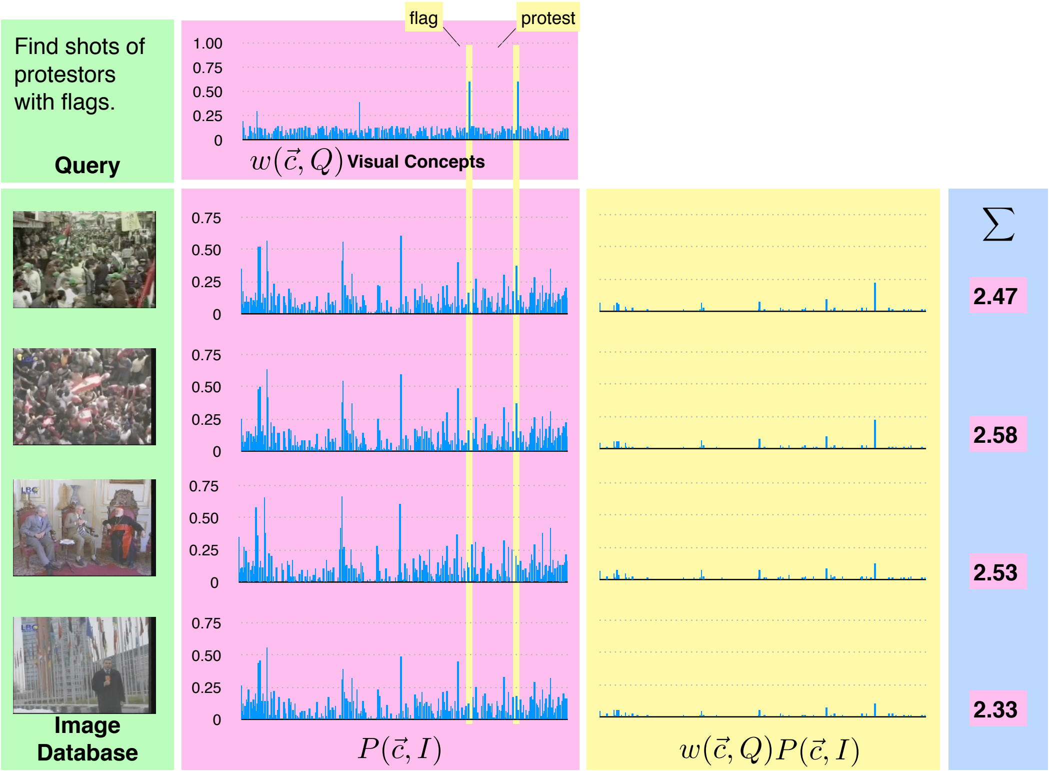


Figure 3.3: Visualization of the concept-space representations of an example query and set of images using a mapping based on lexical relationships. The method gives some weight to all keywords.

subject to errors in human judgment. For example, a term with many different senses might cause a concept to be incorrectly chosen. Or, the exclusion of a common synonym might cause the concept to be incorrectly missed. The following methods aim to address these problems.

3.2.2.2 Lexical Relationships

Several systems have proposed to utilize additional peripherally related concepts by leveraging ontological relationships. Here, an existing ontology is leveraged to infer concepts that may be related to the query. Most typically, this is done by mapping concept names into WordNet and measuring the semantic distance (such as the

Resnick [61] or Lesk [6] distances) between query terms and the various concepts, ultimately suggesting the concepts most similar to the query. For example, a query for "basketball" might be easily mapped to a "sports" concept detector.

As we discussed in the previous section, a starting point for such a system will have to be a set of terms, $T_i = \{t_1, t_2, ...t_N\}$, associated with each concept, c_i and a set of terms, $Q = \{q_1, q_2, ...q_M\}$, given in the query. Instead of looking for exact occurrences of matching terms between the sets T_i and Q, lexical methods make softer scores based on the lexical similarity. Given some method for determining lexical similarity between two words, w_1, w_2 (like Resnick or Lesk), $sim(w_1, w_2)$, we can calculate this similarity pairwise between all of the terms in Q and T_i, $sim(q_j, t_k)$. If there are N terms in T_i and M terms in Q, then we will come up with $M \times N$ similarities overall. The weight $w(c_i, Q)$ associated with a concept given the query can be derived from this set of similarities through any number of operations, perhaps by taking the mean or the median. The most logical approach seems to be to take the maximum similarity value, since this captures the best fit between the query terms and the concept terms. This will yield a continuously-valued set of weights for various concepts. The hope is that highly-related concepts get high weights, peripherally-related ones get medium weights, and unrelated ones get near-zero weights.

This approach is highly effective for a small-scale vocabulary (of, perhaps, a few dozen concepts), because such a small vocabulary is likely to not have specific concepts, like a "basketball" detector, but a more general "sports" detector would be sufficiently helpful instead. However, its impact can degrade sharply with its application to large-scale vocabularies (with hundreds of concepts). In such cases, a "basketball" query might be easily answered with an available "basketball" concept. The inclusion of the "sports" concept would only serve to degrade the overall performance, since the detection performance of the "sports" detector for the specific "basketball" concept is much worse than the "basketball" concept, itself. Similarly, the method might also suggest other semantically-related concepts, such as "baseball," "football," "soccer," all of which would not be helpful in this case. A contributing factor in the case of large vocabularies is that the sum of miniscule weights for many unrelated concepts may actually overpower the weights for "good" related concepts. This effect is demonstrated in Figure 3.3. The structure of the figure is analogous to Figure 3.2, showing the concept-based representations of queries and images. Here, however, the weights for the concepts given by the query are spread across all of the concepts, which seems to degrade the degree to which the approach can effectively discriminate between relevant and irrelevant images. The effects of these many lightly-weighted concepts might be mitigated by zeroing out lower weights below a certain threshold.

Another (and perhaps more grave) part of the problem here is that WordNet is designed for lexical relationships, which often do not amount to visual relationships or visual co-occurrence. A "car" is lexically similar to a "boat," since both are types of vehicles, but the two rarely occur in the same visual scene. The benefits of such fuzzy mappings are larger in small vocabularies, since finding any matching concept at all is better than finding none and the likelihood for having marginal concepts eclipse correct concepts is much lower. However, when the vocabulary grows very large in size, the need for peripheral concepts is diminished, since many more concepts exist and the likelihood of an exact match is much higher. Likewise, the likelihood of finding incorrect and detrimental concepts is also high.

3.2.2.3 Statistical Relationships

Recent work in [57] presents a statistical approach to concept selection. Here, each video shot in a training corpus is associated with both text terms (found within in a temporal window of the shot in the text transcript) and visual concepts (which are manually annotated). Statistical correlations between the text terms and visual concepts are then mined. Incoming text query keywords can then be matched against this data and correlated concepts can be found and selected for performing the search.

In a training corpus of video shots, we can look at the annotated content of the video shots to see which concepts $\vec{c}$ are present in the shot as well as the terms $\vec{t}$ present in the speech recognition transcript concurrent with the video shot, where $\vec{c}$ and $\vec{t}$ are fixed-length vectors representing the presence or absence of concepts and the counts of given terms, respectively. Over many example shots, the cooccurrence of concepts and terms can be used to estimate the conditional probability, $P(c_i|t_j)$, of a concept c_i being present in a shot given that term t_j is present in the speech transcript. So, given the incoming query terms $Q = \{q_1, q_2, ...q_M\}$, we might estimate the weight $w(c_i, Q)$ associated with a concept given the query via the calculated relationships between c_i and other terms by summing over the conditional probabilities of c_i, given the incoming terms, such that $w(c_i, Q) = \frac{\sum_{j=1}^{M} P(c_i|q_j)}{M}$. This, again, will yield a continuously-valued set of weights for various concepts, with the intention of having higher weights point towards higher concept/query similarity.

Some of same set of problems from lexical relationships emerge in statistical relationships, however. Again, the large concept vocabularies are difficult to use with such methods. The sheer number of concepts may lead to problems with low-weight concepts overpowering high-weight ones. Plus, peripheral relationships might be less useful if more-direct ones are available.

Furthermore, statistical relationships present their own problems. It is not always necessarily true that statistical relationships between visual concepts and textual are strong or meaningful. If they were, then, perhaps simple text-based search over speech transcripts would be sufficiently powerful for multimedia search.

3.2.2.4 Proposed Reranking Approach

Given the generally observed poor performance of lexical and statistical models in scenarios with large concept lexicons, our experiments will primarily focus on simple dictionary-based approaches. However, we also intuitively believe that dictionary-based approaches have limitations in their inability to adequately utilize important peripheral relationships, so we also propose a new reranking based approach to discovering such concepts in this chapter. In this section, we discuss the intuition and the details of our system. In the next section we will evaluate and analyze its performance.

In this new reranking-based approach, we somewhat eschew the basic model for concept-based retrieval that we gave in Equation 3.1. Specifically, we choose not to approach the problem from the standpoint of having to map from query keywords to weightings for visual concepts, which essentially means ignoring $w(c_i, Q)$. Instead, we propose to represent each image as point defined by its vector of concept, $P(\vec{c}, I)$, and suggest that a decision boundary in this high dimensional space could be learned to separate the relevant and irrelevant images for some given query and some labeled "training" examples. Since, in retrieval, such training examples are unavailable, we resort to using the results of some other search method (such as a text-based search or even a dictionary-based concept search) as a set of pseudo-labels that we utilize for learning a discriminative model over $P(\vec{c}, I)$.

As we have seen in earlier discussions of lexical and statistical expansion models, the presence of many concepts can dampen the impact of the methods. A key hurdle in this process is the need to cut down the possible space of meaningful concepts. We accomplish this by applying pseudo-labels to search results, as discussed above. We take the top-returned results to be pseudo-positives and sample pseudo-negatives from the lower-ranked results, thereby deriving a pseudo-labeling for the target relevant images, R, which is simply binary: pseudo-positive or pseudo-negative. For these images, we also have detector scores, C, for each of the concepts in the lexicon. C quantizes the normalized scores of the concept detectors, $P(\vec{c}, I)$, into 20 bins, which is empirically found to be a reasonable number. We determine the most meaningful concepts for the query by taking the mutual information between the pseudo-labels and the concepts:

$$I(R;C) = \sum_{r \in R} \sum_{c \in C} P(r,c) \log \frac{P(r,c)}{P(r)P(c)}, \tag{3.3}$$

where $P(R,C)$, $P(R)$, and $P(C)$ are all estimated by counting frequencies in the sampled set of shots. We take the concepts with the highest mutual information with the pseudo-labeled results as the most discriminative and drop the less-discriminative ones.[2]

This process leaves us with an abbreviated feature space, $P(\vec{c}_1, I)$, which only includes the most-discriminative concepts. We then propose to learn a discriminative model over this space. Since support vector machine (SVM) classifiers have been proven to be powerful in light-weight learning applications [58] and learning combi-

[2] Mutual information can also be calculated over continuous variables as: $I(R;C) = \int_R \int_C P(r,c) \log \frac{P(r,c)}{P(r)P(c)} dcdr$; however this can be problematic in our scenario, since R is necessarily binary. We find it more practical to quantize C, which also helps in computation by mitigating factors due to small sample sizes, which make it difficult to estimate a continuous value for C.

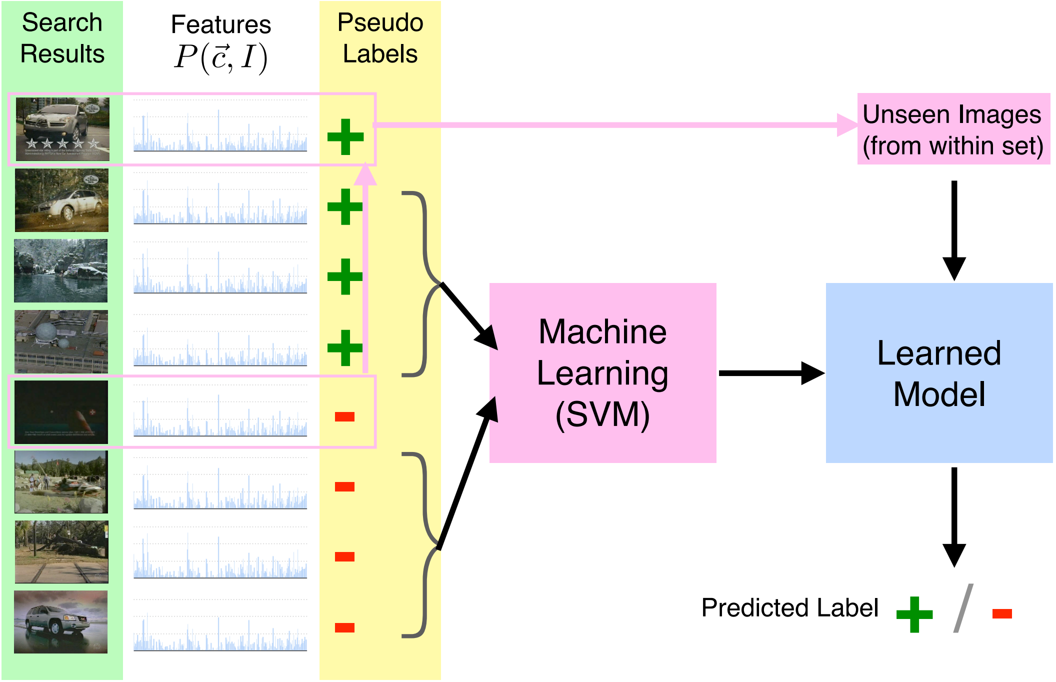


Figure 3.4: Architecture of proposed framework for context fusion in concept detection and search. Given a set of pre-trained concept detectors, related concepts are discovered through measuring the mutual information between their resulting concept scores and pseudo-labels given by an initial detection/search model result, such as a text search. Pseudo-labels and discovered related concepts are leveraged to re-order and refine the initial detection result.

nations of visual concepts [69], we opt to utilized them in our framework. We learn SVM classifiers using $P(\vec{c}_1, I)$ as the input space and R as the set of target labels. Since we are applying this method for reranking, we essentially need to apply our classifiers to the same set of data that we have learned the models from. To make this approach valid, we apply a cross-validation approach, where the list of results to be reranked is randomly divided into three partitions. An SVM is learned using two of the partitions and then used to score the left-out partition. This is repeated three times until all partitions have been scored. Finally, the reranked score, S_{Rerank} is given as the average of the initial scores from the input search engine, $S_{Initial}$, and the scores provided by the SVM models, S_{SVM}.

$$S_{Rerank} = \frac{S_{Initial} + S_{SVM}}{2} \tag{3.4}$$

An overall system flow for this system is shown in Figure 3.4.

In our experiments, we will have an input space of 374 concepts. We use mutual information to cut this space down to 75 concepts for learning the SVM models for reranking. We arrive at this value by testing various levels of concept usages (i.e. 10, 25, 50, etc.) and observing the final performance. The setting for 75 concepts is the strongest, so we only report on those results here. The pseudo-positives are the top-1200 shots returned by the initial search engine. This number is chosen, since the objective is to arrive at the top-1000 results to use for scoring. The pseudo-negatives are 3600 results randomly selected from outside of that set of pseudo-positives. This value is selected to provide adequate information about the pseudo-negative class without overpowering the SVM models with too many negative examples. Prior experience has shown us that a ratio of 3 negative examples per given positive example is sufficient for this criterion.

The process is lightweight and highly general and provides significant improvement in the TRECVID automatic video search task.

3.3 Experiments and Evaluation

We intend to evaluate the utility of concept-based search in terms of a few criteria: (1) the contributions that it can make in a multimodal search sytem, (2) the impact of the size of the concept lexicon, and (3) the efficacy of our proposed reranking framework for incorporating many peripherally-related features. In this section, we will discuss the details of our experiments and analyze the results.

3.3.1 Data Set

We conduct our experiments using the data from the NIST TRECVID 2005 and 2006 video retrieval benchmarks [66], which includes over 300 hours of broadcast news video from English, Chinese, and Arabic sources. The data is accompanied with speech recognition and machine translation transcripts in English. In each year, 24 query topics are provided with ground truth relevance labels collected through a pooling process which are distributed after the benchmark.

Our experiments also rely on the The Large-Scale Concept Ontology for Multimedia (LSCOM) [56], which was developed in a series of workshops in 2005, leveraging input from intelligence personnel, librarians, and computer vision researchers to assemble a list of more than 2000 visual concepts relevant to intelligence applications in the domain of broadcast news. These concepts were selected based on a number of criteria determined by each of the interested parties. In particular, the concepts were chosen to be “useful,” meaning that they would be relevant to intelligence analysis tasks. Secondly, the concepts were chosen to be “observable,” meaning that human observers would easily be able to determine their presence by visually inspecting images or videos. And finally, the concepts were also selected to be “detectable,” meaning that it would be feasible for state-of-the-art computer vision algorithms to automatically detect the concepts in the near future. Human annotators then labeled the concepts over a collection of 80 hours of broadcast news video by inspecting 61,901 still keyframes from automatically-detected shots and marking each concept as being either present or absent. The extreme expense in labor for this task led to only 449 of the total 2000 concepts being annotated over the collection. The concepts to be annotated were selected primarily based on the “useful,” “observable,” “detectable,” criteria mentioned above, but also based on an intuition about the projected “frequency” of the concept. Concepts with only one or two positive examples in the set would be of little value and were, therefore, ignored. The collection also provides a few forms of hierarchical organization. In the simplest version, the concepts are organized into a few broad classes, such as “people,” “locations,” “objects,” and “events.” In a more in-depth organization, the concepts are mapped into the Cyc ontology and can be used in reasoning and artificial intelligence applications. The collection of concepts also includes a lighter-scale version, known as LSCOM-lite [55], which is a subset of 39 of the concepts, which were made available prior to the release of the full LSCOM and which were used as a test set in the 2005 TRECVID [66] evaluation.

3.3.2 Concept Detection

A critical component in a concept-based search system is a library of automatic visual concept detectors. In concept detection, the objective is to build automatic detectors for arbitrary visual concepts, given a large set (on the order of hundreds or thousands) of ground-truth labels for the concept. To achieve this, we applied a generic visual-only framework which can be applied to any concept without any specific tuning. For simplicity, we choose to represent shots as single still keyframe images; however, in principle, more involved temporal models could be applied for event-based concepts. The concept model is built using SVMs over three visual features: color moments on a 5-by-5 grid, Gabor textures over the whole image, and an edge direction histogram. The resulting scores of each of the three SVM models over the test keyframe images are then averaged to give a fused concept detection score. We have found the accuracy of such baseline detectors satisfactory over a large set of concepts [11], especially for scenes and objects. The concept detection framework is discussed more in [84, 11]. The performance of these detectors relative to state-of-the-art methods is also further discussed in [36].

We apply the concept detection framework for 374 of the 449 concepts from the LSCOM [56] annotations, excluding only the concepts with too few positive examples (fewer than 10) to adequately train the models. This set includes all of the LSCOM-lite annotations.

3.3.3 Concept-based Search in Query-Class-Dependent Multimodal Search System

Our search system is designed to combine each of the core component search methods (text search, concept-based search, and visual example-based search) through a weighted sum, where the weights of each method are determined by the class of the query. An overview of the system is shown in Figure 3.5. We have conducted some experiments and analysis to measure the effects of each of the component methods on search performance.

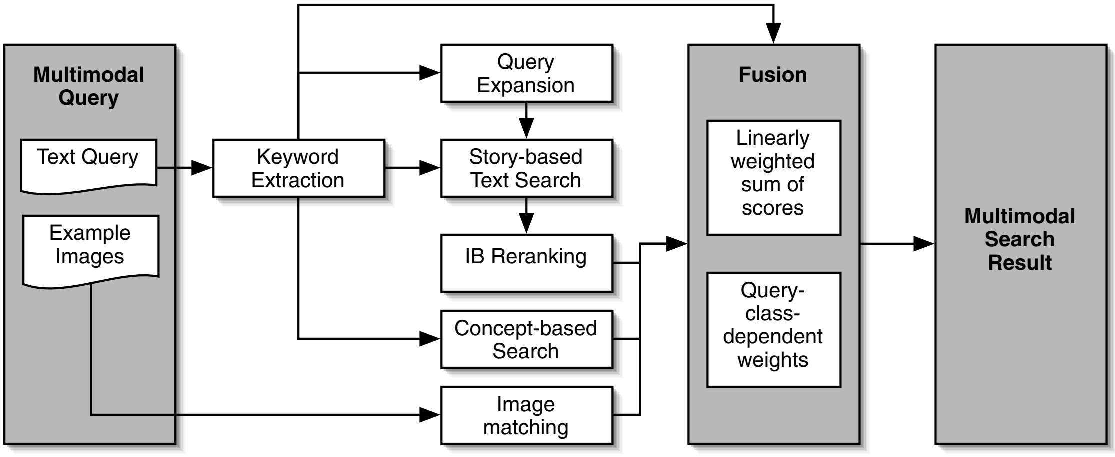


Figure 3.5: Architecture of multimodal video search system.

3.3.3.1 Multimodal Search Tools

We use many of the same six search tools proposed in the previous chapter: text-based search (though, here, we have two varieties: with and without story segmentation), query expansion (which encompasses WordNet- and Google-based expansion, along with pseudo-relevance feedback) and query-by-example image search. We exclude Person-X search, since we find that it is mostly superfluous, given text-based search.

We also add a few new tools. Most prominently (and sensibly), we utilize concept-based search. Specifically, we use the dictionary-based approach with binary weightings for concepts as described in Section 3.2.2.1 and the shots indexed using the method described in Section 3.2.1. We also utilize a query-by-text-example method [11], which is similar to query-by-image-example, but utilizes the text stories around provided example query shots. Finally, we also include information bottleneck (IB) reranking [30], which reranks the results based on the recurrence of visual patterns.

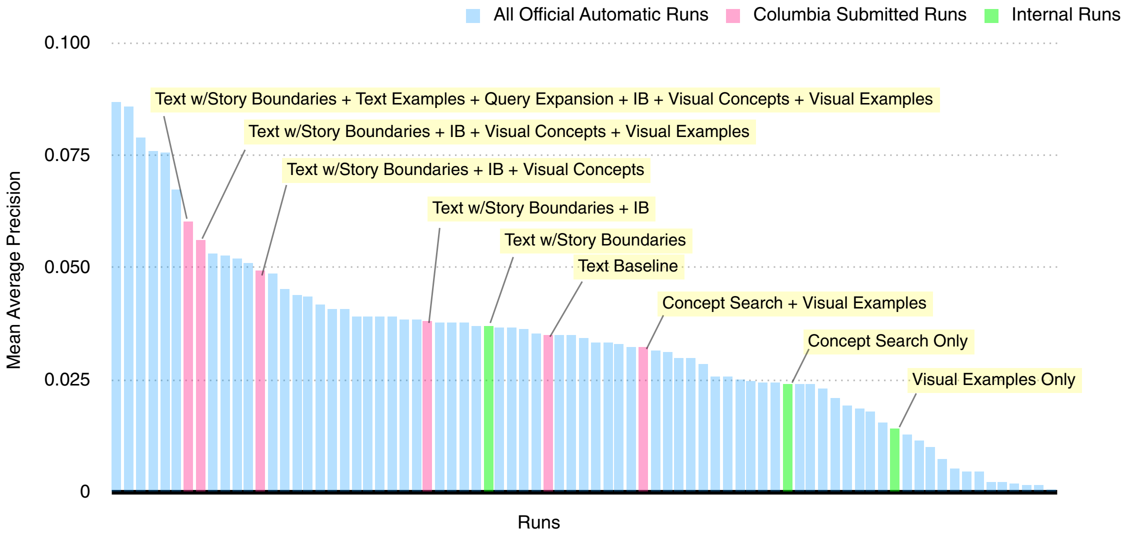


Figure 3.6: Performances of our automatic runs and all official automatic search submissions in TRECVID 2006.

