Web-scale Multimedia Search for Internet Video Content

by

Lu Jiang

Ph.D Thesis Proposal

Thesis Committee:
Dr. Alex Hauptmann, Carnegie Mellon University
Dr. Teruko Mitamura, Carnegie Mellon University
Dr. Louis-Philippe Morency, Carnegie Mellon University
Dr. Tat-Seng Chua, National University of Singapore

Language Technologies Institute

October 2015
Abstract

The Internet has been witnessing an explosion of video content. According to a Cisco study, video content accounted for 64% of all the world’s internet traffic in 2014, and this percentage is estimated to reach 80% by 2019. Video data are becoming one of the most valuable sources to assess information and knowledge. However, existing video search solutions are still based on text matching (text-to-text search), and could fail for the huge volumes of videos that have little relevant metadata or no metadata at all. The need for large-scale and intelligent video search, which bridges semantic gap between the user’s information need and the video content, seems to be urgent.

In this thesis, we propose an accurate, efficient and scalable search method for video content. As opposed to text matching, the proposed method relies on automatic video content understanding, and allows for intelligent and flexible search paradigms over the video content (text-to-video and text&video-to-video search). It provides a new way to look at content-based video search from finding a simple concept like “puppy” to searching a complex incident like “a scene in urban area where people running away after an explosion”. To achieve this ambitious goal, we propose several novel methods focusing on accuracy, efficiency and scalability in the novel search paradigm. First, we introduce a novel self-paced curriculum learning theory that allows for training more accurate semantic concepts. Second, we propose a novel and scalable approach to index semantic concepts that can significantly improve the search efficiency with minimum accuracy loss. Third, we design a novel video reranking algorithm that can boost accuracy for video retrieval.

The extensive experiments demonstrate that the proposed methods are able to surpass state-of-the-art accuracy on multiple datasets. In addition, our method can efficiently scale up the search to hundreds of millions videos, and only takes about 0.2 second to search a semantic query on a collection of 100 million videos, 1 second to process a hybrid query over 1 million videos. Based on the proposed methods, we implement E-Lamp Lite, the first of its kind large-scale semantic search engine for Internet videos. According to National Institute of Standards and Technology (NIST), it achieved the best accuracy in the TRECVID Multimedia Event Detection (MED) 2013 and 2014, the most representative task for content-based video search. To the best of our knowledge, E-Lamp Lite is the first content-based semantic search system that is capable of indexing and searching a collection of 100 million videos.
Acknowledgements

The acknowledgements and the people to thank go here, don’t forget to include your project advisor.

This work was partially supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20068. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number OCI-1053575. It used the Blacklight system at the Pittsburgh Supercomputing Center (PSC).

Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.
# Contents

Abstract i
Acknowledgements ii

## 1 Introduction
1.1 Research Challenges and Solutions 3
1.2 Social Validity 5
1.3 Proposal Overview 6
1.4 Thesis Statement 8
1.5 Key Contributions of the Thesis 8

## 2 Related Work
2.1 Content-based Image Retrieval 10
2.2 Copy Detection 11
2.3 Semantic Concept / Action Detection 11
2.4 Multimedia Event Detection 12
2.5 Content-based Video Semantic Search 12
2.6 Comparison 12

## 3 Indexing Semantic Features
3.1 Introduction 13
3.2 Related Work 15
3.3 Method Overview 16
3.4 Concept Adjustment 17
3.4.1 Distributional Consistency 19
3.4.2 Logical Consistency 20
3.4.3 Discussions 22
3.5 Inverted Indexing & Search 24
3.5.1 Video Search 24
3.6 Experiments 26
3.6.1 Setups 26
3.6.2 Performance on MED 27
3.6.3 Comparison to State-of-the-art on MED 29
3.6.4 Comparison to Top-k Thresholding on MED 30
3.6.5 Accuracy of Concept Adjustment 30
3.6.6 Performance on YFCC100M 31
3.7 Summary .......................................................... 33

4 Semantic Search .................................................. 34
  4.1 Introduction ..................................................... 34
  4.2 Related Work ................................................... 34
  4.3 Semantic Search ................................................ 35
    4.3.1 Semantic Query Generation ............................... 35
    4.3.2 Retrieval Models ......................................... 38
  4.4 Experiments .................................................... 39
    4.4.1 Setups ..................................................... 39
    4.4.2 Semantic Matching in SQG ............................... 40
    4.4.3 Modality/Feature Contribution ........................... 42
    4.4.4 Comparison of Retrieval Methods ........................ 43
  4.5 Summary ........................................................ 44

5 Hybrid Search .................................................... 45
  5.1 Introduction ................................................... 45
  5.2 Related Work ................................................... 45
  5.3 Scalable Few Example Search ................................. 45
  5.4 Experiments .................................................... 45

6 Video Reranking ................................................ 46
  6.1 Introduction ................................................... 46
  6.2 Related Work ................................................... 49
  6.3 MMPRF .......................................................... 50
  6.4 SPaR ............................................................. 52
    6.4.1 Learning with Fixed Pseudo Labels and Weights .... 54
    6.4.2 Learning with Fixed Classification Parameters ..... 56
    6.4.3 Convergence and Relation to Other Reranking Models 59
  6.5 Experiments .................................................... 60
    6.5.1 Setups ..................................................... 60
    6.5.2 Comparison with Baseline methods ........................ 62
    6.5.3 Impact of Pseudo Label Accuracy ......................... 63
    6.5.4 Comparison of Weighting Schemes ........................ 64
    6.5.5 Experiments on Hybrid Search ........................... 66
    6.5.6 Experiments on Web Query Dataset ........................ 66
    6.5.7 Runtime Comparison .................................... 66
  6.6 Summary ........................................................ 67

7 Building Semantic Concepts by Self-paced Curriculum Learning 69
  7.1 Introduction ................................................... 69
  7.2 Related Work ................................................... 71
    7.2.1 Curriculum Learning ...................................... 71
    7.2.2 Self-paced Learning ....................................... 71
    7.2.3 Survey on Weakly Training ............................... 73
  7.3 Theory .......................................................... 73
    7.3.1 Model ...................................................... 73
    7.3.2 Relationship to CL and SPL ............................... 76
Chapter 1

Introduction

We are living in an era of big data: three hundred hours of video are uploaded to YouTube every minute; social media users are posting 12 millions videos on Twitter every day. According to a Cisco study, video content accounted for 64% of all the world’s internet traffic in 2014, and this percentage is estimated to reach 80% by 2019. The explosion of video data is creating impacts on many aspects of society. The big video data is important not because there is a lot of it but because increasingly it is becoming a valuable source for insights and information, e.g. telling us about things happening in the world, giving clues about a person’s preferences, pointing out places, people or events of interest, providing evidence about activities that have taken place [1].

An important approach of acquiring information and knowledge is through video retrieval. However, existing large-scale video retrieval methods are still based on text-to-text matching, in which the query words are matched against the textual metadata generated by the uploader [2]. The text-to-text search method, though simple, is of minimum functionality because it provides no understanding about the video content. As a result, the method proves to be futile in many scenarios, in which the metadata are either missing or less relevant to the visual video content. According to a recent study [3], 66% videos on a social media site called Twitter Vine are not associated with meaningful metadata (hashtag or a mention), which suggests on an average day, around 8 million videos may never be watched again just because there is no way to find them. The phenomenon is more severe for the even larger amount of videos that are captured by mobile phones, surveillance cameras and wearable devices that end up not having any metadata at all. Comparable to the days in the late 1990s, when people usually got lost in the rising sea of web pages, now they are overwhelmed by the vast amounts of videos, but lack powerful tools to discover, not to mention to analyze, meaningful information in the video content.
In this thesis, we seek the answer to a fundamental research question: how to satisfy information needs about video content at a very large scale. We embody this fundamental question into a concrete problem called Content-Based Video Semantic Retrieval (CBVSR), a category of content-based video retrieval problem focusing on semantic understanding about the video content, rather than on textual metadata nor on low-level statistical matching of color, edges, or the interest points in the content. A distinguishing characteristic about the CBVSR method is the capability to search and analyze videos based on semantic (and latent semantic) features that can be automatically extracted from the video content. The semantic features are human interpretable multimodal tags about the video content such as people (who were involved in the video), objects (what objects were seen), scenes (where did it take place), actions and activities (what happened), speech (what did they say), visible text (what characters were spotted).

The CBVSR method advances traditional video retrieval methods in many ways. It enables a more intelligent and flexible search paradigm that traditional metadata search would never achieve. A simple query in CBVSR may contain a single object about, say, “a puppy” or “a desk”, and a complex query may describe a complex activity or incident, e.g. “changing a vehicle tire”, “attempting bike tricks in the forest”, “a group of people protesting an education bill”, “a scene in urban area where people running away after an explosion”, and so forth. In this thesis, we consider the following two types of queries:

**Definition 1.1.** (Semantic Query and Hybrid Query) Queries only consisting of semantic features (e.g. people, objects, actions, speech, visible text, etc.) or a text description about semantic features are called **semantic queries**. Queries consisting of both semantic features and a few video examples are called **hybrid queries**. As video examples are usually provided by users on the fly, according to NIST [4], we assume there are at most 10 video examples in a hybrid query.

A user may formulate a semantic query in terms of a few semantic concept names or a natural language description of her information need (See Chapter 4). According to the definition, the semantic query provides an approach for text-to-video search, and the hybrid query offers a mean for text&video-to-video search. Semantic queries are important as, in a real-world scenario, users often start the search without any video example. A query consisting only of a few video examples is regarded as a special case of the hybrid query. Example 1.1 illustrates an example of formulating the queries for birthday party.

**Example 1.1.** Suppose our goal is to search the videos about birthday party. In the traditional text query, we have to search the keywords in the user-generated metadata, such as titles and descriptions, as shown in Fig. 1.1(a). For videos without any metadata,
there is no way to find them at all. In contrast, in a semantic query we might look for visual clues in the video content such as “cake”, “gift” and “kids”, audio clues like “birthday song” and “cheering sound”, or visible text like “happy birthday”. See Fig. 1.1(b). We may alternatively input a sentence like “videos about birthday party in which we can see cake, gift, and kids, and meanwhile hear birthday song and cheering sound.”

Semantic queries are flexible and can be further refined by Boolean operators. For example, to capture only the outdoor party, we may add “AND outdoor’ to the current query; to exclude the birthday parties for a baby, we may add “AND NOT baby”. Temporal relation can also be specified by a temporal operator. For example, suppose we are only interested in the videos in which the opening of presents are seen before consuming the birthday cake. In this case, we can add a temporal operator to specify the temporal occurrence of the two objects “gift” and “cake”.

After watching some of the retrieved videos for a semantic query, the user is likely to select a few interesting videos, and to find more relevant videos like these [5]. This can be achieved by issuing a hybrid query which adds the selected videos to the query. See Fig. 1.1(c). Users may also change the semantic features in the hybrid query to refine or emphasize certain aspects in the selected video examples. For example, we may add “AND birthday song” in the hybrid query to find more videos not only similar to the video examples but also have happy birthday songs in their content.

Figure 1.1: Comparison of text, semantic and hybrid query on “birthday party”.

1.1 Research Challenges and Solutions

The idea of CBVSR sounds appealing but, in fact, it is a very challenging problem. It introduces several novel issues that have not been sufficiently studied in the literature, such as the issue of searching complex query consisting of multimodal semantic features and video examples, the novel search paradigm entirely based on video content understanding, and efficiency issue for web-scale video retrieval. As far as this thesis is concerned, we confront the following research challenges:
1. **Challenges on accurate retrieval for complex queries.** A crucial challenge for any retrieval system is achieving a reasonable accuracy, especially for the top-ranked documents or videos. Unlike other problems, the data in this problem are real-world noisy and complex Internet videos, and the queries are of complex structures containing both texts and video examples. How to design intelligent algorithms to obtain state-of-the-art accuracy is a challenging issue.

2. **Challenges on efficient retrieval at very large scale.** Processing video proves to be a computationally expensive operation. The huge volumes of Internet video data brings up a key research challenge. How to design efficient algorithms that are able to search hundreds of millions of video within the maximum recommended waiting time for a user, i.e. 2 seconds [6], while maintaining maximum accuracy becomes a critical challenge.

3. **Challenges on interpretable results.** A distinguishing characteristic about CBVSR is that the retrieval is entirely based on semantic understanding about the video content. A user should have some understanding of why the relevant videos are selected, so that she can modify the query to better satisfy her information need. In order to produce accountable results, the model should be interpretable. However, how to build interpretable models for content-based video retrieval is still unclear in the literature.

Due to the recent advances in the fields of computer vision, machine learning, multimedia and information retrieval, it becomes increasingly interesting to consider addressing the above research challenges. In analogy to building a rocket spaceship, we are now equipped with powerful cloud computing infrastructures (structural frame) and big data (fuel). What is missing is a rocket engine that provides driving force and reaches the target. In our problem, the engine is essentially a collection of effective algorithms that can solve the above challenges. To this end, we propose the following novel methods:

1. To address the challenges on accuracy, we explore the following three aspects. In Chapter 4, we systematically study a number of query generation methods, which translate a user query to a system query that can be handled by the system, and retrieval algorithms to improve the accuracy for semantic query. In Chapter 6, we propose a cost-effective reranking algorithm called self-paced reranking. It optimizes a concise mathematical objective and provides notable improvement for both semantic and hybrid queries. In Chapter 7, we propose a theory of self-paced curriculum learning, and then apply it to training more accurate semantic concept detectors.

2. To address the challenges on efficiency and scalability, in Chapter 3 we propose a semantic concept adjustment and indexing algorithm that provides a foundation
for efficient search over 100 millions of videos. In Chapter 5, we propose a search algorithm for hybrid queries that can efficiently search a collection of 100 million videos, whereas without significant loss on accuracy.

3. To address the challenges on interpretability, we design algorithms to build interpretable models based on semantic (and latent semantic) features. In Chapter 4, we provide a semantic justification that can explain the reasoning of selecting relevant videos for the semantic query. In Chapter 5, we discuss an approach that can explain the reasoning behind the search for results retrieved by a hybrid query.

The above proposed methods are extensively verified on a number of large-scale challenging datasets. Experimental results demonstrate that the proposed method can exceed state-of-the-art accuracy across a number of datasets. Furthermore, it can efficiently scale up the search to hundreds of millions of Internet videos. It only takes about 0.2 second to search a semantic query on a collection of 100 million videos, and 1 second to handle a hybrid query over 1 million videos.

Based on the proposed methods, we implement E-Lamp Lite, the first of its kind large-scale semantic search engine for Internet videos. According to National Institute of Standards and Technology (NIST), it achieved the best accuracy in the TRECVID Multimedia Event Detection (MED) 2013 and 2014, one of the most representative and challenging tasks for content-based video search. To the best of our knowledge, E-Lamp Lite is also the first content-based video retrieval system that is capable of indexing and searching a collection of 100 million videos.

1.2 Social Validity

The problem studied in this thesis is fundamental. The proposed methods can potentially benefit a variety of related tasks such as video summarization [7], video recommendation, video hyperlinking [8], social media video stream analysis [9], in-video advertising [10], etc. A direct usage is augmenting existing metadata search paradigms for video. Our method provides a solution to control video pollution on the web [11], which results from introduction into the environment of (i) redundant, (ii) incorrect, noisy, imprecise, or manipulated, or (iii) undesired or unsolicited videos or meta-information (i.e., the contaminants). The pollution can cause harm or discomfort to the members of the social environment of a video sharing service, e.g. opportunistic users can pollute the system spreading video messages containing undesirable content (i.e., spam); users can also associate metadata with videos in attempt to fool text-to-text search methods to achieve high ranking positions in search results. The new search paradigms in the
proposed method can be used to identify such polluted videos so as to alleviate the pollution problem. Another application is about in-video advertising. Currently, it may be hard to place in-video advertisements as the user-generated metadata typically does not describe the video content, let alone concept occurrences in time. Our method provides a solution by formulating this information need as a semantic query and putting ads into the relevant videos [10]. For example, a sport shoe company may use the query “(running OR jumping) AND parkour AND urban scene” to find parkour videos in which the promotional shoe ads can be put.

Furthermore, our method provides a feasible solution of finding information in the videos without any metadata. Analyzing video content helps automatically understanding about what happened in the real life of a person, an organization or even a country. This functionality is crucial for a variety of applications. For example, finding videos in social streams that violate either legal or moral standards; analyzing videos captured by a wearable device, such as Google Glass, to assist the user’s cognitive process on a complex task [12]; searching specific events captured by surveillance cameras or even devices that record other of types of signals.

Finally, the theory and insights in the proposed methods may inspire the development of more advanced methods. For example, the insight in our web-scale method may guide the design of the future search or analysis systems for video big data [13]. The proposed reranking method can be also used to improve the accuracy of image retrieval [14]. The self-paced curriculums learning theory may inspire other machine learning methods on other problems, such as matrix factorization [15].

1.3 Proposal Overview

In this thesis, we model a CBVSR problem as a retrieval problem, in which given a query that complies with Definition 1.1, we are interested in finding a ranked list of relevant videos based on the semantic understanding about the video content. To solve this problem, we incorporate a two-stage framework as illustrated in Fig. 1.2.

The offline stage is called semantic indexing, which aims at extracting semantic features in the video content and indexing them for efficient online search. It usually involves the following steps: a video clip is first represented by the low-level features that capture the local appearance, texture or acoustic statistics in the video content, represented by a collection of local descriptors such as interest points or trajectories. State-of-the-art low-level features include dense trajectories [16] and convolutional Deep Neural Network (DNN) features [17] for visual modality, and Mel-frequency cepstral coefficients
Introduction

(MFCCs) [18] and DNN features for audio modality [19, 20]. The low-level features are then input into the off-the-shelf detectors to extract the semantic features\(^1\). The semantic features, also known as high-level features, are human interpretable tags, each dimension of which corresponds to a confidence score of detecting a concept or a word in the video [21]. The visual/audio concepts, Automatic Speech Recognition (ASR) [19, 20] and Optical Character Recognition (OCR) are four types of semantic features considered in this thesis. After extraction, the high-level features will be adjusted and indexed for the efficient online search. The offline stage can be trivially paralleled by distributing the videos over multiple cores\(^2\).

The second stage is an online stage called video search. We employ two modules to process the semantic query and the hybrid query. Both modules consist of a query generation and a multimodal search step. A user can express a query in the form of a text description and a few video examples. The query generation for semantic query is to map the out-of-vocabulary concepts in the user query to their most relevant alternatives in the system vocabulary. For the hybrid query, the query generation also involves training a classification model using the video examples. The multimodal search component aims at retrieving a ranked list using the multimodal features. This step is a retrieval process for the semantic query and a classification process for the hybrid query. Afterwards, we can refine the results by reranking the videos in the initial ranked list. This process is known as reranking or Pseudo-Relevance Feedback (PRF) [24]. The basic idea is to first select a few videos and assign assumed labels to them. The samples with assumed labels are then used to build a reranking model using semantic and low-level features to improve the initial ranked list.

\(^1\)Here we assume we are given the off-the-shelf detectors. Chapter 7 will introduce approaches to build the detectors.

\(^2\)In this thesis, we do not discuss the offline video crawling process. This problem can be solved by vertical search engines crawling techniques [22, 23]
The quantity (relevance) and quality of the semantic concepts are two factors in affecting performance. The relevance is measured by the coverage of the concept vocabulary to the query, and thus is query-dependent. For convenience, we name it quantity as a larger vocabulary tends to increase the coverage. Quality determines the accuracy of the detector. To increase both the criteria, We propose a novel self-paced curriculum learning theory that allows for training more accurate semantic concepts over noisy datasets. The theory is inspired by the learning process of humans and animals that gradually proceeds from easy to more complex samples in training.

The reminder of this thesis will discuss the above topics in more details. In Chapter 2, We first briefly review related problems on video retrieval. In Chapter 3, we propose a scalable semantic indexing and adjustment method for semantic feature indexing. We then discuss the query generation and the multimodal search for semantic queries and hybrid queries in Chapter 4 and Chapter 5, respectively. The reranking method will be presented in 6. Finally we will introduce the method for training robust semantic concepts in Chapter 5. The conclusions and future work will be presented in the last chapter.

1.4 Thesis Statement

In this thesis, we approach a fundamental problem of acquiring semantic information in video content at a very large scale. We address the problem by proposing an accurate, efficient, and scalable method that can search the content of a billion of videos by semantic concepts, speech, visible texts, video examples, or any combination of these elements.

1.5 Key Contributions of the Thesis

To summarize, the contributions of the thesis are as follows:

1. The first-of-its-kind framework for web-scale content-based search over hundreds of millions of Internet videos [ICMR’15]. The proposed framework supports text-to-video, video-to-video, and text&video-to-video search [MM’12].

3. A novel reranking algorithm that is cost-effective in improving performance. It has a concise mathematical objective to optimize and useful properties that can be theoretically verified [MM’14, ICMR’14].

4. A consistent and scalable concept adjustment method representing a video by a few salient and consistent concepts that can be efficiently indexed by the modified inverted index [MM’15].

5. (Proposed Work) a novel efficient search method for the hybrid query.

Based on the above contributions, we implement E-Lamp Lite, the first of its kind large-scale semantic search engine for Internet videos. To the best of our knowledge, E-Lamp Lite is also the first content-based video retrieval system that is capable of indexing and searching a collection of 100 million videos.
Chapter 2

Related Work

Traditional content-based video retrieval methods have successfully demonstrated promising results in many real-world applications. Existing methods in related problems greatly enlightens our approach. In this chapter, we briefly review some related problems. Our goal is to analyze their similarity and difference to the proposed CBVSR.

2.1 Content-based Image Retrieval

Given a query image, a content-based image retrieval method is to find identical or visually similar images in a large image collection. Similar images are images about the same object despite possibly changes in image scale, viewpoint, lighting and partial occlusion. The method is a type of query-by-example search, where the query is usually represented by a single image. Generally, the solution is to first extract the low-level descriptors within a image such as SIFT \[25\] or GIST \[26\], encode them into a numerical vector by, for example, bag-of-visual-words \[27\] or fisher vector \[28\], and finally index the feature vectors for efficient online search using min-hashing or LSH \[29\]. The content-based image retrieval method can be extended to search the key frames in a video clip. But we still regard it as a special case of image retrieval. Sivic et al. introduced a video frame retrieval system called Video Google \[30\]. The system can be used to retrieve similar video key frames for a query image. Another application is to search the key frames about a specific instance such as an image about a person, a logo or a landmark. In some cases, users can select a region of interest in an image, and use it as a query image \[31\].

The content-based image retrieval method only utilizes the low-level descriptors that carry little semantic meaning. It is able to retrieve an instance of object as a result of
local descriptors matching, without realizing what is the object. Therefore, it is good at finding visually similar but not necessarily semantically similar images. Content-based image retrieval is a well-studied problem. There have been some commercial image retrieval system available such as Google image search. State-of-the-art image retrieval systems can efficiently handle more than 100 million images [32].

[todo: finish the section] [todo: cite more related work]

### 2.2 Copy Detection

The goal of video copy detection is to detect a segment of video derived from another video, usually by means of various transformations such as addition, deletion, modification (of aspect, color, contrast or encoding) camcording, etc [4]. The query in this method is a video segment called copy. This problem is sometimes also known as near duplicate video detection. The method relies on low-level visual and acoustic features without semantic understanding about the content. This problem is easier than the content-based image retrieval problem as the query and the relevant videos are essentially the same video with insignificant changes. It is a well-solved problem. State-of-the-art methods can handle web-scale videos with very high accuracy.

[todo: finish the section] [todo: cite more related work]

### 2.3 Semantic Concept / Action Detection

The goal of which is to search the occurrence of a single concept. A concept can be regarded as a visual or acoustic semantic tag on people, objects, scenes, actions, etc. in the video content [33]. The difficulty here is training robust and accurate detector. Though the output is high-level features, in the indexing method there are all based on low-level features.

Semantic search relies on understanding about the video content.

This line of study first emerged in a TRECVID task called Semantic Indexing [34].

A pair of concepts [35]

Papers in semantic concept detection in news video [36].

Action recognition papers [37–41].

[todo: finish the section] [todo: cite more related work]
Table 2.1: Comparison of video retrieval problem.

<table>
<thead>
<tr>
<th>Property</th>
<th>CBIR</th>
<th>Copy Detection</th>
<th>Semantic Indexing</th>
<th>MED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query</td>
<td>An image</td>
<td>A video segment</td>
<td>A concept name</td>
<td>A sentence and/or a few example videos</td>
</tr>
<tr>
<td>Retrieved Results</td>
<td>An image</td>
<td>A video segment</td>
<td>A concept name</td>
<td>A sentence and/or a few example videos</td>
</tr>
</tbody>
</table>

### 2.4 Multimedia Event Detection

with the advance in semantic concept detection, people started to focus on searching more complex queries called events. An event is more complex than a concept as it usually involves people engaged in process-driven actions with other people and/or objects at a specific place and time [21]. For example, the event “rock climbing” involves a climber, mountain scenes, and the action climbing. The relevant videos may include videos about outdoor bouldering, indoor artificial wall climbing or snow mountain climbing. A benchmark task on this topic is called TRECVID Multimedia Event Detection (MED) [18, 42]. Its goal is to provide a video-to-video search scheme. MED is a challenging problem, and the biggest collection in TRECVID only contains around 200 thousand videos.

### 2.5 Content-based Video Semantic Search

The CBVSR problem is similar to MED but advances it in the following ways. First, the queries can be simple complex queries consisting of both text description of semantic features and video examples. Second, the search is solely based on semantic understanding about the content rather than low-level features matching. Finally, the problem scale is orders-of-magnitude larger than that of MED.

Multimodal search related papers [43]. [todo: finish the section] [todo: cite more related work]

### 2.6 Comparison

[todo: finish the section] [todo: cite more related work]

what kind of questions can the method answers? What is the input query? What is scalability in 2 second? multimodal or single modality?
Chapter 3

Indexing Semantic Features

3.1 Introduction

Semantic indexing aims at extracting semantic features in the video content and indexing them for efficient online search. In this chapter, we introduce the method for extracting and indexing semantic features from the video content, focusing on adjusting and indexing semantic concepts.

We consider indexing four types of semantic features in this thesis: visual concepts, audio concepts ASR and OCR. ASR provides acoustic information about videos. It especially benefits finding clues in close-to-camera and narrative videos such as “town hall meeting” and “asking for directions”. OCR captures the text characters in videos with low recall but high precision. The recognized characters are often not meaningful words but sometimes can be a clue for fine-grained detection, e.g. distinguishing videos about “baby shower” and “wedding shower”. ASR and OCR are text features, and thus can be conveniently indexed by the standard inverted index. The automatically detected text words in ASR and OCR in a video, after some preprocessing, can be treated as text words in a document. The preprocessing includes creating a stop word list for ASR from the English stop word list. The stop word lists for ASR includes utterances like “uh”, “you know”, etc. For OCR, due to the noise in word detection, we need to remove the words that do not exist in the English vocabulary.