3.3.3.2 Query-class-dependent Multimodal Search

The various search tools are applied for any given search query and combined through a weighted combination of scores. The weights for the various methods, though, are different depending upon the type of the query, using a query-class-dependent model [44]. Through examination of the performance of our tools over the TRECVID 2005 queries, we decided upon a set of five query classes: “named person,” “sports,” “concept,” “named person with concept,” and “other.”

3.3.3.3 Evaluation and Analysis

We found much support for the utility of concept-based search in the TRECVID 2006 set of query topics. Of the 24 total query topics, 17 had direct matches to related concepts in the LSCOM lexicon (i.e., one of the query terms matched one of the keywords associated with visual concept). Of the 7 remaining topics, 4 were for named persons, which were purposely excluded from the LSCOM lexicon, and only 3 were for objects or concepts which were truly outside of the lexicon. We also found that among the 17 concept-related queries, there were on average 2.5 matched concepts per query.

In Figure 3.6, we see the performance of each of our component search tools, as well as the fusion of many subsets of individual tools and all of the official TRECVID 2006 automatic search submissions. The concept-based search method was used in combination with the text search baseline, along with several other basic components, such as story segmentation, query expansion, and visual example-based search, with each of the components applied differently for various classes of queries. In total, all of the components used together provided an improvement of 70% relative to the text baseline. The concept-based search accounted for the largest individual portion of this improvement, improving 30% relative over a story-based text search. The degree of improvement is particularly large for a few queries with strong concept detectors, such as “Find shots of one or more soccer goalposts” (14% relative improvement), “Find shots with soldiers of police escorting a prisoner” (70% improvement), “Find shots of soldiers or police with weapons and vehicles” (40% improvement), or “Find shots of tall buildings” (many fold increase). These results confirm that a large lexicon of pre-trained concept detectors can significantly enable content-based video search, particularly when the concept detectors are strong.

3.3.4 Evaluating the Impact of a Very Large Lexicon of Visual Concept Detectors

The usage of such a large collection of visual concept detectors is still a relatively unexplored space and leaves many open questions. We conducted some experiments and analysis to evaluate the effects of using 374 concepts as opposed to a more typically-sized lexicon (in particular, the LSCOM-Lite [55] set of 39 concept detectors). In effect, we are increasing the size of concept lexicon ten fold over the conventionally-sized lexicon and we would like to investigate the influence that this change has on search performance and the number of queries which can leverage concept detection scores. We conduct our experiments by running two different versions of the concept-based search system. In one version, we simply use the concept-based search method described above. In the other version, we take the components from the first version and simply limit the range of possible matched concepts to only be the 39 LSCOM-Lite concepts, instead of the full 374-concept set.

We find that increasing the concept lexicon from 39 concepts to 374 concepts increases the mean average precision (MAP) of concept-based search from .0191 to .0244, a relative change of about 30%. This is a significant improvement; however, it is interesting that it is not proportional in magnitude to the additional cost of increasing the concept lexicon by ten fold. Further investigation shows that the 39 concepts are able to be used in 12 of the 24 query topics with about 1.8 concepts being useful for each of those matched queries. In comparison, the 374 concepts are able to be used in 17 of the 24 queries with about 2.5 concepts being useful for each matched query. Therefore, it seems that the increase in the size of the concept lexicon gives increases in query coverage and concept usage in the neighborhood of 40%-50%, for this set of query topics.

Two other factors are also likely limiters on the benefits seen from increasing the size of the concept lexicon: (1) the design of the concept lexicons and (2) the design of the query topic set.

The first factor (the design of the concept lexicons) implies that the 39 LSCOM-Lite concepts are not simply a random subset of the 374 LSCOM concepts. Instead, they are a carefully selected group, designed by selecting the concepts which would be most practical for video indexing, in terms of their utility for search, the anticipated quality of automatic detectors, and their frequency in the data set. Indeed, we can observe that a lot of these design goals are shown to be true in the data. For example, we compare the frequency of concepts in each set over the development data. The 374 LSCOM concepts have, on average, 1200 positive training examples per concept, while the 39 LSCOM-Lite concepts have, on average, 5000 positive training examples per concept. This may be a reflection of Heap's Law [29], which characterizes the rate of discovery of terms in a vocabulary by inspecting terms in corpus documents: there is expected to be a diminishing rate of discovery over time. The LSCOM-Lite concepts may reflect the more-easily discovered concepts, which also happen to be frequent and useful for retrieval. Another way to look at the data is that more than half of the 374 LSCOM concepts are less frequent than the least frequent LSCOM-Lite concept. Such a disparity in availability of training data is expected to lead to a similar disparity in concept detector performances. We evaluate the concept detectors over a validation subset from the TRECVID 2005 development set and find that this detection disparity is also present. The 374 LSCOM concepts have a MAP of 0.26, while the 39 LSCOM-Lite concepts have a MAP of 0.39. In fact, roughly a quarter of the 374 LSCOM concepts have average precision values very close to zero.

The second factor (the design of the query topic set) is due to the influence that the concept lexicons have on the selection of query topics by NIST. An important goal of the TRECVID benchmark at large is to evaluate the utility of pre-trained concept detectors in video search. To achieve this goal, the benchmark organizers include many queries which should theoretically be well-suited for using concept detection results; however, the LSCOM-Lite set of 39 concepts is the standard set used in TRECVID and our larger set is non-standard, so the queries are designed to make

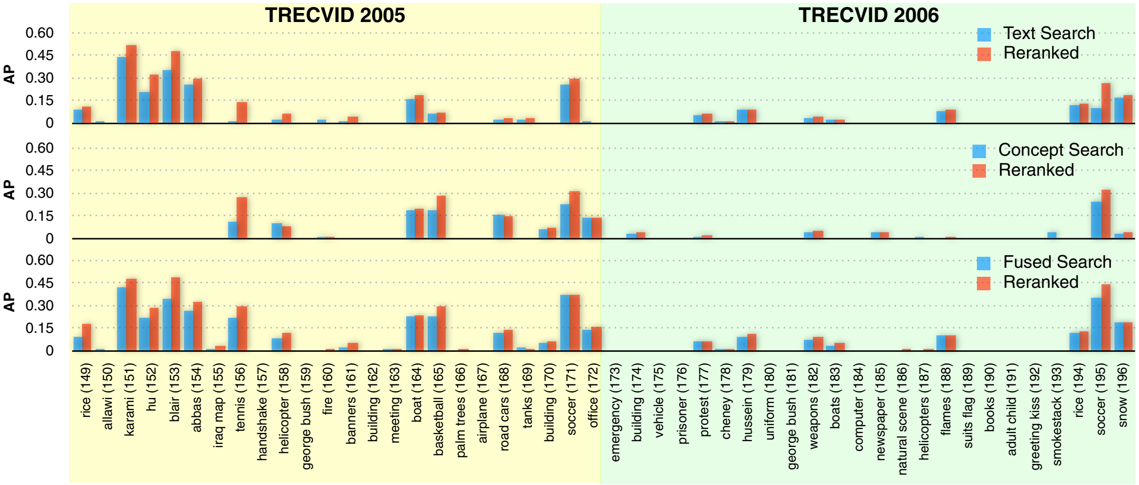


Figure 3.7: Average precisions of baseline and reranked search methods for each query in TRECVID 2005 and 2006. Text search shown in the top graph. Concept-based search in the middle. Fused multimodal search in the bottom.

use of the 39-concept set, without much regard to the content of our 374-concept set. This skews the evaluation queries in a way that is important to extending the opportunity to use concepts in search to as many participating groups as possible, but also potentially limits the impact that a large-scale concept lexicon might have.

3.3.5 Concept-based Reranking

We now move to the task of applying our reranking method described in Section 3.4.2 to the results of these multimodal search runs in order to leverage a large number of related visual concepts.

Figure 3.7 shows the results of applying the reranking method to text, concept-based, and fused searches for each query in the TRECVID 2005 and 2006 automatic search task. We see small but significant and steady increases in nearly every query topic, on average improving upon the baseline by between 15% and 30%. For the dominant group of search topics (Concept), the proposed method achieves an encouraging improvement of 12%-15%.

Varying levels of improvement can be influenced by a number of factors, such as the quality of the initial search results (results that are too noisy will not give enough information to meaningfully rerank, while extremely strong results will be difficult to improve upon) and the availability of un-tapped concept detectors to reinforce the search result. The effects of these factors are discussed in the following section.

3.3.5.1 Class-Dependency

The multimodal fused search result (and reranking) shown at the bottom of Figure 3.7 is generated using a different weighted summation of the text and concept-based search scores depending upon the class of the query. In this application, we use the five pre-defined query classes listed in Section 3.3.3.2: **Named Person**, **Sports**, **Concept**, **Person+Concept**, and **Other**. Examples from each class are given below. Each incoming query is automatically classified into one of these classes using some light language processing, like part-of-speech tagging, named entity extraction, or matching against keyword lists, as described in [11]. The performance of each search method over each class is shown in Table 3.1.

Named Person queries (such as "Find shots of Dick Cheney") frequently have the most room for improvement by the reranking method, especially when the initial text search results are strong (as is the case in the TRECVID 2005 set). These queries rely solely on text search arrive at an initial ranking. Luckily, though, text search tends to give a very strong initial result, with many positives appearing near the top of the list. These results are also loaded with false positives, since all shots within a matching story are blindly returned. The application of pseudo-labeling can help sort this out, since shots relevant to the the query will likely come from the same news event featuring the sought-after person reported across various news

			Text Search			Concept-based Search			Fused Multimodal		
Class	**Set**	**#**	**base**	**rerank**	**% imp.**	**base**	**rerank**	**% imp.**	**base**	**rerank**	**% imp.**
Named Person	TV05	6	0.231	0.293	26.7%	0.000	0.000	0.0%	0.232	0.294	26.7%
	TV06	4	0.065	0.070	8.0%	0.000	0.000	0.0%	0.065	0.070	8.0%
Sports	TV05	3	0.116	0.174	50.0%	0.182	0.297	62.8%	0.276	0.325	17.8%
	TV06	1	0.109	0.268	145.5%	0.251	0.326	29.8%	0.358	0.445	24.3%
Concept	TV05	11	0.029	0.032	12.6%	0.066	0.066	0.0%	0.064	0.074	15.7%
	TV06	16	0.029	0.033	14.3%	0.019	0.019	1.1%	0.037	0.042	12.6%
Person + Concept	TV05	2	0.007	0.002	-68.5%	0.003	0.002	-19.3%	0.012	0.018	48.0%
	TV06	0	0.000	0.000	0.0%	0.000	0.000	0.0%	0.000	0.000	0.0%
Other	TV05	2	0.013	0.026	99.0%	0.003	0.004	25.6%	0.014	0.035	146.2%
	TV06	3	0.002	0.002	0.0%	0.015	0.015	0.0%	0.002	0.002	0.0%
All	TV05	24	0.087	0.112	28.7%	0.054	0.068	27.1%	0.124	0.153	22.9%
	TV06	24	0.033	0.042	27.4%	0.024	0.028	14.3%	0.049	0.056	15.0%

Table 3.1: The number of queries in each class (#), with the MAP of the baseline method (base), reranking result (reranking), and relative improvement (% imp.) over each of the available unimodal tools (text search and concept-based search) and the fused multimodal result.

sources. This will result in the occurrence of repeating scenes with shared contextual concept clues, while the false positives in the top-returned results will probably be indistinguishable from the pseudo-negatives that were sampled, resulting in the false positives being (correctly) pushed down the list.

Sports queries (like "Find shots of basketball courts") also have significant room for improvement from reranking. These queries rely fairly equally on concept-based and text search methods and the two are highly complimentary. When reranking is applied to sports queries, we expect a behavior much like the named person queries: the initial result is already very strong, so the context clues discovered via reranking are highly reliable. However, we also observe that the results from the initial search (particularly in the fused case) can be very strong, leaving little room (or need) for improvement via reranking.

Concept queries are quite different in character from either named person or sports queries. These queries are found to have keywords matching concepts in our lexicon (such as "Find shots of boats") and therefore rely mostly on concept-based search with some weight also on text search. We might expect that the degree of improvement for each concept-type query provided by reranking over the concept-based search, alone, might be similar to the improvement if the method was simply applied to the concept detection task; however, concept-type queries actually tend to be quite different from just the combination of one or two known concepts. They may also contain keywords which are not matched to any concepts, so it can be difficult to build a strong initial search result using concept-based search alone. In fact, we see that concept-based search for these queries is not improved significantly by reranking. However, if concept-based search is fused with text search, a stronger initial result is obtained for the query and reranking provides improvements over the fused multimodal result of about 15%.

Person+Concept queries (which includes queries matching the criteria for *both* the named person and concept classes, like "Find shots of George Bush leaving a vehicle") and **Other** queries (which includes any query not meeting the criteria for any of the other four classes) are harder to draw conclusions for, due to the limited number of queries fitting in to these classes. Given our examples, it seems that these two classes still lack viable methods for getting an initial search result, yielding a poor baseline and making it difficult to discover related concepts. These queries have low performance and reranking does not offer improvement.

3.3.5.2 Related Concepts

Beyond simply observing the degrees in improvement experienced through reranking in context-based concept fusion, it is also interesting to examine the exact sources and manifestations of the contextual relationships between concepts and query topics. As mentioned in Equation 3.3, the degree of correlation between a target semantic concept and any given concept detector in the lexicon is determined by measuring the mutual information between the two, giving concepts that are both negatively and positively correlated with the target concept. We can further distinguish between positively and negatively correlated concepts by utilizing the *pointwise mutual information*:

$$I_P(r;c) = \log \frac{P(r,c)}{P(r)P(c)}, \tag{3.5}$$

where if $I_P(r =$pseudo-postive$; c =$postive$)$ is greater than $I_P(r =$pseudo-postive$; c =$ negative$)$, then the concept is considered to be positively correlated with the semantic target. Otherwise, it is considered to be negatively correlated. This approach has been used in prior works to determine the utility of concepts in answering search queries [49]; however that analysis was applied on ground truth relevance labels and

Original Query	**Positive Concepts**	**Negative Concepts**
(151) Find shots of Omar Karami, the former prime minister of Lebannon.	Government Leader, Adult, Meeting, Furninture, Sitting	Overlaid Text, Commercial Advertisement, Female Person
(152) Find shots of Hu Jintao, president of the People's Republic of China.	Asian People, Government Leader, Suits, Group, Powerplants	Commercial Advertisement, Flags, Single Female Person, Logos Full Screen
(158) Find shots of a helicopter in flight.	Exploding Ordnance, Weapons, Explosion Fire, Airplane, Smoke, Sky	Person, Civilian Person, Face, Talking, Male Person, Sitting
(164) Find shots of boats or ships.	Waterscape, Daytime Outdoor, Sky, Explosion Fire, Exploding Ordnance	Person, Civilian Person, Face, Adult, Suits, Ties
(179) Find shots of Saddam Hussein.	Court, Politics, Lawyer, Sitting, Government Leader, Suits, Judge	Desert, Demonstration or Protest, Crowd, Mountain, Outdoor, Animal
(182) Find shots of soldiers or police with weapons and military vehicles.	Military, Machine Guns, Desert, Rocky Ground, Residential Buildings	Single Person, Corporate Leader, Actor, Female Person, Entertainment

Table 3.2: Concepts found to be strongly positively or negatively correlated to some example queries through reranking.

concept annotations. In this work, we are measuring pseudo-labels and automatic concept detection scores.

Some interesting examples of concepts found to be related to query topics are shown in Table 3.2. Scanning through this table, we can observe concept relationships coming from a number of different mechanisms.

The first, and perhaps most obvious, type of relationship is essentially the discovery of **generally present** relationships, such as “Government Leader” for “Hu Jintao” and “Omar Karami” or “Waterscape” for “boats or ships.” These are the relationships that we would expect to find, as many search topics have direct relationships with concepts and many topics might only occur in settings with a specific concept present. Virtually all of the negatively correlated concepts also fall into this class (or a variant of the class for generally *not* present concepts). In fact, we see that the negative relationships are dominated by detectable production artifacts, such as graphics or commercials which are rarely positively associated with typical search topics.

The second type of relationship is a **news story** relationship. In this relationship, scenes containing the target topic occur as part of some news story, where additional related concepts can be discovered that are unique to this news story, but not generally true across all time frames. The named person queries typically display these relationships. The “Hu Jintao” topic is found to be related to the “Powerplants” concept, not because Hu Jintao is typically found in a powerplant setting, but because the search set contained news stories about a visit to powerplants. Similarly, the “Saddam Hussein” topic is found to be related to the “Court,” “Judge,” and “Lawyer” concepts, though this relationship is only true during the time frame of the search set, during which Saddam Hussein was on trial. Also, the “Find shots of boats or ships” topic is found to be related to the “Explosion Fire” concept. There are no examples of boats with fires in the training data; however, the search set contains a news story about an occurrence of a boat fire. This second class of contextual relationship is uniquely discoverable by only the reranking method, since external training sets are likely to be constrained in time and from different time periods, making it impossible to predict new relationships arising in emerging news stories.

A third type of relationship is a **mistaken** relationship. In this relationship, the system can discover context cues from an erroneous concept detector, which end up being beneficial despite the mistaken relationship. For example, it is found that the "Helicopter" topic is related to the "Airplane" concept; however, this relationship is actually false: these two concept typically do not occur in the same scene. The "Airplane" concept detector is not perfect and it turns out that the errors it makes tend to include helicopters since both have similar low-level appearances, so this mistaken relationship between concepts ends up being correct and beneficial with respect to the effects of imperfect detectors.

3.3.6 Computational Complexity

Basic approaches to concept search are light-weight and can run in less than 1 second at query time against a database of 80 hours of video (or 60,000 shot keyframe images), which is sufficient for real-time interaction. The newly-proposed reranking approach requires 15 seconds to run against the same database on an Intel Xeon 3.0 workstation, with the method implemented in Matlab. The delay is primarily due to the cost of learning multiple SVM models, which may be challenging to overcome. This may be unsatisfactory for real-time interaction, though this may be dealt with by pre-computing results for popular queries, thereby gaining the accuracy benefits without the time-delay drawbacks. By applying standard indexing techniques, like inverted indexes, these retrieval-time speeds should be largely invariant to increases in the number of concepts in the lexicon and the number of documents in the search set.

The major computational issues in concept-based search arise from training the concept detectors and indexing image collections. Training collections of concept detectors is a large task, which can often be measured in terms of months or even years, when we are dealing with lexicons on the scale of hundreds or thousands of concepts. Applying the detectors to a set of images can also become problematic: the cost grows with the size of the lexicon. There are some newer approaches that hierarchically apply detectors to mitigate this problem [76].

Scaling to larger collections, such as images on the web, presents a number of issues. As the size of the image collection increases (and the number and diversity of incoming queries also increases), the number of concepts needed to adequately cover the content of the collection should also increase. This presents the same speed issue mentioned above. However, if these issues can be addressed, then fast query-time speeds should be maintainable.

3.4 Related Work

In addition to the query-class-dependent multimodal search discussed the previous chapter and the concept-based search methods reviewed in Section 3.2, this work also draws upon a number of other related works in context fusion and reranking.

3.4.1 Context Fusion

As we have seen, it is intuitive that knowledge of the detection results for a large lexicon of concepts can be useful for refining the detection of individual concepts.

This context-based concept fusion approach has been explored and exploited in several prior works. The Discriminative Model Fusion (DMF) method [69] generates a model vector based on the detection score of the individual detectors and an SVM is then trained to refine the detection of the original concepts. In [34], the DMF model is modified and extended to incorporate active labeling from a user. These early approaches suffer from limitations in the concept lexicons, which are typically only on the order of a few dozen concepts. The results typically show improvement on only some of the concepts, while degrading others, requiring techniques for predicting when the context fusion will succeed.

The recent releases of much larger concept lexicons [56, 73], which contain hundreds of concepts, has renewed interest in context-based concept fusion. This is explored further in [71], where hundreds of high-level features, which can be difficult to detect, are modeled using cues resulting from more-easily-detected mid-level features, such as “sky” or “person” or “outdoors.” Another work in context fusion with large lexicons [35] uses an approach called a Boosted Conditional Random Field (BCRF). This framework captures the contextual relationships by a Conditional Random Field, where each node is a concept and the edges are the relationships. Detection scores of 374 concepts generated by a baseline detection system are taken as input observations. Through graph learning, the detection results for each of the target concepts are refined.

These approaches are fully supervised and require explicit knowledge of the target semantic concept and ground-truth labels in order to discover relationships with other concepts. While this constraint is fine for concept detection, where many labels are available, it is unclear how these approaches could cover the unsupervised conditions in search.

Work in using concept detectors for context in search has been much more limited. Basic applications have included methods of filtering out or boosting certain concept types depending on the type of query, such as weighting the "face" concept for searches for people, or a "sports" concept for sports queries, or filtering out the "anchor" concept across all searches [11, 27]. These approaches provide small gains in improvement and are very difficult to scale up to effectively utilize large lexicons of hundreds of concepts.

The primary successes in leveraging concept lexicons for search have been in using direct matches between query keywords and concept descriptions to find a few concepts directly related to the query. This approach suffers from limitations in the breadth of concepts that can be applied to a query. Some empirical research suggests that it may be highly beneficial to incorporate many peripherally related concepts instead of just a few concepts that are tightly related [49]. Some approaches have incorporated lexical relationships between query terms and concepts through knowledgebases; however, these relationships are often shaky, sometimes degrading search performance by uncovering relationships that are ill-suited for video retrieval [12, 9, 71]. For example, a knowledge network might tell you that "boat" is related to "airplane" since both are types of vehicles; however, in natural images, these concepts are rarely visually co-occurrent, since they occur in very different settings. This approach is explored in [12] and [9] for concept lexicons of only a few dozen concepts, showing small improvement over direct matches, alone. This is likely due to the sparse and orthogonal concept space. In [71], the approach is applied using a lexicon of several hundred concepts, and the indirect ontology-based mapping is shown to seriously degrade search performance, when compared to direct matching, likely due to the increased risk of false selection of concepts. Our experience confirms this effect on large-lexicon context fusion for search, therefore, to use large lexicons, it is necessary to engineer methods that uncover the visual co-occurrence of related concepts with the target instead of less meaningful lexical relationships.

3.4.2 Reranking

Reranking is rooted in pseudo-relevance feedback (PRFB) for text search [10]. In PRFB, an initial text search is conducted over text documents and the top-ranked documents are assumed to be true, or pseudo-positives. Additional terms discovered in the pseudo-positive documents are then added to the query and the search is re-run, presumably providing greater clarity of the semantic target and refining the search results. An extension of PRFB from the text domain to video search is to simply apply the method to text searches over text modalities from video sources (such as the speech recognition transcripts) [13]. Further extensions have used text search to obtain pseudo-*negative* examples from video collections and used those results for example-based search with user-provided positive images [81].

In [30, 22, 23], the authors extract *both* pseudo-positives and pseudo-negative image examples from text search results, requiring no user-provided examples. It is found that for many types of queries, this reranking approach outperforms approaches requiring the input of example images from the user. The intuition is that many relationships between search topics and peripheral concepts change according to the news stories in the time frame of the search set and such relationships may not be present in the provided example images or external training data. Reranking can discover the salient relationships that are present in the the data to be searched.

These applications of PRFB and reranking, however, are all applied to low-level features, such as text token frequencies or color and texture image features. In a parallel work [80], the use of large sets of concept detectors to uncover contextual cues for refining search results is explored, using initial query results as a hypothesis and re-ordering those results through a weighted summation of 75 pre-trained concept detectors. The reranking mechanism used is called "probabilistic local context analysis," and functions by assuming the top-returned results are positive, while others are negative, and treats the weights on pre-defined concept detectors as latent variables to be learned. The approaches and experiments used in this chapter and [80] are structured similarly. Our proposed approach includes two steps: the selection of relevant concepts and the construction of discriminative classifiers to rerank the initial search results. This is significantly different from the generative model used in [80], where latent aspects involving weighted sums of concepts are found. One potential drawback is the lack of clear semantics associated with each latent aspect, unlike the explicit relations discovered in our approach. Furthermore, discriminative classifiers have been found to be more effective than generative models in context fusion and reranking.

3.5 Summary and Future Work

We have evaluated the efficacy of concept-based retrieval in a multimodal search system and explored the implications of varying lexicon sizes. We have found that, indeed, concept-based search is a powerful retrieval tool and can be among the single largest-contributing components in a multimodal search system. We further find that that increasing the size of the repository of available concepts can have significant impact on the quality of retrieval results; however, the magnitude of retrieval improvement is often much smaller than the need effort for increasing the lexicon.

We have further presented a new framework for incorporating contextual cues from large sets of pre-computed concept detectors for refining and reranking search results. The proposed model differs from conventional contextual fusion models in that it requires no training data for discovering contextual relationships and instead mines an initial hypothesis ranking to find related concepts and reorder the original ranking. This unsupervised implementation allows the framework to be applied to search results, where training data for contextual relationships is typically not present, and where past approaches have been far more conservative, using only a few concept detectors per query. The reranking approach enables robust utilization of dozens of concept detectors for a single query to discover rich contextual relationships. To the best of our knowledge, this is the first work using a large pool of 374 concept detectors for unsupervised search reranking.

We find that the reranking method is powerful for search tasks, where the method improves 15%-30% in MAP on the TRECVID 2005 and 2006 search tasks. The method is particularly successful for "Named Person," "Sports," and "Concept" queries, where the initial search result is reasonably strong. The method is shown to discover many of the direct concept relationships that would likely be discoverable using a supervised approach with training data, while also discovering relationships that are entirely present only in the search set, due to the temporal growth and decay of news cycles, and would therefore not be discoverable in a temporally separated training set. The method does not improve over queries where the initial search results perform poorly. Future research should focus on these difficult queries.

Chapter 4

Improved Search through Mining Multimedia Reuse Patterns

4.1 Introduction

In previous chapters in this thesis, we have discussed a variety of approaches to indexing and searching multimedia collections, which have relied primarily on either the visual content of the images and videos (modeled either as low-level features or high-level concepts) or the text surrounding the media on either a Web page or broadcast news transcripts.

The shortcomings of these existing approaches lie in their assumption that multimedia collections simply consist of a flat collection of documents, with little internal structure. The fact, however, is that many interesting collections consist of media that is carefully produced by a human author. And now, many modern collections are produced by dozens, hundreds, or even millions of authors. Photographs for news magazines are selected independently by teams of editors and footage for the evening news is curated by broadcast news producers. The interesting result is that each instance of an image or video from within the final product is not necessarily unique. Indeed, newsworthy imagery is often iconic and shared across many differ-

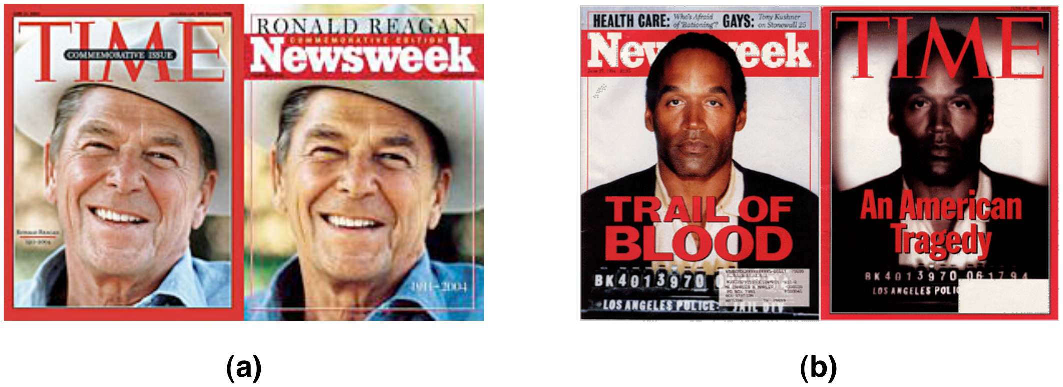


Figure 4.1: Examples of *Newsweek* and *Time* covers using the same cover image.

ent publications or broadcasts, and each time a news editor chooses to use a piece of media, they provide it with a sort of stamp of approval, indicating their belief that it is important.

For example, Ronald Reagan, the 40th President of the United States, died on June 5, 2004. His death was without doubt the largest news story of that week, if not that year, and received around-the-clock media attention. Naturally, the nation's two largest weekly news magazines, *Time* and *Newsweek*, ran photographs of Reagan on their front covers. Somewhat surprisingly, however, both magazines selected the same exact photograph. The covers are shown in Figure 4.1a. This was not first time that these two magazines had matching covers and it will most likely not be the last. In Figure 4.1b, we see the covers of *Time* and *Newsweek* from June of 1994, almost exactly 10 years before the Reagan covers, showing O.J. Simpson's mug shot. While these duplicate uses of images can be embarrassing for magazine editors, they represent an encapsulation of the collective psyche at specific points in time, from fond remembrance of a popular President to the tragic downfall of a national celebrity.

The selection and use of these images necessarily accords a degree of importance to them, which we might be able to leverage to wade through the vast number of images available: there are likely thousands of existing photographs of Ronald Reagan and O.J. Simpson, but the ones selected by magazine editors are likely to be the most iconic or most relevant. All of this has implications for search and retrieval across multimedia collections.

In more recent history, the ability of individuals to select and reuse media has moved away from being only the privilege of an elite few and has come into the hands of virtually everyone. The entire world wide Web is authored by legions of amateurs and their behaviors in selecting and posting media to personal websites or blogs similarly reinforces the subjective value of each piece of media.

In this chapter, we propose that multimedia reuse patterns within multiple-author collections give rise to an important signal about the relative authority or importance of each individual piece of media. We then aim to exploit this emergent internal structure for the purposes of improving relevance for search applications and increasing the quality of knowledge representation in summarization and exploration. We demonstrate this principle in the application domain of image search on the Web. We find that image search results contain many repeated instances of various images, which may be cropped, scaled, or contain additional overlayed text or other image content. By using currently-available duplicate detection techniques, we can decrease the amount of redundancy in the returned results by suppressing repeated (duplicate) images in the results returned to the user. More interestingly, we further find that the most-frequently repeated images within the result set are often comparatively more-relevant than less-frequently repeated ones. So, in addition to decreasing redundancy, we can also improve the relevance of the results by pushing highly-repeated images upwards in the rankings.

The primary contribution of this work is a method for mining multimedia reuse patterns from Web image search results and improving the overall performance.

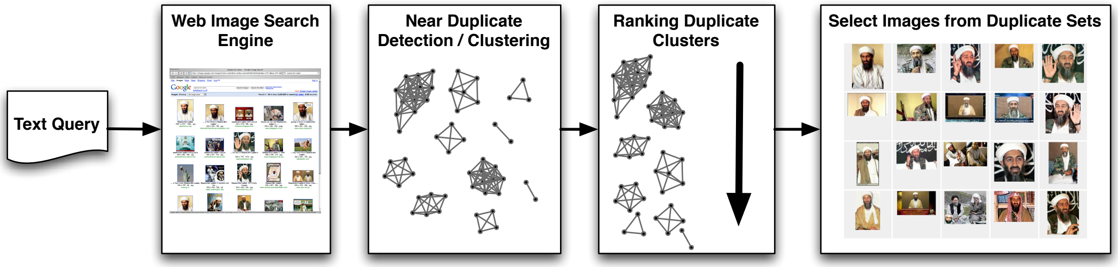

Text Query
Web Image Search Engine
Near Duplicate Detection / Clustering
Ranking Duplicate Clusters
Select Images from Duplicate Sets

Figure 4.2: System architecture for duplicate reranking framework.

We will see this applied across a broad range of real image search queries on a commercial Web search engine. The results show a significant improvement of 19% in the number of unique positive images on the first page of returned search results.

The remainder of the chapter is organized as follows. We discuss the experiments and analyze their results in Sections 4.3 and 4.4. In Section 4.5 review relevant prior work and we offer a summary and directions for future work in 4.6. We begin by presenting the details of the proposed algorithm.

4.2 Reranking with Duplicate Structures

The framework of our proposed approach is shown in Figure 4.2. Incoming text search queries are passed to a generic Web image search engine. The top 1000 results returned by the engine are then used to discover authoritative images and to rerank the returned image set. This is achieved by detecting the presence of pairs of copies of images in the set of returned images and constructing a graph, where the images are the nodes and the copy detection results are binary edges between the nodes. In typical cases, when duplicate detection is precise enough, the various sets of duplicates appear as disjoint sets in the graph, which can be taken as clusters. These clusters are then ranked (according either to the initial ranks of the images within the clusters or by the size of the cluster, or by any other method) and then

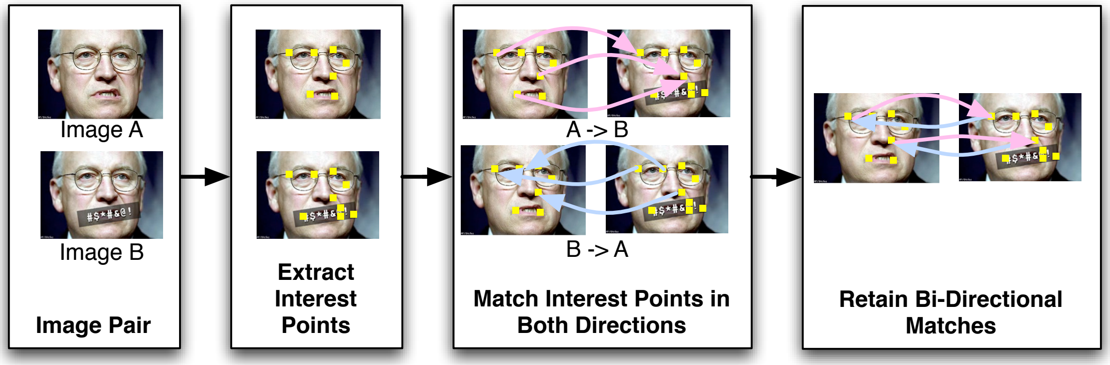


Image A
Image B
Image Pair
Extract Interest Points
A -> B
B -> A
Match Interest Points in Both Directions
Retain Bi-Directional Matches

Figure 4.3: Architecture of copy detection system employing ambiguity rejection.

one image can be selected from each cluster to generate a result set to return to the user.

4.2.1 Copy Detection

Copy detection has seen a great deal of interest and a number of workable proposed solutions. In this work, however, we opt for a simple approach, which we first applied to detecting repeated views of landmarks [43]. An overview of the approach is shown in Figure 4.3.

In the duplicate detection algorithm, SIFT [52] descriptors are extracted for each image. These features capture local geometric properties around interest points within the image. SIFT descriptors are invariant against a number of distortions, such as scaling and rotation, and robust against a number of other transformations. They are highly distinctive and occurrences of descriptors of a real-world point represented across different images can be matched with very high precision. Figures 5.9a and 5.9b (in the following chapter) show examples of automatically detected SIFT interest points. Given two images, each with a set of SIFT interest points and associated descriptors, we use a straight-forward approach, sometimes known as ambiguity rejection, to discover correspondences between interest points. Intuitively, in order to decide if two SIFT descriptors capture the same object, we need to measure the distance between the two descriptors and apply some threshold to that similarity in order make a binary decision. In ambiguity rejection, this threshold is set on a case-by-case basis, essentially requiring that, for a given SIFT descriptor in an image, the nearest matching point in a second image is considered a match only if the Euclidean distance between the two descriptors is less than the distance between the first descriptor and all other points in the second image by a given threshold. To ensure symmetry, we also find matching points using a reverse process, matching from the second image against the first image. When a pair of points is found to be a candidate both through matching the first image against the second *and* through matching the second image against the first, then we take the candidate match as a set of corresponding points between the two images. The intuition behind this ambiguity rejection approach is that matching points will be highly similar to each other and highly dissimilar to all other points. A version of this method is included in a freely-available SIFT toolkit [65].

4.2.2 Clustering

Once these correspondences are determined between points in various images in the set, we detect duplicate pairs when the number of point-wise correspondences between the two images exceeds a threshold. In our experiments, we have set this threshold equal to 50, since some of our initial observations have shown that this yields precise detection with very few false alarms.[1] The result is a graph of connections between images in the candidate set based on the duplication detection

[1] We have observed variations in the ideal threshold for different types of queries. In particular, locations and landmarks benefit from lower thresholds, since the variation between repeated images might be due to varying angles of the same scene, rather than manipulated versions of the same photograph. For simplicity, we have kept the threshold consistent across all queries; however, adapted the threshold based on the query may provide further improvements.

between the images. In typical queries, such as the one shown in Figure 4.5, there are several connected components in this graph, representing the different sets of duplicate images.

4.2.3 Ranking Images

Given the set of images returned by the image search engine and the automatically extracted network of duplicate connections, we need to arrive at an appropriate ordering of the images to be returned to the user. In this section, we outline three possible alternatives that we have evaluated.

4.2.3.1 Baseline

In the baseline case, we simply trust the results of the original image search engine.

4.2.3.2 Collapse

In the collapse method, we improve upon the baseline by removing all detected duplicates from the results. It is equivalent to traversing down the results list and detecting duplicate pairs with all higher-ranked results. If a duplicate pair is found, the lower-ranked result is simply "collapsed" into a duplicate cluster and essentially hidden behind the higher-ranked result. This approach should successfully hide redundancy, but will do little to remove irrelevant results.

4.2.3.3 Rerank

The rerank approach applies the method we've discussed earlier to re-order the duplicate clusters and to represent each cluster by selecting the most canonical image from the set. Specifically, we order the clusters according to the number of images that they contain. Larger clusters are ranked higher. Also, we select the image that

300 movie, andy warhol, ansel adams, ayad allawi, barack obama, barcelona, benjamin netanyahu, berlin, bernice abbott, bill clinton, bill richardson, bobby jindal, borat, boris yeltsin, caravaggio, charlie crist, che, chelsea, chet edwards, chris dodd, chuck hagel, cindy mccain, condoleezza rice, dennis kucinich, dick cheney, dorothea lange, dubai, earth day, el greco, evan bayh, francisco de goya, fred thompson, george w bush, georgia o keeffe, ghost rider, gilbert stuart, giotto, global warming, hawaii, henri cartier bresson, henry hyde, hillary clinton, hu jintao, iceberg, jackson pollack, jack reed, jasper johns, jeff wall, joan miro, joaquin sorolla, joe biden, joe lieberman, john edwards, john f kennedy, john kerry, john mccain, knut, kurt vonnegut, leonardo da vinci, london, madrid, mahmoud abbas, mahmoud ahmadinejad, margaret bourke white, mark rothko, mark souder, mexico, michaelangelo, mike gravel, mike huckabee, mitt romney, morgan freeman, new york city, omar karami, osama bin laden, pablo picasso, paris france, pope john paul ii, raphael, robert mapplethorpe, ronald reagan, ron paul, rudy giuliani, saddam hussein, salvador dali, sam donaldson, sandro botticelli, san francisco, shaha riza, shrek, southern cross, spiderman 3, titian, tom ridge, tony blair, virginia tech, washington dc, world map, yasser arafat, yuri gagarin

Figure 4.4: Query terms used in the evaluation.

has the highest degree of connectivity (i.e. the most copy detections) to represent the cluster. Our experience has led us to hypothesize that larger clusters tend to contain more canonical images, while smaller clusters are rarely-used images. Similarly, we can see that the highly-connected images within a duplicate set are more likely to be canonical or representative of the cluster. The less-connected images are more likely to be manipulated outliers and are probably not the best representatives. So, the “rerank” approach sets out to simultaneously hide redundant duplicate images and provide more relevant results.

4.3 Experiments

4.3.1 Experimental Data

We conduct experiments using a diverse set of queries culled from a variety of sources. Among these sources are the queries in the Google image search Zeitgeist[2], which lists popular queries entered into the engine in recent history, and the query topics used over the past seven years in the TRECVID video search evaluation [66]. In the end, we arrive at a list of 100 image search queries, spanning a range of categories including named persons (such as politicians, public figures / celebrities, locations, events in the news, films, and artists). The list of queries is shown in Figure 4.4. The queries are then run through the Yahoo! image search engine and the top 1000 resulting images (the maximum number that can be displayed) are downloaded. We keep this set of results as a baseline and then apply the proposed reranking methods on top of this initial set to experimentally observe the gains in performance.

4.3.2 Evaluation

Since most image search engines display 20 resulting images on the first returned page, we evaluate each of our ranking methods across the top 20 results that each approach returns. We classify errors in the results as either being due to redundancy (a given image is a duplicate of a higher-ranked image and contribues nothing to the results) or irrelevance (the image does not satisfy the query). We evaluate redundancy and irrelevance separately in order to gauge the unique contributions of various approaches. We then introduce a metric that we call "Unique Positives @ 20" (UP@20) which is the proportion of the images in the top 20 returned results

[2] http://www.google.com/press/zeitgeist.html

	% redundant	**% irrelevant**	**UP@20**
Baseline	8%	12%	80%
Collapse	1%	13%	86%
Rerank	0%	5%	95%

Table 4.1: Performance of the proposed methods.

that are neither redundant nor irrelevant.

4.4 Results and Analysis

Our experiments have shown a number of benefits from the proposed approach over traditional Web search results. Here, we will present quantitative evaluations for the improvements in relevance along with some qualitative examples illustrating other increases in search quality.

4.4.1 Retrieval Performance

Table 4.1 shows the results of the evaluation in terms of retrieval performance. We see that the "collapse" method practically eliminates the occurrence of redundant images in the top results; however, it does not affect the number of irrelevant results (in fact, by removing redundant images, it actually makes more room for irrelevant ones and increases their occurrence slightly). The end effect is a very modes relative increase of approximately 8% in terms of unique positives in the top 20. The "rerank" method, on the other hand, maintains this decrease in redundancy but also successfully limits the number of irrelevant images in the results, providing an increase of 19% relative to the baseline in terms of unique positives.

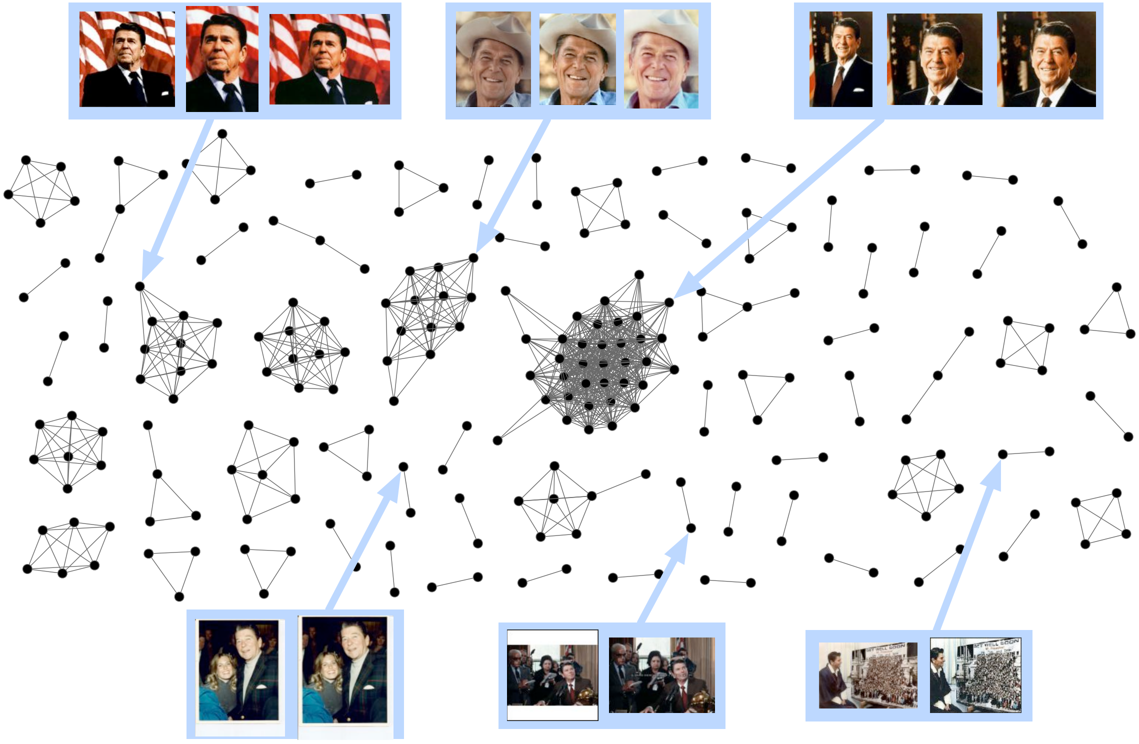


Figure 4.5: Duplicate detection graph all images returned for a query.