How to index semantic concepts is an open question. Existing methods index a video by the raw concept detection score that is dense and inconsistent [8, 14, 44–48]. This solution is mainly designed for analysis and search over a few thousand of videos, and cannot scale to big data collections required for real world applications. Even though a modern text retrieval system can already index and search over billions of text documents, the
task is still very challenging for semantic video search. The main reason is that semantic concepts are quite different from the text words, and semantic concept indexing is still an understudied problem. Specifically, concepts are automatically extracted by detectors with limited accuracy. The raw detection score associated with each concept is inappropriate for indexing for two reasons. First, the distribution of the scores is dense, i.e. a video contains every concept with a non-zero detection score, which is analogous to a text document containing every word in the English vocabulary. The dense score distribution hinders effective inverted indexing and search. Second, the raw score may not capture the complex relations between concepts, e.g. a video may have a “puppy” but not a “dog”. This type of inconsistency can lead to inaccurate search results.

To address this problem, we propose a novel step called concept adjustment that aims at producing video (and video shot) representations that tend to be consistent with the underlying concept representation. After adjustment, a video is represented by a few salient and consistent concepts that can be efficiently indexed by the inverted index. In theory, the proposed adjustment model is a general optimization framework that incorporates existing techniques as special cases. In practice, as demonstrated in our experiments, the adjustment increases the consistency with the ground-truth concept representation on the real world TRECVID dataset. Unlike text words, semantic concepts are associated with scores that indicate how confidently they are detected. We propose an extended inverted index structure that incorporates the real-valued detection scores and supports complex queries with Boolean and temporal operators.

Compared to existing methods, the proposed method exhibits the following three benefits. First, it advances the text retrieval method for video retrieval. Therefore, while existing methods fail as the size of the data grows, our method is scalable, extending the current capability of semantic search by a few orders of magnitude while maintaining state-of-the-art performance. Our experiments validate this argument. Second, we propose a novel component called concept adjustment in a common optimization framework with solid probabilistic interpretations. Finally, our empirical studies shed some light on the tradeoff between efficiency and accuracy in a large-scale video search system. These observations will be helpful in guiding the design of future systems on related tasks.

The experimental results are promising on three datasets. On the TRECVID Multimedia Event Detection (MED), our method achieves comparable performance to state-of-the-art systems, while reducing its index by a relative 97%. The results on the TRECVID Semantic Indexing dataset demonstrate that the proposed adjustment model is able to generate more accurate concept representation than baseline methods. The results on the largest public multimedia dataset called YCCC100M [49] show that the method is capable of indexing and searching over a large-scale video collection of 100 million
Internet videos. It only takes 0.2 seconds on a single CPU core to search a collection of 100 million Internet videos. Notably, the proposed method with reranking is able to achieve by far the best result on the TRECVID MED 0Ex task, one of the most representative and challenging tasks for semantic search in video.

### 3.2 Related Work

With the advance in object and action detection, people started to focus on searching more complex queries called events. An event is more complex than a concept as it usually involves people engaged in process-driven actions with other people and/or objects at a specific place and time [21]. For example, the event “rock climbing” involves video clips such as outdoor bouldering, indoor artificial wall climbing or snow mountain climbing. A benchmark task on this topic is called TRECVID Multimedia Event Detection (MED). Its goal is to detect the occurrence of a main event occurring in a video clip without any user-generated metadata. MED is divided into two scenarios in terms of whether example videos are provided. When example videos are given, a state-of-the-art system first train classifiers using multiple features and fuse the decision of the individual classification results [50–58].

This thesis focuses on the other scenario named zero-example search (0Ex) where no example videos are given. 0Ex mostly resembles a real world scenario, in which users start the search without any example. As opposed to training an event detector, 0Ex searches semantic concepts that are expected to occur in the relevant videos, e.g. we might look for concepts like “car”, “bicycle”, “hand” and “tire” for the event “changing a vehicle tire”. A few studies have been proposed on this topic [14, 44–48]. A closely related work is detailed in [59], where the authors presented their lessons and observations in building a state-of-the-art semantic search engine for Internet videos. Existing solutions are promising but only for a few thousand videos because they cannot scale to big data collections. Therefore, the biggest collection in existing studies contains no more than 200 thousand videos [4, 59].

Deng et al. [60] recently introduced label relation graphs called Hierarchy and Exclusion (HEX) graphs. The idea is to infer a representation that maximizes the likelihood and do not violate the label relation defined in the HEX graph.
3.6 Experiments

3.6.1 Setups

Dataset and evaluation: The experiments are conducted on two TRECVID benchmarks called Multimedia Event Detection (MED): MED13Test and MED14Test [4]. The performance is evaluated by several metrics for a better understanding, which include: P@20, Mean Reciprocal Rank (MRR), Mean Average Precision (MAP), and MAP@20, where the MAP is the official metric used by NIST. Each set includes 20 events over 25,000 test videos. The official NIST’s test split is used. We also evaluate each experiment on 10 randomly generated splits to reduce the split partition bias. All experiments are conducted without using any example or text metadata.

Features and queries: Videos are indexed by semantic features including semantic visual concepts, ASR, and OCR. For semantic concepts, 1,000 ImageNet concepts are trained by the deep convolution neural networks [61]. The remaining 3,000+ concepts are directly trained on videos by the self-paced learning pipeline [70, 71] on around 2 million videos using improved dense trajectories [16]. The video datasets include Sports [72], Yahoo Flickr Creative Common (YFCC100M) [49], Internet Archive Creative Common (IACC) [4] and Do It Yourself (DIY) [73]. The details of these datasets can be found in Table 3.1. The ASR module is built on EESEN and Kaldi [19, 20, 74]. OCR is extracted by a commercial toolkit. Three sets of queries are used: 1) Expert queries are obtained by human experts; 2) Auto queries are automatically generated by the Semantic Query Generation (SQG) methods in [59] using ASR, OCR and visual concepts; 3) AutoVisual queries are also automatically generated but only includes the visual concepts. The Expert queries are used by default.

Configurations: The concept relation released by NIST is used to build the HEX graph for IACC features [33]. The adjustment is conducted at the video-level average (\( p = 1 \) in Eq. (3.1)) so no shot-level exclusion relations are used. For other concept features, since there is no public concept relation specification, we manually create the HEX graph. The HEX graphs are empty for Sports and ImageNet features as there is no evident hierarchical and exclusion relation in their concepts. We cluster the concepts based on the correlation of their training labels, and include concepts that frequently co-occur together into a group. The parameters are tuned on a validation sets, and then are fixed across all experiment datasets including MED13Test, MED14Test and YFCC100M. Specifically, the default parameters in Eq. (3.1) are \( p = 1 \), \( \alpha = 0.95 \). \( \beta \) is set as the top \( k \) detection scores in a video, and is different for each type of features: 60

\[^{1}\text{http://www-nlpir.nist.gov/projects/tv2012/tv11.sin.relations.txt}\]
TABLE 3.1: Summary of the semantic concept training sets. ImageNet features are trained on still images, and the rest are trained on videos.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Samples</th>
<th>#Classes</th>
<th>Category</th>
<th>Example Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIY</td>
<td>72,000</td>
<td>1,601</td>
<td>Instructional videos</td>
<td>Yoga, Juggling, Cooking</td>
</tr>
<tr>
<td>IACC</td>
<td>600,000</td>
<td>346</td>
<td>Internet archive videos</td>
<td>Baby, Outdoor, Sitting down</td>
</tr>
<tr>
<td>YFCC100M</td>
<td>800,000</td>
<td>609</td>
<td>Amateur videos on Flickr</td>
<td>Beach, Snow, Dancing</td>
</tr>
<tr>
<td>ImageNet</td>
<td>1,000,000</td>
<td>609</td>
<td>Still images</td>
<td>Bee, Corkscrew, Cloak</td>
</tr>
<tr>
<td>Sports</td>
<td>1,100,000</td>
<td>487</td>
<td>Sports videos on YouTube</td>
<td>Bullfighting, Cycling, Skiing</td>
</tr>
</tbody>
</table>

CVX optimization toolbox [62] is used to solve the model in Eq. (3.1). Eq. (3.10) is used as the retrieval model for concept features, where $k_1 = 1.2$ and $b = 0.75$.

### 3.6.2 Performance on MED

We first examine the overall performance of the proposed method. Table 3.2 lists the evaluation metrics over the two benchmarks on the standard NIST split and on the 10 randomly generated splits. The performance is reported over three set of queries: Expert, Auto, and AutoVisual.

Table 3.3 compares the performance of the raw and the adjusted representation on the 10 splits of MED13Test. Raw lists the performance of indexing the raw score by dense matrices; Adjusted lists the performance of indexing the adjusted concepts by the proposed index which preserves the real-valued scores. As we see, although Raw is slightly better than Adjusted, its index in the form of dense matrices is more than 33 times bigger than the inverted index in Adjusted. The comparison substantiates that the adjusted representation has comparable performances with the raw representation but can be indexed by a much smaller index.

An interesting observation is that Adjusted outperforms Raw on 8 out of 20 events on MED13Test (see Table E.1). We inspected the results and found that concept adjustment can generate more consistent representations. Fig. 3.3 illustrates raw and adjusted concepts on three example videos. Since the raw score is dense, we only list the top ranked concepts. As we see, the noisy concept in the raw detection may be removed by the logical consistency, e.g. “snow” in the first video. The missed concept may be recalled by logical consistencies, e.g. “vehicle” in the third video is recalled by “ground vehicle”. The frequently co-occurring concepts may also be recovered by distributional consistencies, e.g. “cloud” and “sky” in the second video. Besides, we also found that Boolean queries can boost the performance. For example, in “E029: Winning a race without a vehicle”, the query of relevant concepts such as swimming, racing or marathon can achieve an AP of 12.5. However, the Boolean query also containing “AND NOT” concepts such as car racing or horse riding can achieve an AP of 24.5.
Table 3.2: Overview of the system performance.

(a) Performance on the NIST’s split

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Query</th>
<th>Evaluation Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>P@20</td>
</tr>
<tr>
<td>MED13Test</td>
<td>Expert</td>
<td>0.355</td>
</tr>
<tr>
<td></td>
<td>Auto</td>
<td>0.243</td>
</tr>
<tr>
<td></td>
<td>AutoVisual</td>
<td>0.125</td>
</tr>
<tr>
<td>MED14Test</td>
<td>Expert</td>
<td>0.228</td>
</tr>
<tr>
<td></td>
<td>Auto</td>
<td>0.150</td>
</tr>
<tr>
<td></td>
<td>AutoVisual</td>
<td>0.120</td>
</tr>
</tbody>
</table>

(b) Average Performance on the 10 splits

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Query</th>
<th>Evaluation Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>P@20</td>
</tr>
<tr>
<td>MED13Test</td>
<td>Expert</td>
<td>0.325</td>
</tr>
<tr>
<td></td>
<td>Auto</td>
<td>0.253</td>
</tr>
<tr>
<td></td>
<td>AutoVisual</td>
<td>0.126</td>
</tr>
<tr>
<td>MED14Test</td>
<td>Expert</td>
<td>0.219</td>
</tr>
<tr>
<td></td>
<td>Auto</td>
<td>0.148</td>
</tr>
<tr>
<td></td>
<td>AutoVisual</td>
<td>0.117</td>
</tr>
</tbody>
</table>

Table 3.3: Comparison of the raw and the adjusted representation on the 10 splits.

<table>
<thead>
<tr>
<th>Method</th>
<th>Index</th>
<th>Evaluation Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P@20</td>
<td>MRR</td>
</tr>
<tr>
<td>MED13 Raw</td>
<td>385M</td>
<td>0.312</td>
</tr>
<tr>
<td>MED13 Adjusted</td>
<td>11.6M</td>
<td>0.325</td>
</tr>
<tr>
<td>MED14 Raw</td>
<td>357M</td>
<td>0.233</td>
</tr>
<tr>
<td>MED14 Adjusted</td>
<td>12M</td>
<td>0.219</td>
</tr>
</tbody>
</table>

Figure 3.3: Comparison of raw and adjusted concepts.

The parameters $\alpha$ and $\beta$ in Eq. (3.1) control the magnitude of sparsity in the concept adjustment, i.e. the percentage of concepts with nonzero scores in a video representation. A sparse representation reduces the size of indexes but hurts the performance at the same time. As we will see later, $\beta$ is more important than $\alpha$ in affecting the performance. Therefore, we fix $\alpha$ to 0.95 and study the impact of $\beta$. Fig. 3.4 illustrates the tradeoff between accuracy and efficiency on the 10 splits of MED13Test. By tuning $\beta$, we obtain
different percentages of nonzero concepts in a video representation. The $x$-axis lists the percentage in the log scale. $x = 0$ indicates the performance of ASR and OCR without semantic concept features. We discovered that we do not need many concepts to index a video, and a few adjusted concepts already preserve significant amount of information for search. As we see, the best tradeoff in this problem is 4% of the total concepts (i.e. 163 concepts). Further increasing the number of concepts only leads to marginal performance gain.

![Figure 3.4: The impact of parameter $\beta$. $x = 0$ indicates the performance of ASR and OCR without semantic concepts.](image)

3.6.3 Comparison to State-of-the-art on MED

We then compare our best result with the published results on MED13Test. The experiments are all conducted on the NIST’s split, and thus are comparable to each other. As we see in Table 3.4, the proposed method has a comparable performance to the state-of-the-art methods. Notably, the proposed method with one iteration of reranking [14] is able to achieve the best result. The comparison substantiates that our method maintains state-of-the-art accuracy. It is worth emphasizing that the baseline methods may not scale to big data sets, as the dense matrices are used to index all raw detection scores [14, 47, 59].

<table>
<thead>
<tr>
<th>Method</th>
<th>Year</th>
<th>MAP ($\times$ 100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Composite Concepts [45]</td>
<td>2014</td>
<td>6.4</td>
</tr>
<tr>
<td>Tag Propagation [46]</td>
<td>2014</td>
<td>9.6</td>
</tr>
<tr>
<td>MMPRF [44]</td>
<td>2014</td>
<td>10.1</td>
</tr>
<tr>
<td>Clauses [48]</td>
<td>2014</td>
<td>11.2</td>
</tr>
<tr>
<td>Multi-modal Fusion [47]</td>
<td>2014</td>
<td>12.6</td>
</tr>
<tr>
<td>SPARe [14]</td>
<td>2014</td>
<td>12.9</td>
</tr>
<tr>
<td>E-Lamp FullSys [59]</td>
<td>2015</td>
<td>20.7</td>
</tr>
<tr>
<td><strong>Our System</strong></td>
<td>2015</td>
<td>18.3</td>
</tr>
<tr>
<td><strong>Our System + reranking</strong></td>
<td>2015</td>
<td><strong>20.8</strong></td>
</tr>
</tbody>
</table>
3.6.4 Comparison to Top-k Thresholding on MED

We compare our full adjustment model with its special case top-

k thresholding on MED14Test. Theorem 3.4 indicates that the top-

k thresholding results are optimal solutions of our model in special cases. The experiments are conducted using IACC SIN346 concept features that have large HEX graphs. We select the features because large HEX graphs help compare the difference between the two methods. Table 3.5 lists the average performance across 20 queries. We set the parameter \(k\) (equivalently \(\beta\)) to be 50, and 60. As the experiment only uses 346 concepts, the results are worse than our full systems using 3000+ concepts.

**Table 3.5:** Comparison of the full adjustment model with its special case Top-

k Thresholding on the 10 splits of MED14Test.

<table>
<thead>
<tr>
<th>Method</th>
<th>(k)</th>
<th>Evaluation Metric</th>
<th>(P@20)</th>
<th>MRR</th>
<th>MAP@20</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Model</td>
<td>50</td>
<td>P@20</td>
<td>0.0392</td>
<td>0.137</td>
<td>0.0151</td>
<td>0.0225</td>
</tr>
<tr>
<td>Top-k</td>
<td>50</td>
<td></td>
<td>0.0342</td>
<td>0.0986</td>
<td>0.0117</td>
<td>0.0218</td>
</tr>
<tr>
<td>Our Model</td>
<td>60</td>
<td>P@20</td>
<td>0.0388</td>
<td>0.132</td>
<td>0.0158</td>
<td>0.0239</td>
</tr>
<tr>
<td>Top-k</td>
<td>60</td>
<td></td>
<td>0.0310</td>
<td>0.103</td>
<td>0.0113</td>
<td>0.0220</td>
</tr>
</tbody>
</table>

As we see, the full adjustment model improves the accuracy and outperforms Top-

k thresholding in terms of \(P@20\), MRR and MAP@20. We inspected the results and found that the full adjustment model can generate more consistent representations (See Fig. 3.3). The results suggest that the full model outperforms the special model in this problem.

3.6.5 Accuracy of Concept Adjustment

Generally the comparison in terms of retrieval performance depends on the query words. A query-independent way to verify the accuracy of the adjusted concept representation is by comparing it to the ground truth representation. To this end, we conduct experiments on the TRECVID Semantic Indexing (SIN) IACC set, where the manually labeled concepts are available for each shot in a video. We use our detectors to extract the raw shot-level detection score, and then apply the adjustment methods to obtain the adjusted representation. The performance is evaluated by Root Mean Squared Error (RMSE) to the ground truth concepts for the 1,500 test shots in 961 videos.

We compare our adjustment method with the baseline methods in Table 3.6, where HEX Graph indicates the logical consistent representation [60] on the raw detection scores (i.e. \(\beta = 0\)), and Group Lasso denotes the representation yield by Eq. (3.1) when \(\alpha = 0\). We tune the parameter in each baseline method and report its best performance. As the ground truth label is binary, we let the adjusted scores be binary in all methods. As we
see, the proposed method outperforms all baseline methods. We hypothesize the reason is that our method is the only one that combines the distributional consistency and the logical consistency.

We study the parameter sensitivity in the proposed model. Fig. 3.5 plots the RMSE under different parameter settings. Physically, $\alpha$ interpolates the group-wise and within-group sparsity, and $\beta$ determines the number of concepts in a video. As we see, the parameter $\beta$ is more sensitive than $\alpha$, and accordingly we fix the value of $\alpha$ in practice. Note the parameter $\beta$ is also an important parameter in the baseline methods including thresholding and top-$k$ thresholding.

<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Score</td>
<td>7.671</td>
</tr>
<tr>
<td>HEX Graph Only</td>
<td>8.090</td>
</tr>
<tr>
<td>Thresholding</td>
<td>1.349</td>
</tr>
<tr>
<td>Top-$k$ Thresholding</td>
<td>1.624</td>
</tr>
<tr>
<td>Group Lasso</td>
<td>1.570</td>
</tr>
<tr>
<td><strong>Our method</strong></td>
<td><strong>1.236</strong></td>
</tr>
</tbody>
</table>

Table 3.6: Comparison of the adjusted representation and baseline methods on the TRECVID SIN set. The metric is Root Mean Squared Error (RMSE).

![Figure 3.5: Sensitivity study on the parameter $\alpha$ and $\beta$ in our model.](image)

### 3.6.6 Performance on YFCC100M

We apply the proposed method on YFCC100M, the largest public multimedia collection that has ever been released [49]. It contains about 0.8 million Internet videos (approximately 12 million key shots) on Flickr. For each video and video shot, we extract the improved dense trajectory, and detect 3,000+ concepts by the off-the-shelf detectors in Table 3.1. We implement our inverted index based on Lucene [76], and a similar configuration described in Section 3.6.1 is used except we set $b = 0$ in the
BM25 model. All experiments are conducted without using any example or text metadata. It is worth emphasizing that as the dataset is very big. The offline video indexing process costs considerable amount of computational resources in Pittsburgh supercomputing center. To this end, we share this valuable benchmark with our community [http://www.cs.cmu.edu/~lujiang/0Ex/mm15.html](http://www.cs.cmu.edu/~lujiang/0Ex/mm15.html).

To validate the efficiency and scalability, we duplicate the original videos and video shots, and create an artificial set of 100 million videos. We compare the search performance of the proposed method to a common approach in existing studies that indexes the video by dense matrices [47, 59]. The experiments are conducted on a single core of Intel Xeon 2.53GHz CPU with 64GB memory. The performance is evaluated in terms of the memory consumption and the online search efficiency. Fig. 3.6(a) compares the in-memory index as the data size grows, where the x-axis denotes the number of videos in the log scale, and the y-axis measures the index in GB. As we see, the baseline method fails when the data reaches 5 million due to lack of memory. In contrast, our method is scalable and only needs 550MB memory to search 100 million videos. The size of the total inverted index on disk is only 20GB. Fig. 3.6(b) compares the online search speed. We create 5 queries, run each query 100 times, and report the mean runtime in milliseconds. A similar pattern can be observed in Fig. 3.6 that our method is much more efficient than the baseline method and only costs 191ms to process a query on a single core. The above results verify scalability and efficiency of the proposed method.

![Figure 3.6:](image)

As a demonstration, we use our system to find relevant videos for commercials. The search is on 800 thousand Internet videos. We download 30 commercials from the Internet, and manually create 30 semantic queries only using semantic visual concepts. See detailed results in Table E.3. The ads can be organized in 5 categories. As we see, the performance is much higher than the performance on the MED dataset in Table 3.2. The improvement is a result of the increased data volumes. Fig. 3.7 plots the top 5 retrieved videos are semantically relevant to the products in the ads. The results suggest that our method may be useful in enhancing the relevance of in-video ads.
Table 3.7: Average performance for 30 commercials on the YFCC100M set.

<table>
<thead>
<tr>
<th>Category</th>
<th>#Ads</th>
<th>Evaluation Metric</th>
<th>P@20</th>
<th>MRR</th>
<th>MAP@20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sports</td>
<td>7</td>
<td></td>
<td>0.88</td>
<td>1.00</td>
<td>0.94</td>
</tr>
<tr>
<td>Auto</td>
<td>2</td>
<td></td>
<td>0.85</td>
<td>1.00</td>
<td>0.95</td>
</tr>
<tr>
<td>Grocery</td>
<td>8</td>
<td></td>
<td>0.84</td>
<td>0.93</td>
<td>0.88</td>
</tr>
<tr>
<td>Traveling</td>
<td>3</td>
<td></td>
<td>0.96</td>
<td>1.00</td>
<td>0.96</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>10</td>
<td></td>
<td>0.65</td>
<td>0.85</td>
<td>0.74</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>30</strong></td>
<td><strong>P@20</strong></td>
<td><strong>0.81</strong></td>
<td><strong>0.93</strong></td>
<td><strong>0.86</strong></td>
</tr>
</tbody>
</table>

Figure 3.7: Top 5 retrieved results for 3 example ads on the YFCC100M dataset.

3.7 Summary

This chapter proposed a scalable solution for large-scale semantic search in video. The proposed method extends the current capability of semantic video search by a few orders of magnitude while maintaining state-of-the-art retrieval performance. A key in our solution is a novel step called concept adjustment that aims at representing a video by a few salient and consistent concepts which can be efficiently indexed by the modified inverted index. We introduced a novel adjustment model that is based on a concise optimization framework with solid interpretations. We also discussed a solution that leverages the text-based inverted index for video retrieval. Experimental results validated the efficacy and the efficiency of the proposed method on several datasets. Specifically, the experimental results on the challenging TRECVID MED benchmarks validate the proposed method is of state-of-the-art accuracy. The results on the largest multimedia set YFCC100M set verify the scalability and efficiency over a large collection of 100 million Internet videos.
Chapter 4

Semantic Search

4.1 Introduction

In this chapter, we study the multimodal search process for semantic queries. The process is called semantic search, which is also known as zero-example search [4] or 0Ex for short, as zero examples are provided in the query. Searching by semantic queries is more consistent with human’s understanding and reasoning about the task, where an relevant video is characterized by the presence/absence of certain concepts rather than local points/trajectories in the example videos.

We will focus on two subproblems, namely semantic query generation and multimodal search. The semantic query generation is to map the out-of-vocabulary concepts in the user query to their most relevant alternatives in the system vocabulary. The multimodal search component aims at retrieving a ranked list using the multimodal features. We empirically study the methods in the subproblems and share our observations and lessons in building such a state-of-the-art system. The lessons are valuable because of not only the effort in designing and conducting numerous experiments but also the considerable computational resource to make the experiments possible. We believe the shared lessons may significantly save the time and computational cycles for others who are interested in this problem.

4.2 Related Work

A representative content-based retrieval task, initiated by the TRECVID community, is called Multimedia Event Detection (MED) [4]. The task is to detect the occurrence of a main event in a video clip without any textual metadata. The events of interest
are mostly daily activities ranging from “birthday party” to “changing a vehicle tire”. The event detection with zero training examples (0Ex) resembles the task of semantic search. 0Ex is an understudied problem, and only few studies have been proposed very recently [10, 44–48, 59, 77]. Dalton et al. [77] discussed a query expansion approach for concept and text retrieval. Habibian et al. [45] proposed to index videos by composite concepts that are trained by combining the labeled data of individual concepts. Wu et al. [47] introduced a multimodal fusion method for semantic concepts and text features. Given a set of tagged videos, Mazloom et al. [46] discussed a retrieval approach to propagate the tags to unlabeled videos for event detection. Singh et al. [78] studied a concept construction method that utilizes pairs of automatically discovered concepts and then prunes those concepts that are unlikely to be helpful for retrieval. Jiang et al. [14, 44] studied pseudo relevance feedback approaches which manage to significantly improve the original retrieval results. Existing related works inspire our system.

4.3 Semantic Search

4.3.1 Semantic Query Generation

Users can express a semantic query in a variety of forms, such as a few concept names, a sentence or a structured description. The Semantic Query Generation (SQG) component translates a user query into a multimodal system query, all words of which exist in the system vocabulary. A system vocabulary is the union of the dictionaries of all semantic features in the system. The system vocabulary, to some extent, determines what can be detected and thus searched by a system. For ASR/OCR features, the system vocabulary is usually large enough to cover most words in user queries. For semantic visual/audio concepts, however, the vocabulary is usually limited, and addressing the out-of-vocabulary issue is a major challenge for SQG. The mapping between the user and system query is usually achieved with the aid of an ontology such as WordNet and Wikipedia. For example, a user query “golden retriever” may be translated to its most relevant alternative “large-sized dog”, as the original concept may not exist in the system vocabulary.

For example, in the MED benchmark, NIST provides a user query in the form of an event-kit description, which includes a name, definition, explanation and visual/acoustic evidences. Table 4.1 shows the user query (event kit description) for the event “E011 Making a sandwich”. Its corresponding system query (with manual inspection) after SQG is shown in Table 4.2. As we see, SQG is indeed a challenging task as it involves understanding of text descriptions written in natural language.
Table 4.1: User query (event-kit description) for the event “Making a sandwich”.

<table>
<thead>
<tr>
<th>Event name</th>
<th>Making a sandwich</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>Constructing an edible food item from ingredients, often including one or more slices of bread plus fillings</td>
</tr>
<tr>
<td>Explication</td>
<td>Sandwiches are generally made by placing food items on top of a piece of bread, roll or similar item, and placing another piece of bread on top of the food items. Sandwiches with only one slice of bread are less common and are called &quot;open face sandwiches&quot;. The food items inserted within the slices of bread are known as &quot;fillings&quot; and often include sliced meat, vegetables (commonly used vegetables include lettuce, tomatoes, onions, bell peppers, bean sprouts, cucumbers, and olives), and sliced or grated cheese. Often, a liquid or semi-liquid &quot;condiment&quot; or &quot;spread&quot; such as oil, mayonnaise, mustard, and/or flavored sauce, is drizzled onto the sandwich or spread with a knife on the bread or top of the sandwich fillers. The sandwich or bread used in the sandwich may also be heated in some way by placing it in a toaster, oven, frying pan, countertop grilling machine, microwave or grill. Sandwiches are a popular meal to make at home and are available for purchase in many cafes, convenience stores, and as part of the lunch menu at many restaurants.</td>
</tr>
</tbody>
</table>

Table 4.2: System query for the event “E011 Making a sandwich”.