4.4.2 Emergent Duplicate Graph Structures

Further investigation into the emergent duplicate graph structures and clusters found in image search results can show greater insight into the successes of the proposed methods. In Figure 4.5, we show a graph of all duplicate images found in the results from a Web image search. Each node is an image returned by the search engine and an edge is drawn between any two given nodes if they have been detected as a duplicate pair. (Images without any duplicate pairs have been removed). In the figure, we have highlighted the content of three of the largest clusters and three of the smallest clusters by extracting and displaying a few example images from each. In this example, we can see quite clearly that the largest clusters (the ones that contain the most-repeated images) tend to be qualitatively more "iconic," containing popular photographs or official portraits. In particular, we see that the

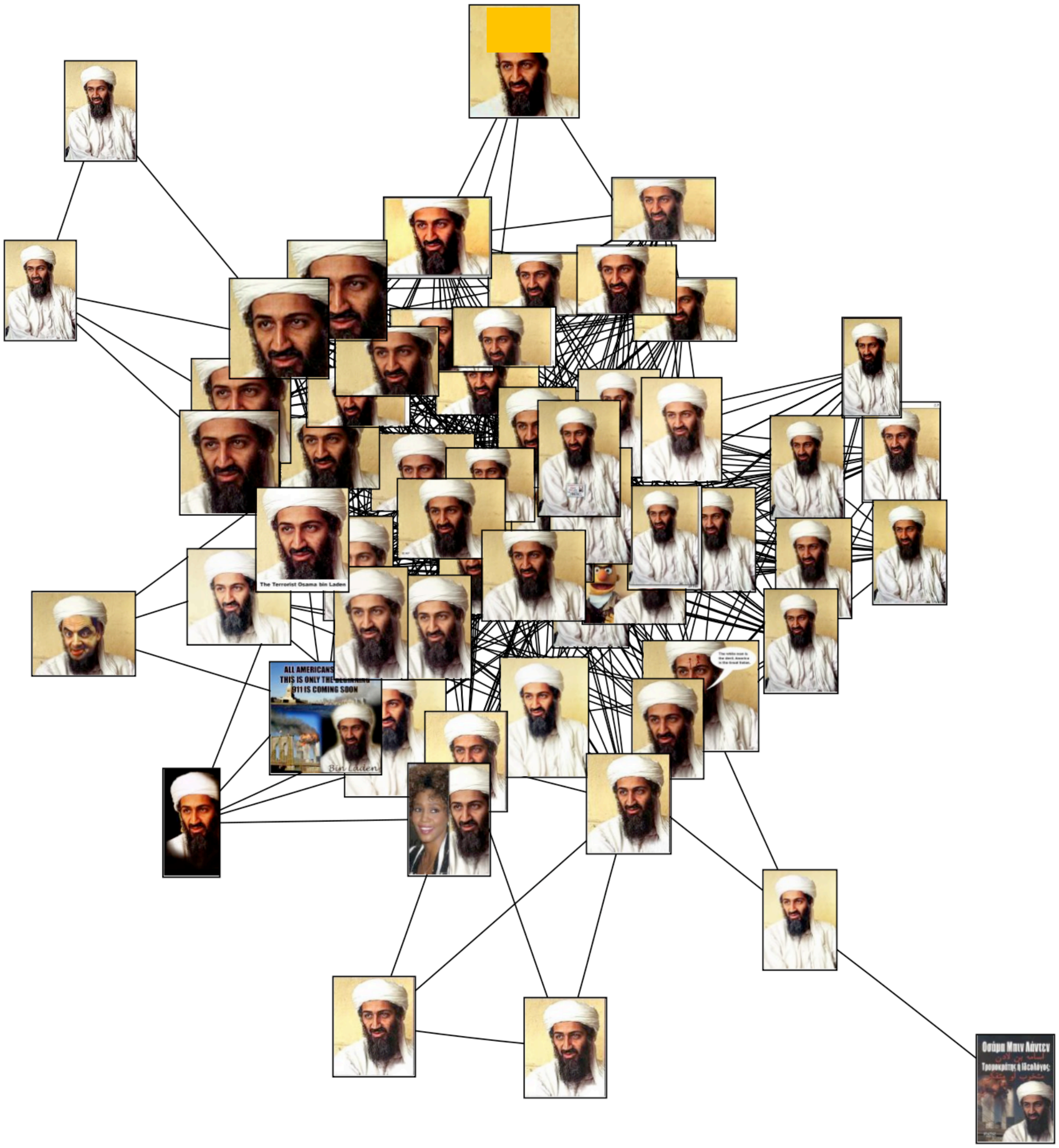
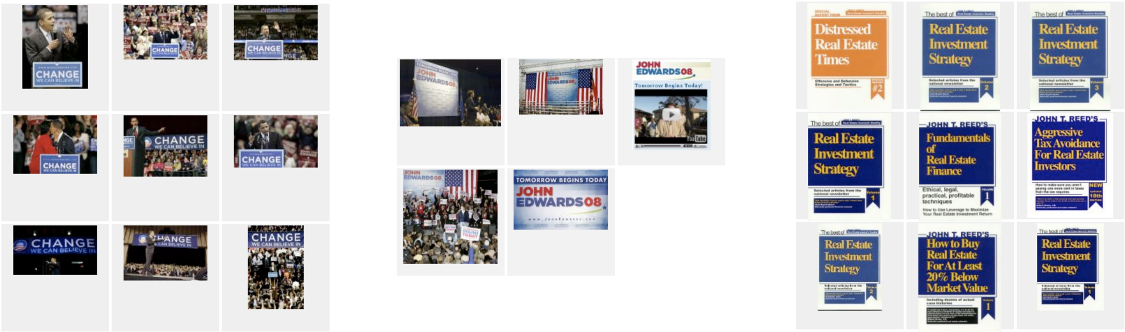


Figure 4.6: Duplicate detection graph for a single cluster.

photograph used for the covers of *Newsweek* and *Time* to cover the story of Ronald Reagan's death, referenced earlier in Figure 4.1, is contained in one of these large clusters in response the the Web image search for "ronald reagan." In contrast, the smaller clusters tend to contain much-less iconic images, which are often still relevant to the query, but may be of more obscure scenes or events. Here, we are seeing that there are often times variable degrees of relevancy. All six image clusters highlighted in Figure 4.5 are relevant to the query "ronald reagan," but the larger clusters are qualitatively more relevant. Arguably, then, the proposed approach not only increases the number of relevant results returned to the user, but also places the most-relevant images higher in the image search results.

In Figure 4.6, we delve further into the internal structure of each of the duplicate image clusters found by the algorithm. Here, the images corresponding to each node are rendered in the graph and edges are drawn where a duplicate pair has been detected. By virtue of the spring model [20] used to draw the graph on a flat, 2-dimensional surface, the most highly-connected nodes tend to be pushed toward the center of the graph and the less-connected nodes are pushed outwards. We can see rather clearly in the figure that the images in the center of the graph tend to contain portions of the original image that are less visually manipulated than the images pushed towards the edges. This is a result of a certain way in which the duplicate detection algorithm can fail. If the content of a particular image is significantly altered, there is a greatly reduced chance that enough matching interest points will be found between it and any other given image from within the duplicate cluster. From this, we can conclude that it is a reasonable approach to select the most highly-connected image as a representative image for the cluster and to hide the others, since these images will contain the most original content and other, highly-manipulated images, can be hidden. In the next chapter we will revisit the


CHANGE
WE CAN BELIEVE IN
JOHN EDWARDS08
TOMORROW BEGINS TODAY
Distressed Real Estate Times
Real Estate Investment Strategy
JOHN T. REED'S
Fundamentals of Real Estate Finance
Aggressive Tax Avoidance For Real Estate Investors
How to Buy Real Estate For At Least 20% Below Market Value

(a) (b) (c)

Figure 4.7: Examples of errors incurred by the duplicate reranking approach.

issue of image manipulation in greater detail.

4.4.3 Sources of Error

This duplicate-based reranking approach is, of course, subject to some sources of error. For example, in Figures 4.7a and b, we see the second-largest clusters found for “barack obama” and “john edwards,” respectively. Here, these images are not truly duplicates. Instead, they contain banners or placards that are repeated in various different settings. Such an error might be alleviated by altering the duplicate detection algorithm by requiring a more global match between two images. Or, the clustering approach might reject clusters where images are only linked by small local segments within each image. In Figure 4.7c, we see a small sampling of the images found in the largest cluster for the politician “jack reed.” Here, there is apparently an author with a similar name and his series of books have repeated visual objects across them. This results in a cluster that is irrelevant, but is pushed to the top of the results because it is artificially large (since many of the detected duplicates are false alarms). Again, this may be alleviated using constraints on the duplicate detection approach, as described above. Or, we may propose to model users and their contexts to disambiguate between queries for persons of the same name. For example, the search history of the user may indicate whether he or she is interested in real estate or politics, which can provide insight on which of these clusters are more appropriate.

4.4.4 Computational Complexity

The reranking system that we have proposed takes approximately 2 hours to process and rerank the top-1000 results from an image search query on an Intel Xeon 3.0 GHz workstation, with the method implemented in Matlab. This is, of course, unacceptable for standard human interactions with Web image search systems. The primary problem is the need to compute pair-wise copy detection scores between all of the resulting images. This might be mitigated by applying faster search techniques for duplicate images. For example, a fast pre-filtering step (based on simple global features can be applied before the aforementioned local point matching copy detection method. Or alternative methods based on efficiently indexing images using fingerprints can be applied [74]. An implementation on the web might also just pre-compute the results for the most frequently handled queries, thereby gaining the accuracy enhancements without causing delays for the user.

4.5 Related Work

Our proposed work should be understood in the context of several prior works, most prominently in near-duplicate and copy detection and search reranking.

The challenge of identifying whether or not two images are copies of each other has been addressed in recent research efforts in image near-duplicate detection [89]. As the term “near” would imply, these methods aim to detect pairs where the duplication may not be exact. As a necessity, these methods must be robust against a variety of distortions, such as cropping, scale, and overlay. Additionally, many proposed approaches also aim to be robust against versions of the same scene being captured by multiple cameras from slightly different vantage points. These methods have seen great success in terms of accuracy, so much so that some recent work has turned toward issues of speed and complexity in order to bring duplicate detection into real-world systems [14]. A similar problem is video copy detection [24], which aims to find repeated occurrences of the same video clip in various streams and locations, without requiring the flexibility of near-duplicate detection. This is necessary for applications, such as copyright protection on video sharing Web services. In our work, we do not seek to re-invent near-duplicate or copy detection. Instead, we stand on top of existing work and incorporate it as a component in our proposed system.

We reviewed reranking work extensively in section 3.4 and will not repeat that review here. A further work that should be mentioned, however, is [37], which is similar to our proposed method and operates in the same domain of Web image search. Like in our work, the authors propose to construct a graph between image search results. Whereas our graph is binary and determined by duplicate detection results, the graph in this other work has weighted edges, where the weights of edges are determined by the number of corresponding matching points found between the images. The authors propose to apply the PageRank algorithm [59] to this graph, which is essentially taking a random walk over the graph and using the stationary probability as the final ordering of the results. Our work is similar to prior work in reranking in that we notice recurring patterns in image search results and mine these patterns to push the recurring patterns upwards in the rankings. Ultimately, the distinguishing factor between our work and prior work in reranking, however, is our acknowledgment of the existence of *duplicates* and *copies* in the search set. Previous works, then, will increase the relevance of image search results at the cost of increasing the repetition and redundancy. This is because previous works have observed the utility of exploiting repeated visual patterns in visual search results, but have not observed that these repeated patterns are often specifically caused by repeated appearances of copies of images. Our proposed method has taken these approaches a step further and dealt directly with copies of images. We use the repetition to discover relevant images in addition to suppressing the repeated versions of images. Thus the approach will increase relevance while simultaneously decreasing redundancy.

4.6 Summary and Future Work

We have proposed a new approach for reranking image search results based on the detection of copies of images throughout the initial results. The novel contribution here is the suggestion when multiple authors have chosen to duplicate, manipulate, and reuse a particlar image that confers upon the image a certain degree of importance or authority and ought to be ranked higher in the search results. This stands in contrast to previous proposals of using copy detection to simply hide redundant results. Indeed, the proposed method is trivially combined with this copy hiding approach to simultaneously decrease redundancy and improve relevance. We find that the proposed approach provides a 19% relative increase in performance, measured as the number of unique relevant images in the top 20 returned results. We further observe that the higher-ranked images for many queries are subjectively more canonical, which may increase perception of search quality.

Chapter 5

Making Sense of Iconic Content in Search Results: Tracing Image Manipulation Histories

5.1 Introduction

In the previous chapter, we have seen that the proliferation of the World Wide Web and the continued growth and simplification of Web publishing tools has made it amazingly easy for any of us to share and publish our ideas online. These stories and ideas that we create are frequently enhanced by the inclusion of photographs and images which support the viewpoints and messages of the authors. Indeed, many of us select these images from existing repositories and, in many cases, through the use of photo editing software, the images can become highly manipulated, with portions added, removed, or otherwise altered. How do these manipulations affect the meanings conveyed by the images and the documents that they accompany? If we are cognizant of these effects, can we mine online image repositories to gain a better understanding of the beliefs held by image authors? Can we use this knowledge to design systems that enhance image browsing and retrieval?

In Figure 1, we see two of the world's most reproduced photographs along with some examples of how image manipulations can have significant impact on the edito-

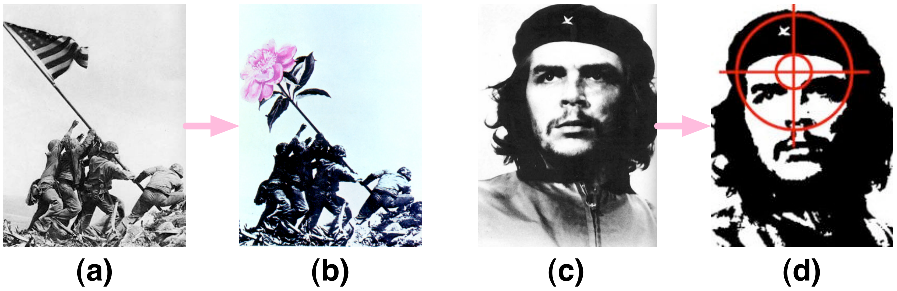


Figure 5.1: Image meaning can change through manipulation.

rial position being conveyed by the image. In Figure 5.1a, we have the famous photograph, *Raising the Flag on Iwo Jima*, which was taken by Joe Rosenthal shortly after the World War II battle in Iwo Jima in 1945. Immediately, the image was reproduced in newspapers all across the United States. By and large, the intention of distributing the image was to spread national pride and convey a highly patriotic and supportive view of the United States' use of military force. In Figure 5.1b, we see a photomontage made for an anti-Vietnam War poster in 1969 by Ronald and Karen Bowen, which replaces the flag in the soldiers' hands with a giant flower. The objective here is to take the originally pro-war image and subvert its meaning into anti-war imagery.

In Figure 1c, we have Alberto Korda's 1960 photograph, *Guerrillero Heroico* (Heroic Guerrilla), which depicts Ernesto "Che" Guevara, the Argentinean doctor-turned-revolutionary. The use of the word "heroic" in the photo title implies that the image represents a positive position towards its subject. Indeed, the image has been co-opted by many to express an identification with Guevara's revolutionary actions or his idealism. Of course, Guevara is also a controversial figure. Many take offense to the popularity of the image on the grounds of their opposition to Guevara's militancy or his role in installing the current communist regime in Cuba. In Figure 1d, we see the manifestation of this anti-Guevara sentiment in the form

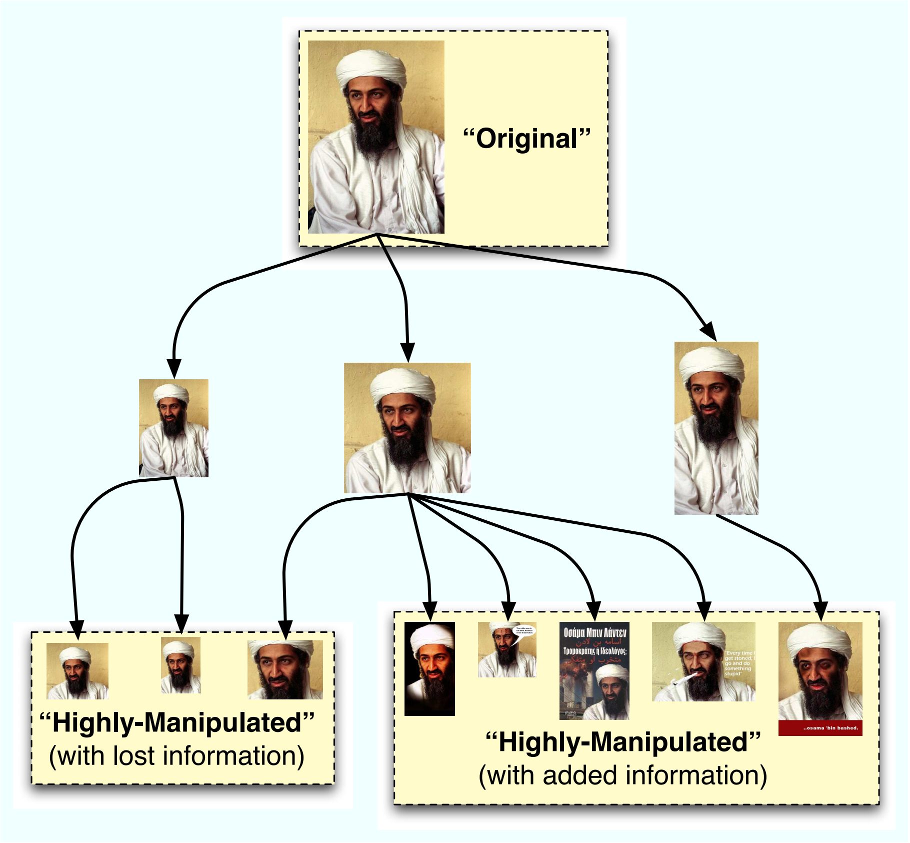


Figure 5.2: Hypothetical Visual Migration Map (VMM) showing the manipulation history of an image.

of an image manipulation. By placing a target over the subject's face, the author of the image effectively places Guevara in the sights of an assassin and turns Guevara's history of violence against him in clear criticism.

5.1.1 Visual Migration Map

These image manipulations do not exist in a vacuum, of course. Each image has a history and context that grows over time. We hypothesize that it is not the typical case that users are exposed initially to some original version of the image and decide to derive an image directly from the original. Instead, users may be exposed to some version of the image that is already derivative in some respect. Perhaps it is cropped, scaled, or modified with overlays. And frequently they may also be exposed to text surrounding the image, which conveys a story and shapes the user's interpretation of the image. So, in effect, it is unlikely that every manipulated image is directly descended from the original image. Instead, each image is likely the result of many generations of manipulations and changes in meaning and context. In Figure 5.2, we see a hypothetical view of what the manipulation history, or visual migration map (VMM), of an image might look like. At the top, we have the original image. Children of images can be spawned through any number of manipulations. And then those children can, in turn, spawn more children. We expect this tree to have a number of characteristics that can be helpful for image exploration or search. These characteristics, the "original" and "highly-manipulated" images are shown in Figure 5.2.

Original versions of images would be thought to be the closest to the very first instance of the photograph. These would be highest-resolution, contain the largest crop area, and be subject to the least manipulation. These versions may be the most relevant for some cases in image search.

Highly-manipulated images are the ones falling the most generations away from the "original." On the one hand, there are images which are highly-manipulated in terms of simply having information removed, through excessive cropping and down-scaling. These are sometimes not of much interest, since they do nothing to enhance or change the meaning conveyed in the original version. On the other hand, there are images which are highly-manipulated in the sense that they have a great deal of external information overlayed on top of the original image. These are actually quite likely to be of interest to the user, since the meanings of the images may have been significantly altered.

In this work, we develop a framework and component techniques for automating the image exploration process. We contend that, given many manipulated instances of a single image, and information about the visual migration map between these instances, we can identify points along the evolution of the image which may be of particular value to a viewer or a searcher. In particular, we hypothesize that the above-described "original" and "highly-manipulated" versions of the image will be interesting to users. We take the stance, however, that it is difficult, if not impossible, to obtain a true history of image manipulations, since, without explicit information from image authors, we simply have no definitive proof of a parent-child relationship between any two images (i.e. we cannot know if one image was directly derived from the other). Furthermore, it is infeasible to obtain every single copy of an image. A Web crawler might be able to find and index nearly all the images on the Web, but, surely there are many images that were never digitized or posted online, which leaves gaps in the image history.

Nonetheless, we suggest that the results of a Web image search for many important people or photographs will yield many dozens of copies of important images, which is sufficient for mining manipulation histories. We further propose that, despite the fact that we cannot truly know the parent-child relationships between two images, the low-level pixel content of images gives significant clues about plausible parent-child relationships. This entire process can be automated. The VMMs that emerge from this automated process are different from the true VMM of the image in many ways. Specifically, the parent-child relationships are merely *plausible*, and not necessarily true. An equally important aspect is that *implausible* manipulations (where no links are created) are detected much more definitively. If an image is not in the ancestor path of a manipulated image, then there must be information in the image extending beyond what's contained in the higher level image. Given these characteristics, we find that these automatically-constructed VMMs have many of

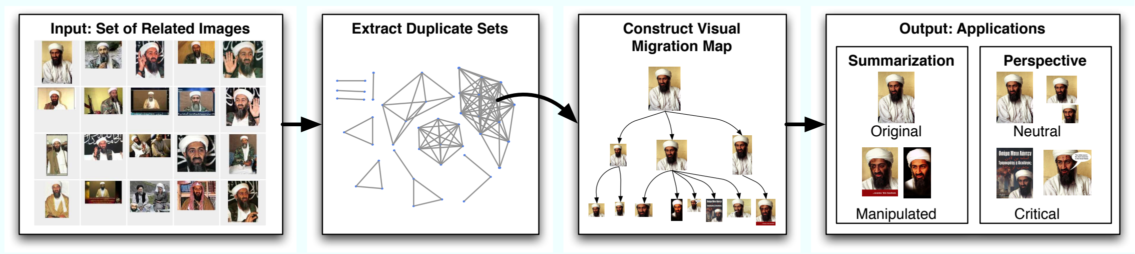


Figure 5.3: Proposed Internet Image Archaeology System (IIAS) framework. Given a set of related images, sets of duplicates or edited copies of various kinds can be extracted. For each set of edited copies, a visual migration map, representing the history of the image can be leveraged to summarize or explore the images and ideological perspectives contained within the set.

the important characteristics necessary for building search and browsing applications and the "original" and "highly-manipulated" images found are exactly the images of interest that we are looking for.