<table>
<thead>
<tr>
<th>Event ID</th>
<th>Name</th>
<th>Category</th>
<th>Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td>sin346_133</td>
<td>food</td>
<td>man made thing, food</td>
<td>very relevant</td>
</tr>
<tr>
<td>sin346_183</td>
<td>kitchen</td>
<td>structure building, room</td>
<td>very relevant</td>
</tr>
<tr>
<td>yfcc609_505</td>
<td>cooking</td>
<td>utensil tool</td>
<td>very relevant</td>
</tr>
<tr>
<td>sin346_261</td>
<td>room</td>
<td>structure building, room</td>
<td>relevant</td>
</tr>
<tr>
<td>sin346_28</td>
<td>bar_pub</td>
<td>structure building, room, commercial building</td>
<td>relevant</td>
</tr>
<tr>
<td>yfcc609_145</td>
<td>lunch</td>
<td>food, meal</td>
<td>relevant</td>
</tr>
<tr>
<td>yfcc609_92</td>
<td>dinner</td>
<td>food, meal</td>
<td>relevant</td>
</tr>
<tr>
<td>ASR</td>
<td>ASR_long</td>
<td>cheese, condiment, knife, plate, utensil, slice, toast, spread, cut, dish</td>
<td>relevant</td>
</tr>
<tr>
<td>OCR</td>
<td>OCR_short</td>
<td>sandwich</td>
<td>relevant</td>
</tr>
</tbody>
</table>

The first step in SQG is to parse negations in the user query in order to recognize counter-examples. The recognized examples can be either discarded or associated with a “NOT” operator in the system query. We found that adding counter examples using the Boolean NOT operator tends to improve performance. For example, in the query “Winning a race without a vehicle”, the query including only relevant concepts such as swimming, racing or marathon can achieve an AP of 12.57. However, the query
also containing “AND NOT” concepts such as car racing, horse riding or bicycling can achieve an AP of 24.50.

Given an event-kit description, a user query can be represented the event name (1-3 words) or the frequent words in the event-kit description (after removing the template and stop words). This user query can be directly used as the system query for ASR/OCR features as their vocabularies are sufficiently large. For visual/audio concepts, the query are used to map the out-of-vocabulary words to their most relevant concepts in the system vocabulary. To this end, we study the following classical mapping algorithms to map a word in the user query to the concept in the system vocabulary:

**Exact word matching**: A straightforward mapping is matching the exact query word (usually after stemming) against the concept name or the concept description. Generally, for unambiguous words, this method has high precision but low recall.

**WordNet mapping**: This mapping calculates the similarity between two words in terms of their distance in the WordNet taxonomy. The distance can be defined in various ways such as structural depths in the hierarchy [79] or shared overlaps between synonymous words [80]. Among the distance metrics, we found the structural depths yields more robust results [79]. WordNet mapping is good at capturing synonyms and subsumption relations between two nouns.

**PMI mapping**: The mapping calculates the Point-wise Mutual Information (PMI) [81] between two words. Suppose $q_i$ and $q_j$ are two words in a user query, we have:

$$\text{PMI}(q_i; q_j) = \log \frac{P(q_i, q_j | C_{ont})}{P(q_i | C_{ont})P(q_j | C_{ont})}, \quad (4.1)$$

where $P(q_i | C_{ont}), P(q_j | C_{ont})$ represent the probability of observing $q_i$ and $q_j$ in the ontology $C_{ont}$ (e.g. a collection of Wikipedia articles), which is calculated by the fraction of the document containing the word. $P(q_i, q_j | C_{ont})$ is the probability of observing the document in which $q_i$ and $q_j$ both occur. PMI mapping assumes that similar words tend to co-occur more frequently, and is good at capturing frequently co-occurring concepts (both nouns and verbs).

**Word embedding mapping**: This mapping learns a word embedding that helps predict the surrounding words in a sentence [82, 83]. The learned embedding, usually by neural network models, is in a lower-dimensional continuous vector space. The cosine coefficient between two words is often used to measure their distance. It is fast and also able to capture the frequent co-occurred words in similar contexts.

We found that discriminating the mapping relevance in a query may increase the performance. In other words, the calculated relevance can be used to weight query terms.
For example, the relevance can be categorized into discrete levels according to their relevance to the user query. Table 4.2 have three levels of relevance: “very relevant”, “relevant” and “slightly relevant”, and the levels are assigned to weight of 2.0, 1.0 and 0.5, respectively. We manually modified the relevance produced by the above automatic mapping algorithms. In this way, we can observe an absolute 1-2% improvement over the same query with no weights.

4.3.2 Retrieval Models

Given a system query, the multimodal search component aims at retrieving a ranked list for each modality. We are interested in leveraging the well-studied text retrieval models for video retrieval. This strategy allows us to utilize the infrastructure built for text retrieval. There is no single retrieval model that can work the best for all modalities. As a result, our system incorporates several classical retrieval models and applies them to their most appropriate modalities. Let \( Q = q_1, \ldots, q_n \) denote a system query. A retrieval model ranks videos by the score \( s(d|Q) \), where \( d \) is a video in the video collection \( C \). We study the following retrieval models:

**Vector Space Model (VSM):** This model represents both a video and a query as a vector of the words in the system vocabulary. The common vector representation includes generic term frequency (tf) and term frequency-inverse document frequency (tf-idf) \[84\]. \( s(d|Q) \) derives from either the dot product or the cosine coefficient between the video and the query vector.

\[
s(d|Q) = \sum_{i=1}^{n} \log \left| C \right| - df(q_i) + \frac{1}{2} \frac{tf(q_i, d)(k_1+1)}{tf(q_i, d)+k_1(1-b+b\frac{len(d)}{len})} \tag{4.2}
\]

where \( |C| \) is the total number of videos in the collection. \( df(\cdot) \) calculates the document frequency for a given word in the collection; \( tf(q_i, d) \) calculates the raw term frequency for the word \( q_i \) in the video \( d \). Unlike text retrieval, in which document frequencies and term frequencies are integers, in multimedia retrieval, these statistics can be real values as concepts are associated with real-valued detection scores. \( len(d) \) calculates the sum of concept or word detection scores in the video \( d \), and \( \text{Len} \) is the average video length in the collection. \( k_1 \) and \( b \) are two model parameters to tune \[85\]. In the experiments, we set \( b = 0.75 \), and tune \( k_1 \) in \([1.2, 2.0]\).
Language Model-JM Smoothing (LM-JM): The score is considered to be generated by a unigram language model [86]:

\[
s(d|Q) = \log P(d|Q) \propto \log P(d) + \sum_{i=1}^{n} \log P(q_i|d),
\]

(4.3)

where \( P(d) \) is usually assumed to be following the uniform distribution, i.e. the same for every video, and can be dropped in the retrieval model. In some cases, we can encode prior information about a video into \( P(d) \), such as its view count, length, and viralness [87]. \( P(q_i|d) \) is calculated from:

\[
P(q_i|d) = \lambda \frac{tf(q_i, d)}{\sum_w tf(w, d)} + (1 - \lambda) P(q_i|C),
\]

(4.4)

where \( w \) enumerates all the words or the concepts in a given video. \( P(q_i|C) \) is a smoother that can be calculated by \( df(q_i)/|C| \). As we see, Eq. (4.4) linearly interpolates the maximum likelihood estimation (first term) with the collection model (second term) by a coefficient \( \lambda \). The parameter is usually tuned in the range of \([0.7, 0.9]\). This model is good for retrieving long text queries, e.g. the frequent words in the event kit description.

Language Model-Dirichlet Smoothing (LM-DL): This model adds a conjugate prior (Dirichlet distribution) to the language model:

\[
P(q_i|d) = \frac{tf(q_i, d) + \mu P(q_i|C)}{\sum_w tf(w, d) + \mu},
\]

(4.5)

where \( \mu \) is a coefficient balancing the likelihood model and the conjugate prior. It is usually tuned in \([0, 2000]\) [86]. This model is good for short text queries, e.g. the event name.

4.4 Experiments

4.4.1 Setups

Dataset and evaluation: The experiments are conducted on two TRECVID benchmarks called Multimedia Event Detection (MED): MED13Test and MED14Test [4]. The performance is evaluated by the official metric Mean Average Precision (MAP). Each set includes 20 events over 25,000 test videos. The official NIST’s test split is used. We also evaluate each experiment on 10 randomly generated splits to reduce the bias brought by the split partition. The mean and 90% confidence interval are reported. All experiments are conducted without using any example or text metadata.
Features and queries: Videos are indexed by semantic features including semantic visual concepts, ASR, and OCR. The same semantic features described in Section 3.6.1 are used in the experiments, except here the features are represented by the raw detection scores before the adjustment. See the details of concept features in Table 3.1. we share our features and experimental results on the benchmark http://www.cs.cmu.edu/~lujiang/0Ex/icmr15.html.

The user query is the event-kit description. For ASR/OCR, the automatically generated event name and description representations are directly used as the system query. The system query for semantic concepts is obtained by a two-step procedure: a preliminary mapping is automatically generated by the discussed mapping algorithms. The results are then examined by human experts to figure out the final system query. We call these queries Expert queries. Besides, we also study the queries automatically generated by the mapping algorithm called “Auto SQG” in Section 4.4.2.

Configurations: In the multimodal search component, by default, the LM-JM model ($\lambda = 0.7$) is used for ASR/OCR for the frequent-words in the event-kit description. BM25 is used for ASR [88] and OCR features for the event name query (1-3 words), where $k_1 = 1.2$ and $b = 0.75$. Both the frequent-words query and the event name query are automatically generated without manual inspection. While parsing the frequent words in the event-kit description, the stop and template words are first removed, and words in the evidence section are counted three times. After parsing, the words with the frequency $\geq 3$ are then used in the query. VSM-tf model is applied to all semantic concept features.

In the SQG component, the exact word matching algorithm finds the concept name in the frequent event-kit words (frequency $\geq 3$). The WordNet mapping uses the distance metrics in [79] as the default metric. We build an inverted index over the Wikipedia corpus (about 6 million articles), and use it to calculate the PMI mapping. To calculate the statistics, we online issue queries to the index. A pre-trained word embedding trained on Wikipedia [83] is used to calculated the word embedding mapping.

4.4.2 Semantic Matching in SQG

We apply the SQG mapping algorithms in Section 4.3.1 to map the user query to the concepts in the vocabulary. The experiments are conducted only using semantic concept features. We use two metrics to compare these mapping algorithms. One is the precision of the 5 most relevant concepts returned by each algorithm. We manually assess the relevance for 10 events (E031-E040) on 4 concept features (i.e. 200 pairs for each mapping algorithm). The other is the MAP obtained by the 3 most relevant concepts. Table 4.3
lists the results, where the last column lists the runtime of calculating the mapping between 1,000 pairs of words. The second last row (Fusion) indicates the average fusion of the results of all mapping algorithms. As we see, in terms of P@5, PMI is slightly better than others, but it is also the slowest because its calculation involves looking up a index of 6 million text documents in Wikipedia. Fusion of all mapping results yields a better P@5.

We then combine the automatically mapped semantic concepts with the automatically generated ASR and OCR query. Here we assume users have specified which feature to use for each query, and SQG is used to automatically find relevant concepts or words in the specified features. We obtain this information in the event-kit description. For example, we will use ASR when we find words “narration/narrating” and “process” in the event-kit description. An event that has these words in the description tends to be an instructional event, such as “Making a sandwich” and “Repairing an appliance”, in which the spoken words are more likely to be detected accurately. Our result can be understood as an overestimate of a fully-automatic SQG system, in which users do not even need to specify the feature. As we see in Table 4.3, PMI performs the best on MED13Test whereas on MED14Test it is Exact Word Matching. The fusion of all mapping results (the second last row) improves the MAP on both the datasets. We then fine-tune the parameters of the mapping fusion and build our AutoSQG system (the last row).

As we see, AutoSQG only achieves about 55% of the full system’s MAP. Several reasons account for the performance drop: 1) the concept name does not accurately describe what is being detected; 2) the quality of mapping is limited (P@5=0.42); 3) relevant concepts are not necessarily discriminative concepts. For example, “animal” and “throwing ball” appear to be relevant to the query “playing a fetch”, but the former is too general and the latter is about throwing a baseball which is visually different; “dog” is much less discriminative than “group of dogs” for the query “dog show”. The results suggest that the automatic SQG is not well-understood. The proposed automatic mappings are still very preliminary, and could be further refined by manual inspection. A significant drawback of current mapping algorithms is representing a concept as a few words. However, in our manual assessment, we regard a concept as a multimodal document that includes a name, description, category, reliability (accuracy) and examples of the top detected video snippet.
Table 4.3: Comparison of SQG mapping algorithms.

<table>
<thead>
<tr>
<th>Mapping Method</th>
<th>MAP</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P@5</td>
<td>MED13Test</td>
</tr>
<tr>
<td>Exact Word Matching</td>
<td>0.340</td>
<td>9.66</td>
</tr>
<tr>
<td>WordNet</td>
<td>0.330</td>
<td>7.86</td>
</tr>
<tr>
<td>PMI</td>
<td>0.355</td>
<td>9.84</td>
</tr>
<tr>
<td>Word Embedding</td>
<td>0.335</td>
<td>8.79</td>
</tr>
<tr>
<td>Mapping Fusion</td>
<td>0.420</td>
<td>10.22</td>
</tr>
<tr>
<td>AutoSQGSys</td>
<td>-</td>
<td>12.00</td>
</tr>
</tbody>
</table>

4.4.3 Modality/Feature Contribution

Table 4.4 compares the modality contribution for semantic search, where each run represents a certain configuration. The MAP is evaluated on MED14Test and MED13Test. As we see, visual modality is the most contributing modality, which by itself can recover about 85% MAP of the full system. ASR and OCR provide complementary contribution to the full system but prove to be much worse than the visual features. The event-level comparison can be found in Table E.4. The experimental results also justify the rationale of multimodal search.

To understand the feature contribution, we conduct leave-one-feature-out experiments. The performance drop, after removing the feature, can be used to estimate its contribution to the full system. As we see in Table 4.5, the results show that every feature provides some contribution. As the feature contribution is mainly dominated by a number of discriminative events, the comparison is more meaningful at the event-level (see Table E.5 and Table E.6), where one can tell that, for example, the contribution of ImageNet mainly comes from three events E031, E015 and E037. Though the MAP drop varies on different events and datasets, the average drop on the two datasets follows: Sports > ImageNet > ASR > IACC > YFCC > DIY > OCR. The biggest contributor Sports is also the most computationally expensive feature to train. In fact, the above order of semantic concepts highly correlates to #samples in their datasets, which suggests the rationale of training concepts over big data sets.

Table 4.4: Comparison of modality contribution for semantic search.

<table>
<thead>
<tr>
<th>Run</th>
<th>MED13Test</th>
<th>MED14Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-split</td>
<td>10-splits</td>
</tr>
<tr>
<td>FullSys</td>
<td>20.75</td>
<td>19.47±1.19</td>
</tr>
<tr>
<td>VisualSys</td>
<td>18.31</td>
<td>18.30±1.11</td>
</tr>
<tr>
<td>ASRSys</td>
<td>6.53</td>
<td>6.90±0.74</td>
</tr>
<tr>
<td>OCRSys</td>
<td>2.04</td>
<td>4.14±0.07</td>
</tr>
</tbody>
</table>
Table 4.5: Comparison of feature contribution for semantic search.

<table>
<thead>
<tr>
<th>SysID</th>
<th>Visual Concepts</th>
<th>MAP 1-split</th>
<th>MAP 10-splits</th>
<th>MAP Drop(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MED13/IACC</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>18.93</td>
<td>18.61±1.13</td>
<td>9%</td>
</tr>
<tr>
<td>MED13/Sports</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>15.67</td>
<td>14.68±0.92</td>
<td>25%</td>
</tr>
<tr>
<td>MED13/YFCC</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>15.67</td>
<td>14.68±0.92</td>
<td>13%</td>
</tr>
<tr>
<td>MED13/DIY</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>19.96</td>
<td>18.70±1.19</td>
<td>4%</td>
</tr>
<tr>
<td>MED13/ImageNet</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>18.18</td>
<td>16.58±1.18</td>
<td>12%</td>
</tr>
<tr>
<td>MED13/OCR</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>18.48</td>
<td>17.78±1.10</td>
<td>11%</td>
</tr>
<tr>
<td>MED13/ASR</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>20.59</td>
<td>19.12±1.20</td>
<td>1%</td>
</tr>
<tr>
<td>MED13/OCR</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>18.34</td>
<td>17.79±1.95</td>
<td>11%</td>
</tr>
<tr>
<td>MED14/IACC</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>13.93</td>
<td>12.47±1.93</td>
<td>32%</td>
</tr>
<tr>
<td>MED14/Sports</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>20.05</td>
<td>18.55±2.13</td>
<td>3%</td>
</tr>
<tr>
<td>MED14/YFCC</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>20.40</td>
<td>18.42±2.22</td>
<td>1%</td>
</tr>
<tr>
<td>MED14/DIY</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>16.37</td>
<td>13.21±1.91</td>
<td>20%</td>
</tr>
<tr>
<td>MED14/ImageNet</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>18.36</td>
<td>17.62±1.84</td>
<td>11%</td>
</tr>
<tr>
<td>MED14/ASR</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>20.43</td>
<td>18.86±2.20</td>
<td>1%</td>
</tr>
<tr>
<td>MED14/OCR</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>20.43</td>
<td>18.86±2.20</td>
<td>1%</td>
</tr>
</tbody>
</table>

4.4.4 Comparison of Retrieval Methods

Table 4.6 compares the retrieval models on MED14Test using representative features such as ASR, OCR and two types of visual concepts. As we see, there is no single retrieval model that works the best for all features. For ASR and OCR words, BM25 and Language Model with JM smoothing (LM-JM) yield the best MAPs. An interesting observation is that VSM can only achieve 50% MAP of LM-JM on ASR (2.94 versus 5.79). This observation suggests that the role of retrieval models in semantic search is substantial. For semantic concepts, VSM performs no worse than other models. We hypothesize that it is because the dense raw concept representation, i.e. every dimension has a nonzero value, and this representation is quite different from sparse text features.

To verify this hypothesis, we apply the (top-\(k\)) concept adjustment to the Sports feature. We increase the parameter \(k\) proportional to the size of vocabulary. As we see, BM25 and LM exhibit better performance in the sparse representations. The results substantiate our hypothesis classical text retrieval algorithms also work for adjusted concept features.

Table 4.6: Comparison of retrieval models on MED14Test using ASR, OCR, Sports and IACC.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ASR</td>
<td>1</td>
<td>2.94</td>
<td>1.26</td>
<td>3.43</td>
<td><strong>5.79</strong></td>
<td>1.45</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>2.67</td>
<td>1.49</td>
<td>3.03</td>
<td><strong>4.26</strong></td>
<td>1.14</td>
</tr>
<tr>
<td>OCR</td>
<td>1</td>
<td>0.56</td>
<td>0.47</td>
<td><strong>1.47</strong></td>
<td>1.02</td>
<td>1.22</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>2.50</td>
<td>2.38</td>
<td><strong>4.52</strong></td>
<td>3.80</td>
<td>4.07</td>
</tr>
<tr>
<td>Sports</td>
<td>1</td>
<td><strong>9.21</strong></td>
<td>8.97</td>
<td>8.83</td>
<td>8.75</td>
<td>7.57</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td><strong>10.61</strong></td>
<td>10.58</td>
<td>10.13</td>
<td>10.25</td>
<td>9.04</td>
</tr>
<tr>
<td>IACC</td>
<td>1</td>
<td>3.49</td>
<td><strong>3.52</strong></td>
<td>2.44</td>
<td>2.96</td>
<td>2.06</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>2.88</td>
<td>2.77</td>
<td>2.05</td>
<td>2.45</td>
<td>2.08</td>
</tr>
</tbody>
</table>
Table 4.7: Study of retrieval performance using the adjusted concept features (Sports) on MED14Test.

<table>
<thead>
<tr>
<th>Density</th>
<th>VSM-tf</th>
<th>BM25</th>
<th>LM-JM</th>
<th>LM-DP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>9.06</td>
<td>9.58</td>
<td>9.99</td>
<td>9.38</td>
</tr>
<tr>
<td>2%</td>
<td>9.93</td>
<td>10.12</td>
<td>10.14</td>
<td>10.07</td>
</tr>
<tr>
<td>4%</td>
<td>10.34</td>
<td>10.36</td>
<td>10.26</td>
<td>10.38</td>
</tr>
<tr>
<td>16%</td>
<td>10.60</td>
<td>10.45</td>
<td>10.03</td>
<td>9.89</td>
</tr>
<tr>
<td>100%</td>
<td>10.61</td>
<td>10.13</td>
<td>10.25</td>
<td>9.04</td>
</tr>
</tbody>
</table>

4.5 Summary

In this chapter, we studied semantic search. We focused on two subproblems called semantic query generation and multimodal search. The proposed method goes beyond conventional text-to-text matching, and allows for semantic search without any textual metadata or example videos. We shared our compelling insights on a number of empirical studies. From the experimental results, we found that 1) retrieval models may have substantial impacts to the search result. A reasonable strategy is to incorporate multiple models and apply them to their appropriate features/modalities; 2) automatic query generation for queries in the form of event-kit descriptions is still very challenging. Combining mapping results from various mapping algorithms and applying manual examination afterward is the best strategy known so far.

The methods studied in this chapter is merely a first effort towards semantic search in Internet videos. The proposed method can be improved in various ways, e.g. by incorporating more accurate visual and audio concept detectors, by studying more appropriate retrieval models, by exploring search interfaces or interactive search schemes. As shown in our experiments, the automatic semantic query generation is not well understood. Closing the gap between the manual and automatic query may point to a promising direction.
Chapter 5

Hybrid Search

Propose a new methods for

5.1 Introduction

5.2 Related Work

5.3 Scalable Few Example Search

5.4 Experiments
Chapter 6

Video Reranking

6.1 Introduction

Reranking is a technique to improve the quality of search results [89]. The intuition is that the initial ranked result brought by the query has noise which can be refined by the multimodal information residing in the retrieved documents, images or videos. For example, in image search, the reranking is performed based on the results of text-to-text search, in which the initial results are retrieved by matching images’ surrounding texts [90]. Studies show that reranking can usually yield improvement of the initial retrieved result [91, 92]. Reranking by multimodal content-based search is still an understudied problem. It is more challenging than reranking by text-to-text search in image search, since the content features not only come from multiple modalities but also are much more noisy. In this chapter, we will introduce two content-based reranking methods, and discuss how they can be united in the same algorithm.

In a generic reranking method, we would first select a few videos, and assign assumed labels to them. Since no ground-truth label is used, the assumed labels are called “pseudo labels”. The samples with pseudo labels are used to build a reranking model. The statistics collected from the model is used to improve the initial ranked list. Most existing reranking or Pseudo-Relevance Feedback (PRF) methods are designed to construct pseudo labels from a single ranked list, e.g. from the text search [24, 93, 94] or the visual image search [95, 96]. Due to the challenge of multimedia retrieval, features from multiple modalities are usually used to achieve better performance [21, 56]. However, performing multimodal reranking is an important yet unaddressed problem. The key challenge is to jointly derive a pseudo label set from multiple ranked lists. Although reranking may not be a novel idea, reranking by multimodal content-based search is
clearly understudied and worthy of exploration, as existing studies mainly concentrate on text-to-text search.

Besides, an important step in this process is to assign weights to the samples with pseudo labels. The main strategy in current reranking methods is to assign binary (or predefined) weights to videos at different rank positions. These weighting schemes are simple to implement, yet may lead to suboptimal solutions. For example, the reranking methods in [44, 96, 97] assume that top-ranked videos are of equal importance (binary weights). The fact is that, however, videos ranked higher are generally more accurate, and thus more “important”, than those ranked lower. The predefined weights [94] may be able to distinguish importance but they are derived independently of reranking models, and thus may not faithfully reflect the latent importance. For example, Fig. 6.1 illustrates a ranked list of videos about “birthday party”, where all videos will be used as positive in reranking; the top two are true positive; the third video is a negative but closely related video on wedding shower due to the common concepts such as “gift”, “cake” and “cheering”; the fourth video is completely unrelated. As illustrated, neither binary nor predefined weights reflects the latent importance residing in the videos. Another important drawback of binary or predefined weighting is that since the weights are designed based on empirical experience, it is unclear where does, or even whether, the process would converge.

An ideal reranking method would consider the multimodal features and assign appropriate weights in a theoretically sound manner. To this end, we propose two content-based reranking models. The first model is called MultiModal Pseudo Relevance Feedback (MMPRF) which conducts the feedback jointly on multiple modalities leading to a consistent joint reranking model. MMPRF utilizes the ranked lists of all modalities and combines them in a principled approach. MMPRF is a first attempt that leverages both
high-level and low-level features for semantic search in a CBVSR system. As we know, it is impossible to use low-level features for semantic search, it is impossible to map the text-like query to the low-level feature without any training data. MMPRF circumvents the difficulty by transferring this problem into a supervised problem on pseudo labels.

The second model is called Self-Paced Reranking (SPaR) which assigns weights adaptively in a self-paced fashion. The method is established on the self-paced learning theory [98, 99]. The theory is inspired by the learning process of humans and animals, where samples are not presented randomly but organized in a meaningful order which illustrates from easy to gradually more complex examples [98]. In the context of reranking problems, easy samples are the top-ranked videos that have smaller loss. As opposed to utilizing all samples to learn a model simultaneously, the proposed model is learned gradually from easy to more complex samples. As the name “self-paced” suggests, in every iteration, SPaR examines the “easiness” of each sample based on what it has already learned, and adaptively determines their weights to be used in the next iteration.

SPaR represents a general multimodal reranking method. MMPRF is a special case of the proposed method that only uses the binary weighting. Compared with existing reranking methods, SPaR has the following three benefits. First, it is established on a solid theory, and of useful properties that can be theoretically verified. For example, SPaR has a concise mathematical objective to optimize, and its convergence property can be theoretically proved. Second, SPaR represents a general framework for reranking on multimodal data, which includes other methods [44, 97, 100], such as MMPRF, as special cases. The connection is useful because once an existing method is modeled as a special case of SPaR, the optimization methods discussed in this chapter become immediately applicable to analyze, and even solve the problem. Third, SPaR offers a compelling insight into reranking by multimodal content-based search [44, 45, 101], where the initial ranked lists are retrieved by content-based search.

The experimental results show promising results on several challenging datasets. As for semantic search, on the MED dataset, MMPRF and SPaR significantly improve the state-of-the-art baseline reranking methods with statistically significant differences; SPaR also outperforms the state-of-the-art reranking methods on an image reranking dataset called Web Query. For hybrid search, SPaR yields statistically significant improvements over the initial search results.
6.2 Related Work

The pseudo labels are usually obtained from a single modality in the literature. On the text modality, reranking, usually known as PRF, has been extensively studied. In the vector space model, the Rocchio algorithm [24] is broadly used, where the original query vector is modified by the vectors of relevant and irrelevant documents. Since a document’s true relevance judgment is unavailable, the top-ranked and bottom-ranked documents in the retrieved list are used to approximate the relevant and irrelevant documents. In the language model, PRF is usually performed with a Relevance Model (RM) [93, 102]. The idea is to estimate the probability of a word in the relevance model, and feed the probability back to smooth the query likelihood in the language model. Because the relevance model is unknown, RM assumes the top-ranked documents imply the distribution of the unknown relevance model. Several extensions have been proposed to improve RM. For example, instead of using the top-ranked documents, Lee et al. proposed a cluster-based resampling method to select better feedback documents [103]. Cao et al. explored a supervised approach to select good expansion terms based on a pre-trained classifier [104].

Reranking has also been shown to be effective in image and video retrieval. Yan et al. proposed a classification-based PRF [95–97], where the query image and its most dissimilar images are used as pseudo samples. The idea is to train an imbalanced SVM classifier, biased towards negative pseudo samples, as true negatives are usually much easier to find. In [94], the pseudo negatives, sampled from the ranked list of a text query, are first grouped into several clusters and the clusters’ conditional probabilities are fed back to alter the initial ranked list. Similar to [103], the role of clustering is to reduce the noise in the initial text ranked list. In [100, 105], the authors incorporated pseudo labels into the learning to rank paradigm. The idea is to learn a ranking function by optimizing the pair-wise or list-wise orders between pseudo positive and negative samples. In [106], the relevance judgments over the top-ranked videos are provided by users. Then an SVM is trained using visual features represented in the Fisher vector. However, the manual inspection of the search results is prohibited in many problems.

Existing reranking methods are mainly performed based on text-to-text search results, i.e. the initial ranked list is retrieved by text/keyword matching [105, 107]. In terms of the types of the reranking model, these methods can be categorized into Classification, Clustering, Graph and LETOR (LEarning-TO-Rank) based reranking. In Classification-based reranking [97], a classifier is trained upon the pseudo label set, and then tested on retrieved videos to obtain a reranked list. Similarly, in LETOR-based reranking [108] instead of a binary classifier, a ranking function is learned by the pair-wise [100] or list-wise [91, 105] RankSVM. In Clustering-based reranking [94], the retrieved videos are
aggregated into clusters, and the clusters’ conditional probabilities of the pseudo samples are used to obtain a reranked list. The role of clustering is to reduce the noise in the initial reranking. In Graph-based reranking [109, 110], the graph of retrieved samples needs to be first constructed, on which the initial ranking scores are propagated by methods such as the random walk [21], under the assumption that visually similar videos usually have similar ranks. Generally, reranking methods, including the above methods, are unsupervised methods. There also exist some studies on supervised reranking [90, 107]. Although reranking may not be a novel idea, reranking by multimodal content-based search is clearly understudied and worthy of exploration. Only a few methods have been proposed to conduct reranking based on content-based search results without examples (or training data).

6.3 MMPRF

The intuition behind MMPRF is that the relevant videos can be modeled by a joint discriminative model trained on all modalities. Suppose $d_j$ is a video in the collection, the probability of it being relevant can be calculated from the posterior $P(y_j|d_j; \Theta)$, where $y_j$ is the (pseudo) label for $j$th video, and $\Theta$ denotes the parameter in the joint model. In PRF methods on unimodal data, the partial model is trained on a single modality [95, 96]. We model the ranked list of each modality by its partial model, and our goal is to recover a joint model from these partial models. Formally, we use logistic regression as the discriminative model. For $i$th modality, the probability of a video being relevant can be calculated from

$$P(y_j|d_j; \Theta_i) = \frac{1}{1 + \exp\{-\theta_i^T w_{ij}\}}, \quad (6.1)$$

where $w_{ij}$ represents the video $d_j$’s feature vector from the $i$th modality. $\Theta_i = \theta_i$ denotes the model parameter vector for the $i$th modality. For a clearer notation, the intercept parameter $b$ is absorbed into the vector $\theta_i$. According to [95], the parameters $\Theta_i$ can be independently estimated using the top ranked $k^+$ samples and the bottom ranked $k^-$ samples in the $i$th modality, where $k^+$ and $k^-$ control the number of pseudo positive and pseudo negative samples, respectively.

However, the models estimated independently on each modality can be inconsistent. For example, a video may be used as a pseudo positive in one modality but as a pseudo negative in another modality. An effective approach to find the consistent pseudo label set is by Maximum Likelihood Estimation (MLE) with respect to the label set likelihood over all modalities. Formally, let $\Omega$ denotes the union of feedback videos of all modalities.
Our objective is to find a pseudo label set that maximizes:

\[
\text{arg max}_y \sum_{i=1}^m \ln L(y; \Omega, \Theta_i)
\]

s.t. \(|y|_1 \leq k^+; y \in \{0, 1\}^{|\Omega|}

where \(|\Omega|\) represents the total number of unique pseudo samples, and \(y = [y_1, ..., y_{|\Omega|}]^T\) represents their pseudo labels. \(L(y; \Omega, \Theta_i)\) is the likelihood of the label set \(y\) in the \(i\)th modality. The sum of likelihood in Eq. (6.2) indicates that each label in the pseudo label set needs to be verified by all modalities and the desired label set satisfies the most modalities. The selection process is analogous to voting, where every modality votes using the likelihood and the better the labels fit a modality, the higher the likelihood is, and vice versa. The set with the highest votes is selected as the pseudo label set. Because each pseudo label is validated by all modalities, the false positives in a single modality may be corrected during the voting. This property is unavailable when only a single modality is considered.

To solve Eq. (6.2), we rewrite the logarithmic likelihood using Eq. (6.1)

\[
\ln L(y; \Omega, \Theta_i) = \ln \prod_{d_j \in \Omega} P(y_j|d_j, \Theta_i)^{y_j}(1 - P(y_j|d_j, \Theta_i))^{1-y_j})
= \sum_{j=1}^{|\Omega|} y_j \theta_i^T w_{ij} - \theta_i^T w_{ij} - \ln(1 + \exp{-\theta_i^T w_{ij}})
\]

As can be seen, the problem of finding the pseudo label set with the maximum likelihood has been transferred to an integer programming problem, where the objective function is the sum of logarithmic likelihood across all modalities and the pseudo labels are restricted to be binary numbers.

As mentioned above, \(\theta_i\) can be independently estimated using the top-ranked and bottom-ranked samples in the \(i\)th modality. \(w_{ij}\) is the known feature vector. Plugging Eq. (6.3) back to Eq. (6.2) and dropping the constants, the objective function in Eq. (6.3) becomes

\[
\text{arg max}_y \sum_{i=1}^m \ln L(y; \Omega, \Theta_i) = \text{arg max}_y \sum_{i=1}^m \sum_{j=1}^{|\Omega|} y_j \theta_i^T w_{ij}.
\]

As can be seen, the problem of finding the pseudo label set with the maximum likelihood has been transferred to an integer programming problem, where the objective function is the sum of logarithmic likelihood across all modalities and the pseudo labels are restricted to be binary numbers.

The pseudo negative samples can be randomly sampled from the bottom-ranked samples, as suggested in [94, 95]. In the worst case, suppose \(n\) pseudo negative samples
are randomly and independently sampled from a collection of samples, and the probability selecting a false negative sample is \( p \). Let the random variable \( X \) represents the experiment of selecting pseudo negative samples, then the random variable follows the binomial distribution, i.e. \( X \sim B(n, p) \). It is easy to calculate the probability of selecting at least 99% true negatives by

\[
F(X \leq 0.01n) = \sum_{i=0}^{\lfloor 0.01n \rfloor} \binom{n}{i} p^i (1-p)^{n-i},
\]

where \( F \) is the binomial cumulative distribution function. \( p \) is usually very small as the number of negative videos is usually far more than that of positive videos. For example, on the MED dataset, \( p = 0.003 \), and if \( n = 100 \), the probability of randomly selecting at least 99% true negatives is 0.963. This result suggests that randomly sampled pseudo negatives seems to be sufficiently accurate on the MED dataset.

If the objective function in Eq. (6.2) is calculated from:

\[
\ln \hat{L}(y; \Omega, \Theta_i) = E[y|\Omega, \Theta_i] = \sum_{j=1}^{[\Omega]} y_j P(y_j|d_j, \Theta_i),
\]

then the optimization problem in Eq. (6.2) can be solved by the late fusion [111], i.e. the scores in different ranked lists are averaged (or summed) and then the top \( k^+ \) videos are selected as pseudo positives. It is easy to verify this yields optimal \( y \) for Eq. (6.2). In fact, late fusion is a common method to combine information within multiple modalities. Eq. (6.2) provides a theoretical justification for the simple method i.e. rather than maximizing the sum of likelihood, one can alternatively maximize the sum of expected values. Note the problem in Eq. (6.2) is tailored to select a small number of accurate labels as opposed to producing a good ranked list in general. Empirically, we observed selecting pseudo positives by the likelihood is better than the expected value when the multiple ranked lists are generated by different retrieval algorithms, e.g. BM25, TFIDF, or Language Model. This is because the distributions of those ranked list (even after normalization) can be quite different. A pain late fusion may produce a biased estimation. In MLE model, estimating \( \Theta \) in Eq. 6.1 first can put the parameter back into the same scale.

### 6.4 SPaR

Self-paced Reranking is a general reranking framework for multimedia search. Given a dataset of \( n \) samples with features extracted from \( m \) modalities, let \( x_{ij} \) denote the feature of the \( i^{th} \) sample from the \( j^{th} \) modalities, e.g., feature vectors extracted from different
channels of a video. \( y_i \in \{-1, 1\} \) represents the pseudo label for the \( i^{th} \) sample whose values are assumed as the true labels are unknown to reranking methods. The kernel SVM is used to illustrate the algorithm due to its robustness and decent performance in reranking \([96]\). We will discuss how to generalize it to other models in Section 6.4.3.

Let \( \Theta_j = \{w_j, b_j\} \) denote the classifier parameters for the \( j^{th} \) modality, which includes a coefficient vector \( w_j \) and a bias term \( b_j \). Let \( v = [v_1, ..., v_n]^T \) denote the weighting parameters for all samples. Inspired by the self-paced learning \([99]\), suppose \( n \) is the total number of samples; \( m \) is the total number of modalities; the objective function \( \mathcal{E} \) can be formulated as:

\[
\min_{\Theta_1, ..., \Theta_m, y, v} \mathcal{E}(\Theta_1, ..., \Theta_m, v, y; C, k) = \sum_{j=1}^{m} \min_{\Theta_j, y, v} \mathcal{E}(\Theta_j, v, y; C, k)
\]

\[
= \min_{y, v, w_1, ..., w_m, \{\ell_{ij}\}} \frac{1}{2} \sum_{j=1}^{m} \|w_j\|^2 + C \sum_{i=1}^{n} \sum_{j=1}^{m} v_i \ell_{ij} + mf(v; k)
\]

s.t. \( \forall i, \forall j, y_i (w_j^T \phi(x_{ij}) + b_j) \geq 1 - \ell_{ij}, \ell_{ij} \geq 0 \)

\[
y \in \{-1, +1\}^n, v \in [0, 1]^n,
\]

where \( \ell_{ij} \) is the hinge loss, calculated from:

\[
\ell_{ij} = \max\{0, 1 - y_i \cdot (w_j^T \phi(x_{ij}) + b_j)\}.
\]

\( \phi(\cdot) \) is a feature mapping function to obtain non-linear decision boundaries. \( C (C > 0) \) is the standard regularization parameter trading off the hinge loss and the margin. \( \sum_{j=1}^{m} v_i \ell_{ij} \) represents the weighted loss for the \( i^{th} \) sample. The weight \( v_i \) reflects the sample’s importance, and when \( v_i = 0 \), the loss incurred by the \( i^{th} \) sample is always zero, i.e. it will not be selected in training.

\( f(v; k) \) is a regularization term that specifies how the samples are selected and how their weights are calculated. It is called the self-paced function as it determines the specific learning scheme. There is an \( m \) in front of \( f(v; k) \) as \( \sum_{j=1}^{m} f(v; k) = mf(v; k) \). \( f(v; k) \) can be defined in various forms which will be discussed in Section 6.4.2. The objective is subjected to two sets of constraints: the first set of constraints in Eq. (6.7) is the soft margin constraint inherited from the conventional SVM. The second constraints in Eq. (6.7) define the domains of pseudo labels and their weights, respectively.

Eq. (6.7) turns out to be difficult to optimize directly due to its non-convexity and complicated constraints. However, it can be effectively optimized by Cyclic Coordinate Method (CCM) \([112]\). CCM is an iterative method for non-convex optimization, in which the variables are divided into a set of disjoint blocks, in this case two blocks, i.e. classifier parameters \( \Theta_1, ..., \Theta_m \), and pseudo labels \( y \) and weights \( v \). In each iteration, a block of
variables can be optimized while keeping the other block fixed. Suppose $\mathbb{E}_{\Theta}$ represents the objective with the fixed block $\Theta_1, \ldots, \Theta_m$, and $\mathbb{E}_{y,v}$ represents the objective with the fixed block $y$ and $v$. Eq. (6.7) can be solved by the algorithm in Fig. 6.2. In Step 2, the algorithms initializes the starting values for the pseudo labels and weights. Then it optimizes Eq. (6.7) iteratively via Step 4 and 5, until convergence is reached.

Fig. 6.2 provides a theoretical justification for reranking from the optimization perspective. Fig. 6.3 lists general steps for reranking that have one-to-one correspondence with the steps in Fig. 6.2. The two algorithms present the same methodology from two perspectives. For example, optimizing $\Theta_1, \ldots, \Theta_m$ can be interpreted as training a reranking model. In the first few iterations, Fig. 6.2 gradually increases the $1/k$ to control the learning pace, which, correspondingly, translates to adding more pseudo positives [44] in training the reranking model.

Fig. 6.2 and Fig. 6.3 offer complementary insights. Fig. 6.2 theoretically justifies Fig. 6.3 on the convergence and the decrease of objective. On the other hand, the empirical experience from studying Fig. 6.3 offers valuable advices on how to set starting values from the initial ranked lists, which is less concerned in the optimization perspective. According to Fig. 6.3, to use SPaR one needs to alternate between two steps: training reranking models and determining the pseudo samples and their weights for the next iteration. We will discuss how to optimize $\mathbb{E}_{y,v}$ (training reranking models on pseudo samples) in Section 6.4.1, and how to optimize $\mathbb{E}_{\Theta}$ (selecting pseudo samples and their weights based on the current reranking model) in Section 6.4.2.

6.4.1 Learning with Fixed Pseudo Labels and Weights

With the fixed $y, v$, Eq. (6.7) represents the sum of weighted hinge loss across all modalities, i.e,
As mentioned, $v_i \ell_{ij}$ is the discounted hinge loss of the $i$th sample from the $j$th modality. Eq. (6.9) represents a non-conventional SVM as each sample is associated with a weight reflecting its importance. Eq. (6.9) is non-trivial to optimize directly due to its complex constraints. As a result, we introduce a method that finds the optimum solution for Eq. (6.9). The objective of Eq. (6.9) can be decoupled, and each modality can be optimized independently. Now consider the $j$th modality ($j = 1, ..., m$). We introduce Lagrange multipliers $\lambda$ and $\alpha$, and define the Lagrangian of the problem as:

$$
\Lambda(w_j, b_j, \alpha, \lambda) = \frac{1}{2} \|w_j\|_2^2 + C \sum_{i=1}^{n} v_i \ell_{ij} + \sum_{i=1}^{n} \alpha_{ij} (1 - \ell_{ij} - y_i w_j^T \phi(x_{ij}) - y_i b_j) + \sum_{i=1}^{n} \lambda_{ij} \ell_{ij}
$$

s.t. $\forall i, \alpha_{ij} \geq 0, \lambda_{ij} \geq 0$. (6.10)

Since only the $j$th modality is considered, $j$ is a fixed constant. The Slater’s condition trivially holds for the Lagrangian, and thus the duality gap vanishes at the optimal solution. According to the KKT conditions [113], the following conditions must hold for its optimal solution:

$$
\nabla_{w_j} \Lambda = w_j - \sum_{i=1}^{n} \alpha_{ij} y_i \phi(x_{ij}) = 0, \quad \nabla_{b_j} \Lambda = \sum_{i=1}^{n} \alpha_{ij} y_i = 0,
$$

$$
\forall i, \frac{\partial \Lambda}{\partial \ell_{ij}} = C v_i - \alpha_{ij} - \lambda_{ij} = 0.
$$

(6.11)

According to Eq. (6.11), $\forall i, \lambda_{ij} = C v_i - \alpha_{ij}$, and since Lagrange multipliers are nonnegative, we have $0 \leq \alpha_{ij} \leq C v_i$. Substitute these inequations and Eq. (6.11) back into Eq. (6.10), the problem’s dual form can be obtained by:
\[
\max_{\alpha} \sum_{i=1}^{n} \alpha_{ij} - \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_{ij} \alpha_{kj} y_i y_k \kappa(x_{ij}, x_{kj}),
\]
\[
\text{s.t. } \sum_{i=1}^{n} y_i \alpha_{ij} = 0, 0 \leq \alpha_{ij} \leq C v_i,
\]

where \(\kappa(x_{ij}, x_{kj}) = \phi(x_{ij})^T \phi(x_{kj})\) is the kernel function. Compared with the dual form of a conventional SVM, Eq. (6.12) imposes a sample-specific upper-bound on the support vector coefficient. A sample’s upper-bound is proportional to its weight, and therefore a sample with a smaller weight \(v_i\) is less influential as its support vector coefficient is bounded by a small value of \(C v_i\). Eq. (6.12) degenerates to the dual form of conventional SVMs when \(v = 1\). According to the Slater’s condition, strong duality holds, and therefore Eq. (6.10) and Eq. (6.12) are equivalent problems. Since Eq. (6.12) is a quadratic programming problem in its dual form, there exists a plethora of algorithms to solve it [113].

### 6.4.2 Learning with Fixed Classification Parameters

With the fixed classification parameters \(\Theta_1, ..., \Theta_m\), Eq. (6.7) becomes:

\[
\min_{y, v} \mathbb{E}_{\Theta}(y, v; k) = \min_{y, v} C \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} \ell_{ij} + mf(v; k)
\]
\[
\text{s.t. } y \in \{-1, +1\}^n, v \in [0, 1]^n.
\]

The goal of Eq. (6.13) is to learn not only the pseudo labels \(y\) but also their weights \(v\). Note, as discussed in Section 6.3, the pseudo negative samples can be randomly sampled. In this section, the learning process focuses on pseudo positive samples. Learning \(y\) is easier as its optimal values are independent of \(v\). We first optimize each pseudo label by:

\[
y^*_i = \arg\min_{y_i = \{+1, -1\}} \mathbb{E}_{\Theta}(y, v) = \arg\min_{y_i = \{+1, -1\}} C \sum_{j=1}^{m} \ell_{ij},
\]

where \(y^*_i\) denotes the optimum for the \(i\)th pseudo label. Solving Eq. (6.14) is simple as all labels are independent with each others in the sum, and each label can only take binary values. Its global optimum can be efficiently obtained by enumerating each \(y_i\). For \(n\) samples, we only need to enumerate \(2^n\) times. In practice, we may need to tune the model to ensure there are a number of pseudo positives.
Having found the optimal $y$, the task switches to optimizing $v$. Recall $f(v; k)$ is the self-paced function, and in [99], it is defined as the $l_1$ norm of $v \in [0, 1]^n$.

$$f(v; k) = -\frac{1}{k}||v||_1 = -\frac{1}{k} \sum_{i=1}^{n} v_i. \quad (6.15)$$

Substituting Eq. (6.15) back into Eq. (6.13), the optimal $v^* = [v_1^*, ..., v_n^*]^T$ is then calculated from

$$v_i^* = \begin{cases} 
1 & \frac{1}{m} \sum_{j=1}^{m} C \ell_{ij} < \frac{1}{k} \\
0 & \frac{1}{m} \sum_{j=1}^{m} C \ell_{ij} \geq \frac{1}{k}.
\end{cases} \quad (6.16)$$

The underlying intuition of the self-paced learning can be justified by the closed-form solution in Eq. (6.16). If a sample’s average loss is less than a certain threshold, $1/k$ in this case, it will be selected, or otherwise not be selected, as a training example. The parameter $k$ controls the number of samples to be included in training. Physically, $1/k$ corresponds to the “age” of the model. When $1/k$ is small, only easy samples with small loss will be considered. As $1/k$ grows, more samples with larger loss will be gradually appended to train a “mature” reranking model.

As we see in Eq. (6.16), the variable $v$ takes only binary values. This learning scheme yields a hard weighting as a sample can be either selected ($v_i = 1$) or unselected ($v_i = 0$). The hard weighting is less appropriate in our problem as it cannot discriminate the importance of samples, as shown in Fig. 6.4. Correspondingly, the soft weighting, which assigns real-valued weights, reflects the latent importance of samples in training more faithfully. The comparison is analogous to the hard/soft assignment in Bag-of-Words quantization, where an interest point can be assigned either to its closest cluster (hard), or to a number of clusters in its vicinity (soft). We discuss three of them, namely, linear, logarithmic and mixture weighting. Note that the proposed functions may not be optimal as there is no single weighting scheme that can always work the best for all datasets.

**Linear soft weighting:** Probably the most common approach is to linearly weight samples with respect to their loss. This weighting can be realized by the following self-paced function:

$$f(v; k) = \frac{1}{k} \left( \frac{1}{2} ||v||_2^2 - \sum_{i=1}^{n} v_i \right). \quad (6.17)$$

Considering $v_i \in [0, 1]$, the close-formed optimal solution for $v_i \ (i = 1, 2, ..., n)$ can be written as:

$$v_i^* = \begin{cases} 
-k\left(\frac{1}{m} \sum_{j=1}^{m} C \ell_{ij}\right) + 1 & \frac{1}{m} \sum_{j=1}^{m} C \ell_{ij} < \frac{1}{k} \\
0 & \frac{1}{m} \sum_{j=1}^{m} C \ell_{ij} \geq \frac{1}{k}.
\end{cases} \quad (6.18)$$
Figure 6.4: Comparison of different weighting schemes ($k = 1.2$, $k' = 6.7$). Hard Weighting assigns binary weights. The figure is divided into 3 colored regions, i.e. “white”, “gray” and “black” in terms of the loss.

Similar as the hard weighting in Eq. (6.16), the weight is 0 for the samples whose average loss is larger than $1/k$; Otherwise, the weight is linear to the loss (see Fig. 6.4).

**Logarithmic soft weighting:** The linear soft weighting penalizes the weight linearly in terms of the loss. A more conservative approach is to penalize the weight logarithmically, which can be achieved by the following function:

$$f(v; k) = \sum_{i=1}^{n} (\zeta v_i - \frac{\zeta v_i}{\log \zeta}),$$

(6.19)

where $\zeta = (k - 1)/k$ and $k > 1$. The closed-form optimal is then given by:

$$v^*_i = \begin{cases} \frac{1}{\log \zeta} \log (\frac{1}{m} \sum_{j=1}^{m} C\ell_{ij} + \zeta) & \frac{1}{m} \sum_{j=1}^{m} C\ell_{ij} < \frac{1}{k} \\ 0 & \frac{1}{m} \sum_{j=1}^{m} C\ell_{ij} \geq \frac{1}{k}. \end{cases}$$

(6.20)

**Mixture weighting:** Mixture weighting is a hybrid of the soft and the hard weighting. One can imagine that the loss range is divided into three colored areas, as illustrated in Fig. 6.4. If the loss is either too small (“white” area) or too large (“black” area), the hard weighting is applied. Otherwise, for the loss in the “gray” area, the soft weighting is applied. Compared with the soft weighting scheme, the mixture weighting tolerates small errors up to a certain point. To define the start of the “gray” area, an additional parameter $k'$ is introduced. Formally,

$$f(v; k, k') = -\zeta \sum_{i=1}^{n} \log (v_i + \zeta k),$$

(6.21)
where $\zeta = \frac{1}{k' - k}$ and $k' > k > 0$. The closed-form optimal solution is given by:

$$v^*_i = \begin{cases} 1 & \frac{1}{m} \sum_{j=1}^{m} C_{ij} \leq \frac{1}{k'} \\ 0 & \frac{1}{m} \sum_{j=1}^{m} C_{ij} \geq \frac{1}{k} \\ \sum_{j=1}^{m} C_{ij} - k\zeta & \text{otherwise} \end{cases}$$

(6.22)

Eq. (6.22) tolerates any loss lower than $1/k'$ by assigning the full weight. It penalizes the weight by the inverse of the loss for samples in the “gray” area which starts from $1/k'$ and ends at $1/k$ (see Fig. 6.4). The mixture weighting has the properties of both hard and soft weighting schemes. The comparison of these weighting schemes is listed in the toy example below.

**Example 6.1.** Suppose we are given six samples from two modalities. The hinge loss of each sample calculated by Eq. (6.8) is listed in the following table, where Loss1 and Loss2 column list the losses w.r.t. the first and the second modality, whereas “Avg Loss” column lists the average loss. The last four columns present the weights calculated by Eq. (6.16), Eq. (6.18), Eq. (6.20) and Eq. (6.22) where $k = 1.2$ and $k' = 6.7$.

<table>
<thead>
<tr>
<th>ID</th>
<th>Loss1</th>
<th>Loss2</th>
<th>Avg Loss</th>
<th>Hard</th>
<th>Linear</th>
<th>Log</th>
<th>Mixture</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.08</td>
<td>0.02</td>
<td>0.05</td>
<td>1</td>
<td>0.940</td>
<td>0.853</td>
<td>1.000</td>
</tr>
<tr>
<td>2</td>
<td>0.15</td>
<td>0.09</td>
<td>0.12</td>
<td>1</td>
<td>0.856</td>
<td>0.697</td>
<td>1.000</td>
</tr>
<tr>
<td>3</td>
<td>0.50</td>
<td>0.50</td>
<td>0.50</td>
<td>1</td>
<td>0.400</td>
<td>0.226</td>
<td>0.140</td>
</tr>
<tr>
<td>4</td>
<td>0.96</td>
<td>0.70</td>
<td>0.83</td>
<td>1</td>
<td>0.004</td>
<td>0.002</td>
<td>0.001</td>
</tr>
<tr>
<td>5</td>
<td>0.66</td>
<td>1.02</td>
<td>0.84</td>
<td>0</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>6</td>
<td>1.30</td>
<td>1.10</td>
<td>1.20</td>
<td>0</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

As we see, Hard produces less reasonable solutions, e.g. the difference between the first (ID=1) and the fourth sample (ID=4) is 0.78 and they share the same weight 1; on the contrary, the difference between the fourth and the fifth sample is only 0.01, but suddenly they have totally different weights. This abrupt change is absent in other weighting schemes. Log is a more prudent scheme than Linear as it diminishes the weight more rapidly. Among all weighting schemes, Mixture is the only one that tolerates small errors.

### 6.4.3 Convergence and Relation to Other Reranking Models

The proposed SPaR has some useful properties. The following lemma proves that the optimum solution can be obtained for the proposed self-paced functions.

**Lemma 6.1.** For the self-paced functions in Section 6.4.2, the proposed method finds the optimal solution for Eq. (6.13).
The following theorem proves the convergence of algorithm 6.2.