5.1.2 Internet Image Archaeology

We test this visual migration map framework in the context of a larger internet image archaeology system (IIAS), shown in Figure 5.3. The IIAS framework takes in a set of related images (such as the results of a Web image search), finds candidate sets of duplicate images derived from common sources, and then automatically extracts the visual migration map. The VMM can then be applied to many interesting applications, such as finding particular versions of the image or exploring the ideological perspectives that the images convey.

We find that these search results frequently contain many repeated instances of the same image with a variety of manipulations applied. We examine the most-repeated images within these results and develop a series of detectors to automatically determine if particular edits (such as scaling, cropping, insertion, overlay, or color removal) are present. We find that many of these edits (scaling and color removal) are detectable with precision and recall both above 90%. The remaining edits are sometimes detectable, with precision and recall values in the range of 60-80%. Using these atomic detectors, we construct a plausible edit history for the set of images. We find that, despite the errors in the individual edit detectors, the system constructs manipulation histories that are highly similar to histories manually constructed by humans. Furthermore, the automatic histories can be used to surface "interesting" images from within the set. We show that the system has strong performance in retrieving "original" and "manipulated" images.

The unique contribution of this work is a framework for an Internet image archaeology system, which can be used to surface interesting images and viewpoints from a set of related images. The IIAS framework's key component is the visual migration map, which automatically extracts the manipulation history from a set of copies of an image. A key insight in this work is that image manipulations are directional, meaning that a positive detection result for one of these manipulations implies that one image might have been derived from the other. If all of the detectors agree about the direction of an edit, then we can establish plausible parent-child relationships between images. Across many different images, these relationships give rise to a graph structure representing an approximation of the image manipulation history, which can in turn be used to surface interesting images in exploration tasks.

The remainder of this chapter is organized as follows. We discuss the proposed framework and system in Section 5.2 and give details about automatic component technologies and implementations in Section 5.3. In Section 5.4, we discuss our experimental results and propose some applications in Section 5.5. In Section 5.6, we discuss related work and offer some conclusions and thoughts on future direction in Section 5.7.

5.2 Mining Manipulation Histories

Given a set of related images, (e.g., the top results of an image search), we would like to discover interesting images and ideological perspectives. To achieve this, we first extract the visual migration map, which will surface cues about the history of these images. We propose that such a rich understanding of the history of related images can be uncovered from the image content via pairwise comparisons between each of the related images in the context of a plurality of instances of the image. In this section, we lay out the intuition that we will rely upon to automatically detect VMMs.

5.2.1 Pairwise Image Relationships

Given two images, we can ask: "are these images derived from the same source?" This implies that a single photograph was taken to capture a scene and that various copies of images can then be derived from this source via a series of physical or digital manipulation operations. The figures throughout this chapter give many examples of the typical appearances of pairs of images derived from the same source. As discussed in the previous chapter, it is sufficiently feasible to detect whether or not two images are derived from the same source using existing copy detection approaches [24, 89]. What is interesting about these related copies of images is that the operations applied to derive new copies give rise to artifacts within the image content which can tell us a great deal about the history of the image.

5.2.1.1 Directional Manipulations

Once we have established that two images are copies of each other, it remains for us to determine whether or not one is descended from the other. The key intuition

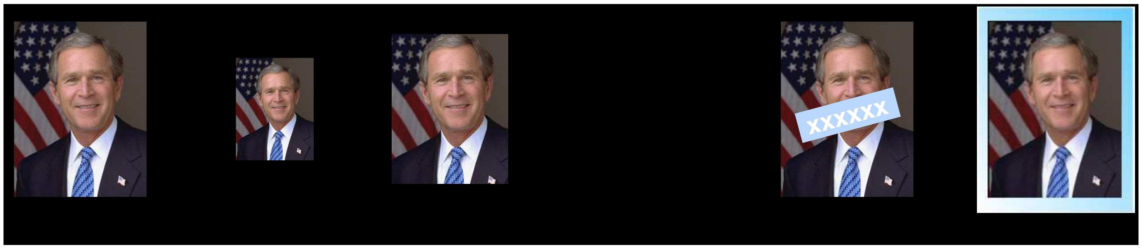


Figure 5.4: Example outcomes (b-f) resulting from possible manipulations on an original image (a).

behind this work is that image manipulations are directional: it is only possible to derive the more-manipulated image from the less-manipulated image. Below, we have enumerated a number of possible edits that can be detected between a pair of images and the directionality implied by each manipulation. Visual examples of each are shown in Figure 5.4.

- **Scaling** is the creation of a smaller, lower-resolution version of the image by decimating the larger image. In general, the smaller-scale image is assumed to be derived from the larger-scale image, as this usually results in preservation of image quality.

- **Cropping** is the creation of a new image out of a subsection of the original image. The image with the smaller crop area is assumed to have been derived from the image with the larger crop area.

- **Grayscale** is the removal of color from an image. We generally assume that the grayscale images are derived from color images (or other grayscale images).

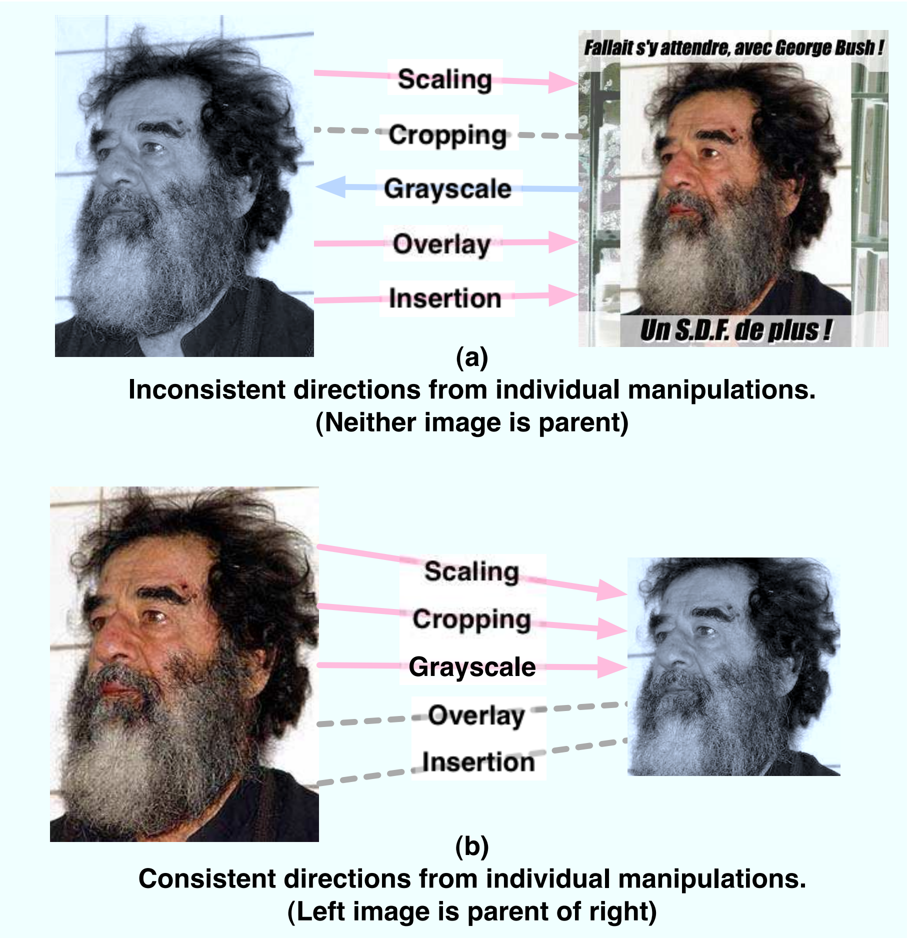
- **Overlay** is the addition of text information or some segment of an external image on top of the original image. It is generally assumed that the image containing the overlay is derived from an image where the overlay is absent.

- **Insertion** is the process of inserting the image inside of another image. Typical examples might be creating an image with two distinct images placed side by side or by inserting the image in some border with additional external information. It is assumed that the image resulting from the insertion is derived from the other image.

Of course, there are exceptions in the directions of each of these manipulations: it is possible, though not ideal, to scale images *up*, or an overlay could be *removed* with retouching software. Still, we assume the directions that we have specified are true in most cases. This list is also not exhaustive. There are other types of manipulations due to format changes (such as JPEG to Tiff), compression quality, colorization, and contrast enhancement. For the purposes of this investigation, we settle on these five above-described manipulations through a survey of several thousand image pairs. We notate the observed differences between images and retain types of manipulations that are *frequent* (appear often in this data set) *observable* (can be identified by a human annotator) and *detectable* (it is feasible to build an automatic detection system for the manipulation). Later, we will show that this set of manipulations is sufficient for building compelling applications.

5.2.1.2 Checking Manipulation Consistency

We will present methods for automatically detecting the above-described manipulations in the following section, but first we will discuss how to use those manipulations to discover possible manipulation relationships between two images. If we have the directions of the five above-mentioned manipulations for images A and B, it remains for us to evaluate whether or not they make sense all together. Each method can give one of three possible results about the parent-child relationship between the



(a)
Inconsistent directions from individual manipulations.
(Neither image is parent)

(b)
Consistent directions from individual manipulations.
(Left image is parent of right)

Figure 5.5: Examples of multiple directional manipulation detectors and their relative levels of consistency.

two images: 1) the manipulation indicates A is the parent (or ancestor) of B, 2) the manipulation indicates that B is the parent (or ancestor) of A, or 3) the manipulation is not present, giving no information about the relationship. If these detection results all agree on the directionality, then it is plausible that there is a parent-child relationship present. If not, then most likely there is no such relationship. Also note a parent-child relationship does not assert one image is the immediate source from which the other one is derived. There could be intermediate generations of copies between the two.

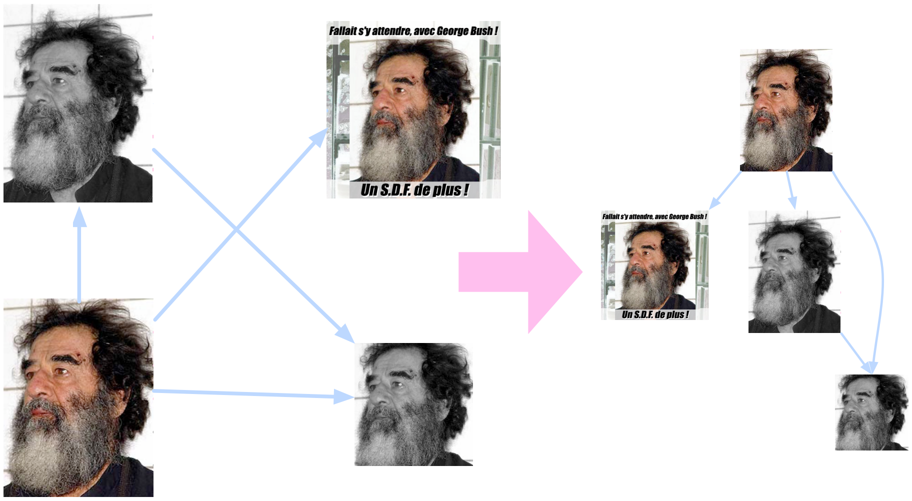

Fallait s'y attendre, avec George Bush !
Un S.D.F. de plus !
Fallait s'y attendre, avec George Bush !
Un S.D.F. de plus !

Figure 5.6: An emerging graph structure from multiple pair-wise checks of manipulation consistency

In Figure 5.5, we show some examples of how image manipulations can either be consistent or inconsistent. At the top, in Figure 5.5a, we show a case where the detectors are giving conflicting stories. The scaling, overlay, and insertion cues indicate that the right image is the child, while the grayscale cues would suggest just the opposite. (Also, there appears to have been no cropping). The contradictory stories being told by the manipulation detectors indicates to us that neither image is derived from the other. It is more likely that each is derived from some other parental image, and the two are cousins or siblings in the manipulation history.

In Figure 5.5b, we see an example where the individual cues are in agreement. The scaling, grayscale, and cropping cues indicate that the right image is derived from the left one, while no insertion or overlay effects appear to be present. The cumulative effect of all of these detections (and their agreement) is that it is plausible that the left image is, indeed, the parent of the right one.

Across many instances of these pair-wise relationship checks, we can give rise to a graph structure that approximates the plausible manipulation history of the image. Such a result is shown in Figure 5.6

5.2.2 Contextual Manipulation Cues

The precise directionality of certain types of manipulations cannot be detected from a single pair of images, alone. Comparing Figures 5.4a and 5.4e, a human can easily tell that 5.4e contains an overlay. A machine, however, would only be able to discern that the two images differ in the region of the overlay, but would not necessarily be able to infer which image contains the original content and which one contains the overlay. By considering all of the images in Figure 5.4, an automated algorithm would see that most images have content similar to Figure 5.4a in the region of the overlay, which would imply that Figure 5.4e is the outlier, and is likely to have been the result of an overlay manipulation. We will later utilize this cue to distinguish between ambiguous or difficult-to-detect operations, such as cropping, overlay, and insertion.

This context provided by a plurality of instances of the image is also needed to obtain information about the manipulation history. After we have detected each of the manipulation cues and evaluated their consistency, we are essentially left with a consensus-based decision about the existence (and direction) of parent-child relationships between pairs of images. We take each of the images to be nodes and form directed edges between nodes based on these detected parent-child relationships (as shown in Figure 5.7). The interpretation of this graph is that, where a directed edge exists between two image nodes, it is plausible that a series of manipulations resulted in one image being derived from the other.

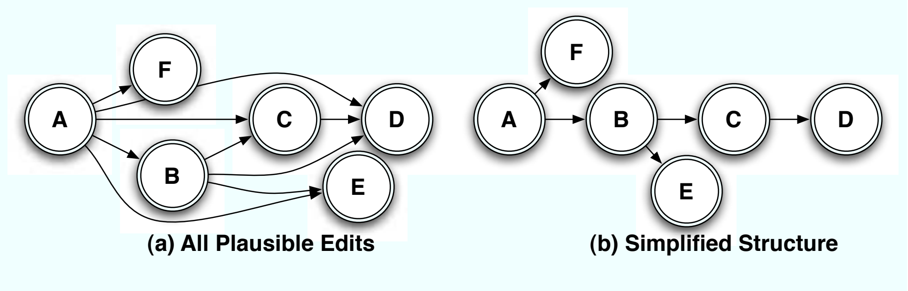


Figure 5.7: Simplification of redundancy in VMMs.

5.2.3 Visual Migration Maps

Recall that the graph showing the plausible manipulation relationships between images is called a Visual Migration Map. Depending upon the nature of the original pool of images used to conduct this manipulation history detection, the emergent structure can be quite different. If the pool is diverse (perhaps drawn from Web image search results), then we would expect to find several different connected components of different (non-copied) images found within the pool. Graphically speaking, these connected components are sets of nodes that are only connected to each other and not to any other portion of the graph. Intuitively speaking, the connected components are clusters where all of the nodes are (manipulated) copies of a given image, but every other connected component represents a set of copies for a different image. If the pool is rather homogeneous (perhaps a human manually provided a set of known copies), then we would expect to find a single connected component covering all of the images. Regardless of the structure of the pool at large, each individual connected component leads to a resulting VMM.

In general, there will be redundancy in the graph structure of each VMM. In practice, our detection approach will result in a structure like Figure 5.7a, since it is plausible for an image to have been derived from its parent or its parent's parent, there will be links formed between an image and each of its ancestors.

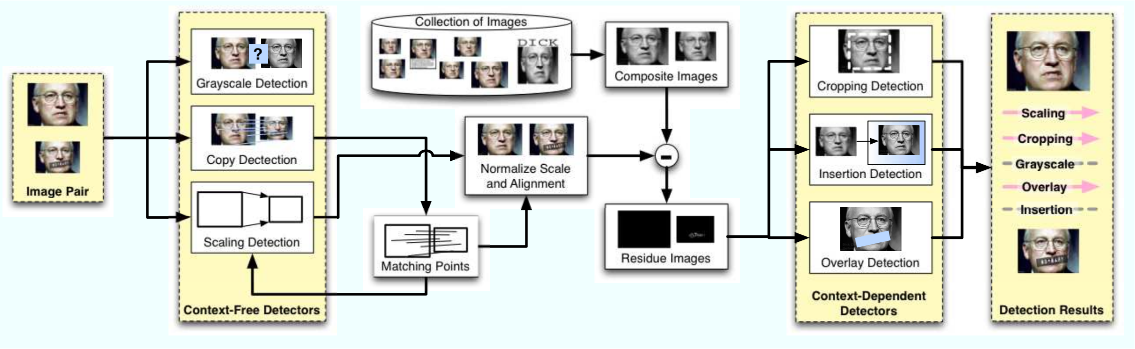


Figure 5.8: Proposed system architecture for automatically detecting various types of image manipulations. In the first stage, copy, scaling, and grayscale effects can be detected using only the two images. The scaling details can then be used to align the images against other available versions of the image to infer details about cropping, insertion, and overlays.

In determining the actual VMM of an image, either path between two images is equally plausible. From a practical point of view, however, we are equally unsure of how truthful either path is. We simplify the structure by always choosing the longest path between two images, resulting in structure similar to 5.7. This is not necessarily better than any other graph simplification, but it is practical in that it retains important aspects, such as the sink and source nodes, and assumes that each image inherits the manipulation history of its parents.

Similarly, our automatic detectors may be faulty and result in cycles in the graph. We can handle these cycles by the edges causing the cycle based on some criteria (e.g., the confidence score in detecting each manipulation operation). This may lead us astray from the true VMM of the image, but the resulting structure will remain sufficient for image discovery applications in the IIAS framework. In our experiments, we do not find cycles in our automatic VMMs.

5.3 Automatic Manipulation Detection Components

In this section, we will discuss specific automatic algorithms that we have used to implement the manipulation detection methods that we will use to infer information about the parent-child relationships between images. A framework for the system is shown in Figure 5.8. We divide the detection methods into *context-free* detectors, where all we need is the two images and the manipulation detection can be done directly, and *context-dependent* detectors, where we need to gather information from other images within the set to determine the exact nature of the edits occurring.

5.3.1 Context-Free Detectors

5.3.1.1 Copy Detection

The first step in the automatic system is to ascertain whether or not the two images are copies of each other, namely the two images are derived from the same source image through distinct manipulation operations. In principle, any generic image near-duplicate method can be applied here. In our current implementation, we adopt the simple but sufficiently effective method for copy detection discussed in Section 4.2.1. Here we also need to retain the spatial locations of matching points and below we elaborate on this further. Each image has a set of SIFT descriptors, I_S, where each descriptor is a tuple of $(x_i,y_i,\mathbf{f})$, where x_i and y_i are the spatial X-Y coordinates of the interest point in the image, and $\mathbf{f}$ is the 128-dimensional SIFT feature vector describing the local geometric appearance surrounding the interest point. Given two images, A and B, we exhaustively search all pairwise point matches between the images. We detect a matching set when the euclidean distance between the points' features, $D(f_{A,i};f_{B,j})$ falls below a given threshold. Matching points between A and B are then retained in a set, $\mathbb{M}_{A,B}$, which consists of a set of tuples,

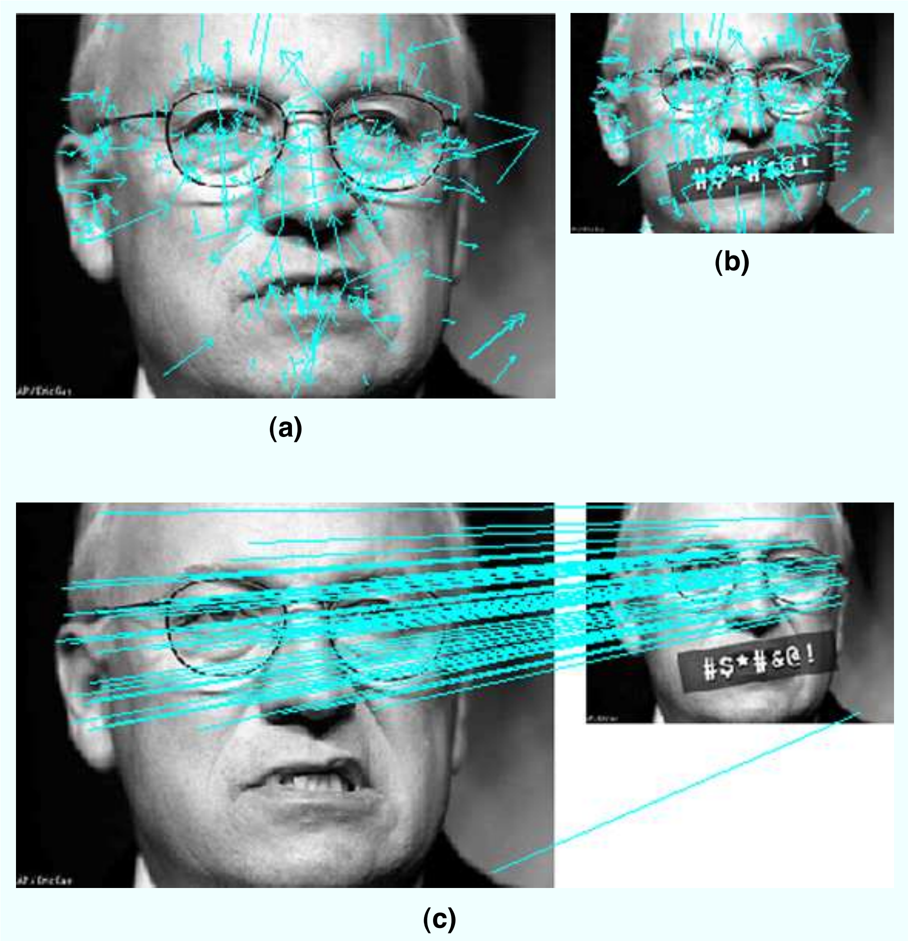


Figure 5.9: Examples of SIFT descriptors detected in two images (a) and (b) and the sets of matching points detected between the two sets (c).

$(x_{A,i}, y_{A,i}, x_{B,j}, x_{B,j})$, marking the locations of the matching points in each image. Figure 5.9c shows the resulting pairs of matching points detected across two images. We then apply a threshold on the number of matching points, $\mathbb{M}_{A,B}$, in order to get a binary copy detection result. In our experiments, we have set this threshold equal to fifty, since some of our initial observations have shown that this yields precise detection. (For comparison, each image in our data set contains between several hundred and a thousand interest points.)

5.3.1.2 Scaling

An important piece of information that emerges from the above-described copy detection approach is the set of matching points, $\mathbb{M}_{A,B}$. Assuming no image rotation is involved, the scaling factor between the two images, $SF_{A,B}$, can be estimated directly as the ratio in the spatial ranges of the X-Y locations of the matching points:

$$SF_{A,B} = \frac{max(x_A) - min(x_A)}{max(x_B) - min(x_B)} \tag{5.1}$$

The same estimate can be computed for the Y-dimension to account for disproportionate scaling. We apply a threshold to $SF_{A,B}$ in order to arrive at a binary detection of scaling. A more principled approach might be to apply random sample consensus (RANSAC) [18], which iteratively estimates a mathematical model given sample data points in the presence of noise and has been frequently used for image registration in computer vision and remote sensing to account for various transformations such as scaling, translation, and rotation.