**Theorem 6.2.** The algorithm in Fig. 6.2 converges to a stationary solution for any fixed $C$ and $k$.

A general form of Eq. (6.7) is written as

$$\min_{\Theta_1, \ldots, \Theta_m, v, y} E(\Theta_1, \ldots, \Theta_m, v, y; k) = \min_{\Theta_1, \ldots, \Theta_m, v, y} \sum_{i=1}^{n} \sum_{j=1}^{m} v_i \text{Loss}(x_{ij}; \Theta_j) + mf(v; k)$$

(6.23)

s.t. Constraints on $\Theta_1, \ldots, \Theta_m, y \in \{-1, +1\}^n, v \in [0, 1]^n$,

where $\text{Loss}(x_{ij}; \Theta_j)$ is a general function of the loss incurred by the $i^{th}$ sample against the $j^{th}$ modality, e.g., it is defined as the sum of the hinge loss and the margin in Eq. (6.7). The constraints on $\Theta_1, \ldots, \Theta_m$ are the constants in the specific reranking model. Alg. 6.2 is still applicable to solve Eq. (6.23). In theory, Eq. (6.23) can be used to find both pseudo positive, pseudo negative samples, and their weights. In practice, we recommend only learning pseudo positive samples and their weights by Eq. (6.23).

Eq. (6.23) represents a general reranking framework, which includes existing reranking methods as special cases. For example, generally, when $\text{Loss}$ takes the negative likelihood of Logistic Regression, and $f(v; k)$ takes Eq. (6.15) (hard weighting scheme), SPaR corresponds to MMPRF. When $\text{Loss}$ is the hinge loss, $f(v; k)$ is Eq. (6.15), the pseudo labels are assumed to be $+1$, and there is only one modality, SPaR corresponds to Classification-based PRF [96, 97]. Given $\text{Loss}$ and constraints on $\Theta$ are from pair-wise RankSVM, SPaR can degenerate to LETOR-based reranking methods [100].

### 6.5 Experiments

#### 6.5.1 Setups

**Dataset, query and evaluation:** We conduct experiments on the TRECVID Multimedia Event Detection (MED) set including around 34,000 videos on 20 Pre-Specified events. The used queries are semantic queries discussed in the previous chapters. The performance is evaluated on the MED13Test consisting of about 25,000 videos, by the official metric Mean Average Precision (MAP). The official test split released by NIST is
used. No ground-truth labeled videos are used in all experiments. In the baseline comparison, we evaluate each experiment 10 times on randomly generated splits to reduce the bias brought by the partition. The mean and 90% confidence interval are reported.

Features: The used semantic features include Automatic Speech Recognition (ASR), Optical Character Recognition (OCR), Semantic INdexing (SIN) and DCNN (Deep Convolutional Neural Network). SIN and DCNN include 346 visual concepts and 1,000 visual objects trained on TRECVID and ImageNet sets. Two types of low-level features are used: dense trajectories and MFCCs.

Baselines: The proposed method is compared against the following baselines: 1) Without Reranking is a plain retrieval method without Reranking, and the language model with Jelinek-Mercer smoothing is used. 2) Rocchio is a classical reranking model for vector space model under tf-idf representation. 3) Relevance Model is a well-known reranking method for text, and the variant with the i.i.d. assumption is used. 4) CPRF(Classification-based PRF) is a seminal PRF-based reranking method. Following , SVM classifiers with the $\chi^2$ kernel are trained using the top-ranked and bottom-ranked videos. 5) Learning to Rank is a LETOR-based method. Following , it is trained using the pairwise constraints derived from the pseudo-positives and pseudo-negatives. A LambdaMART in the RankLib toolkit is used to train the RankSVM model; The parameters of all methods, including the proposed SPaR, are tuned on a third dataset that shares no overlap with our development set.

Model Configuration: Alg. 6.2 is used to solve MMPRF and SPaR. In MMPRF, lp_solve is used to solve the linear/integer programming problem. The regression with the elastic net regularization is used to estimate the parameters of the partial models. Linear and $\chi^2$ kernel is used for dense trajectory and MFCCs features. By default, 10 pseudo positive samples are selected by Eq. (6.2) in MMPRF MLE model. A hundred of pseudo-negatives were randomly sampled from the bottom of the fused ranked list. For a fair comparison, we fix the pseudo negative samples used in all baseline methods.

In SPaR, Eq. (6.12) is solved by the quadratic programming package “quadprog”, in which the parameter $C$ is fixed to 1 and the $\phi$ is set as the $\chi^2$ explicit feature map. By default, Eq. (6.21) is used. The initial values of the pseudo positive labels and weights are derived by MMPRF. Since, according to , pseudo negative samples have little impact on the MAP, Eq. (6.12) is only used to learn pseudo positive samples.
6.5.2 Comparison with Baseline methods

We first examine the overall MAP in Table 6.1, in which the best result is highlighted. As we see, MMPRF significantly outperforms the baseline method without PRF. SPaR outperforms all baseline methods by statistically significant differences. For example, on the NIST’s split, it increases the MAP of the baseline without reranking by a relative 230% (absolute 9%), and the second best method MMPRF by a relative 28% (absolute 2.8%). Fig. 6.6 plots the AP comparison on each event, where the x-axis represents the event ID and the y-axis denotes the average precision. As we see, SPaR outperforms the baseline without reranking on 18 out of 20 events, and the second best MMPRF on 15 out of 20 events. The improvement is statistically significant at the p-level of 0.05, according to the paired t-test. Fig. 6.5 illustrates the top retrieved results on two events that have the highest improvement. As we see, the videos retrieved by SPaR are more accurate and visually coherent.

We observed two reasons accounting for the improvement brought by MMPRF. First, MMPRF explicitly considers multiple modalities and thus can produce a more accurate pseudo label set. Second, the performance of MMPRF is further improved by leveraging both high-level and low-level features. The improvement of SPaR stems from the capability of adjusting weights of pseudo samples in a reasonable way. For example, Fig. 6.7 illustrates the weights assigned by CPRF and SPaR on the event “E008 Flash Mob Gathering”. Three representative videos are plotted where the third (ID=3) is true positive, and the others (ID=1,2) are negative. The tables on the right of Fig. 6.7 list their pseudo labels and weights in each iteration. Since the true labels are unknown to the methods, in the first iteration, both methods made mistakes. In Conventional Reranking, the initial pseudo labels and learned weights stay unchanged thereafter.
Figure 6.6: The AP comparison with the baseline methods. The MAP across all events is available in Table 6.1.

Figure 6.7: Weights changed by CPRF and SPaR on representative videos in different iterations.

However, SPaR adaptively adjusts their weights as the iteration grows, e.g. it reduces the overestimated weights of videos (ID=1,2) in iteration 2 and 3 probably because of their dissimilarity from other pseudo positive videos.

We found two scenarios where SPaR and MMPRF can fail. First, when the initial top-ranked samples retrieved by queries are completely off-topic. SPaR and MMPRF may not recover from the inferior starting values, e.g. the query brought by “E022 Cleaning an appliance” are off-topic (all videos are on all cooking in kitchen). Second, SPaR and MMPRF may not help when the features used in reranking are not discriminative to the queries, e.g. for “E025 Marriage Proposal”, our system lacks of meaningful detectors such as “stand on knees”. Therefore even if 10 true positives are used, the AP is still bad.

6.5.3 Impact of Pseudo Label Accuracy

To study the impact of pseudo label accuracy, we conduct the following experiments, where the pseudo positive samples are simply selected by the top $k^+$ samples in the ranked lists of individual features and the fusion of all features. Figure 6.8(a) illustrates
Table 6.1: MAP (× 100) comparison with the baseline methods across 20 Pre-Specified events.

<table>
<thead>
<tr>
<th>Method</th>
<th>NIST’s split</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Reranking</td>
<td>3.9</td>
</tr>
<tr>
<td>Rocchio</td>
<td>5.7</td>
</tr>
<tr>
<td>Relevance Model</td>
<td>2.6</td>
</tr>
<tr>
<td>CPRF</td>
<td>6.4</td>
</tr>
<tr>
<td>Learning to Rank</td>
<td>3.4</td>
</tr>
<tr>
<td>MMPRF</td>
<td>10.1</td>
</tr>
<tr>
<td>SPaR</td>
<td>12.9</td>
</tr>
</tbody>
</table>

Figure 6.8: The correlation between pseudo label accuracy and MAP. Each point represents an experiment with pseudo samples with certain accuracy.

the result in a scatter plot where the x-axis represents the accuracy of pseudo positives and the y-axis represents the MAP. As can be seen, there is a strong correlation between the MAP and the accuracy of pseudo-positives. The average Pearson correlation is 0.93. We also conduct a similar experiment on pseudo negative samples, where the pseudo positive samples are fixed and the pseudo negative samples are randomly selected from the bottom of the initial ranked list. The experiments are conducted five times and the result is shown in Figure 6.8(b). As we see, the precision is always larger than 0.980 as false negatives are difficult to find. This observation agrees with the analytical result in Section 6.3. Given such highly accurate pseudo negatives, the impact of pseudo negatives on MAP seems to be marginal. In summary, the results demonstrate that the accuracy of pseudo positive samples has a substantial impact on the MAP. The impact of pseudo negative samples, however, appears to be negligible.

6.5.4 Comparison of Weighting Schemes

Section 6.4.2 discusses four weighting schemes including the conventional hard weighting and the proposed three soft weighting schemes. The following two predefined schemes are also included for comparison: 1) Interpolation is a commonly used weighting scheme
which assigns weights linearly to a sample’s rank order [94, 105]:

\[ v_i = \frac{1}{m} \sum_{j=1}^{m} \left( 1 - \frac{\text{rank}(x_{ij})}{N} \right), \]  

where \( N \) is the number of total pseudo samples. The weight for the first sample is 1.0, and 0.0 for the last. \( \text{rank}(\cdot) \) returns the sample’s rank order in its list. 2) Inverse Rank assigns a sample’s weight based on its inverse rank order. The weight \( v_i \) equals the average inverse rank across \( m \) modalities:

\[ v_i = \frac{1}{m} \sum_{j=1}^{m} \frac{1}{\text{rank}(x_{ij})}. \]  

We conduct experiments with different weighting schemes and plot their MAPs in Fig. 6.9, where the x-axis denotes the iteration, and the y-axis is the MAP. The same step size is used in all methods. As we see, SPaR with the proposed soft weighting schemes, including linear, log and mixture weighting, outperforms the binary and the predefined weighting across iterations. Among them, the mixture weighting is slightly better than others, suggesting the rationale for tolerating small errors on this dataset. However, it needs an additional parameter to tune. The MAPs of the proposed soft weighting schemes seem to be robust and less sensitive to the iteration change. The MAP drop, in all reranking methods, seems to be related to the nature of the MED dataset as the similar pattern can be observed in other reranking methods. Nevertheless, SPaR still outperforms the binary, predefined weights and the baseline methods in Table 6.1.
6.5.5 Experiments on Hybrid Search

[TODO: add more experimental results]

6.5.6 Experiments on Web Query Dataset

To verify SPaR’s performance on image search, we conduct experiments on a web image query dataset consisting of 71,478 images from 353 queries, retrieved by a search engine named Exalead. For each query, the top ranked images generated by Exalead are provided, along with the true label for every image. The dataset is representative as the 353 queries cover a broad range of topics. The performance is evaluated by the non-interpolated MAP, as used in [90]. MAP@100 is also included for comparison. Note that as the initial result contains a single modality.

Following [91, 92], densely sampled SIFT are extracted. A codebook of 1,024 centroids is constructed. Spatial Tiling [121] is used to further improve the performance. We compare SPaR with the state-of-the-art reranking methods. SPaR is configured in a similar way as discussed in Section 6.5.1, and provided initial text-based search results are used. Following [91, 92], the parameters are tuned on a validation set consisting of a subset of 55 queries.

We examine the overall MAP in Table 6.2. “-” denotes that the number is unavailable in the cited paper. As we see, SPaR achieves the promising MAP among state-of-the-art reranking methods, including Graph-based [109], LETOR-based [91, 105], Classification-based [97] and even supervised reranking methods [90, 107], in terms of both MAP and MAP@100. A similar pattern can be observed that SPaR significantly boosts the MAP of plain retrieval without reranking, and obtain comparable or even better performance than the baseline methods. Generally, SPaR improves about 84% queries over the method without reranking. Since the initial ranked lists are retrieved by text matching, this result substantiates the claim that SPaR is general and applicable to reranking by text-based search.

6.5.7 Runtime Comparison

To empirically verify the efficiency of SPaR and MMPRF, we compare the runtime (second/query) in a single iteration. The experiments are conducted on Intel Xeon E5649 @ 2.53GHz with 16GB memory and the results are listed in Table 6.3. To test the speed of Rocchio and Relevance Model, we built our own inverted index on the Web Query dataset, and issue the query against the index. The reranking in MED, which
Table 6.2: MAP and MAP@100 comparison with baseline methods on the Web Query dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
<th>MAP@100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Reranking [90]</td>
<td>0.569</td>
<td>0.431</td>
</tr>
<tr>
<td>CPRF [97]</td>
<td>0.658</td>
<td>-</td>
</tr>
<tr>
<td>Random Walk [109]</td>
<td>0.616</td>
<td>-</td>
</tr>
<tr>
<td>Bayesian Reranking [91, 105]</td>
<td>0.658</td>
<td>0.529</td>
</tr>
<tr>
<td>Preference Learning Model [91]</td>
<td>-</td>
<td>0.534</td>
</tr>
<tr>
<td>BVLS [92]</td>
<td>0.670</td>
<td>-</td>
</tr>
<tr>
<td>Query-Relative(visual) [90]</td>
<td>0.649</td>
<td>-</td>
</tr>
<tr>
<td>Supervised Reranking [107]</td>
<td>0.665</td>
<td>-</td>
</tr>
<tr>
<td>SPaR</td>
<td>0.672</td>
<td>0.557</td>
</tr>
</tbody>
</table>

Table 6.3: Runtime Comparison in a single iteration.

<table>
<thead>
<tr>
<th>Method</th>
<th>MED</th>
<th>Web Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rocchio</td>
<td>5.3</td>
<td>2.0</td>
</tr>
<tr>
<td>Relevance Model</td>
<td>7.2</td>
<td>2.5</td>
</tr>
<tr>
<td>Learning to Rank</td>
<td>178</td>
<td>22.3</td>
</tr>
<tr>
<td>CPRF</td>
<td>145</td>
<td>10.1</td>
</tr>
<tr>
<td>MMPRF</td>
<td>149</td>
<td>10.1</td>
</tr>
<tr>
<td>SPaR</td>
<td>158</td>
<td>12.2</td>
</tr>
</tbody>
</table>

is conducted on semantic features, is slower because it involves multiple features and modalities. As we see, SPaR’s overhead over CPRF is marginal on the both sets. This result suggests SPaR and MMPRF is inexpensive. Note the implementations for all methods reported here are far from optimal, which involve a number of programming languages. We will report the runtime of the accelerated pipeline in Section 8.

6.6 Summary

In this chapter, we proposed two approaches for multimodal reranking, namely Multi-Modal Pseudo Relevance Feedback (MMPRF) and Self-Paced Reranking (SPaR). Unlike existing methods, the reranking is conducted using multiple ranked lists. In MMPRF, we formulated the pseudo label construction problem as maximum likelihood estimation and maximum expected value estimation problems, which can be solved by existing linear/integer programming algorithms. By training a joint model on the pseudo label set, MMPRF leverages low-level features and high-level features for multimedia event detection without any training data. SPaR reveals the link between reranking and an optimization problem that can be effectively solved by self-paced learning. The proposed SPaR is general, and can be used to theoretically explain other reranking methods including MMPRF. Experimental results validate the efficacy and the efficiency of the
proposed methods on several datasets. The proposed methods consistently outperforms
the plain retrieval without reranking, and obtains decent improvements over existing reranking methods.
Chapter 7

Building Semantic Concepts by Self-paced Curriculum Learning

7.1 Introduction

Concept detectors is the key in a CBVSR system as it not only affects what can be searched by semantic search but also determines the video representation in hybrid search. Concept detectors can be trained on still images or videos. The latter is more desirable due to the minimal domain difference and the capability for action and audio detection. In this chapter, we explore a semantic concept training method using self-paced curriculum learning. The theory has been used in the reranking method SPaR in Section 6.4. In this chapter, we will formally introduce the general form of the theory and discuss its application on semantic concept training. We approach this problem based on recently proposed theories called Curriculum learning [98] and self-paced learning [99]. The theories have been attracting increasing attention in the field of machine learning and artificial intelligence. Both the learning paradigms are inspired by the learning principle underlying the cognitive process of humans and animals, which generally start with learning easier aspects of a task, and then gradually take more complex examples into consideration. The intuition can be explained in analogous to human education in which a pupil is supposed to understand elementary algebra before he or she can learn more advanced algebra topics. This learning paradigm has been empirically demonstrated to be instrumental in avoiding bad local minima and in achieving a better generalization result [122–124].

A curriculum determines a sequence of training samples which essentially corresponds to a list of samples ranked in ascending order of learning difficulty. A major disparity between curriculum learning (CL) and self-paced learning (SPL) lies in the derivation of
the curriculum. In CL, the curriculum is assumed to be given by an oracle beforehand, and remains fixed thereafter. In SPL, the curriculum is dynamically generated by the learner itself, according to what the learner has already learned.

The advantage of CL includes the flexibility to incorporate prior knowledge from various sources. Its drawback stems from the fact that the curriculum design is determined independently of the subsequent learning, which may result in inconsistency between the fixed curriculum and the dynamically learned models. From the optimization perspective, since the learning proceeds iteratively, there is no guarantee that the predetermined curriculum can even lead to a converged solution. SPL, on the other hand, formulates the learning problem as a concise biconvex problem, where the curriculum design is embedded and jointly learned with model parameters. Therefore, the learned model is consistent. However, SPL is limited in incorporating prior knowledge into learning, rendering it prone to overfitting. Ignoring prior knowledge is less reasonable when reliable prior information is available. Since both methods have their advantages, it is difficult to judge which one is better in practice.

In this chapter, we discover the missing link between CL and SPL. We formally propose a unified framework called **Self-paced Curriculum Learning** (SPCL). SPCL represents a general learning paradigm that combines the merits from both the CL and SPL. On one hand, it inherits and further generalizes the theory of SPL. On the other hand, SPCL addresses the drawback of SPL by introducing a flexible way to incorporate prior knowledge. This chapter offers a compelling insight on the relationship between the existing CL and SPL methods. Their relation can be intuitively explained in the context of human education, in which SPCL represents an “instructor-student collaborative” learning paradigm, as opposed to “instructor-driven” in CL or “student-driven” in SPL. In SPCL, instructors provide prior knowledge on a weak learning sequence of samples, while leaving students the freedom to decide the actual curriculum according to their learning pace. Since an optimal curriculum for the instructor may not necessarily be optimal for all students, we hypothesize that given reasonable prior knowledge, the curriculum devised by instructors and students together can be expected to be better than the curriculum designed by either part alone.
7.2 Related Work

7.2.1 Curriculum Learning

Bengio et al. proposed a new learning paradigm called curriculum learning (CL), in which a model is learned by gradually including from easy to complex samples in training so as to increase the entropy of training samples [98]. Afterwards, Bengio and his colleagues presented insightful explorations for the rationality underlying this learning paradigm, and discussed the relationship between CL and conventional optimization techniques, e.g., the continuation and annealing methods [125, 126]. From human behavioral perspective, evidence have shown that CL is consistent with the principle in human teaching [122, 123].

The CL methodology has been applied to various applications, the key in which is to find a ranking function that assigns learning priorities to training samples. Given a training set \( D = \{(x_i, y_i)\}_{i=1}^{n} \), where \( x_i \) denotes the \( i^{th} \) observed sample, and \( y_i \) represents its label. A curriculum is characterized by a ranking function \( \gamma \). A sample with a higher rank, i.e., smaller value, is supposed to be learned earlier.

The curriculum (or the ranking function) is often derived by predetermined heuristics for particular problems. For example, in the task of classifying geometrical shapes, the ranking function was derived by the variability in shape [98]. The shapes exhibiting less variability are supposed to be learned earlier. In [122], the authors tried to teach a robot the concept of “graspability” - whether an object can be grasped and picked up with one hand, in which participants were asked to assign a learning sequence of graspability to various object. The ranking is determined by common sense of the participants. In [127], the authors approached grammar induction, where the ranking function is derived in terms of the length of a sentence. The heuristic is that the number of possible solutions grows exponentially with the length of the sentence, and short sentences are easier and thus should be learn earlier.

The heuristics in these problems turn out to be beneficial. However, the heuristical curriculum design may lead to inconsistency between the fixed curriculum and the dynamically learned models. That is, the curriculum is predetermined a priori and cannot be adjusted accordingly, taking into account the feedback about the learner.

7.2.2 Self-paced Learning

To alleviate the issue of CL, Koller’s group [99] designed a new formulation, called self-paced learning (SPL). SPL embeds curriculum design as a regularization term into the
learning objective. Compared with CL, SPL exhibits two advantages: first, it jointly optimizes the learning objective together with the curriculum, and therefore the curriculum and the learned model are consistent under the same optimization problem; second, the regularization term is independent of loss functions of specific problems. This theory has been successfully applied to various applications, such as action/event detection [70], reranking [14], domain adaption [124], dictionary learning [128], tracking [129] and segmentation [130].

Formally, let $L(y_i, g(x_i, w))$ denote the loss function which calculates the cost between the ground truth label $y_i$ and the estimated label $g(x_i, w)$. Here $w$ represents the model parameter inside the decision function $g$. In SPL, the goal is to jointly learn the model parameter $w$ and the latent weight variable $v = [v_1, \cdots, v_n]^T$ by minimizing:

$$
\min_{w, v \in [0,1]^n} E(w, v; \lambda) = \sum_{i=1}^{n} v_i L(y_i, g(x_i, w)) - \lambda \sum_{i=1}^{n} v_i, \tag{7.1}
$$

where $\lambda$ is a parameter for controlling the learning pace. Eq. (7.1) indicates the loss of a sample is discounted by a weight. The objective of SPL is to minimize the weighted training loss together with the negative $l_1$-norm regularizer $-\|v\|_1 = -\sum_{i=1}^{n} v_i$ (since $v_i \geq 0$). A more general regularizer consists of both $\|v\|_1$ and the sum of group-wise $\|v\|_2$ [70].

ACS (Alternative Convex Search) is generally used to solve Eq. (7.1) [112]. ACS is a special case of Cyclic Coordinate Method (CCM) [112] discussed in Section 6.4. It is an iterative method for biconvex optimization, in which the variables are divided into two disjoint blocks. In each iteration, a block of variables are optimized while keeping the other block fixed. With the fixed $w$, the global optimum $v^* = [v_1^*, \cdots, v_n^*]$ can be easily calculated by:

$$
v_i^* = \begin{cases} 
1, & L(y_i, g(x_i, w)) < \lambda, \\
0, & \text{otherwise}.
\end{cases} \tag{7.2}
$$

There exists an intuitive explanation behind this alternative search strategy: first, when updating $v$ with a fixed $w$, a sample whose loss is smaller than a certain threshold $\lambda$ is taken as an “easy” sample, and will be selected in training ($v_i^* = 1$), or otherwise unselected ($v_i^* = 0$); second, when updating $w$ with a fixed $v$, the classifier is trained only on the selected “easy” samples. The parameter $\lambda$ controls the pace at which the model learns new samples, and physically $\lambda$ corresponds to the “age” of the model. When $\lambda$ is small, only “easy” samples with small losses will be considered. As $\lambda$ grows, more samples with larger losses will be gradually appended to train a more “mature” model.
This strategy complies with the heuristics in most CL methods \cite{98, 122}. However, since the learning is completely dominated by the training loss, the learning may be prone to overfitting. Moreover, it provides no way to incorporate prior guidance in learning. To the best of our knowledge, there has been no studies to incorporate prior knowledge into SPL, nor to analyze the relation between CL and SPL.

7.2.3 Survey on Weakly Training

[add the survey of training concept detectors using web data]

7.3 Theory

7.3.1 Model

An ideal learning paradigm should consider both prior knowledge known before training and information learned during training in a unified and sound framework. Similar to human education, we are interested in constructing an “instructor-student collaborative” paradigm, which, on one hand, utilizes prior knowledge provided by instructors as a guidance for curriculum design (the underlying CL methodology), and, on the other hand, leaves students certain freedom to adjust to the actual curriculum according to their learning paces (the underlying SPL methodology). This requirement can be realized through the following optimization model. Similar in CL, we assume that the model is given a curriculum that is predetermined by an oracle. Following the notation defined above, we have:

\[
\min_{w, v} \mathbb{E}(w, v; \lambda, \Psi) = \sum_{i=1}^{n} v_i L(y_i, g(x_i, w)) + f(v; \lambda)
\]  
\text{s.t. } v \in \Psi  

(7.3)

where \( v = [v_1, v_2, \ldots, v_n]^T \) denote the weight variables reflecting the samples’ importance. \( f \) is called self-paced function which controls the learning scheme; \( \Psi \) is a feasible region that encodes the information of a predetermined curriculum. A curriculum can be mathematically described as:

**Definition 7.1** (Total order curriculum). For training samples \( X = \{x_i\}_{i=1}^{n} \), a total order curriculum, or curriculum for short, can be expressed as a ranking function:

\[
\gamma : X \rightarrow \{1, 2, \ldots, n\},
\]
where $\gamma(x_i) < \gamma(x_j)$ represents that $x_i$ should be learned earlier than $x_j$ in training, $\gamma(x_i) = \gamma(x_j)$ denotes there is no preferred learning order on the two samples.

**Definition 7.2 (Curriculum region).** Given a predetermined curriculum $\gamma(\cdot)$ on training samples $X = \{x_i\}_{i=1}^n$ and their weight variables $v = [v_1, \cdots, v_n]^T$. A feasible region $\Psi$ is called a curriculum region of $\gamma$ if

1. $\Psi$ is a nonempty convex set;
2. for any pair of samples $x_i, x_j$, if $\gamma(x_i) < \gamma(x_j)$, it holds that $\int_{\Psi} v_i \, dv > \int_{\Psi} v_j \, dv$, where $\int_{\Psi} v_i \, dv$ calculates the expectation of $v_i$ within $\Psi$. Similarly if $\gamma(x_i) = \gamma(x_j)$, $\int_{\Psi} v_i \, dv = \int_{\Psi} v_j \, dv$.

The two conditions in Definition 7.2 offer a realization for curriculum learning. Condition 1 ensures the soundness for calculating the constraints. Condition 2 indicates that samples to be learned earlier should have larger expected values. The curriculum region physically corresponds to a convex region in the high-dimensional space. The area inside this region confines the space for learning the weight variables. The shape of the region weakly implies a prior learning sequence of samples, where the expected values for favored samples are larger. For example, Figure 7.1(b) illustrates an example of feasible region in 3D where the $x, y, z$ axis represents the weight variable $v_1, v_2, v_3$, respectively. Without considering the learning objective, we can see that $v_1$ tends to be learned earlier than $v_2$ and $v_3$. This is because if we uniformly sample sufficient points in the feasible region of the coordinate $(v_1, v_2, v_3)$, the expected value of $v_1$ is larger. Since prior knowledge is missing in Eq. (7.1), the feasible region is a unit hypercube, i.e. all samples are equally favored, as shown in Figure 7.1(a). Note the curriculum region should be confined within the unit hypercube since the constraints $v \in [0, 1]^n$ in Eq. (7.3).

![Figure 7.1](image_url)

*Figure 7.1: Comparison of feasible regions in SPL and SPCL.*

Note that the prior learning sequence in the curriculum region only weakly affects the actual learning sequence, and it is very likely that the prior sequence will be adjusted by the learners. This is because the prior knowledge determines a weak ordering of samples that suggests what should be learned first. A learner takes this knowledge into account, but has his/her own freedom to alter the sequence in order to adjust to
Table 7.1: Comparison of different learning approaches.

<table>
<thead>
<tr>
<th></th>
<th>CL</th>
<th>SPL</th>
<th>Proposed SPCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comparable to human</td>
<td>Instructor-driven</td>
<td>Student-driven</td>
<td>Instructor-student</td>
</tr>
<tr>
<td>learning</td>
<td>Prior knowledge</td>
<td>Learning objective</td>
<td>collaborative</td>
</tr>
<tr>
<td>Curriculum design</td>
<td>Multiple</td>
<td>Single</td>
<td>Learning objective</td>
</tr>
<tr>
<td>Learning schemes</td>
<td>Heuristic approach</td>
<td>Gradient-based</td>
<td>+ prior knowledge</td>
</tr>
<tr>
<td>Iterative training</td>
<td></td>
<td></td>
<td>Gradient-based</td>
</tr>
</tbody>
</table>

the learning objective. See Example 7.1. Therefore, SPCL represents an “instructor-student-corporative” learning paradigm.

Compared with Eq. (7.1), SPCL generalizes SPL by introducing a regularization term. This term determines the learning scheme, i.e., the strategy used by the model to learn new samples. In human learning, we tend to use different schemes for different tasks. Similarly, SPCL should also be able to utilize different learning schemes for different problems. Since the existing methods only include a single learning scheme, we generalize the learning scheme and define:

Definition 7.3 (Self-paced function). A self-paced function determines a learning scheme. Suppose that $v = [v_1, \cdots, v_n]^T$ denotes a vector of weight variable for each training sample and $\ell = [\ell_1, \cdots, \ell_n]^T$ are the corresponding loss. $\lambda$ controls the learning pace (or model “age”). $f(v; \lambda)$ is called a self-paced function, if

1. $f(v; \lambda)$ is convex with respect to $v \in [0, 1]^n$.
2. When all variables are fixed except for $v_i$ and $\ell_i$, $v_i^*$ decreases with $\ell_i$, and it holds that $\lim_{\ell_i \to 0} v_i^* = 1, \lim_{\ell_i \to \infty} v_i^* = 0$.
3. $\|v\|_1 = \sum_{i=1}^n v_i$ increases with respect to $\lambda$, and it holds that $\forall i \in [1, n]$, $\lim_{\lambda \to 0} v_i^* = 0, \lim_{\lambda \to \infty} v_i^* = 1$.

where $v^* = \arg\min_{v \in [0,1]^n} \sum v_i\ell_i + f(v; \lambda)$, and denote $v^* = [v_1^*, \cdots, v_n^*]$.

The three conditions in Definition 7.3 provide a definition for the self-paced learning scheme. Condition 2 indicates that the model inclines to select easy samples (with smaller losses) in favor of complex samples (with larger losses). Condition 3 states that when the model “age” $\lambda$ gets larger, it should incorporate more, probably complex, samples to train a “mature” model. The convexity in Condition 1 ensures the model can find good solutions within the curriculum region.

It is easy to verify that the regularization term in Eq. (7.1) satisfies Definition 7.3. In fact, this term corresponds to a binary learning scheme since $v_i$ can only take binary values, as shown in the closed-form solution of Eq. (7.2). This scheme may be less appropriate in the problems where the importance of samples needs to be discriminated. In fact,
there exist a plethora of self-paced functions corresponding to various learning schemes. We will detail some of them in the next section.

Inspired by the algorithm in [99], we employ a similar ACS algorithm to solve Eq. (7.3). Algorithm 2 takes the input of a predetermined curriculum, an instantiated self-paced function and a stepsize parameter; it outputs an optimal model parameter $w$. First of all, it represents the input curriculum as a curriculum region that follows Definition 2, and initializes variables in their feasible region. Then it alternates between two steps until it finally converges: Step 4 learns the optimal model parameter with the fixed and most recent $v^*$; Step 5 learns the optimal weight variables with the fixed $w^*$. In first several iterations, the model “age” is increased so that more complex samples will be gradually incorporated in the training. For example, we can increase $\lambda$ so that $\mu$ more samples will be added in the next iteration. According to the conditions in Definition 7.3, the number of complex samples increases along with the growth of the number iteration. Step 4 can be conveniently implemented by existing off-the-shelf supervised learning methods. Gradient-based or interior-point methods can be used to solve the convex optimization problem in Step 5. According to [112], the alternative search in Algorithm 2 converges as the objective function is monotonically decreasing and is bounded from below.

**Algorithm 2: Self-paced Curriculum Learning.**

```plaintext
input: Input dataset $D$, predetermined curriculum $\gamma$, self-paced function $f$ and a stepsize $\mu$
output: Model parameter $w$

1. Derive the curriculum region $\Psi$ from $\gamma$;
2. Initialize $v^*$, $\lambda$ in the curriculum region;
3. while not converged do
   4. Update $w^* = \arg\min_w E(w, v^*; \lambda, \Psi)$;
   5. Update $v^* = \arg\min_v E(w^*, v; \lambda, \Psi)$;
   6. if $\lambda$ is small then increase $\lambda$ by the stepsize $\mu$;
4. end
8. return $w^*$
```

### 7.3.2 Relationship to CL and SPL

SPCL represents a general learning framework which includes CL and SPL as special cases. SPCL degenerates to SPL when the curriculum region is ignored ($\Psi = [0, 1]^n$), or equivalently, the prior knowledge on predefined curriculums is absent. In this case, the learning is totally driven by the learner. SPCL degenerates to CL when the curriculum region (feasible region) only contains the learning sequence in the predetermined curriculum. In this case, the learning process neglects the feedback about learners, and is dominated by the given prior knowledge. When information from both sources are
available, the learning in SPCL is collaboratively driven by prior knowledge and learning objective. Table 7.1 summarizes the characteristics of different learning methods. Given reasonable prior knowledge, SPCL which considers the information from both sources tend to yield better solutions. Example 7.1 shows a case in this regard.

7.3.3 Implementation

The definitions discussed above provide a theoretical foundation for SPCL. However, we still need concrete self-paced functions and curriculum regions to solve specific problems. To this end, this section discusses some implementations that follow Definition 7.2 and Definition 7.3. Note that there is no single implementation that can always work the best for all problems. Our purpose is to argument the implementations in the literature, and to help enlighten others to further explore this interesting direction.

Curriculum region implementation: We suggest an implementation induced from a linear constraint for realizing the curriculum region: \( a^T v \leq c \), where \( v = [v_1, \cdots, v_n]^T \) are the weight variables in Eq. (7.3), \( c \) is a constant, and \( a = [a_1, \cdots, a_n]^T \) is a \( n \)-dimensional vector. The linear constraints is a simple implementation for curriculum region that can be conveniently solved. It can be proved that this implementation complies with the definition of curriculum region.

**Theorem 7.4.** For training samples \( X = \{x_i\}_{i=1}^n \), given a curriculum \( \gamma \) defined on it, the feasible region, defined by,

\[
\Psi = \{v | a^T v \leq c\}
\]

is a curriculum region of \( \gamma \) if it holds: 1) \( \Psi \cap v \in [0, 1]^n \) is nonempty; 2) \( a_i < a_j \) for all \( \gamma(x_i) < \gamma(x_j) \); \( a_i = a_j \) for all \( \gamma(x_i) = \gamma(x_j) \).

Self-paced function implementation: Similar to the scheme human used to absorb knowledge, a self-paced function determines a learning scheme for the model to learn new samples. Note the self-paced function is realized as a regularization term, which is independent of specific loss functions, and can be easily applied to various problems. Since human tends to use different learning schemes for different tasks, SPCL should also be able to utilize different learning schemes for different problems. Inspired by a study in [14], this section discusses some examples of learning schemes.

**Binary scheme:** This scheme in is used in [99]. It is called binary scheme, or “hard” scheme, as it only yields binary weight variables.

\[
f(v; \lambda) = -\lambda ||v||_1 = -\lambda \sum_{i=1}^n v_i,
\]

(7.4)
**Linear scheme:** A common approach is to linearly discriminate samples with respect to their losses. This can be realized by the following self-paced function:

\[
f(v; \lambda) = \frac{1}{2} \lambda \sum_{i=1}^{n} (v_i^2 - 2v_i),
\]

in which \( \lambda > 0 \). This scheme represents a “soft” scheme as the weight variable can take real values.

**Logarithmic scheme:** A more conservative approach is to penalize the loss logarithmically, which can be achieved by the following function:

\[
f(v; \lambda) = \sum_{i=1}^{n} \zeta v_i - \frac{\zeta v_i}{\log \zeta},
\]

where \( \zeta = 1 - \lambda \) and \( 0 < \lambda < 1 \).

**Mixture scheme:** Mixture scheme is a hybrid of the “soft” and the “hard” scheme [14]. If the loss is either too small or too large, the “hard” scheme is applied. Otherwise, the soft scheme is applied. Compared with the “soft” scheme, the mixture scheme tolerates small errors up to a certain point. To define this starting point, an additional parameter is introduced, i.e. \( \lambda = [\lambda_1, \lambda_2]^T \). Formally,

\[
f(v; \lambda) = -\zeta \sum_{i=1}^{n} \log(v_i + \frac{1}{\lambda_1} \zeta),
\]

where \( \zeta = \frac{\lambda_1 \lambda_2}{\lambda_1 - \lambda_2} \) and \( \lambda_1 > \lambda_2 > 0 \).

**Theorem 7.5.** The binary, linear, logarithmic and mixture scheme function are self-paced functions.

It can be proved that the above functions follow Definition 7.3. The name of the learning scheme suggests the characteristic of its solution. The curve in Fig. 6.4 illustrates the characteristics of the learning schemes. When the curriculum region is not a unit hypercube, the closed-form solution, such as Eq. (7.2) cannot be directly used. Gradient-based methods can be applied. As \( \mathbb{E}_w \) is convex, the local optimal is also the global optimal solution for the subproblem.

**Example 7.1.** Given six samples \( a, b, c, d, e, f \). In the current iteration, the losses for these samples are \( \ell = [0.1, 0.2, 0.4, 0.6, 0.5, 0.3] \), respectively. A latent ground-truth curriculum is listed in the first row of the following table, followed by the curriculum of CL, SPL and SPCL. For simplicity, binary scheme is used in SPL and SPCL where \( \lambda = 0.8333 \). If two samples with the same weight, we rank them in ascending order of their losses, in order to break the tie. The Kendall’s rank correlation is presented in the last column.
Building Semantic Concepts by Self-paced Curriculum Learning

<table>
<thead>
<tr>
<th>Method</th>
<th>Curriculum</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground-Truth</td>
<td>a, b, c, d, e, f</td>
<td>-</td>
</tr>
<tr>
<td>CL</td>
<td>b, a, d, e, c, f</td>
<td>0.73</td>
</tr>
<tr>
<td>SPL</td>
<td>b, a, f, c, e, d</td>
<td>0.46</td>
</tr>
<tr>
<td>SPCL</td>
<td>a, b, c, d, e, f</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The curriculum region used is a linear constraint $\mathbf{a}^T \mathbf{v} \leq 1$, where $\mathbf{a} = [0.1, 0.0, 0.4, 0.3, 0.5, 1.0]^T$. In the implementation, we add a small constant $10^{-7}$ in the constraints for optimization accuracy. The constraint follows Definition 2 in the paper. As shown, both CL and SPL yield the suboptimal curriculum, e.g. their correlations are only 0.73 and 0.46. However, SPCL exploits the complementary information in CL and SPL, and devises an optimal curriculum. Note that CL recommends to learn b before a, but SPCL disobeys this order in the actual curriculum. The final solution of SPCL is $\mathbf{v}^* = [1.00, 1.00, 1.00, 0.88, 0.47, 0.00]$.

When the predetermined curriculum is completely wrong, SPCL may still be robust to the inferior prior knowledge given reasonable curriculum regions are applied. In this case, the prior knowledge should not be encoded as strong constraints. For example, in the above example, we can use the following curriculum region to encode the completely incorrect predetermined curriculum: $\mathbf{a}^T \mathbf{v} \leq 6.0$, where $\mathbf{a} = [2.3, 2.2, 2.1, 2.0, 1.7, 1.5]^T$.

<table>
<thead>
<tr>
<th>Method</th>
<th>Curriculum</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CL</td>
<td>f, e, d, c, b, a</td>
<td>-1.00</td>
</tr>
<tr>
<td>SPL</td>
<td>a, b, f, c, e, d</td>
<td>0.46</td>
</tr>
<tr>
<td>SPCL</td>
<td>a, f, b, c, e, d</td>
<td>0.33</td>
</tr>
</tbody>
</table>

As we see, even though the predetermined curriculum is completely wrong (correlation -1.00), the proposed SPCL still obtains reasonable curriculum (correlation 0.33). This is because SPCL is able to leverage information in both prior knowledge and learning objective. The optimal solution of SPCL is $\mathbf{v}^* = [1.00, 0.91, 0.10, 0.00, 0.00, 1.00]$.

In the above learning schemes, samples in a curriculum are selected solely in terms of “easiness”. In this section, we reveal that diversity, an important aspect in learning, should also be considered. Ideal learning should utilize not only easy but also diverse examples that are sufficiently dissimilar from what has already been learned. This can be intuitively explained in the context of human education. A rational curriculum for a pupil not only needs to include examples of suitable easiness matching her learning pace, but also, importantly, should include some diverse examples on the subject in order for her to develop more comprehensive knowledge. Likewise, learning from easy and diverse samples is expected to be better than learning from either criterion alone. To this end, we propose the following learning scheme.
Diverse learning scheme: Diversity implies that the selected samples should be less similar or clustered. An intuitive approach for realizing this is by selecting samples of different groups scattered in the sample space. We assume that the correlation of samples between groups is less than that of within a group. This auxiliary group membership is either given, e.g. in object recognition frames from the same video can be regarded from the same group, or can be obtained by clustering samples.

This aim can be mathematically described as follows. Assume that the training samples \(X = \{(x_1, \cdots, x_n) \in \mathbb{R}^{m \times n}\}^b\) are partitioned into \(b\) groups: \(X^{(1)}, \cdots, X^{(b)}\), where columns of \(X^{(j)} \in \mathbb{R}^{m \times n_j}\) correspond to the samples in the \(j^{th}\) group, \(n_j\) is the sample number in the group and \(\sum_{j=1}^{b} n_j = n\). Accordingly denote the weight vector as \(v = [v^{(1)}, \cdots, v^{(b)}]\), where \(v^{(j)} = (v^{(j)}_1, \cdots, v^{(j)}_{n_j})^T \in [0, 1]^{n_j}\). The diverse learning scheme on one hand needs to assign nonzero weights of \(v\) to easy samples as the hard learning scheme, and on the other hand requires to disperse nonzero elements across possibly more groups \(v^{(i)}\) to increase the diversity. Both requirements can be uniformly realized through the following optimization model:

\[
\min_{w, v} E(w, v; \lambda, \gamma) = \sum_{i=1}^{n} v_i L(y_i, f(x_i, w)) - \lambda \sum_{i=1}^{n} v_i - \gamma \|v\|_{2,1}, \quad \text{s.t.} \; v \in [0, 1]^n, \quad (7.8)
\]

where \(\lambda, \gamma\) are the parameters imposed on the easiness term (the negative \(l_1\)-norm: \(-\|v\|_1\)) and the diversity term (the negative \(l_{2,1}\)-norm: \(-\|v\|_{2,1}\)), respectively. As for the diversity term, we have:

\[
-\|v\|_{2,1} = -\sum_{j=1}^{b} \|v^{(j)}\|_2. \quad (7.9)
\]

The new regularization term consists of two components. One is the negative \(l_1\)-norm inherited from the hard learning scheme in SPL, which favors selecting easy over complex examples. The other is the proposed negative \(l_{2,1}\)-norm, which favors selecting diverse samples residing in more groups. It is well known that the \(l_{2,1}\)-norm leads to the group-wise sparse representation of \(v\) [64], i.e. non-zero entries of \(v\) tend to be concentrated in a small number of groups. Contrariwise, the negative \(l_{2,1}\)-norm should have a counter-effect to group-wise sparsity, i.e. nonzero entries of \(v\) tend to be scattered across a large number of groups. In other words, this anti-group-sparsity representation is expected to realize the desired diversity. Note that when each group only contains a single sample, Eq. (D.16) degenerates to Eq. (7.1).

Unlike the convex regularization term above, the term in diverse learning scheme is non-convex. A challenge is optimizing \(v\) with a fixed \(w\) becoming a non-convex problem. To this end, we propose a simple yet effective algorithm for extracting the global optimum of this problem when the curriculum is as in SPL, i.e. \(\Psi = [0, 1]^n\). Algorithm 3 takes
as input the groups of samples, the up-to-date model parameter $w$, and two self-paced parameters, and outputs the optimal $v$ of $\min_v E(w, v; \lambda, \gamma)$. The global minimum is proved in Appendix D:

**Theorem 7.6.** Algorithm 3 attains the global optimum to $\min_v E(w, v)$ for any given $w$ in linearithmic time.

Algorithm 3: Algorithm for Solving $\min_v E(w, v; \lambda, \gamma)$.

```
input : Input dataset $D$, groups $X^{(1)}, \ldots, X^{(b)}$, $w$, $\lambda$, $\gamma$
output: The global solution $v = (v^{(1)}, \ldots, v^{(b)})$ of $\min_v E(w, v; \lambda, \gamma)$.
for $j = 1$ to $b$ do // for each group
    Sort the samples in $X^{(j)}$ as $(x^{(j)}_1, \ldots, x^{(j)}_{n_j})$ in ascending order of their loss values $L$;
    Accordingly, denote the labels and weights of $X^{(j)}$ as $(y^{(j)}_1, \ldots, y^{(j)}_{n_j})$ and $(v^{(j)}_1, \ldots, v^{(j)}_{n_j})$;
    for $i = 1$ to $n_j$ do // easy samples first
        if $L(y^{(j)}_i, f(x^{(j)}_i, w)) < \lambda + \gamma \sqrt{\frac{1}{\sqrt{i+\sqrt{i-1}}}}$ then $v^{(j)}_i = 1$ // select this sample
        else $v^{(j)}_i = 0$; // not select this sample
    end
return $v$
```

As shown, Algorithm 3 selects samples in terms of both the easiness and the diversity. Specifically:

- Samples with $L(y_i, f(x_i, w)) < \lambda$ will be selected in training ($v_i = 1$) in Step 5. These samples represent the “easy” examples with small losses.

- Samples with $L(y_i, f(x_i, w)) > \lambda + \gamma$ will not be selected in training ($v_i = 0$) in Step 6. These samples represent the “complex” examples with larger losses.

- Other samples will be selected by comparing their losses to a threshold $\lambda + \gamma \sqrt{\frac{1}{\sqrt{i+\sqrt{i-1}}}}$, where $i$ is the sample’s rank w.r.t. its loss value within its group. The sample with a smaller loss than the threshold will be selected in training. Since the threshold decreases considerably as the rank $i$ grows, Step 5 penalizes samples monotonously selected from the same group.

**Example 7.2.** We study a tractable example that allows for clearer diagnosis in Fig. 7.2, where each keyframe represents a video sample on the event “Rock Climbing” of the TRECVID MED data [4], and the number below indicates its loss. The samples are clustered into four groups based on the visual similarity. A colored block on the right shows a curriculum selected by Algorithm 3. When $\gamma = 0$, as shown in Fig. 7.2(a), SPLD, which is identical to SPL, selects only easy samples (with the smallest losses) from a single cluster. Its curriculum thus includes duplicate samples like $b, c, d$ with the same loss value. When $\lambda \neq 0$ and $\gamma \neq 0$ in Fig. 7.2(b), SPLD balances the easiness and the diversity, and produces a reasonable and diverse curriculum: $a, j, g, b$. Note that
even if there exist 3 duplicate samples b, c, d, SPLD only selects one of them due to the decreasing threshold in Step 5 of Algorithm 3. Likewise, samples e and j share the same loss, but only j is selected as it is better in increasing the diversity. In an extreme case where $\lambda = 0$ and $\gamma \neq 0$, as illustrated in Fig. 7.2(c), SPLD selects only diverse samples, and thus may choose outliers, such as the sample n which is a confusable video about a bear climbing a rock. Therefore, considering both easiness and diversity seems to be more reasonable than considering either one alone. Physically the parameters $\lambda$ and $\gamma$ together correspond to the “age” of the model, where $\lambda$ focuses on easiness whereas $\gamma$ stresses diversity.

7.3.4 Limitations and Practical Observations

We observed a number of limitations of the current SPCL model. First, the fundamental learning philosophy of SPL/CL/SPCL is 1) learning needs to be conducted iteratively using samples organized in a meaningful sequence; 2) models are becoming more complex in each iteration. However, the learning philosophy may not applicable to every learning problem. For example, in many problems where training data, especially small training data, are are carefully selected and the spectrum of learning difficulty of training samples is controlled. We found the proposed theory may not outperform the conventional training methods. Second, the performance of SPCL can be unstable to the random starting values. This phenomenon can be intuitively explained in the context of education, it is impossible for students to predetermine what to learn before they actually learning anything. To address this, the curriculum needs to be meaningful so that it can provide some supervision in the first few iterations. However, precisely deriving curriculum region from prior knowledge seems to be an open question. Third, the age parameters $\lambda$ are very important hyperparameters to tune. In order to tune the
parameters, the proposed theory requires a labeled validation set that follows the same underlying distribution of the test set. Intuitively, it is analogous to the mock exam whose purposes are to let students realize how well they would perform on the real test data, and more importantly have a better idea of what to study.

In implementation, we found some engineering tricks to apply the theory to real-world problems. First, the parameters $\lambda$ (and $\gamma$ in the diverse learning scheme) should be tuned by the statistics collected from the ranked samples, as opposed to the absolute values. For example, instead of setting $\lambda$ to an absolute value, we rank samples by their loss in increasing order, then set $\lambda$ as the loss of the top $n$th sample. As a result, the top $n - 1$ samples will be used in training. The $n$th sample will have 0 weights and will not be used in training, and so does the samples ranked after it. In the next iteration we may increase $\lambda$ to be the loss of the top $1.5n$ sample. This strategy avoids selecting too many or too few samples at a single iteration and seems to be robust. Second, for unbalanced datasets, two sets of parameter $\lambda$ were introduced: $\lambda_+$ for positive and $\lambda_-$ for negative samples in order to pace positive and negative separately. This trick lead to a balance training data set in each iteration. Third, for the convex loss function $L$ in the off-the-shelf model, if we use the same training sets, we will end up with the same model, irrespective of iterative steps. In this case, at each iteration, we should test our model on the validation set, and determine when to terminate the training process. The converged model on a subset of training samples may perform better than the model trained on the whole training set. For example, Lapedriza et al. found training detectors using a subset samples can yield better results. For non-convex loss function in the off-the-shelf model, the sequential steps affect the final model. Therefore, the early stopping is not necessary.

7.4 Experiments using Diverse Scheme

We name SPCL with diverse learning scheme SPLD. We present experimental results on two tasks: event detection and action recognition. We demonstrate that our approach outperforms SPL on three real-world challenging datasets.

SPLD is compared against four baseline methods: 1) RandomForest is a robust bootstrap method that trains multiple decision trees using randomly selected samples and features [131]. 2) AdaBoost is a classical ensemble approach that combines the sequentially trained “base” classifiers in a weighted fashion [132]. Samples that are misclassified by one base classifier are given greater weight when used to train the next classifier in sequence. 3) BatchTrain represents a standard training approach in which a model is trained simultaneously using all samples; 4) SPL is a state-of-the-art method that
trains models gradually from easy to more complex samples [99]. The baseline methods are a mixture of the well-known and the state-of-the-art methods on training models using sampled data.

### 7.4.1 Event Detection

Given a collection of videos, the goal of MED is to detect events of interest, e.g. “Birthday Party” and “Parade”, solely based on the video content. The task is very challenging due to complex scenes, camera motion, occlusions, etc. The experiments are conducted on the largest collection on event detection: TRECVID MED13Test, which consists of about 32,000 Internet videos. There are a total of 3,490 videos from 20 complex events, and the rest are background videos. For each event, 10 positive examples are given to train a detector, which is tested on about 25,000 videos. The official test split released by NIST (National Institute of Standards and Technology) is used. A Deep Convolutional Neural Network is trained on 1.2 million ImageNet challenge images from 1,000 classes [75] to represent each video as a 1,000-dimensional vector. Algorithm 3 is used. By default, the group membership is generated by the spectral clustering, and the number of groups is set to 64. Following [124], LibLinear is used as the solver in Step 4 of Algorithm 3 due to its robust performance on this task. The performance is evaluated using MAP as recommended by NIST. The parameters of all methods are tuned on the same validation set.

Table 7.2 lists the overall MAP comparison. To reduce the influence brought by initialization, we repeated experiments of SPL and SPLD 10 times with random starting values, and report the best run and the mean (with the 95% confidence interval) of the 10 runs. The proposed SPLD outperforms all baseline methods with statistically significant differences at the $p$-value level of 0.05, according to the paired t-test. It is worth emphasizing that MED is very challenging and 26% relative (2.5 absolute) improvement over SPL is a notable gain. SPLD outperforms other baselines on both the best run and the 10 runs average. RandomForest and AdaBoost yield poorer performance. This observation agrees with the study in literature [4] that SVM is more robust on event detection.

<table>
<thead>
<tr>
<th>Run Name</th>
<th>RandomForest</th>
<th>AdaBoost</th>
<th>BatchTrain</th>
<th>SPL</th>
<th>SPLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Run</td>
<td>3.0</td>
<td>2.8</td>
<td>8.3</td>
<td>9.6</td>
<td>12.1</td>
</tr>
<tr>
<td>10 Runs Average</td>
<td>3.0</td>
<td>2.8</td>
<td>8.3</td>
<td>8.6±0.42</td>
<td>9.8±0.45</td>
</tr>
</tbody>
</table>

BatchTrain, SPL and SPLD are all performed using SVM. Regarding the best run, SPL boosts the MAP of the BatchTrain by a relative 15.6% (absolute 1.3%). SPLD yields another 26% (absolute 2.5%) over SPL. The MAP gain suggests that optimizing
Figure 7.3: The validation and test AP in different iterations. Top row plots the SPL result and bottom shows the proposed SPLD result. The x-axis represents the iteration in training. The blue solid curve (Dev AP) denotes the AP on the validation set, the red one marked by squares (Test AP) denotes the AP on the test set, and the green dashed curve denotes the Test AP of BatchTrain which remains the same across iterations.

objectives with the diversity is inclined to attain a better solution. Fig. 7.3 plots the validation and test AP on three representative events. As illustrated, SPLD attains a better solution within fewer iterations than SPL, e.g. in Fig. 7.3(a) SPLD obtains the best test AP (0.14) by 6 iterations as opposed to AP (0.12) by 11 iterations in SPL. Studies have shown that SPL converges fast, while this observation further suggests that SPLD may lead to an even faster convergence. We hypothesize that it is because the diverse samples learned in the early iterations in SPLD tend to be more informative. The best Test APs of both SPL and SPLD are better than BatchTrain, which is consistent with the observation in [133] that removing some samples may be beneficial in training a better detector. As shown, Dev AP and Test AP share a similar pattern justifying the rationale for parameters tuning on the validation set.