We can utilize the above estimation to normalize the scales and align the positions of two images. It implies that A is $SF_{A,B}$ times larger than B, so we can generate B', which is at the same scale as A', by scaling and interpolating its pixels by a factor of $SF_{A,B}$. In addition, simple shift operations can be performed to align the interest points (and corresponding pixel content) of B' with A. Such scale normalization and position alignment can then be later utilized to conduct pixel-level comparisons to detect many types of manipulation artifacts.

5.3.1.3 Color Removal

To implement the color removal detector, we start by estimating whether each individual image is in grayscale. In the trivial case, the image is stored as a grayscale image, so we can see unambiguously that the image is contained in a single channel of grayscale intensity information. This case seems to account for roughly 50% of the grayscale images that we encounter. The remaining images are grayscale, but are stored in regular three-channel files (such as RGB or YUV). For these cases, we analyze the differences between the red, green, and blue channel values for each pixel. For grayscale images, we expect these differences to be zero. We calculate the mean over all of these channel differences over every pixel in the image and take images below a certain threshold to be grayscale. (The median or mode of this distribution may be less prone to error, but in our implementation, which may be explored later.) This has the benefit of causing tinted or other "semi-colored" images to be classified as grayscale, which is the right choice in many situations. Once we know whether each of the two images are in color or in grayscale, we can then estimate the direction of the color removal edit.

5.3.2 Context-Dependent Detectors

The nature of certain types of manipulations cannot be detected directly from just a pair of images. Consider, for example, the case of two images: one is an original instance of the image and the other contains a portion of overlayed image content. Given just the two images, we could most likely compare pixel intensities and discover that there is some area where the images have different content, but how can we know which image is the original and which one is the derived version? Consider the images in Figure 5.4. If we only had the images in 5.4a and 5.4b, all we would know is that one has a larger crop area than the other. But, if we only had images in 5.4a and 5.4f, we would reach the same conclusion: one has a larger crop area than the other. Given just two images, we can detect if one has a smaller crop area than the other, but what does that actually tell us? Perhaps the smaller image resulted

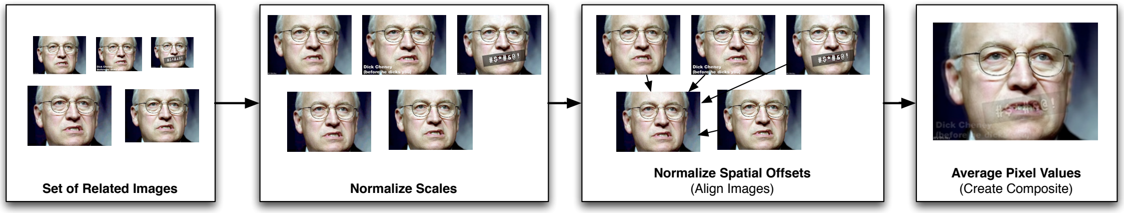


Figure 5.10: Creation of a composite image from multiple scaled and aligned copies.

from cropping the larger image, or maybe the larger image resulted from inserting the smaller image in a larger scene.

To address these problems, we look at image differences in the context of all of the other copies of the image that we have available. We consider the set of all images, $\mathbb{I}$, within a connected component obtained via analysis of the copy graph construction described in Section 5.2.2. Suppose that we would like to evaluate an image I_A. The image has a set of "neighbors," $\mathbb{I}_{A,N}$, which are the images that have been detected as copies of I_A. Within this set, we can use the method described in Section 5.3.1.2 to normalize the scales and offsets between each image in $I_{A,N}$ and I_A, yielding a scaled-shifted version of the images, $I_{A;N''}$, such that they can be composited on top of each other with pixel-wise correspondences. We can construct a composite image:

$$I(x,y)_{A,comp} = \frac{\sum \mathbb{I}(x,y)_{A,N''}}{|\mathbb{I}(x,y)_{A,N}|} \tag{5.2}$$

where each pixel in the composite image, $I_{A,comp}$ is essentially the average of the values of the corresponding pixels in the images in the neighbor set $I_{A,N}$. This composite image gives us contextual information about the typical appearances of areas of the image across many different copies of the image. An example of this process of creating a composite image is shown in Figure 5.10. We can compare the content of I_A against the composite content in $I_{A,comp}$ to find regions of I_A that are

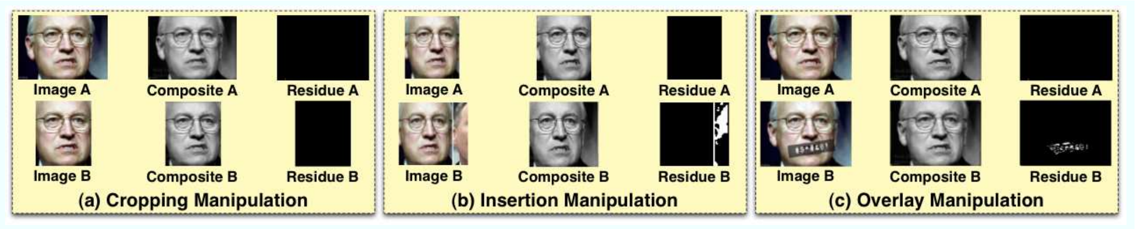


Figure 5.11: Examples of residue images in cropping, insertion, and overlay manipulations.

atypical. We do this by finding the residue between the two:

$$I_{A,res} = |I_A - I_{A,comp}| \tag{5.3}$$

where the residue image, $I_{A,res}$, is the absolute value of the pixel-wise difference between the image and the composite of its neighbors. We apply a threshold to $I_{A,res}$ to binarize it. Pixels falling below the threshold are valued at 0 (similar to the composite image) and the other are valued at 1 (different from the composite). Each image has a unique set of neighbors, so a different composite image is calculated for each image in the set. Now, if we wish to compare I_A against some other image, I_B, we can similarly produce composite and residue images $I_{B,comp}$ and $I_{B,res}$ and use the residue images $I_{A,res}$ and $I_{B,res}$ as proxies for evaluating the pair I_A and I_B.

In Figure 5.11, we see some examples of the appearances of the composite and residue images. The composite images sometimes still show traces of manipulations that are present in other images, but are largely true to the original content of the image. The key intuition behind the resulting residue images is that they are such that we expect that areas that are consistent with the original image will be black and areas that are inconsistent will be white. These residue images are then powerful tools that can be used to disambiguate the directions of overlay manipulations or to clarify the differences between crops and insertions. We will discuss the specifics of these manipulations in the following sections.

5.3.2.1 Cropping and Insertion

In Figure 5.11a, we see an example of how a crop manipulation would manifest itself in terms of the composite and residue images that drive our detection system. We can see from the example quite plainly that the image content in the larger-crop-area image is consistent with the composite image, which is reflected in the darkness of the residue image. This is consistent with a cropping operation. In Figure 5.11b, on the other hand, we see an example of an insertion operation. Here, the content of larger-crop-area image is significantly different from the composite image, which is reflected in the many white areas of the residue image. This is consistent with an insertion operation. In summary, candidates for cropping and insertion are discovered by finding image pairs with significant differences in image area. Cropping and insertion can then be disambiguated by examining the properties of the residue image in the out-of-crop region.

5.3.2.2 Overlay

In Figure 5.11c, we see an example of the composite and residue images that would be seen in the case of an overlay. We see that the overlay image has a region that is highly different from the original, which is reflected in a concentration of white pixels in the residue image. This is consistent with the existence of an overlay operation. Here, we can also see the relationship between the overlay and the insertion operations. In particular, they both exhibit image regions with high dissimilarity to the composite image. However, the areas of difference for overlays are inside the image crop area shared by both images, while these areas of difference are outside the main image crop area in the case of insertion.

5.4 Experiments and Analysis

We have applied the above-described automatic manipulation detectors to several sets of image copies found by applying copy detection to web search results. After each of the individual detectors have returned their results, we can use the consistency checking approaches and the graph construction techniques discussed in Sections 5.2.1.2 and 5.2.3 to use these automatically-detected cues to construct VMMs for each set of images and deliver summaries of their contents to end users. We evaluate our method in the context of Web image search by taking the top results returned to the user from a Web image search engine as our pool of images and extracting the manipulation histories of various highly-reused images contained within the set. Here, we will describe the queries that we have processed and the characteristics of the resulting data, along with the ground-truth manual annotations that we have generated about the manipulation operations associated with the image data. We will also discuss our intrinsic evaluations of the quality of the automatic manipulation detectors (from Section 5.3) and the VMMs that result by explicitly comparing against manually-generated manipulation labels and VMMs. Later, in Section 5.5, we will further evaluate the larger image archaeology system, extrinsically, in terms of its utility for applications in discovering “interesting” images from within the result pool.

5.4.1 Experimental Data

We evaluate the system against images from the Web. We use the same set of 100 queries described in Section 4.3.1 and the corresponding top-1000 image search results. Across this set of 1000 images per query, we then applied a copy detection approach (previously described in Section 4.2.1) to find all copy pairs within these

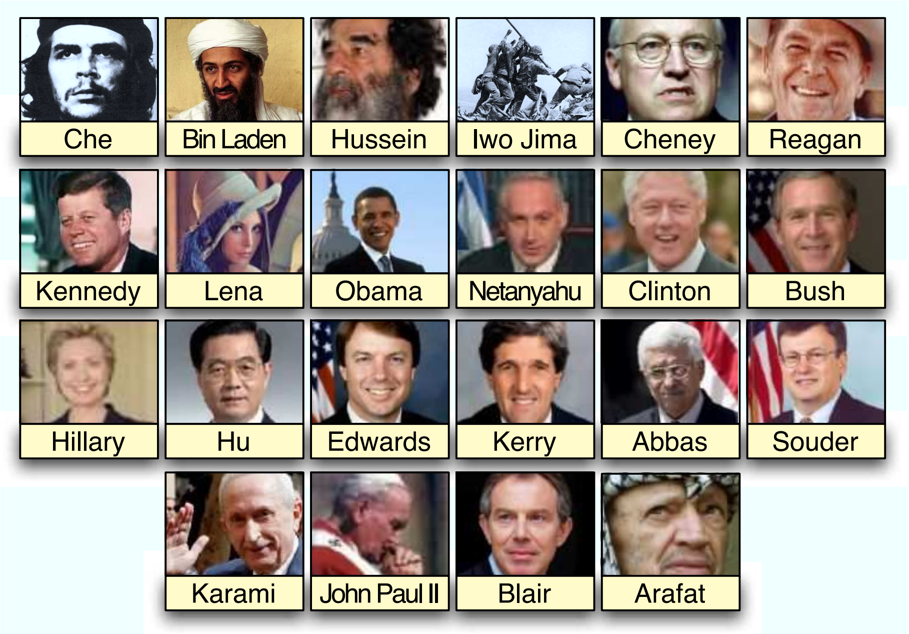


Figure 5.12: Images evaluated in our experiments. We use the proposed IIAS system to discover the plausible manipulation history for each of these iconic images on the Internet.

sets. We form edges between images to construct a copy graph, which typically consists of many different connected components. For each result set, we then take the largest connected component (i.e. the most-copied image) as the set of images to be fed into our manipulation detection and VMM construction algorithms. But, first, we filter down our connected components to only those which contain interesting manipulation patterns. Most classes of queries, such as locations, films, and artists, do not exhibit ideological perspective-changing manipulations. Some classes, such as political figures and celebrities do contain such manipulations. We do this filtering process manually, by visually skimming the contents of the connected components. This process might be automated by detecting the presence of manipulations using some adaptations of the methods that we have discussed. In the end, we evaluate the IIAS system against 22 unique queries, shown in Figure 5.12. For each query, the largest connected component (which we will process) typically contains several dozen copies of the image.

A single human annotator provides ground-truth labels for the manipulations that we wish to detect: copy, scaling, cropping, insertion, overlay, and grayscale. The annotator inspects each pair of images and individually labels whether any of these manipulations are present between the pair. If the manipulation is present, the annotator also labels the directionality of the manipulation (i.e. which image is implied to be derived from the other). Many of these manipulations can be very simple to observe visually. For example, a grayscale image is completely obvious. Overlaying external content and insertion within other images also tend to be quite observable. The degree to which other manipulations can be observed can be subject to the magnitude of the manipulation. In scaling, if one image is decimated by 50% compared to the other, then it should be obvious. A 1% relative scaling would be harder to accurately notice, however. So, as with any human-generated annotation, this data is subject to errors, but we contend that it is still helpful for comparing our automatic approach against manually-generated approaches. Given the individual pair-wise manipulation labels, we can then apply the consistency-checking approach from Section 5.2.1.2 to form parent-child links between images, based on manual (instead of automatic) labels. These links across a set of image copies form a manually-generated VMM, against which we can compare our automatically-generated VMM.

The human annotator also annotates two properties of each individual image: its manipulation status and the viewpoint that it conveys. The first property, the manipulation status, simply reflects whether the image is one of the types of images shown in Figure 5.2 (“original” or “highly-manipulated”). These annotations are gathered by having the annotator scan all of the images within a connected component of images to gather an intuition about the appearance of the original image crop area and content. These classes are largely easy to observe and the annotations are quite reliable. The second property, the viewpoint conveyed by the image, is more subjective and we rely on the content of the original HTML page that referred to the image. We examine these pages and evaluate the viewpoint of the document as either positive (supportive), neutral, or negative (critical) of the subject of the image.

5.4.2 Image Manipulation Detector Performance

We evaluate the core image manipulation detectors by comparing their results against the ground-truth labels given by the human annotator. We evaluate in terms of precision and recall. Precision is defined as the percentage of the automatically detected manipulations returned by our system that are manually labeled as true manipulations in our ground-truth. Recall is defined as the percentage of manually-labeled ground-truth manipulations that are successfully detected by our automatic system. Each of the methods relies on some sort of threshold to make a binary decision. Examples of these thresholds might be the absolute magnitude of the detected manipulation, such as the percentage by which a scaling edit decreased the size of an image or the percentage of the image that is occupied by detected overlay pixels. We scan over different threshold levels and observe the relative shifts in precision and recall.

We see precision-recall curves for each of the detectors in Figure 5.13. All of the basic, context-free detectors (copy detection, scaling, and color removal) have nearly perfect performance, each is able to exceed a precision in the range of 95% with recall in the range of 90%. The context-dependent detectors still leave some room for improvement, however. The most successful among these detectors is the insertion detection method, which retains moderately high precision through most

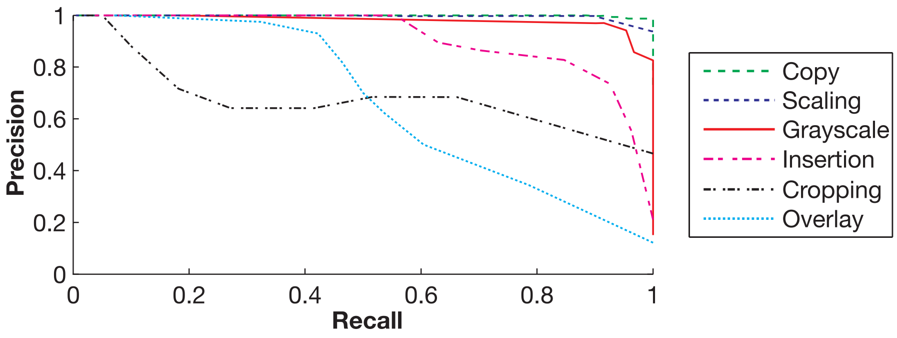


Figure 5.13: Performance of the manipulation detectors.

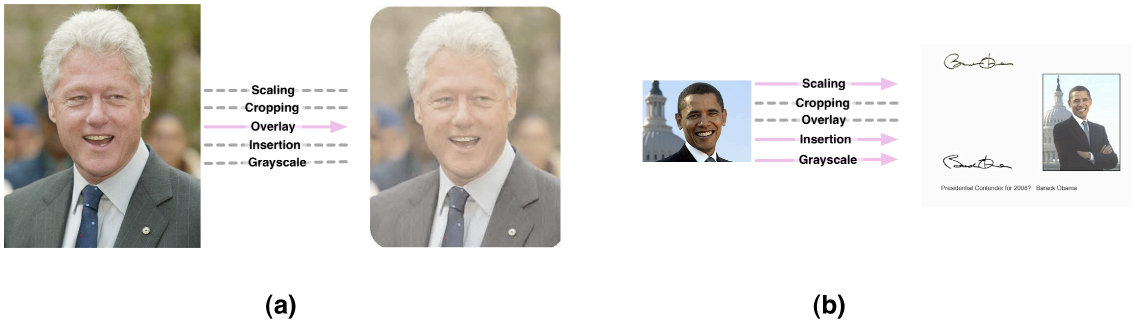


Figure 5.14: Examples of successful (a) and erroneous (b) detections of manipulations between images.

recall ranges. The overlay detection method provides near-perfect precision up to a certain recall level and then falls off precipitously. We find that the size and color contrast of an overlay is causing this effect: given overlays that are large enough and different enough from the original image, then the method performs well. Smaller, less-perceptible overlays still remain as a challenge. Cropping provides fair precision throughout all recall levels. Further observation of the errors typically leads us to mistrust the quality of our manual annotation of cropping effects. In many cases, where the crop is only a few pixels, close inspection would reveal that the machine was correct and the human was in error, so the computed precision-recall value may not reflect the strength of the detector.

In Figure 5.14, we see some results of successful and erroneous detection of manipulations between images. In Figure 5.14a, we can see quite clearly the absences of scaling, cropping, insertion, or grayscale effects. What we do see, however, is the overlay of white rounded edges at each of the four corners. According to our definitions, this is an overlay, and we have detected it successfully. In Figure 5.14b, we see some typical errors that result from some fragile aspects of the proposed system. While the system correctly detects the presence of scaling and the absence of overlay, the other aspects of the edit are incorrectly detected. First, the right image is mostly white or black, with a color image inserted. Our grayscale detector finds this to be a grayscale image because of this black/white dominance. Second, we see here an error induced by our cropping/insertion assumptions. Our system finds that the external crop areas in the right image are due to an insertion, but it fails to account for the fact that the left image has a much smaller crop area. The correct result would indicate that it is infeasible to derive the right image from the left one. A more successful approach to manipulation detection might require a more nuanced examination of the edge cases in cropping and insertion. In particular, we should have predicted cropping on the left-hand image and insertion on the right-hand one, a scenario which is not currently covered in our computational model of cropping and insertion.

The proposed methods all rely upon pair-wise comparisons, which leads to the majority of the complexity in the resulting application. Specifically, copy detection requires pair-wise comparison across all images in the input candidate set, which in our experiments is the 1000 images returned by a Web search result. Once the copy detection is complete across the set, the construction of a VMM for the largest connected component requires a pairwise comparison of only the images within that component (typically several dozen) . In our experiments, on an Intel Xeon 3.0 GHz machine, with our methods implemented in Matlab, it takes 2 hours to conduct

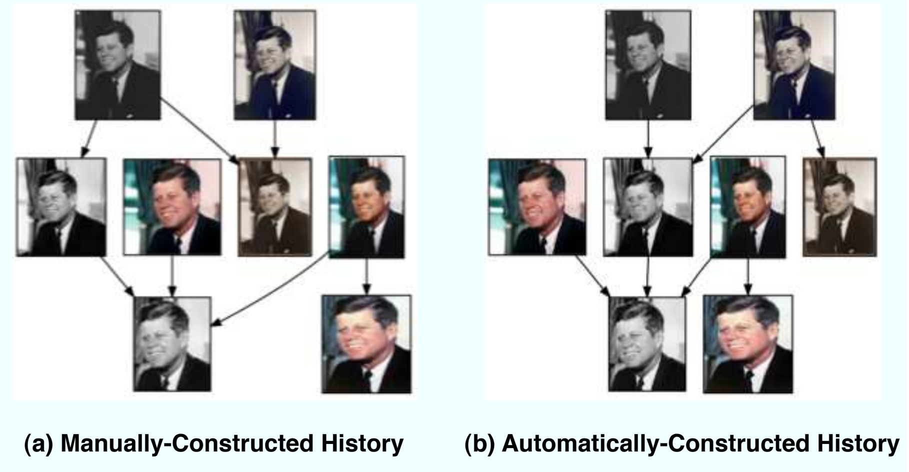

(a) Manually-Constructed History
(b) Automatically-Constructed History

Figure 5.15: Comparison of automatically-produced and manually-produced VMMs. Note the significant agreement.

the pair-wise copy detection over the initial web search results. Once the largest connect component has been found, it approximately 15 minutes to compute the VMM over that subset of images. Scaling towards pairwise comparisons of all images on the Web is obviously infeasible. To apply this method to a dataset of such scale, we would need to reconsider our copy detection approach using faster, constant-time hashing-based indexing approaches. From there, the connected components of copies of images will likely be manageable in size and the above-describe methods could feasibly be used in a massively parallel system to construct VMMs for each component in an offline indexing method.

5.4.3 Constructed Migration Maps

How useful are these manipulation detectors for constructing visual migration maps? To evaluate this, we construct VMMs using two methods: one using the ground-truth pairwise manipulation labels that we have collected by manual annotation and another using manipulation labels that are automatically computed by just trusting the results coming from our imperfect detectors. In Figure 5.15, we can

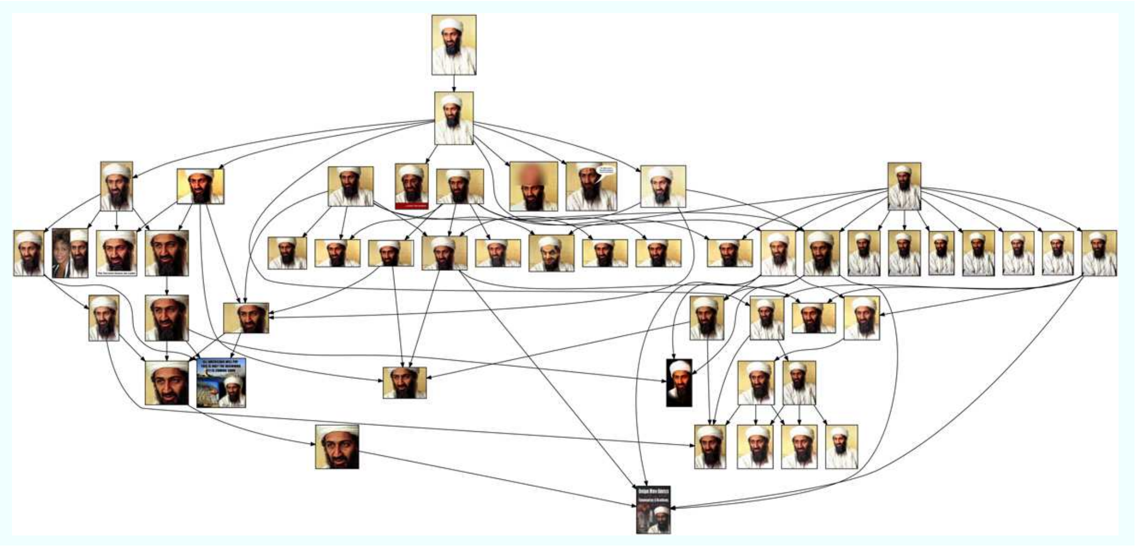


Figure 5.16: Automatically-produced visual migration map for an image of Osama bin Laden.

see a comparison between the resulting VMMs in one of the simpler images in our data set. A careful comparison between the two graphs reveals that there is a great deal of agreement between the two. Across all of the various image sets, we compare the automatic VMMs against the manual VMMs. Specifically, we do this by evaluating the pairwise relationship between each pair of images. We take the correct detection of an edge between an image pair as a true positive and the incorrect detection as a false alarm and evaluate in terms of precision and recall. In our experiments, precision is 92% and recall is 71%, on average. Some further examples of automatically-generated VMMs are shown in Figures 5.16 and 5.17.

5.5 Application Scenarios

The resulting VMMs that we have emerging from this analysis can give us a great deal of information about the qualities of the individual images, which can be used to help navigate and summarize the contents of a pool of related images in search or

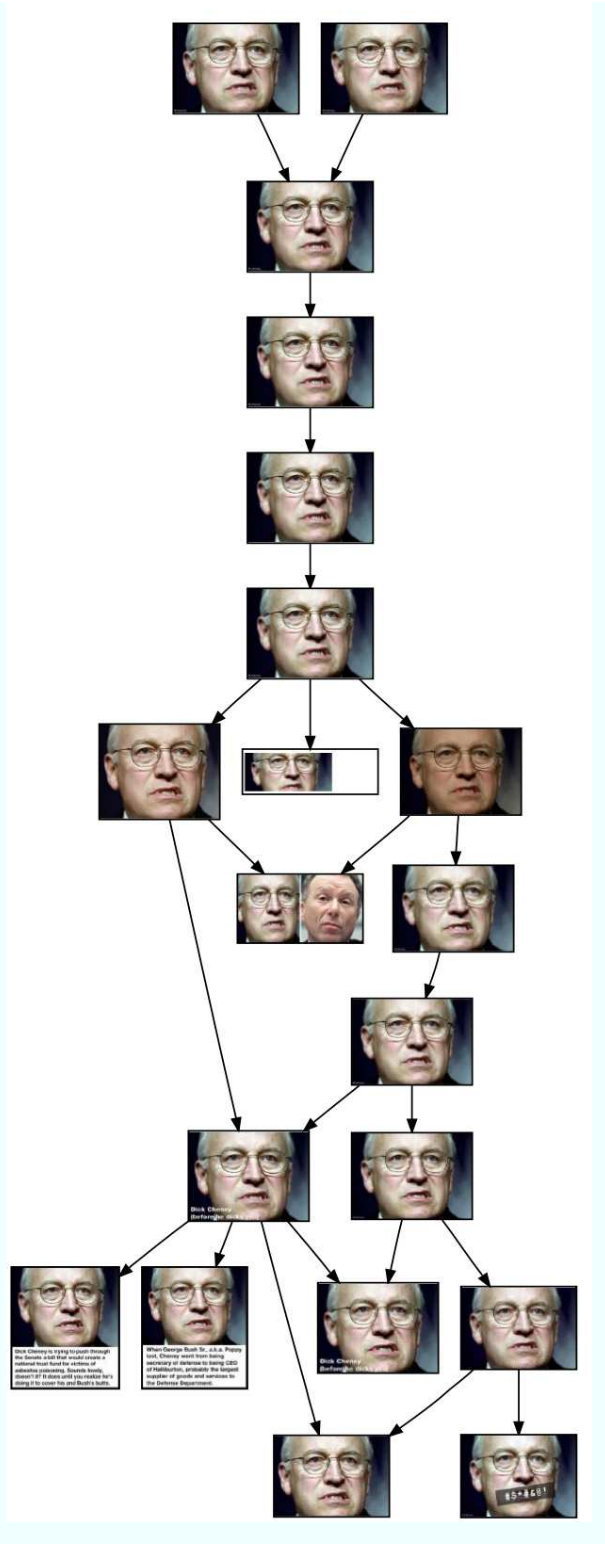

Dick Cheney
Dick Cheney is trying to push through the Senate a bill that would create a national trust fund for victims of asbestos poisoning. Sounds lovely, doesn't it? It does until you realize he's doing it to cover his and Bush's butts.
When George Bush Sr., a.k.a. Poppy lost, Cheney went from being secretary of defense to being CEO of Halliburton, probably the largest supplier of goods and services to the Defense Department.
Dick Cheney
#$*#&@!

Figure 5.17: Automatically-produced visual migration map for an image of Dick Cheney.

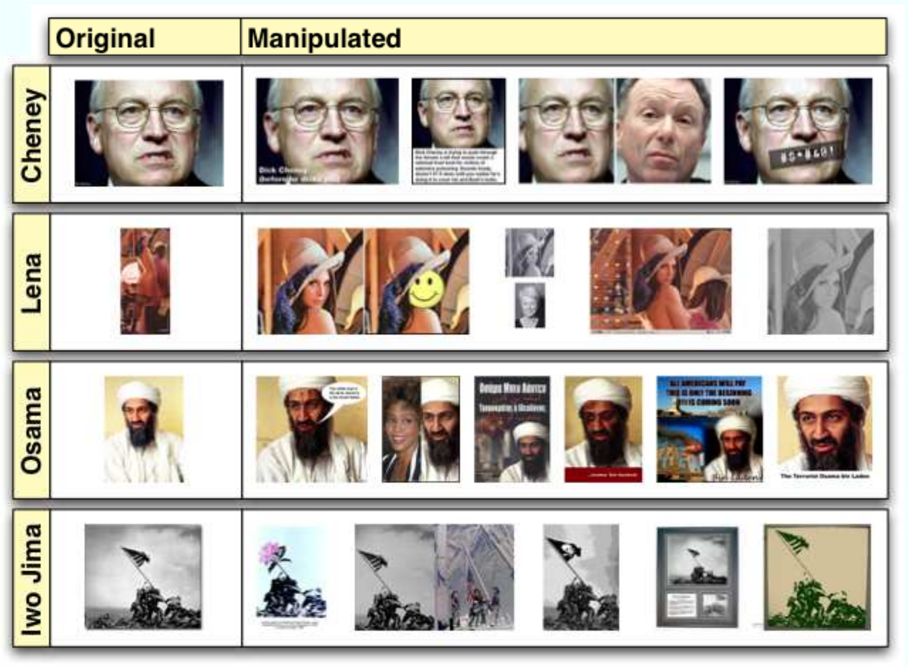


Figure 5.18: Examples of automatically discovered “original” and “manipulated” summaries for several images in our set.

exploration tasks. Most simply, the “original”-type images that we are seeking will be the ones corresponding to source nodes (those with no incoming edges) in the graph structure, while the “highly-manipulated”-type images that we are seeking will be the ones corresponding to the sink nodes (those with no outgoing edges). As we have stated earlier, the types of “highly-manipulated” images that we are most interested in are the ones whose histories include a relatively large amount of information addition, which leads to changes in meaning and context. We can disambiguate between these “information-added” types of highly-manipulated images and the less-desirable “information-subtracted” types by tracing the history from a source node, evaluating the types of manipulations experienced, and determining the relative number of additive versus subtractive operations that have taken place. Given these tools for analyzing the history of the images in the collection, the exact

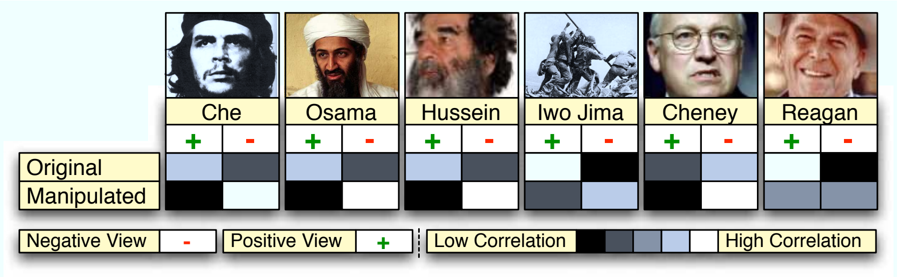


Figure 5.19: Observed correlations between image types and doc- ument perspectives.

mechanisms for feeding the results back to the users can be left to be adapted for the specific tasks at hand. In a search task, where authentic relevant images might be preferred, perhaps the “original”- type images will be most useful to the user and other copies can be suppressed and removed from the results. In an exploration task, where the user may be interested in exploring the different ideological perspectives surrounding a person or issue, the “interesting highly-manipulated” types might be most useful for comparative purposes. These considerations can be left for specific system designs, but they do rely on the ability of the image manipulation history to surface these various kinds of images. In Figure 5.18, we show examples of automatically discovered “original” and “manipulated” summaries which are indeed quite accurate.

5.5.1 Implications Toward Ideological Perspective

A key claim in this work is that the manipulations conducted against a particular instance of an image can change the image’s meaning and reflect the opinion being conveyed by the author. In Figure 5.19, we present a summary of the correlations between image types (“original” or “manipulated”) and the viewpoints represented by the Web pages upon which they appear. Here, we boil the viewpoints down to simply “positive” (for cases in which the author takes a position specifically in favor of image subject, or the author is neutral and takes no position) or “negative” (for cases in which the author takes a position specifically opposed to the image subject). Through many of the examples, including “Che,” “Osama,” “Hussein,” and “Iwo Jima,” we can see that there is, indeed, a correlation between the type of image and the viewpoint of the document. In these cases, the original-version images are highly associated with positive and neutral documents, while the manipulated images are associated with negative documents. This lends some credence to our assertion that the status of the manipulation history of an image can indicate the meaning or ideological perspective that it conveys. With some other images, manipulations are not observed to change the meaning as much. For the “Cheney” image, the original version is already unflattering and is frequently used as-is to convey negative viewpoints. Though, in some cases, it is manipulated, and the resulting images are still associated with negative viewpoints. The “Reagan” image is rarely manipulated in our set, so there is little chance to discover such a correlation.

5.6 Related Work

To the best of our knowledge, this is the first work that has dealt directly with the issue of automatically detecting the manipulation history of an image and utilizing this information to discover important instances of images on the Internet and explore how they communicate different ideological perspectives. There are, however, several works in related fields upon which this work draws some influence.

This work, of course, is dependent upon other research results in copy detection, which were reviewed in Section 4.5. Again, in this work, we do not seek to re-invent near-duplicate or copy detection, and, instead, we stand on top of existing work and incorporate it as a component in our proposed system. We extend beyond copy detection by turning attention specifically towards the ways in which two duplicate images differ and detect the manipulations that the image has been subjected to.

In the field of image forensics, the objective is typically to take a single image and identify whether or not any manipulations have taken place [28, 16]. Approaches in this field may involve checking the consistency of various regions of the image for artifacts induced by the physics or the peculiarities of the acquisition and compression processes. Such cues can be used to identify the camera used to take a photograph or if two regions of a photograph are derived from separate sources. A typical proposed application for such systems would be to verify the authenticity of individual images. Our work differs in that we do not consider single images, instead we evaluate the manipulations of images in the context of a plurality of various instances of the same image, which makes the task easier. We further aim to not only detect the presence of a manipulation, but to also characterize the types and history of manipulations and use that information to enhance browsing and search.

One of the goals of our work is to extract cues about the message conveyed by the photograph and how that may have been subverted through manipulation. Some works have looked at the messages and ideological perspectives inherent in multimedia documents in a more general sense. In [47], the authors have investigated the the differences between documents dealing with the same subject from different ideological standpoints. In text documents, it is shown that documents reflecting different sides of an issue have divergent distributions of divisive keywords. Similarly, in visual documents, the use of differing types of imagery (such as tanks and explosions versus peaceful scenes) can express differing viewpoints [48]. These effects, of course, are exhibited by completely separate documents. In this work, we examine how ideological differences are expressed through manipulation and re-distribution of the same original document.

Also related to this work is the system proposed in [70], which registers a plurality of tourist photographs of popular landmarks against each other using SIFT point matching (like we discussed earlier) along with structure from motion techniques. From these matches, the authors can estimate a quasi-3D view of the scene, estimate the location from which photographs were taken, and ultimately present a reconstructed composite view of the scene. In this dataset, there may be copies and manipulations that could be detected, but no such attempts were made. Similarly, the definitions of operations like "overlay" discussed earlier in our work might be relaxed to detect which images have occlusions of the scene cause by people or other objects and this information might be used to hide or highlight such images, depending upon the application.

In the same vein, our previous work [43, 39] operates on collections of co-located tourist photographs of landmarks and forms clusters of visually-similar images. The system can then extract the most "representative" photographs from the clusters by following the assumption that many users are trying to capture the same important angles and views of the landmark, so the representative photographs will tend to be central to these visually-similar clusters. That work, again, focuses on suppressing the outlier images and makes no attempt to characterize them and possibly bring the interesting outliers to the surface.

Finally, we should note that an interesting aspect of this work is that we focus on structural cues related to patterns that are external to the semantic content of the documents, themselves. Indeed, this bears resemblance to the PageRank [59] algorithm, which utilizes cues from hyperlinks between webpages to infer the authority of each page. If the links are used to form a directed graph, then the stationary probability of a random walk over that graph can yield an estimate of this authority. This idea has recently been transferred to the domain of images by forming graphs over image search results via image similarity and applying the same random walk process [51, 37]. There are still problems with this approach, however, since it is still an open issue as to how to determine the directionality of edges between images in these similarity graphs. Perhaps the directions given by manipulation histories (or their inverses) might be applicable.

5.7 Summary and Future Work

We have proposed an Internet image archaeology system, which can be used to process a given set of related images and find interesting instances of images or ideological perspectives. A key component of this IIAS framework is the visual manipulation map, which assumes that images on the Web are frequently copied, manipulated, and re-used, and that, in the case of iconic or popular images, this behavior can lead to a plurality of instances of the image across many sources. The VMM can acquire knowledge about the shared history among these photographs and the lineage of each instance, leading to intelligent approaches for browsing and summarizing the collection of copied photos. In particular, we believe that specific instances of images (such as the one closest to the original photograph or the versions with the most added external information) will be of the most interest to users. To find these images, we aim to detect parent-child derivation relationships between pairs of images and then construct a plausible VMM.

We have suggested a novel approach of considering pairs of near-duplicate images in the context of the plausible combinations of manipulations that could have resulted in one image being derived from the other. We propose that many types of manipulation operations performed on images are directional, meaning that they either remove information from the image or inject external information into the image. So, there are clues about the parent-child relationship between images encoded in the content of each of the images. We decompose the parent-child relationship detection problem into the task of individually detecting each type of manipulation and its directionality. Given these results, we can then check whether the directions of manipulations are all in agreement or not. We show that the directionality of many types of editing operations can be reasonably detected automatically and that we can construct useful plausible visual migration maps for images and use these cues to conduct archaeological digs against Internet image collections to discover important images from within the set.

Future work may expand upon this work by going into greater depth in the investigation of how users might leverage the discovered history graph to navigate through the collection of images. Other work might explore the relationship between the manipulation status of an image and the ideological content of its associated Web page. Since manipulation is shown to be correlated with ideological differences, natural language processing could be used with this approach to further extract the ideological perspective in multimedia documents. Also, the collection of individual manipulation detectors that we have proposed could be expanded to reflect other aspects of image manipulations.

We have observed that, out of 100 queries submitted to a Web image search engine, only 22 queries (or, about 20%) returned images exhibiting manipulations that could be mined to extract interesting visual migration maps. These migration maps are "interesting" because they contain some images that have been subjected to insertions and overlays, which change the visual content of the image (and therefore reflect varying viewpoints). By contrast, the image sets with "uninteresting" migration maps contain only scaling, cropping, and grayscale manipulations, which do not change the content or ideological position of the image. Most of these interesting highly-manipulated images are related to political figures. It is still unclear how frequently users will find images with interesting visual manipulation maps. In general usage cases, do 20% (or a different percentage) of image queries exhibit interesting patterns? Future work might explore the overall reach of mining manipulations in actual scenarios by leveraging collections of real queries from actual users to understand how well the approach generalizes.

Finally, we also note that one run of the proposed IIAS framework only presents a single snapshot of the ongoing evolution of the meanings and re-use patterns in image collections. So, through running the system periodically over time and tracking the emerging structures, we may uncover temporal aspects of the ways in which visual migration maps present themselves and grow over time. Similarly, image manipulations may spread spatially over different geographical and cultural locations or topologically across Internet-based sub-cultures. In future work, we might take the initial results that we have obtained so far for the initial 100 queries as a seed for capturing the visual migration map maps over an expanded scope on the Web over time and space. Utilizing the content of the web pages encompassing the images within the migration map, we may extract text patterns or hyperlink structures to further probe the web and expand the utility of the novel visual migration map structures.

Chapter 6

Conclusions and Future Work

In this final chapter, we will summarize the conclusions that we have reached in the work presented in this thesis along the four major directions that we have explored – learning models for conducting query-adaptive search over multimodal collections, leveraging large concept vocabularies for semantic search, exploiting the patterns of reuse in multi-author collections, and uncovering the manipulation histories of images. We will then present a few potential areas for extension and applying these research results.

6.1 Summary of Research

This thesis addressed the problem of multimedia retrieval by developing a multimodal search framework and investigating new cues and search methods that could fit within that system. Below, we summarize the specific contributions and findings in each of these areas.

6.1.1 Learning a Multimodal Retrieval Framework

We have begun by developing a multimodal retrieval framework, which can leverage cues from any number of search tools. The general approach in this system calls for each unimodal search method to be applied independently and then to have the results of each search fused to form a multimodal result. We find that a problem arises in such a framework that some cues are more powerful for certain queries and outright detrimental in others. We look towards prior work in query-class-dependent models for search to address these issues by choosing to weight various search methods according to their predicted power for a given query. We find that previous approaches to query-class-dependency are insufficient in that they require human system designers to decide the classes of queries by hand. We therefore propose to discover these models automatically by clustering previous queries with known ground-truth according to their similarity in terms of which search methods perform best and worst for each query. We find that the automatic model for discovering query classes exceeds human-defined models in terms of retrieval performance. We further find that the automatic approach confirms some of the intuitions in hand-defined models (such as the need for "named-person" or "sports" classes of queries), but also that many unintuitive classes can be discovered. The proposed model can grow to incorporate any number of search methods and can be adapted to any new search domain with only new example queries and without the need for any (erroneous) human intervention to develop new query class models.

6.1.2 Leveraging Large-Scale Concept Ontologies for Semantic Search

Given our framework for conducting multimodal search, we have moved on to investigate powerful new techniques that could be incorporated into such a system. In particular, we have taken a thorough look at the use of large-scale concept ontologies for powering semantic visual search. Here, visual concepts serve as a mid-layer representation of semantics. The visual content of images and videos can be mapped into this space by pre-trained concept detectors: the confidence scores of these detectors gives an approximation of the objects, locations, or scenes contained within a visual document. Users could query against this data by offering textual keywords. The textual keywords could be mapped into the space of the visual concepts via textual matches between the keywords provided and the names or descriptions of the visual concept. Such a method is, in many ways, an ideal situation for interacting with multimedia data. The documents are indexed based entirely on their visual content, requiring no external text or annotations and the search interface is based entirely on keywords, relieving the users of the burden of providing example images. We have found, specifically, that concept-based search is among the most powerful tools available for multimedia search. Also, we have proposed a new reranking framework for leveraging many peripherally-related concepts for retrieval and have found that this framework offers significant increases in performance and is able to uncover many unique types of peripherally-related concepts that would otherwise not be found.

6.1.3 Exploiting Multimedia Reuse Trends in Multi-author Collections

We have further proposed a powerful new signal, which can be leveraged within a multimedia retrieval framework, based on the ways in which creators of multimedia documents choose and re-appropriate particular pieces of media. We propose to detect these reuse patterns automatically through existing copy-detection approaches and to use the relative popularity of an image as another signal to be incorporated in search systems by pushing more-popular images higher in the rankings. We find that this signal can significantly improve the relevance of returned image search results. We further find that it can also provide qualitatively “better” relevant images.

6.1.4 Image Manipulation Histories

In the above-mentioned exploration of image reuse patterns, we also found a curious trend that images are not necessarily copied directly and that there are, in fact, manipulations that take place between various versions of images. This implies that, at one point, there was an original image, and that all other images were derived from this image through a series of manipulations on it or on other images derived from it. We propose that the history of an image can be approximated automatically by detecting the presence of certain manipulations, such as scaling, cropping, overlay, color removal, and insertion. We find that many of these manipulation cues can be automatically detected with a high degree of accuracy and that the resulting automatically-detected manipulation history will have many features in common with manually-generated histories. In particular, the most-original images will surface at root nodes and the most highly-divergent images will appear at sink nodes. We find that there is a significant correlation between the appearance of highly-divergent manipulations and changes in the ideological viewpoint presented by the image, which might be leveraged to identify viewpoints of image authors and to aide in the disambiguation of ideological perspectives.

6.2 Future Directions

Each of the potential solutions and techniques that we have investigated in this thesis still have promise for interesting applications in future research efforts. We will discuss these below.

6.2.1 Query Adaptation in Search

We have proposed a system for query-adaptive multimodal retrieval, but what are the specific areas of opportunity for us to explore to move forward? Here, we will highlight the space with room for growth.

6.2.1.1 Fusion models and Quality of Search Components

At the heart of the query-adaptive approach to retrieval is the premise that various individual search approaches can be combined using different fusion models, depending on the query being handled. However, the majority of current multimodal search engines (including our proposed framework) use very simple and unimaginative fusion approaches: typically a weighted sum of scores or ranks. There would seem to be ample opportunity here to explore and exploit the interactions between modalities and search methods with greater subtlety. On the other hand, multimodal fusion can really only improve so much over the individual components available, and there is certainly plenty of work left to be done on the individual search components used in multimedia search systems.

6.2.1.2 Training data

A query-adaptive system is only as good as the queries (and relevance labels) that it has already seen. A major sticking point of multimodal search systems across a variety of domains is that acquiring enough training queries and relevance labels (both in quantity and in quality) is immensely difficult. One proposal that we can put forth is to turn such annotation tasks inside out and to allow communities of contributors to collect, save, and share the results of their search queries.