Fig. 7.4 plots the curriculum generated by SPL and SPLD in a first few iterations on two representative events. As we see, SPL tends to select easy samples similar to what it has already learned, whereas SPLD selects samples that are both easy and diverse to the model. For example, for the event “E006 Birthday Party”, SPL keeps selecting indoor scenes due to the sample learned in the first place. However, the samples learned by SPLD are a mixture of indoor and outdoor birthday parties. For the complex samples, both methods leave them to the last iterations, e.g. the 10th video in “E007”.

7.4.2 Action Recognition

The goal is to recognize human actions in videos. Two representative datasets are used: Hollywood2 was collected from 69 different Hollywood movies [41]. It contains 1,707 videos belonging to 12 actions, splitting into a training set (823 videos) and a test set (884 videos). Olympic Sports consists of athletes practicing different sports collected from YouTube [40]. There are 16 sports actions from 783 clips. We use 649 for training...
and 134 for testing as recommended in [40]. The improved dense trajectory feature is extracted and further represented by the fisher vector [16, 134]. A similar setting discussed in Section 7.4.1 is applied, except that the groups are generated by K-means (K=128).

Table 7.3: MAP (x100) comparison with the baseline methods on Hollywood2 and Olympic Sports.

<table>
<thead>
<tr>
<th>Run Name</th>
<th>RandomForest</th>
<th>AdaBoost</th>
<th>BatchTrain</th>
<th>SPL</th>
<th>SPLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hollywood2</td>
<td>28.20</td>
<td>41.14</td>
<td>58.16</td>
<td>63.72</td>
<td>66.65</td>
</tr>
<tr>
<td>Olympic Sports</td>
<td>63.32</td>
<td>69.25</td>
<td>90.61</td>
<td>90.83</td>
<td>93.11</td>
</tr>
</tbody>
</table>

Table 7.3 lists the MAP comparison on the two datasets. A similar pattern can be observed that SPLD outperforms SPL and other baseline methods with statistically significant differences. We then compare our MAP with the state-of-the-art MAP in Table 7.4. Indeed, this comparison may be less fair since the features are different in different methods. Nevertheless, with the help of SPLD, we are able to achieve the best MAP reported so far on both datasets. Note that the MAPs in Table 7.4 are obtained by recent and very competitive methods on action recognition. This improvement confirms the assumption that considering diversity in learning is instrumental.

Table 7.4: Comparison of SPLD to the state-of-the-art on Hollywood2 and Olympic Sports

<table>
<thead>
<tr>
<th></th>
<th>Hollywood2</th>
<th>Olympic Sports</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vig et al. 2012 [135]</td>
<td>59.4%</td>
<td>BRENDEL ET AL. 2011 [136] 73.7%</td>
</tr>
<tr>
<td>Jiang et al. 2012 [137]</td>
<td>59.5%</td>
<td>JIANG ET AL. 2012 [137] 80.6%</td>
</tr>
<tr>
<td>Jain et al. 2013 [39]</td>
<td>62.5%</td>
<td>GAIDON ET AL. 2012 [138] 82.7%</td>
</tr>
<tr>
<td>Wang et al. 2013 [16]</td>
<td>64.3%</td>
<td>WANG ET AL. 2013 [16] 91.2%</td>
</tr>
<tr>
<td><strong>SPLD</strong></td>
<td><strong>66.7%</strong></td>
<td><strong>SPLD</strong> 93.1%</td>
</tr>
</tbody>
</table>

7.5 Experiments with Noisy Data

[proposed work. Add more experiments]
7.6 Summary

We proposed a novel learning regime called self-paced curriculum learning (SPCL), which imitates the learning regime of humans/animals that gradually involves from easy to more complex training samples into the learning process. The proposed SPCL can exploit both prior knowledge before training and dynamical information extracted during training. The novel regime is analogous to an “instructor-student-collaborative” learning mode, as opposed to “instructor-driven” in curriculum learning or “student-driven” in self-paced learning. We presented compelling understandings for curriculum learning and self-paced learning, and revealed that they can be unified into a concise optimization model. We discussed several concrete implementations in the proposed SPCL framework.

SPCL is a general learning framework, the component of which is of physical interpretation. The off-the-shelf models, such as SVMs, deep neural networks, and regression models, correspond to students. The self-paced functions correspond to learning schemes used by students to solve specific problems. The curriculum region corresponds to the prior knowledge provided from an oracle or an instructor so that learning can be processed in a desired direction.
Chapter 8

Conclusions and Proposed Work

In this thesis, we studied a fundamental research problem of searching semantic information in video content at a very large scale. We proposed several novel methods focusing on improving accuracy, efficiency and scalability in the novel search paradigm. The proposed methods demonstrated promising results on web-scale semantic search for video. The extensive experiments demonstrated that the methods are able to surpass state-of-the-art accuracy on multiple datasets. In addition, our method can efficiently scale up the search to hundreds of millions videos, and only takes about 0.2 second to search a semantic query on a collection of 100 million videos, 1 second to process a hybrid query over 1 million videos. There are two research issues to be addressed. In this section, we will summarize the tasks we plan to complete.

8.1 Evaluation of Final System

8.2 Proposed Tasks

In this section, we aggregate all the tasks we propose to do from each section of the thesis proposal.

8.2.1 Hybrid Search

Semantic and hybrid queries are handled by two methods in the current approach. The method for semantic queries is scalable but the method for hybrid queries is not. We extrapolate there exists a fundamental method that can unify the two methods and provides a scalable solution for hybrid queries.
Appendix A

An Appendix
Appendix B

Terminology

[todo: define frequently used definition here]

**Semantic features** are human interpretable multimodal features occurring in the video content, including speech, people, objects, scenes, actions, visual text. Semantic visual concepts, semantic audio concepts, ASR and OCR are four types of semantic features considered in this thesis.

**high-level features** is an alias of semantic features.
Appendix C

Evaluation Metrics

[todo: define the following metrics] RMSE

MAP

MAP@N

MRR

Precision

Precision@N
Appendix D

Proof

Theorem 3.4: the thresholding and the top-k thresholding results are optimal solutions of Eq. (3.1) in special cases.

Proof. Suppose $v \in [0, 1]^m$ represents the adjusted $m$-dimensional representation, and $d \in [0, 1]^m$ represents the vector $f_p(D)$. Define the regularization term $g(v; \alpha, \beta)$ as:

$$g(v; \alpha, \beta) = \frac{1}{2} \beta^2 \|v\|_0,$$

and denote the objective function value of Eq. (3.1) in the paper as $E(v, \beta)$. Then the optimization problem without constraint is reformulated as:

$$\min_v E(v, \beta) = \frac{1}{2} \|v - d\|_2^2 + \frac{1}{2} \beta^2 \|v\|_0$$

and the objective function value of Eq. (3.1) in the paper as $E(v, \beta)$. Thus the minimum of Eq. (D.1) is obtained at the minimal value of the first term, where $v_i^* = d_i$, and the corresponding optimal objective value is $E(v_i^*, \beta) = \frac{1}{2} \beta^2$.

For $v = 0$, we have that $E(v, \beta) = \frac{1}{2} d_i^2$.\[93\]
It is then easy to deduce that the optimum of Eq. (D.1) can be calculated at:

\[ v_i^* = \begin{cases} 
  d_i, & d_i \geq \beta \\
  0, & \text{otherwise},
\end{cases} \]

and thus the solution of the original problem is

\[ v^* = [v_1^*, v_2^*, \ldots, v_m^*]^T. \]

Denote the \( k \)th largest component of \( d \) is \( \tilde{d}_k \). Then if we want to get the \( k \)-sparse solution of Eq. (3.1), we just need to set that \( \tilde{d}_k \leq \beta < \tilde{d}_{k+1} \). Then the solution of the problem keeps top-\( k \) largest elements of \( d \) while thresholds others to 0. This corresponds the thresholding and the top-\( k \) thresholding results. The proof is completed.

**Theorem 3.3:** the optimal solutions of Eq. (3.1) (before or after normalization) is logically consistent with its given HEX graph.

**Proof.** Suppose \( v \in [0, 1]^m \) represents the adjusted \( m \)-dimensional representation, and \( G = (N, E_h, E_e) \) represents the given HEX graph. For any concept \((n_i, n_j) \in E_h\), according to Algorithm 1, we have \( v_i \geq v_j \). Therefore \( \forall n_k, n_j \in V, n_k \in \alpha(n_j), \) we have \( v_k \geq v_j \), where \( \alpha(n_j) \) is a set of ancestor of \( n_j \) in \( G_h \). This means condition 1 in Definition 3.2 is satisfied.

Suppose \( \exists n_p \in \bar{\alpha}(n_i), \exists n_q \in \bar{\alpha}(n_j) \) s.t. \((n_p, n_q) \in E_e \). According to Algorithm 1, the constraints ensure that if \((n_p, n_q) \in E_h, v_pv_q = 0\), which breaks down into three cases: 1) \( v_p \neq 0, v_q = 0 \), 2) \( v_p = 0, v_q \neq 0 \), and 3) \( v_p = v_q = 0 \). According to the condition 1 in Definition 3.2, we have \( v_p \geq v_i \) and \( v_q \geq v_j \), so for case 1) we have \( v_j \leq v_q = 0 \); for case 2) \( v_i \leq v_p = 0 \); for case 3) \( v_i = 0 \) and \( v_j = 0 \). Therefore in all cases, either \( v_i \) or \( v_j \) is nonzero. When \( i = p \) and \( j = q \), it trivially holds that \( v_pv_q = v_iv_j = 0 \). This means condition 2 in Definition 3.2 is satisfied.

According to Definition 3.2, \( v \) satisfies the two conditions and thus is logically consistent. The normalization method discussed in the paper does not change nonzero scores to zero or vice versa. If \( v \) is consistent with the exclusion relation in the given HEX graph (condition 2), then after normalization it is still consistent. The normalization method multiples a constant factor to each dimension of \( v \) so it would not change the ranking order of the concepts. Therefore the normalization also preserves the hierarchical relation (condition 1). The proof is completed. \( \square \)
Lemma 6.1: for the self-paced functions in Section 6.4.2, the proposed method finds the optimal solution for Eq. (6.13).

Proof. Consider the objective of Eq. (6.13). Suppose \( y^* = [y_1^*, \ldots, y_n^*]^T \) is a solution found by the gradient descent method in Section 6.4.2. According to Eq. (6.14), for \( y_i \in \{-1, +1\} \) and \( \forall v_i \in [0, 1] \), we have:

\[
E_{\Theta}(y_i^*, v_i; k) \leq E_{\Theta}(y_i, v_i; k). \tag{D.2}
\]

Therefore \( \forall y, \forall v \), the following inequations hold:

\[
E_{\Theta}(y^*, v; k) = \sum_{i=1}^{n} E_{\Theta}(y_i^*, v_i; k) \leq \sum_{i=1}^{n} E_{\Theta}(y_i, v_i; k) = E_{\Theta}(y, v; k). \tag{D.3}
\]

In other words, \( y^* \) found by Eq. (6.14) is the global optimum for Eq. (6.13). Now consider the objective with the fixed \( y^* \). The functions in Section 6.4.2, \( f(v) \) are convex functions of \( v \), so Eq. (6.13) is a convex function of \( v \). Suppose that \( v^* \) is a solution found by gradient descent, due to the convexity, \( v^* \) is the global optimum for Eq. (6.13).

Therefore, \( y^*, v^* \) is the global optimal solution for Eq. (6.13).

Theorem 6.2: the algorithm in Fig. 6.2 converges to a stationary solution for any fixed \( C \) and \( k \).

Proof. Let the superscript index the variable value in that iteration, e.g. \( v^{(t)} \) represents the value of \( v \) in the \( t \)th iteration. Denote \( \Theta^{(t)} = \Theta^{(t)}_1, \ldots, \Theta^{(t)}_m \). \( y^{(0)} \) and \( v^{(0)} \) are arbitrary initial values in their feasible regions. As Eq. (6.12) is a quadratic programming problem, the solution \( \Theta^{(t)} \) is the global optimum for \( E_{y,v} \), i.e.

\[
E(\Theta^{(t)}, y^{(t-1)}, v^{(t-1)}) \leq E(\Theta^{(t-1)}, y^{(t-1)}, v^{(t-1)}). \tag{D.4}
\]

According to Lemma 6.1, \( v, y \) are also global optimum for \( E_{\Theta} \), i.e.

\[
E(\Theta^{(t)}, y^{(t)}, v^{(t)}) \leq E(\Theta^{(t)}, y^{(t-1)}, v^{(t-1)}). \tag{D.5}
\]

Substitute Eq. (D.5) back into Eq. (D.4), we have that \( \forall t \geq 1, \)

\[
E(\Theta^{(t)}, y^{(t)}, v^{(t)}) \leq E(\Theta^{(t-1)}, y^{(t-1)}, v^{(t-1)}). \tag{D.6}
\]

Eq. (D.6) indicates that the objective decreases in every iteration. Since the objective \( E \) is the sum of finite elements, it is bounded from below. Consequently, according to [139], it is guaranteed that Alg. 6.2 (an instance of CCM algorithm) converges to a stationary solution of the problem.
**Theorem 7.4:** For training samples $X = \{x_i\}_{i=1}^n$, given a curriculum $\gamma$ defined on it, the feasible region, defined by,

$$\Psi = \{v | a^T v \leq c\} \quad (D.7)$$

is a curriculum region of $\gamma$ if it holds: 1) $\Psi \land v \in [0, 1]^n$ is nonempty; 2) $a_i < a_j$ for all $\gamma(x_i) < \gamma(x_j)$; $a_i = a_j$ for all $\gamma(x_i) = \gamma(x_j)$.

**Proof.** (1) $\Psi \land v \in [0, 1]^n$ is a nonempty convex set.

(2) For $x_i, x_j$ with $\gamma(x_i) < \gamma(x_j)$, denote $\Psi_{ij} = \{v_{ij} | a_{ij}^T v_{ij} \leq c\}$, $a_{ij}/v_{ij}$ the sub-vector of $a/v$ by wiping off its ith and jth elements, respectively, we can then calculate the expected value of $v_i$ on the region $\Psi = \{v | a^T v \leq c\}$ as:

$$E(v_i) = \int_{\Psi} v_i d\Psi = \int_{\Psi_{ij}} \frac{c - a_{ij}^T v_{ij}}{a_i} v_i dv_i dv_j d\Psi_{ij}$$

$$= \int_{\Psi_{ij}} \int_0^{c - a_{ij}^T v_{ij}} \frac{c - a_{ij}^T v_{ij} - a_j v_j}{2a_i^2} v_j dv_j d\Psi_{ij}$$

$$= \int_{\Psi_{ij}} \left( \frac{c - a_{ij}^T v_{ij}}{6a_i^2a_j} \right) dv_{ij}.$$

In the similar way, we can get that:

$$E(v_j) = \int_{\Psi} v_j d\Psi = \int_{\Psi_{ij}} \frac{c - a_{ij}^T v_{ij}}{6a_i^2a_j} dv_{ij}.$$

We thus can get that

$$E(v_i) - E(v_j) = \int_{\Psi_{ij}} \left( \frac{c - a_{ij}^T v_{ij}}{6a_i^2a_j} \right) dv_{ij} (a_j - a_i) > 0.$$

Similarly, we can prove that $\int_{\Psi} v_i d\Psi = \int_{\Psi} v_j d\Psi$ for $\gamma(x_i) = \gamma(x_j)$.

The proof is then completed.

**Theorem 7.5:** The binary, linear, logarithmic and mixture scheme are self-paced functions.

**Proof.** We first prove the above functions satisfying Condition 1 in Definition 7.3, i.e. they are convex with respect to $v \in [0, 1]^n$, where $n$ is the number of samples. As
Appendix

binary, linear, logarithmic and mixture self-paced functions can be decoupled \( f(v; \lambda) = \sum_{i=1}^{n} f(v_i; \lambda) \):

For binary scheme \( f(v_i; \lambda) = -\lambda v_i \):

\[
\frac{\partial^2 f}{\partial^2 v_i} = 0.
\] (D.8)

For linear scheme \( f(v_i; \lambda) = \frac{1}{2} \lambda (v_i^2 - 2v_i) \):

\[
\frac{\partial^2 f}{\partial^2 v_i} = \lambda > 0,
\] (D.9)

where \( \lambda > 0 \).

For logarithmic scheme \( f(v_i; \lambda) = \zeta v_i - \frac{\zeta v_i}{\log \zeta} \):

\[
\frac{\partial^2 f}{\partial^2 v_i} = -\frac{1}{\log \zeta} \frac{\zeta v_i}{\zeta_v} > 0,
\] (D.10)

where \( \zeta = 1 - \lambda \) and \( \lambda \in (0, 1) \).

For mixture scheme \( f(v_i; \lambda) = -\zeta \log(v_i + \frac{1}{\lambda_1} \zeta) \):

\[
\frac{\partial^2 f}{\partial^2 v_i} = \frac{\zeta \lambda_1^2}{(\zeta + \lambda_1 v_i)^2} > 0
\] (D.11)

where \( \lambda = [\lambda_1, \lambda_2] \), \( \zeta = \frac{\lambda_1 \lambda_2}{\lambda_1 - \lambda_2} \), and \( \lambda_1 > \lambda_2 > 0 \).

As the above second derivatives are non-negative, and the sum of convex functions is convex, we have \( f(v; \lambda) \) for binary, linear, logarithmic and mixture scheme are convex.

We then prove the above functions satisfying Condition 2 that is when all variables are fixed except for \( v_i, \ell_i, v_i^* \) decreases with \( \ell_i \).

Denote \( E_w = \sum_{i=1}^{n} v_i \ell_i + f(v; \lambda) \) as the objective with the fixed model parameters \( w \), where \( \ell_i \) is the loss for the \( i^{th} \) sample. The optimal solution \( v^* = [v_1^*, \ldots, v_n^*]^T = \arg \min_{v \in [0,1]^n} E_w \).
For binary scheme:

\[ E_w = \sum_{i=1}^{n} (\ell_i - \lambda)v_i; \]
\[ \frac{\partial E_w}{\partial v_i} = \ell_i - \lambda = 0; \]  \hspace{1cm} (D.12)
\[ \Rightarrow v_i^* = \begin{cases} 
1 & \ell_i < \lambda \\
0 & \ell_i \geq \lambda.
\end{cases} \]

For linear scheme:

\[ E_w = \sum_{i=1}^{n} \ell_i v_i + \frac{1}{2} \lambda(v_i^2 - 2v_i); \]
\[ \frac{\partial E_w}{\partial v_i} = \ell + v_i \lambda - \lambda = 0; \]  \hspace{1cm} (D.13)
\[ \Rightarrow v_i^* = \begin{cases} 
-\frac{1}{\lambda} \ell + 1 & \ell_i < \lambda \\
0 & \ell_i \geq \lambda.
\end{cases} \]

For logarithmic scheme:

\[ E_w = \sum_{i=1}^{n} \ell_i v_i + \zeta v_i - \zeta \log(v_i + 1); \]
\[ \frac{\partial E_w}{\partial v_i} = \ell + \zeta - \zeta v_i = 0; \]  \hspace{1cm} (D.14)
\[ \Rightarrow v_i^* = \begin{cases} 
\frac{1}{\log \zeta} \log(\ell + \zeta) & \ell_i < \lambda \\
0 & \ell_i \geq \lambda.
\end{cases} \]

where \( \zeta = 1 - \lambda \) (0 < \( \lambda \) < 1).

For mixture scheme:

\[ E_w = \sum_{i=1}^{n} \ell_i v_i - \zeta \log(v_i + 1)\lambda; \]
\[ \frac{\partial E_w}{\partial v_i} = \ell - \frac{\zeta \lambda_1}{\zeta + \lambda_1 v_i} = 0; \]  \hspace{1cm} (D.15)
\[ \Rightarrow v_i^* = \begin{cases} 
1 & \ell_i \leq \lambda_2 \\
0 & \ell_i \geq \lambda_1 \\
\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2} & \lambda_2 < \ell_i < \lambda_1
\end{cases} \]

where \( \lambda = [\lambda_1, \lambda_2] \), and \( \zeta = \frac{\lambda_1 \lambda_2}{\lambda_1 - \lambda_2} \), (\( \lambda_1 > \lambda_2 > 0 \)).
By setting the partial gradient to zero we arrive the optimal solution of $v$. It is obvious that $v_i$ is decreasing with respect to $\ell_i$ in all functions. In all cases, we have that \[ \lim_{\ell_i \to 0} v_i^* = 1, \lim_{\ell_i \to \infty} v_i^* = 0. \]

Finally, we prove that the above functions satisfying Condition 3 that is $\|v\|_1$ increases with respect to $\lambda$, and it holds that $\forall i \in [1, n]$, \[ \lim_{\lambda \to 0} v_i^* = 0, \lim_{\lambda \to \infty} v_i^* = 1. \]

It is easy to verify that each individual $v_i^*$ increases with respect to $\lambda$ in their closed-form solutions in Eq. (D.12), Eq. (D.13), Eq. (D.14) and Eq. (D.15) (in mixture scheme, let $\lambda = \lambda_1$ represent the model age). Therefore $\|v\|_1 = \sum_{i=1}^n v_i$ also increases with respect to $\lambda$. In an extreme case, when $\lambda$ approaches positive infinity, we have $\forall i \in [1, n]v_i = 1$, i.e. \[ \lim_{\lambda \to \infty} v_i^* = 1 \] in Eq. (D.12), Eq. (D.13), Eq. (D.14) and Eq. (D.15). Similarly, when $\lambda$ approaches 0, we have \[ \lim_{\lambda \to 0} v_i^* = 0. \]

As binary, linear, logarithmic and mixture scheme satisfy the three conditions, they are all self-paced functions.

The proof is then completed.

\[ \square \]

**Theorem 7.6:** Algorithm 3 attains the global optimum to $\min_v \mathbb{E}(w, v)$ for any given $w$ in linearithmic time.

**Proof.** Given the training dataset $\mathcal{D} = \{(x_1, y_1), \cdots, (x_n, y_n)\}$, where $x_i \in \mathbb{R}^m$ denotes the $i^{th}$ observed sample and $y_i$ denotes its label. Assume that the training samples $X = [x_1, \cdots, x_n]$ are with $b$ groups: $X^{(1)}, \cdots, X^{(b)}$, where $X^{(j)} = (x_1^{(j)}, \cdots, x_{n_j}^{(j)}) \in \mathbb{R}^{m \times n_j}$ corresponds to samples in the $j^{th}$ group, $n_j$ is the sample number in this group and $\sum_{j=1}^b n_j = n$. Accordingly, denote the weight vector as $v = [v^{(1)}, \cdots, v^{(b)}]$, where $v^{(j)} = (v_1^{(j)}, \cdots, v_{n_j}^{(j)})^T \in \mathbb{R}^{n_j}$. The following theorem proves that Algorithm 1 can get the global solution of the following non-convex optimization problem:

\[ \min_{v \in [0, 1]^n} \mathbb{E}(w, v; \lambda, \gamma) = \sum_{i=1}^n v_i L(y_i, f(x_i, w)) - \lambda \sum_{i=1}^n v_i - \gamma \|v\|_{2,1}, \tag{D.16} \]

where $L(y_i, f(x_i, w))$ denotes the loss function which calculates the cost between the ground truth label $y_i$ and the estimated label $f(x_i, w)$, and the $l_{2,1}$-norm $\|v\|_{2,1}$ is the group sparsity of $v$:

\[ \|v\|_{2,1} = \sum_{j=1}^b \|v^{(j)}\|_2. \]

For convenience we briefly rewrite $\mathbb{E}(w, v; \lambda, \gamma)$ and $L(y_i, f(x_i, w))$ as $\mathbb{E}(v)$ and $L_i$, respectively.
The weight vector \( v^* \) outputted from Algorithm 1 attains the global optimal solution of the optimization problem (D.16), i.e.,

\[
\begin{align*}
v^* &= \arg \min_{v \in [0,1]^n} E(v).
\end{align*}
\]

The objective function of (D.16) can be reformulated as the following decoupling forms based on the data cluster information:

\[
E(v) = \sum_{j=1}^{b} E(v^{(j)}), \quad (D.17)
\]

where

\[
E(v^{(j)}) = \sum_{i=1}^{n_j} v_i^{(j)} L_i^{(j)} - \lambda \sum_{i=1}^{n_j} v_i^{(j)} - \gamma \|v^{(j)}\|_2, \quad (D.18)
\]

where \( L_i^{(j)} \) represents the loss value of \( x_i^{(j)} \). It is easy to see that the original problem (D.16) can be equivalently decomposed as a series of the following sub-optimization problems \( (j = 1, \cdots, b) \):

\[
v^{(j)*} = \arg \min_{v^{(j)} \in [0,1]^{n_j}} E(v^{(j)}). \quad (D.19)
\]

\( E(v^{(j)}) \) defined in Eq. (D.18) is a concave function since its first and second terms are linear, and the third term is the negative \( l_{2,1} \) norm, whose positive form is a commonly utilized convex regularizer. It is well known that the minimum solution of a concave function over a polytope can be obtained at its vertices [140]. In other words, for the optimization problem (D.19), it holds that its optimal solution \( v^{(j)*} \in \{0,1\}^{n_j} \), i.e.,

\[
v^{(j)*} = \arg \min_{v^{(j)} \in \{0,1\}^{n_j}} E(v^{(j)}). \quad (D.20)
\]

For \( k = 1, \cdots, n_j \), let’s denote

\[
v^{(j)}(k) = \arg \min_{v^{(j)} \in \{0,1\}^{n_j}} \|v^{(j)}\|_0 = k \quad (D.21)
\]

This means that \( v^{(j)}(k) \) is the optimum of (D.19) if it is further constrained to be with \( k \) nonzero entries. It is then easy to deduce that

\[
v^{(j)*} = \arg \min_{v^{(j)}(k)} E(v^{(j)}(k)). \quad (D.22)
\]
That is, the optimal solution $v^{(j)*}$ of (D.19) can be achieved among $v^{(j)}(1), \ldots, v^{(j)}(n_j)$ at which the minimal objective value is attained.