6.2.1.3 Difficulty prediction

In our model, we have chosen to adapt queries based on their memberships to certain classes. In reality, these classes serve as a proxy for predicting which search methods will be powerful and which will not. A more direct approach may be to predict the power of each search method directly, a prospect which has already been explored to some extent in the text search community [88, 87] and may yield benefits in multimedia search.

6.2.2 Concept-based Search

Concept-based retrieval is a relatively new approach to visual search and still has many possibilities for improvement, as described below.

6.2.2.1 Domain Adaptation

A major hurdle to concept-based indexing systems arises due to domain issues. The visual composition of multimedia documents varies significantly across many different domains, such as images on the Web, broadcast news videos, surveillance videos, medical imaging, and personal consumer photographs. This has significant impact on both the design of the visual ontology and the learned visual models.

Ontology Adaptation A visual ontology designed for one domain may be completely inappropriate for another. Given the large human cost of constructing such a visual ontology, we might hope that the entire process need not be repeated for each new vocabulary in every new application or domain. Can automatic tools, such as query log or metadata mining automatically point designers toward the important visual concepts? When system designers decide to add a concept to a growing ontology, can an automatic system scan existing hierarchical relationships from other existing vocabularies in order to suggest additional concepts (or entire sub-trees), which can be transplanted directly to the system?

Concept Model Adaptation Despite the differences in ontological relationships between domains, many key visual concepts may still be present across any two domains. However, the appearance of the same visual concept across different domains may be extremely divergent. Consider, for example the typical appearance of the "car" concept in a television commercial (either against a studio background or zooming along a slick road). Now, consider the appearance of the same concept in surveillance video (a small, grainy light region in a dark field). In current approaches, we would simply have to gather new sets of annotations for each concept in a new domain in order to learn concepts. This is extremely expensive. It seems that we should be able to leverage annotations and models already learned on one domain and adapt them to the new domain either directly or through the collection of a limited amount of annotation. This is an interesting topic that has already received some research interest [33, 86], but will require much more investigation as the reach of concept-based systems grows.

6.2.2.2 Formulating Conceptual Queries

Perhaps the most fundamental problem in concept-based search systems is the proper mechanism for utilizing concept models in retrieval. We have discussed issues of the scale of the concept lexicon and the development of new ontological relationships, but all of these aspects assume that we will ultimately know how to use the resulting vocabularies and visual models at query time. While there has been promising initial results in this direction, these are based on simple baseline approaches. The expansions towards the use of many concepts to answer queries have not demonstrated truly successful results. The problems here are largely the result of a lack of structured "understanding" of the systems of the interrelationships between visual concepts and spoken language words or concepts. The existing systems bridge the gap between the visual ontology and the English language by using WordNet as a middle ground. This is ultimately inadequate, since the semantic relationships useful for organizing language (synonyms, hypernyms, etc.) are not useful for representing visual compositions. The fact that a concept is a lexical sibling of a concept matching a query term is less powerful than the fact that a concept is likely to be visually co-occurrent with a concept matching a query term. In the end, the solution may only be to design a visual ontology on the scale of WordNet endowed with visual relationships instead of lexical ones. Such a project will, without doubt, require a great deal of effort, but the implications might be profound.

6.2.2.3 Large-Scale Learning

Finally, there is some promise for large-scale learning, which is driven by the use of large volumes of noisily labeled data, as opposed to smaller quantities of precisely labeled data [42, 75, 17, 78]. This approach is comparatively new and, while it shows a lot of promise, it also opens up many more questions. Some of these questions relate to the issues that we have discussed above. There are issues of scale. Current experiments stop way short of the volume of information that is truly available. Large-scale datasets may work for images on the Web, but many other types of media may not have noisy textual labels that we can leverage for learning. There are also issues with creating ontologies. Current experiments rely on human-designed ontological relationships in order to endow the system with some knowledge. If the volume of data (and our capacity to mine it) actually reaches the levels that we hope it will, will we be able to automatically extract more useful ontologies from this data than we can construct by hand through intuition? Clearly, there is an abundance of unstructured information encapsulated in the images created and shared by average users. The development of methods to properly harness this data and shape it into meaningful knowledge remains an exciting open challenge.

6.2.3 Image Replication Patterns

To the best of our knowledge, this is the first work to propose to leverage image reuse patterns to gain insights into the relative importance or popularity of various images and, as such, leaves much room for investigating the implications and possibilities of such approaches. In our proposed system for duplicate-based reranking, we apply duplicate detection to the results of image search queries to find the cumulatively most-popular images. This approach might not be the most-appropriate or most-interesting for all occasions. For example, as news events appear and recede over time, the most-relevant images for people or locations or other topics might not be the ones that are cumulatively the most popular over all time, but rather the ones that are most-recently popular, or "hot." Tracking the trends of image use in time-sensitive sources, such as newspapers and blogs, might give an interesting view into the views of people, places, products, and other topics that are of interest in the current moment.

6.2.4 Image Manipulation Histories

We have also proposed the first system to track or uncover the history of manipulation of an image, which opens many doors for further exploration. We outline some of these directions below.

6.2.4.1 Image Propagation

The image manipulation maps that we have presented are merely snapshots taken at one point in time. In reality, images are changed and reused over time. It would be of particular interest to periodically probe the Internet for the status of an image and create a temporal view of the image's growth over time and propagation over geographical distances or social topologies. Flurries of activity with a particular image at particular points in time, might provide a reflection of the relative interest in the subject of the image as it evolves over time. This investigation would, of course, be limited by our search engine-based approach to finding images. In reality, we are restricted only to images that are provided by image search results. The temporal appearance of images in search results might not reflect the images' actual creation or publication dates. Also, many images may still be omitted.

6.2.4.2 Profiling Authors

In our image manipulation framework, we have taken an image-centric approach to discovering and navigating the various versions of images. One possible different direction is to take a more author-centric approach. How does a particular author tend to use imagery? Are the author's images typically unedited and neutral? Or are they highly-edited, presenting negative viewpoints? Where do these authors images tend to lie in manipulation histories? We could possibly track the relative influence of particular authors based on how often their images are copied and reused by other authors. Or, we might find relationships between authors based on the tendency to copy and reuse each others' images directly.

References

[1] LingPipe Named Entity Tagger. Available at: http://www.alias-i.com/lingpipe/. 2004.

[2] J. Adcock, A. Girgensohn, M. Cooper, T. Liu, L. Wilcox, and E. Rieffel. FXPAL experiments for TRECVID 2004. In *TRECVID 2004 Workshop*, 2004.

[3] Arnon Amir, Janne Argillander, Murray Campbell, Alexander Haubold, Giridharan Iyengar, Shahram Ebadollahi, Feng Kang, Milind R. Naphade, Apostol Natsev, John R. Smith, Jelena Tesic, and Timo Volkmer. IBM Research TRECVID-2005 Video Retrieval System. In *NIST TRECVID workshop*, Gaithersburg, MD, November 2005.

[4] Arnon Amir, Marco Berg, Shih-Fu Chang, Giridharan Iyengar, Ching-Yung Lin, Apostol Natsev, Chalapathy Neti, Harriet Nock, Milind Naphade, Winston Hsu, John R. Smith, Belle Tseng, Yi Wu, and Dongqing Zhang. Ibm research trecvid-2003 video retrieval system. In *TRECVID 2003 Workshop*, Gaithersburg, MD, November 2003.

[5] J.R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C. Shu. The Virage image search engine: An open framework for image management. *Proceedings of SPIE, Storage and Retrieval for Still Image and Video Databases IV*, pages 76–87, 1996.

[6] Satanjeev Banerjee and Ted Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In *CICLing '02: Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, pages 136–145, London, UK, 2002. Springer-Verlag.

[7] Mandis Beigi, Ana B. Benitez, and Shih-Fu Chang. Metaseek: A content-based meta-search engine for images. In *SPIE Conference on Storage and Retrieval for Image and Video Databases VI (IST/SPIE-1998)*, volume 3312, San Jose, CA, January 1998.

[8] Ana B. Benitez, Mandis Beigi, and Shih-Fu Chang. Using relevance feedback in content-based image metasearch. *IEEE Internet Computing*, 2(4):59–69, July 1998.

[9] M. Campbell, S. Ebadollahi, M. Naphade, A. (P.) Natsev, J. R. Smith, J. Tesic, L. Xie, and A. Haubold. IBM Research TRECVID-2006 Video Retrieval System. In *NIST TRECVID workshop*, Gaithersburg, MD, November 2006.

[10] J.G. Carbonell, Y. Yang, R.E. Frederking, R.D. Brown, Y. Geng, and D. Lee. Translingual information retrieval: A comparative evaluation. *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 708–715, 1997.

[11] S.-F. Chang, W. Hsu, W. Jiang, L. Kennedy, D. Xu, A. Yanagawa, and E. Zavesky. Columbia University TRECVID-2006 Video Search and High-Level Feature Extraction. In *NIST TRECVID workshop*, Gaithersburg, MD, November 2006.

[12] Shih-Fu Chang, Winston Hsu, Lyndon Kennedy, Lexing Xie, Akira Yanagawa, Eric Zavesky, and Dongqing Zhang. Columbia University TRECVID-2005

Video Search and High-Level Feature Extraction. In *NIST TRECVID workshop*, Gaithersburg, MD, November 2005.

[13] T.-S. Chua, S.-Y. Neo, K.-Y. Li, G. Wang, R. Shi, M. Zhao, and H. Xu. TRECVID 2004 search and feature extraction task by NUS PRIS. In *TRECVID 2004 Workshop*, 2004.

[14] M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. *Proceedings of the twentieth annual symposium on Computational geometry*, pages 253–262, 2004.

[15] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[16] H. Farid. Detecting Digital Forgeries Using Bispectral Analysis. Technical Report AIM-1657, MIT, 1999.

[17] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from googles image search. *ICCV*, 2005.

[18] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[19] M.S. Flickner, H. Niblack, W. Ashley, J.Q.H. Dom, B. Gorkani, M. Hafner, J. Lee, D. Petkovic, D. Steele, D. Yanker, et al. Query by image and video content: the QBIC system. *Computer*, 28(9):23–32, 1995.

[20] E.R. Gansner and S.C. North. An open graph visualization system and its applications to software engineering. *Software Practice and Experience*, 30(11):1203–1233, 2000.

[21] J. L. Gauvain, L. Lamel, and G. Adda. The LIMSI broadcast news transcription system. *Speech Communication*, 37(1-2):89–102, 2002.

[22] Winston H. Hsu, Lyndon Kennedy, and Shih-Fu Chang. Reranking methods for visual search. *IEEE Multimedia Magazine*, 13(3), 2007.

[23] Winston H. Hsu, Lyndon Kennedy, and Shih-Fu Chang. Video Search Reranking through Random Walk over Document-Level Context Graph. In *ACM Multimedia*, Augsburg, Germany, September 2007.

[24] A. Hampapur, K. Hyun, and R. Bolle. Comparison of sequence matching techniques for video copy detection. In *Conference on Storage and Retrieval for Media Databases*, pages 194–201, 2002.

[25] J.A. Hartigan. *Clustering Algorithms*. John Wiley & Sons, Inc. New York, NY, USA, 1975.

[26] A. Hauptmann, RV Baron, M. Chen, M. Christel, P. Duygulu, C. Huang, R. Jin, W. Lin, T. Ng, N. Moraveji, et al. Informedia at TRECVID 2003: Analyzing and Searching Broadcast News Video. In *TRECVID 2004 Workshop*, 2004.

[27] A. Hauptmann, M.-Y. Chen, M. Christel, C. Huang, W.-H. Lin, T. Ng, N. Papernick, A. Velivelli, J. Yang, R. Yan, H. Yang, and H. D. Wactlar. Confounded expectations: Informedia at TRECVID 2004. In *TRECVID 2004 Workshop*, 2004.

[28] J. He, Z. Lin, L. Wang, and X. Tang. Detecting Doctored JPEG Images Via DCT Coefficient Analysis. *European Conference on Computer Vision*, 2006.

[29] HS Heaps. *Information Retrieval: Computational and Theoretical Aspects.* Academic Press, Inc. Orlando, FL, USA, 1978.

[30] Winston Hsu, Lyndon Kennedy, and Shih-Fu Chang. Video search reranking via information bottleneck principle. In *ACM Multimedia*, Santa Babara, CA, USA, 2006.

[31] Winston Hsu, Lyndon Kennedy, Chih-Wei Huang, Shih-Fu Chang, Ching-Yung Lin, and Giridharan Iyengar. News video story segmentation using fusion of multi-level multi-modal features in trecvid 2003. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Montreal, Canada, May 2004.

[32] Intel. Compute-intensive, highly parallel applications and uses. *Intel Technology Journal*, 9, 2005.

[33] W. Jiang, E. Zavesky, and A. Chang, S.-F. and Loui. Cross-Domain Learning Methods for High-Level Visual Concept Classification. In *IEEE International Conference on Image Processing*, San Diego, California, U.S.A., October 2008.

[34] Wei Jiang, Shih-Fu Chang, and Alexander C. Loui. Active context-based concept fusion with partial user labels. In *IEEE International Conference on Image Processing (ICIP 06)*, Atlanta, GA, USA, 2006.

[35] Wei Jiang, Shih-Fu Chang, and Alexander C. Loui. Context-based Concept Fusion with Boosted Conditional Random Fields. In *IEEE ICASSP*, 2007.

[36] Yu-Gang Jiang, Akira Yanagawa, Shih-Fu Chang, and Chong-Wah Ngo. CU-VIREO374: Fusing Columbia374 and VIREO374 for Large Scale Semantic Concept Detection. Technical report, Columbia University, August 2008.

[37] Y. Jing and S. Baluja. Pagerank for product image search. In *Conference on the World Wide Web*. ACM New York, NY, USA, 2008.

[38] L. Kennedy, S.F. Chang, and A. Natsev. Query-Adaptive Fusion for Multimodal Search. *Proceedings of the IEEE*, 96(4):567–588, 2008.

[39] L.S. Kennedy and M. Naaman. Generating diverse and representative image search results for landmarks. In *World Wide Web*, 2008.

[40] Lyndon Kennedy and Shih-Fu Chang. A Reranking Approach for Context-based Concept Fusion in Video Indexing and Retrieval. In *ACM International Conference on Image and Video Retrieval*, Amsterdam, Netherlands, July 2007.

[41] Lyndon Kennedy and Shih-Fu Chang. Internet image archaeology: automatically tracing the manipulation history of photographs on the web. In *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, October 2008.

[42] Lyndon Kennedy, Shih-Fu Chang, and Igor Kozintsev. To search or to label?: predicting the performance of search-based automatic image classifiers. *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 249–258, 2006.

[43] Lyndon Kennedy, Mor Naaman, Shane Ahern, Rahul Nair, and Tye Rattenbury. How Flickr helps us make sense of the world: context and content in community-contributed media collections. In *MULTIMEDIA '07: Proceedings*

of the 15th international conference on Multimedia, pages 631–640, New York, NY, USA, 2007. ACM Press.

[44] Lyndon Kennedy, Paul Natsev, and Shih-Fu Chang. Automatic discovery of query class dependent models for multimodal search. In *ACM Multimedia*, Singapore, November 2005.

[45] Risi Kondor and Tony Jebara. A kenel between sets of vectors. In *International Conference on Machine Learning*, 2003.

[46] Christina Leslie and Rui Kuang. Fast kernels for inexact string matching. In *Conference on Learning Theory and Kernel Workshop*, 2003.

[47] W.-H. Lin and A. Hauptmann. Are These Documents Written from Different Perspectives? A Test of Different Perspectives Based On Statistical Distribution Divergence. In *Proceedings of the International Conference on Computational Linguistics*, 2006.

[48] W.-H. Lin and A. Hauptmann. Do these news videos portray a news event from different ideological perspectives? In *International Conference on Semantic Computing*, 2008.

[49] W.H. Lin and A. Hauptmann. Which Thousand Words are Worth a Picture? Experiments on Video Retrieval Using a Thousand Concepts. July 2006.

[50] Hugo Liu. MontyLingua: An end-to-end natural language processor with common sense. Available at: http://web.media.mit.edu/˜hugo/montylingua. 2004.

[51] Jingjing Liu, Wei Lai, Xian-Sheng Hua, Yalou Huang, and Shipeng Li. Video search re-ranking via multi-graph propagation. In *MULTIMEDIA '07: Pro-*

ceedings of the 15th international conference on Multimedia, pages 208–217, New York, NY, USA, 2007. ACM.

[52] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[53] K. McDonald and A.F. Smeaton. A Comparison of Score, Rank and Probability-Based Fusion Methods for Video Shot Retrieval. *Image and Video Retrieval. LECTURE NOTES IN COMPUTER SCIENCE*, 3568:61, 2005.

[54] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–244, 1990.

[55] M. Naphade, L. Kennedy, JR Kender, SF Chang, JR Smith, P. Over, and A. Hauptmann. LSCOM-lite: A Light Scale Concept Ontology for Multimedia Understanding for TRECVID 2005. Technical report, IBM Research Tech. Report, RC23612 (W0505-104), May, 2005.

[56] M. Naphade, J.R. Smith, J. Tesic, S.F. Chang, W. Hsu, L. Kennedy, A. Hauptmann, and J. Curtis. Large-Scale Concept Ontology for Multimedia. *IEEE MULTIMEDIA*, pages 86–91, 2006.

[57] Apostol (Paul) Natsev, Alexander Haubold, Jelena Tešić, Lexing Xie, and Rong Yan. Semantic concept-based query expansion and re-ranking for multimedia retrieval. In *MULTIMEDIA '07: Proceedings of the 15th international conference on Multimedia*, pages 991–1000, New York, NY, USA, 2007. ACM.

[58] Apostol (Paul) Natsev, Milind R. Naphade, and Jelena Tesic. Learning the

semantics of multimedia queries and concepts from a small number of examples. In *ACM Multimedia*, pages 598–607, Singapore, 2005.

[59] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Computer Science Department, Stanford University, 1998.

[60] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, July 1980.

[61] P. Resnik. Using information content to evaluate semantic similarity. In *Conference on Artificial Intelligence*, 1995.

[62] S. E. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau. Okapi at TREC4. In *Text REtrieval Conference*, 1992.

[63] Anton Schwaighofer. SVM Toolbox, available at http://www.igi.tugraz.at/aschwaig/software.html. 2002.

[64] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, pages 888–905, 2000.

[65] SIFT demo program, David Lowe. http://www.cs.ubc.ca/˜lowe/keypoints/.

[66] Alan F. Smeaton, Paul Over, and Wessel Kraaij. Evaluation campaigns and trecvid. In *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, New York, NY, USA, 2006. ACM Press.

[67] John R. Smith and Shih-Fu Chang. Visually searching the web for content. *IEEE Multimedia*, 4(3):12–20, July 1997.

[68] J.R. Smith and S.F. Chang. VisualSEEk: a fully automated content-based image query system. *Proceedings of the fourth ACM international conference on Multimedia*, pages 87–98, 1997.

[69] JR Smith, M. Naphade, and A. Natsev. Multimedia semantic indexing using model vectors. *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, 2, 2003.

[70] N. Snavely, S.M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM Transactions on Graphics (TOG)*, 25(3):835–846, 2006.

[71] C. G. M. Snoek, J. C. van Gemert, Th. Gevers, B. Huurnink, D. C. Koelma, M. Van Liempt, O. De Rooij, K. E. A. van de Sande, F. J. Seinstra, A. W. M. Smeulders, A. H.C. Thean, C. J. Veenman, and M. Worring. The MediaMill TRECVID 2006 Semantic Video Search Engine. In *NIST TRECVID workshop*, Gaithersburg, MD, November 2006.

[72] C.G.M. Snoek, J. van Gemert, J.M. Geusebroek, B. Huurnink, D. C. Koelma, G. P. Nguyen, O. deRooij, F. J. Seinstra, A.W.M. Smeulders, C. J.Veenman, and M. Worring. The MediaMill TRECVID 2005 Semantic Video Search Engine. In *NIST TRECVID workshop*, Gaithersburg, MD, November 2005.

[73] C.G.M. Snoek, M. Worring, J.C. van Gemert, J.M. Geusebroek, and A.W.M. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. *ACM Multimedia*, pages 421–430, 2006.

[74] S H. Srinivasan and Neela Sawant. Finding near-duplicate images on the web using fingerprints. In *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, pages 881–884, New York, NY, USA, 2008. ACM.

[75] A. Torralba, R. Fergus, and W. T. Freeman. Tiny images. Technical Report MIT-CSAIL-TR-2007-024, Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, 2007.

[76] Deepak S. Turaga, Brian Foo, Olivier Verscheure, and Rong Yan. Configuring topologies of distributed semantic concept classifiers for continuous multimedia stream processing. In *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, pages 289–298, New York, NY, USA, 2008. ACM.

[77] Vladimir Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

[78] Xin-Jing Wang, Lei Zhang, Feng Jing, and Wei-Ying Ma. Annosearch: Image auto-annotation by search. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1483–1490, Washington, DC, USA, 2006. IEEE Computer Society.

[79] L. Xie, A. Natsev, and J. Tesic. Dynamic Multimodal Fusion in Video Search. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 1499–1502, 2007.

[80] R. Yan. *Probabilistic Models for Combining Diverse Knowledge Sources in Multimedia Retrieval.* PhD thesis, Carnegie Mellon University, 2006.

[81] R. Yan, A. Hauptmann, and R. Jin. Multimedia search with pseudo-relevance feedback. *Intl Conf on Image and Video Retrieval*, pages 238–247, 2003.

[82] R. Yan and A.G. Hauptmann. Probabilistic latent query analysis for combining multiple retrieval sources. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 324–331, 2006.

[83] Rong Yan, Jun Yang, and Alexander G. Hauptmann. Learning query-class dependent weights in automatic video retrieval. In *ACM Multimedia*, 2004.

[84] Akira Yanagawa, Shih-Fu Chang, Lyndon Kennedy, and Winston Hsu. Brief descriptions of visual features for baseline trecvid concept detectors. Technical report, Columbia University, July 2006.

[85] Akira Yanagawa, Winston Hsu, and Shih-Fu Chang. Anchor shot detection in trecvid-2005 broadcast news videos. Technical report, Columbia University, December 2005.

[86] Jun Yang, Rong Yan, and Alexander G. Hauptmann. Cross-domain video concept detection using adaptive svms. In *MULTIMEDIA '07: Proceedings of the 15th international conference on Multimedia*, pages 188–197, New York, NY, USA, 2007. ACM.

[87] E. Yom-Tov, S. Fine, D. Carmel, and A. Darlow. Metasearch and Federation using Query Difficulty Prediction. In *SIGIR 2005 Query Prediction Workshop*, July 2005.

[88] Elad Yom-Tov, Shai Fine, David Carmel, and Adam Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *SIGIR*, 2005.

[89] Dong-Qing Zhang and Shih-Fu Chang. Detecting image near-duplicate by stochastic attributed relational graph matching with learning. In *ACM Multimedia*, New York City, USA, October 2004.