Without loss of generality, we assume that the samples $(x_1^{(j)}, \ldots, x_{n_j}^{(j)})$ in the $j^{th}$ cluster are arranged in the ascending order of their loss values $L_i^{(j)}$. Then for the optimization problem (D.21), we can get that

$$\min_{v^{(j)} \in \{0,1\}^{n_j}} \|v^{(j)}\|_0 = k$$

$$\Leftrightarrow \min_{v^{(j)} \in \{0,1\}^{n_j}} \sum_{i=1}^{n_j} v_i^{(j)} L_i^{(j)} = k$$

since the last two terms in $E(v^{(j)})$ are with constant values under the constraint. Then it is easy to get that the optimal solution $v^{(j)}(k)$ of (D.21) is attained by setting its $k$ entries corresponding to the $k$ smallest loss values $L_i^{(j)}$ (i.e., the first $k$ entries of $v^{(j)}(k)$) as 1 while others as 0, and the minimal objective value is

$$E(v^{(j)}(k)) = \sum_{i=1}^{k} v_i^{(j)} L_i^{(j)} - \lambda k - \gamma \sqrt{k}.$$  
(D.23)

Then let's calculate the difference between any two adjacent elements in the sequence $E(v^{(j)}(1)), \ldots, E(v^{(j)}(n_j))$:

$$diff_k = E(v^{(j)}(k + 1)) - E(v^{(j)}(k)) = L_{k+1}^{(j)} - \lambda - \gamma (\sqrt{k+1} - \sqrt{k})$$

$$= L_{k+1}^{(j)} - (\lambda + \gamma \sqrt{\frac{1}{\sqrt{k+1} + \sqrt{k}}})$$

Since $L_{k}^{(j)}$ (with respect to $k$) is a monotonically increasing sequence while $\lambda + \gamma \sqrt{\frac{1}{\sqrt{k+1} + \sqrt{k}}}$ is a monotonically decreasing sequence, $diff_k$ is a monotonically increasing sequence. Denote $k^*$ as the index where its first positive value is attained (if $diff_k \leq 0$ for all $k = 1, \ldots, n_j - 1, k^* = n_j$). Then it is easy to get that $E(v^{(j)}(k))$ is monotonically decreasing until $k = k^*$ and then it starts to be monotonically increasing. This means that $E(v^{(j)}(k^*))$ gets the minimum among all $E(v^{(j)}(1)), \ldots, E(v^{(j)}(n_j))$. Based on (D.22), we know that the global optimum $v^{(j)*}$ of (D.19) is attained at $v^{(j)}(k^*)$.

By independently calculating the optimum $v^{(j)*}$ for each cluster and then combining them, the global optimal solution $v^*$ of (D.16) can then be calculated. This corresponds
to the process of our proposed Algorithm 3.

The most computational complex step in the above derivation is the sort of \( n_j \) \((1 \leq j \leq b)\) samples. Since \( n_j < n \), the average-case complexity is thus upper bounded by \( O(n \log n) \), assuming that the quick sort algorithm is used.

The proof is completed.
## Appendix E

### Detailed Results

Table E.1: Event-level comparison of AP on the 10 splits of MED13Test.

<table>
<thead>
<tr>
<th>Event ID &amp; Name</th>
<th>Raw + Expert</th>
<th>Adjusted + Expert</th>
<th>Adjusted + Auto</th>
<th>Adjusted + AutoVisual</th>
</tr>
</thead>
<tbody>
<tr>
<td>E006: Birthday party</td>
<td>0.3411</td>
<td>0.2980</td>
<td>0.1207</td>
<td>0.1207</td>
</tr>
<tr>
<td>E007: Changing a vehicle tire</td>
<td>0.0967</td>
<td>0.1667</td>
<td>0.2061</td>
<td>0.0134</td>
</tr>
<tr>
<td>E008: Flash mob gathering</td>
<td>0.2087</td>
<td>0.1647</td>
<td>0.1028</td>
<td>0.1028</td>
</tr>
<tr>
<td>E009: Getting a vehicle unstuck</td>
<td>0.1416</td>
<td>0.1393</td>
<td>0.0569</td>
<td>0.0569</td>
</tr>
<tr>
<td>E010: Grooming an animal</td>
<td>0.0442</td>
<td>0.0479</td>
<td>0.0128</td>
<td>0.0128</td>
</tr>
<tr>
<td>E011: Making a sandwich</td>
<td>0.0909</td>
<td>0.0804</td>
<td>0.2910</td>
<td>0.0709</td>
</tr>
<tr>
<td>E012: Parade</td>
<td>0.4552</td>
<td>0.4685</td>
<td>0.2027</td>
<td>0.2027</td>
</tr>
<tr>
<td>E013: Parkour</td>
<td>0.0498</td>
<td>0.0596</td>
<td>0.0619</td>
<td>0.0525</td>
</tr>
<tr>
<td>E014: Repairing an appliance</td>
<td>0.2731</td>
<td>0.2376</td>
<td>0.2262</td>
<td>0.0234</td>
</tr>
<tr>
<td>E015: Working on a sewing project</td>
<td>0.2022</td>
<td>0.2184</td>
<td>0.0135</td>
<td>0.0045</td>
</tr>
<tr>
<td>E021: Attempting a bike trick</td>
<td>0.0969</td>
<td>0.1163</td>
<td>0.0486</td>
<td>0.0486</td>
</tr>
<tr>
<td>E022: Cleaning an appliance</td>
<td>0.1248</td>
<td>0.1248</td>
<td>0.1248</td>
<td>0.0124</td>
</tr>
<tr>
<td>E023: Dog show</td>
<td>0.7284</td>
<td>0.7288</td>
<td>0.6028</td>
<td>0.6027</td>
</tr>
<tr>
<td>E024: Giving directions to a location</td>
<td>0.0253</td>
<td>0.0252</td>
<td>0.0252</td>
<td>0.0069</td>
</tr>
<tr>
<td>E025: Marriage proposal</td>
<td>0.0748</td>
<td>0.0750</td>
<td>0.0755</td>
<td>0.0011</td>
</tr>
<tr>
<td>E026: Renovating a home</td>
<td>0.0139</td>
<td>0.0061</td>
<td>0.0049</td>
<td>0.0049</td>
</tr>
<tr>
<td>E027: Rock climbing</td>
<td>0.1845</td>
<td>0.1724</td>
<td>0.0668</td>
<td>0.0668</td>
</tr>
<tr>
<td>E028: Town hall meeting</td>
<td>0.1585</td>
<td>0.0898</td>
<td>0.0163</td>
<td>0.0163</td>
</tr>
<tr>
<td>E029: Winning a race without a vehicle</td>
<td>0.1470</td>
<td>0.1697</td>
<td>0.0584</td>
<td>0.0584</td>
</tr>
<tr>
<td>E030: Working on a metal crafts project</td>
<td>0.0673</td>
<td>0.0422</td>
<td>0.0881</td>
<td>0.0026</td>
</tr>
<tr>
<td>MAP</td>
<td>0.1762</td>
<td>0.1716</td>
<td>0.1203</td>
<td>0.0741</td>
</tr>
</tbody>
</table>
Table E.2: Event-level comparison of AP on the 10 splits of MED14Test.

<table>
<thead>
<tr>
<th>Event ID &amp; Name</th>
<th>Raw + Expert</th>
<th>Adjusted + Expert</th>
<th>Adjusted + Auto</th>
<th>Adjusted + AutoVisual</th>
</tr>
</thead>
<tbody>
<tr>
<td>E021: Attempting a bike trick</td>
<td>0.0632</td>
<td>0.0678</td>
<td>0.0814</td>
<td>0.0822</td>
</tr>
<tr>
<td>E022: Cleaning an appliance</td>
<td>0.2634</td>
<td>0.2635</td>
<td>0.2634</td>
<td>0.2636</td>
</tr>
<tr>
<td>E023: Dog show</td>
<td>0.6757</td>
<td>0.6449</td>
<td>0.4387</td>
<td>0.4414</td>
</tr>
<tr>
<td>E024: Giving directions to a location</td>
<td>0.0613</td>
<td>0.0613</td>
<td>0.0614</td>
<td>0.0612</td>
</tr>
<tr>
<td>E025: Marriage proposal</td>
<td>0.0176</td>
<td>0.0174</td>
<td>0.0181</td>
<td>0.0174</td>
</tr>
<tr>
<td>E026: Renovating a home</td>
<td>0.0252</td>
<td>0.0089</td>
<td>0.0043</td>
<td>0.0043</td>
</tr>
<tr>
<td>E027: Rock climbing</td>
<td>0.2082</td>
<td>0.1302</td>
<td>0.0560</td>
<td>0.0560</td>
</tr>
<tr>
<td>E028: Town hall meeting</td>
<td>0.2478</td>
<td>0.0925</td>
<td>0.0161</td>
<td>0.0161</td>
</tr>
<tr>
<td>E029: Winning a race without a vehicle</td>
<td>0.1234</td>
<td>0.1848</td>
<td>0.0493</td>
<td>0.0497</td>
</tr>
<tr>
<td>E030: Working on a metal crafts project</td>
<td>0.1238</td>
<td>0.0616</td>
<td>0.0081</td>
<td>0.0081</td>
</tr>
<tr>
<td>E031: Beekeeping</td>
<td>0.5900</td>
<td>0.5221</td>
<td>0.4217</td>
<td>0.4258</td>
</tr>
<tr>
<td>E032: Wedding shower</td>
<td>0.0834</td>
<td>0.0924</td>
<td>0.0922</td>
<td>0.0905</td>
</tr>
<tr>
<td>E033: Non-motorized vehicle repair</td>
<td>0.5218</td>
<td>0.4525</td>
<td>0.0149</td>
<td>0.0150</td>
</tr>
<tr>
<td>E034: Fixing musical instrument</td>
<td>0.0284</td>
<td>0.0439</td>
<td>0.0439</td>
<td>0.0023</td>
</tr>
<tr>
<td>E035: Horse riding competition</td>
<td>0.3673</td>
<td>0.3346</td>
<td>0.0994</td>
<td>0.0993</td>
</tr>
<tr>
<td>E036: Felling a tree</td>
<td>0.0970</td>
<td>0.0620</td>
<td>0.0108</td>
<td>0.0108</td>
</tr>
<tr>
<td>E037: Parking a vehicle</td>
<td>0.2921</td>
<td>0.2046</td>
<td>0.0313</td>
<td>0.0313</td>
</tr>
<tr>
<td>E038: Playing fetch</td>
<td>0.0339</td>
<td>0.0284</td>
<td>0.0016</td>
<td>0.0014</td>
</tr>
<tr>
<td>E039: Tailgating</td>
<td>0.1429</td>
<td>0.0200</td>
<td>0.0010</td>
<td>0.0010</td>
</tr>
<tr>
<td>E040: Tuning musical instrument</td>
<td>0.1553</td>
<td>0.1553</td>
<td>0.1840</td>
<td>0.0128</td>
</tr>
<tr>
<td>MAP</td>
<td>0.2061</td>
<td>0.1724</td>
<td>0.0994</td>
<td>0.0865</td>
</tr>
</tbody>
</table>

Table E.3: Performance for 30 commercials on the YFCC100 set.

<table>
<thead>
<tr>
<th>ID</th>
<th>Query Name</th>
<th>Commercial Product</th>
<th>Evaluation Metric</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>football and running soccer shoes</td>
<td>0.80 1.00 0.88</td>
<td>Sport</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>auto_racing sport cars</td>
<td>0.70 1.00 0.91</td>
<td>Auto</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>dog_show dog training collars</td>
<td>0.95 1.00 0.97</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>baby stroller/diaper</td>
<td>1.00 1.00 1.00</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>fire burning smoke fire prevention</td>
<td>0.95 1.00 0.96</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>cake birthday cake</td>
<td>0.35 0.50 0.60</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>underwater diving</td>
<td>1.00 1.00 1.00</td>
<td>Sports</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>dog indoor dog food</td>
<td>0.75 1.00 0.67</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>riding_horse horse riding lessons</td>
<td>0.90 1.00 0.93</td>
<td>Sports</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>kitchen_food restaurant</td>
<td>1.00 1.00 1.00</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Christmas decoration</td>
<td>0.80 1.00 0.87</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>dancing dancing lessons</td>
<td>0.90 1.00 0.90</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>bicycling cycling cloth and helmet</td>
<td>0.95 1.00 0.99</td>
<td>Sports</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>car and vehicle car tires</td>
<td>1.00 1.00 1.00</td>
<td>Auto</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>skiing or snowboarding ski resort</td>
<td>0.95 1.00 0.96</td>
<td>Sports</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>parade flags or banners</td>
<td>0.90 1.00 0.96</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>music_band live music show</td>
<td>1.00 1.00 1.00</td>
<td>Grocery</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>busking live show</td>
<td>0.20 1.00 0.50</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>home renovation furniture</td>
<td>0.00 0.00 0.00</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>speaking in front of people</td>
<td>0.65 0.50 0.63</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>sunny_beach vacation by beach</td>
<td>1.00 1.00 1.00</td>
<td>Traveling</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>politicians vote Obama</td>
<td>0.60 1.00 0.63</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>female face makeup</td>
<td>1.00 1.00 1.00</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>cellphone cell phone</td>
<td>0.80 1.00 0.96</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>fireworks fireworks</td>
<td>0.95 1.00 0.96</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>tennis</td>
<td>1.00 1.00 1.00</td>
<td>Sports</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>helicopter helicopter tour</td>
<td>1.00 1.00 1.00</td>
<td>Traveling</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>cooking</td>
<td>0.90 1.00 0.92</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>eiffel_night hotels in Paris</td>
<td>0.90 1.00 0.89</td>
<td>Traveling</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>table tennis ping pong</td>
<td>0.60 1.00 0.85</td>
<td>Sports</td>
<td></td>
</tr>
</tbody>
</table>
Table E.4: Event-level comparison of modality contribution on the NIST split. The best AP is marked in bold.

<table>
<thead>
<tr>
<th>Event ID &amp; Name</th>
<th>FullSys</th>
<th>FullSys+PRF</th>
<th>VisualSys</th>
<th>ASRSys</th>
<th>OCRSys</th>
</tr>
</thead>
<tbody>
<tr>
<td>E006: Birthday party</td>
<td>0.3842</td>
<td>0.3862</td>
<td>0.3673</td>
<td>0.0327</td>
<td>0.0386</td>
</tr>
<tr>
<td>E007: Changing a vehicle tire</td>
<td>0.2322</td>
<td>0.3240</td>
<td>0.2162</td>
<td>0.1707</td>
<td>0.0212</td>
</tr>
<tr>
<td>E008: Flash mob gathering</td>
<td>0.2864</td>
<td>0.4310</td>
<td>0.2864</td>
<td>0.0052</td>
<td>0.0409</td>
</tr>
<tr>
<td>E009: Getting a vehicle unstuck</td>
<td>0.1588</td>
<td>0.1561</td>
<td>0.1588</td>
<td>0.0063</td>
<td>0.0162</td>
</tr>
<tr>
<td>E010: Grooming an animal</td>
<td>0.0782</td>
<td>0.0725</td>
<td>0.0782</td>
<td>0.0166</td>
<td>0.0050</td>
</tr>
<tr>
<td>E011: Making a sandwich</td>
<td>0.1183</td>
<td>0.1304</td>
<td>0.1064</td>
<td>0.2184</td>
<td>0.0682</td>
</tr>
<tr>
<td>E012: Parade</td>
<td>0.5566</td>
<td>0.5319</td>
<td>0.5566</td>
<td>0.0080</td>
<td>0.0645</td>
</tr>
<tr>
<td>E013: Parkour</td>
<td>0.0545</td>
<td>0.0839</td>
<td>0.0448</td>
<td>0.0043</td>
<td>0.0066</td>
</tr>
<tr>
<td>E014: Repairing an appliance</td>
<td>0.2619</td>
<td>0.2989</td>
<td>0.2341</td>
<td>0.2086</td>
<td>0.0258</td>
</tr>
<tr>
<td>E015: Working on a sewing project</td>
<td>0.2068</td>
<td>0.2021</td>
<td>0.2036</td>
<td>0.0866</td>
<td>0.0166</td>
</tr>
<tr>
<td>E021: Attempting a bike trick</td>
<td>0.0635</td>
<td>0.0701</td>
<td>0.0635</td>
<td>0.0006</td>
<td>0.0046</td>
</tr>
<tr>
<td>E022: Cleaning an appliance</td>
<td>0.2634</td>
<td>0.1747</td>
<td>0.0008</td>
<td>0.2634</td>
<td>0.0105</td>
</tr>
<tr>
<td>E023: Dog show</td>
<td>0.6737</td>
<td>0.6610</td>
<td>0.6737</td>
<td>0.0009</td>
<td>0.0303</td>
</tr>
<tr>
<td>E024: Giving directions to a location</td>
<td>0.0614</td>
<td>0.0228</td>
<td>0.0011</td>
<td>0.0614</td>
<td>0.0036</td>
</tr>
<tr>
<td>E025: Marriage proposal</td>
<td>0.0188</td>
<td>0.0270</td>
<td>0.0024</td>
<td>0.0021</td>
<td>0.0188</td>
</tr>
<tr>
<td>E026: Renovating a home</td>
<td>0.0252</td>
<td>0.0160</td>
<td>0.0252</td>
<td>0.0026</td>
<td>0.0023</td>
</tr>
<tr>
<td>E027: Rock climbing</td>
<td>0.2077</td>
<td>0.2001</td>
<td>0.2077</td>
<td>0.1127</td>
<td>0.0038</td>
</tr>
<tr>
<td>E028: Town hall meeting</td>
<td>0.2492</td>
<td>0.3172</td>
<td>0.2492</td>
<td>0.0064</td>
<td>0.0134</td>
</tr>
<tr>
<td>E029: Winning a race without a vehicle</td>
<td>0.1257</td>
<td>0.1929</td>
<td>0.1257</td>
<td>0.0011</td>
<td>0.0019</td>
</tr>
<tr>
<td>E030: Working on a metal crafts project</td>
<td>0.1238</td>
<td>0.1255</td>
<td>0.0608</td>
<td>0.0981</td>
<td>0.0142</td>
</tr>
<tr>
<td>E031: Beekeeping</td>
<td>0.5883</td>
<td>0.6401</td>
<td>0.5883</td>
<td>0.2676</td>
<td>0.0440</td>
</tr>
<tr>
<td>E032: Wedding shower</td>
<td>0.0833</td>
<td>0.0879</td>
<td>0.0459</td>
<td>0.0428</td>
<td>0.0017</td>
</tr>
<tr>
<td>E033: Non-motorized vehicle repair</td>
<td>0.5198</td>
<td>0.5263</td>
<td>0.5198</td>
<td>0.0828</td>
<td>0.0159</td>
</tr>
<tr>
<td>E034: Fixing musical instrument</td>
<td>0.0276</td>
<td>0.0444</td>
<td>0.0170</td>
<td>0.0248</td>
<td>0.0023</td>
</tr>
<tr>
<td>E035: Horse riding competition</td>
<td>0.3077</td>
<td>0.3710</td>
<td>0.3677</td>
<td>0.0013</td>
<td>0.0104</td>
</tr>
<tr>
<td>E036: Felling a tree</td>
<td>0.0968</td>
<td>0.1180</td>
<td>0.0968</td>
<td>0.0020</td>
<td>0.0076</td>
</tr>
<tr>
<td>E037: Parking a vehicle</td>
<td>0.2918</td>
<td>0.2477</td>
<td>0.2918</td>
<td>0.0008</td>
<td>0.0009</td>
</tr>
<tr>
<td>E038: Playing fetch</td>
<td>0.0339</td>
<td>0.0373</td>
<td>0.0339</td>
<td>0.0020</td>
<td>0.0014</td>
</tr>
<tr>
<td>E039: Tailgating</td>
<td>0.1437</td>
<td>0.1501</td>
<td>0.1437</td>
<td>0.0013</td>
<td>0.0388</td>
</tr>
<tr>
<td>E040: Tuning musical instrument</td>
<td>0.1554</td>
<td>0.3804</td>
<td>0.0010</td>
<td>0.1840</td>
<td>0.0677</td>
</tr>
<tr>
<td>MAP (MED13Test E006-E015 E021-E030)</td>
<td>0.2076</td>
<td>0.2222</td>
<td>0.1831</td>
<td>0.0653</td>
<td>0.0203</td>
</tr>
<tr>
<td>MAP (MED14Test E021-E040)</td>
<td>0.2060</td>
<td>0.2205</td>
<td>0.1758</td>
<td>0.0579</td>
<td>0.0147</td>
</tr>
</tbody>
</table>
Table E.5: Event-level comparison of visual feature contribution on the NIST split.

<table>
<thead>
<tr>
<th>Event ID &amp; Name</th>
<th>FullSys</th>
<th>MED/IIAC</th>
<th>MED/Sports</th>
<th>MED/YFCC</th>
<th>MED/DIY</th>
<th>MED/ImageNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>E006: Birthday party</td>
<td>0.3842</td>
<td>0.3797</td>
<td>0.3842</td>
<td>0.2814</td>
<td>0.3842</td>
<td>0.2876</td>
</tr>
<tr>
<td>E007: Changing a vehicle tire</td>
<td>0.2322</td>
<td>0.2720</td>
<td>0.2782</td>
<td>0.1811</td>
<td>0.1247</td>
<td>0.0698</td>
</tr>
<tr>
<td>E008: Flash mob gathering</td>
<td>0.2864</td>
<td>0.1872</td>
<td>0.2864</td>
<td>0.3345</td>
<td>0.2864</td>
<td>0.2864</td>
</tr>
<tr>
<td>E009: Getting a vehicle unstuck</td>
<td>0.1588</td>
<td>0.1070</td>
<td>0.1588</td>
<td>0.1132</td>
<td>0.1588</td>
<td>0.1588</td>
</tr>
<tr>
<td>E010: Grooming an animal</td>
<td>0.0782</td>
<td>0.0902</td>
<td>0.0782</td>
<td>0.0914</td>
<td>0.0474</td>
<td>0.0782</td>
</tr>
<tr>
<td>E011: Making a sandwich</td>
<td>0.1183</td>
<td>0.0926</td>
<td>0.1183</td>
<td>0.1146</td>
<td>0.1183</td>
<td>0.1183</td>
</tr>
<tr>
<td>E012: Parade</td>
<td>0.5666</td>
<td>0.5738</td>
<td>0.5566</td>
<td>0.3007</td>
<td>0.5566</td>
<td>0.5666</td>
</tr>
<tr>
<td>E013: Parkour</td>
<td>0.0545</td>
<td>0.0066</td>
<td>0.0545</td>
<td>0.0545</td>
<td>0.0545</td>
<td>0.0545</td>
</tr>
<tr>
<td>E014: Repairing an appliance</td>
<td>0.2619</td>
<td>0.2247</td>
<td>0.2619</td>
<td>0.1709</td>
<td>0.2619</td>
<td>0.1129</td>
</tr>
<tr>
<td>E015: Working on a sewing project</td>
<td>0.2068</td>
<td>0.2166</td>
<td>0.2068</td>
<td>0.2068</td>
<td>0.1847</td>
<td>0.0712</td>
</tr>
<tr>
<td>E021: Attempting a bike trick</td>
<td>0.0635</td>
<td>0.0635</td>
<td>0.0006</td>
<td>0.0635</td>
<td>0.0635</td>
<td>0.0635</td>
</tr>
<tr>
<td>E022: Cleaning an appliance</td>
<td>0.2634</td>
<td>0.2634</td>
<td>0.2634</td>
<td>0.2634</td>
<td>0.2634</td>
<td>0.2634</td>
</tr>
<tr>
<td>E023: Dog show</td>
<td>0.6737</td>
<td>0.6737</td>
<td>0.6737</td>
<td>0.6737</td>
<td>0.6737</td>
<td>0.6737</td>
</tr>
<tr>
<td>E024: Giving directions to a location</td>
<td>0.0614</td>
<td>0.0614</td>
<td>0.0614</td>
<td>0.0614</td>
<td>0.0614</td>
<td>0.0614</td>
</tr>
<tr>
<td>E025: Marriage proposal</td>
<td>0.0188</td>
<td>0.0188</td>
<td>0.0188</td>
<td>0.0188</td>
<td>0.0188</td>
<td>0.0188</td>
</tr>
<tr>
<td>E026: Renovating a home</td>
<td>0.0252</td>
<td>0.0197</td>
<td>0.0252</td>
<td>0.0252</td>
<td>0.0252</td>
<td>0.0252</td>
</tr>
<tr>
<td>E027: Rock climbing</td>
<td>0.2077</td>
<td>0.2077</td>
<td>0.2077</td>
<td>0.2077</td>
<td>0.2077</td>
<td>0.2077</td>
</tr>
<tr>
<td>E028: Town hall meeting</td>
<td>0.2492</td>
<td>0.0956</td>
<td>0.2492</td>
<td>0.2418</td>
<td>0.2492</td>
<td>0.2492</td>
</tr>
<tr>
<td>E029: Winning a race without a vehicle</td>
<td>0.1257</td>
<td>0.1257</td>
<td>0.1257</td>
<td>0.1257</td>
<td>0.1257</td>
<td>0.1257</td>
</tr>
<tr>
<td>E030: Working on a metal crafts project</td>
<td>0.1238</td>
<td>0.1238</td>
<td>0.1238</td>
<td>0.0998</td>
<td>0.1238</td>
<td>0.1238</td>
</tr>
<tr>
<td>E031: Beekeeping</td>
<td>0.5883</td>
<td>0.5883</td>
<td>0.5883</td>
<td>0.5883</td>
<td>0.5883</td>
<td>0.0012</td>
</tr>
<tr>
<td>E032: Wedding shower</td>
<td>0.0833</td>
<td>0.0833</td>
<td>0.0833</td>
<td>0.0833</td>
<td>0.0924</td>
<td>0.0833</td>
</tr>
<tr>
<td>E033: Non-motorized vehicle repair</td>
<td>0.5198</td>
<td>0.5198</td>
<td>0.4440</td>
<td>0.5198</td>
<td>0.4742</td>
<td>0.4417</td>
</tr>
<tr>
<td>E034: Fixing musical instrument</td>
<td>0.0276</td>
<td>0.0276</td>
<td>0.0276</td>
<td>0.0276</td>
<td>0.0439</td>
<td>0.0276</td>
</tr>
<tr>
<td>E035: Horse riding competition</td>
<td>0.3677</td>
<td>0.3430</td>
<td>0.1916</td>
<td>0.3677</td>
<td>0.3677</td>
<td>0.3677</td>
</tr>
<tr>
<td>E036: Felling a tree</td>
<td>0.0968</td>
<td>0.0275</td>
<td>0.1100</td>
<td>0.0968</td>
<td>0.0968</td>
<td>0.0968</td>
</tr>
<tr>
<td>E037: Parking a vehicle</td>
<td>0.2918</td>
<td>0.1902</td>
<td>0.2918</td>
<td>0.2918</td>
<td>0.2918</td>
<td>0.1097</td>
</tr>
<tr>
<td>E038: Playing fetch</td>
<td>0.0339</td>
<td>0.0339</td>
<td>0.0008</td>
<td>0.0339</td>
<td>0.0339</td>
<td>0.0339</td>
</tr>
<tr>
<td>E039: Tailgating</td>
<td>0.1437</td>
<td>0.0631</td>
<td>0.1437</td>
<td>0.0666</td>
<td>0.1437</td>
<td>0.1437</td>
</tr>
<tr>
<td>E040: Tuning musical instrument</td>
<td>0.1554</td>
<td>0.1554</td>
<td>0.1554</td>
<td>0.1554</td>
<td>0.1554</td>
<td>0.1554</td>
</tr>
</tbody>
</table>

MAP (MED13Test E006-E015 E021-E030) | 0.2075  | 0.1893   | 0.1567     | 0.1814   | 0.1995  | 0.1818        |
MAP (MED14Test E021-E040)           | 0.2060  | 0.1834   | 0.1393     | 0.2005   | 0.2050  | 0.1637        |
Bibliography


[36] Alexander Hauptmann, Rong Yan, Wei-Hao Lin, Michael Christel, and Howard Wactlar. Can high-level concepts fill the semantic gap in video retrieval? a case


