Search based face annotation

Wang, Dayong

2014

Wang, D. (2014). Search based face annotation. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/59861>

<https://doi.org/10.32657/10356/59861>

Search Based Face Annotation

Dayong Wang

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy

2014

Abstract

In the era of big data today, a large amount of facial images are shared over the internet, which have posed a lot of research challenges and created a quantity of opportunities in various real-world applications. An important emerging research problem in this area is automatic face annotation, which aims to automatically tag human names to a query facial image. Unlike the conventional “*model-based face annotation*” techniques, some recent emerging studies have attempted to tackle this problem in the framework “Search-based Face Annotation” (SBFA), which mines massive web facial images that are freely available on the world wide web (WWW) by exploring content-based image retrieval (CBIR) techniques. Due to the noisy nature of web images, the raw labels of web facial images are often noisy, in which some facial images are associated with incorrect/incomplete names. Such kind of raw facial image database is referred to as “weakly labeled web facial image database” in this thesis.

This thesis investigates a comprehensive framework of the “Search-based Face Annotation” paradigm, which generally consists of three main stages: firstly, given a query facial image, it typically involves a pre-processing stage, including face detection, face alignment, and facial feature extraction; subsequently, the input facial image is aligned into a consistent position and represented as a feature vector in the facial feature space; second, it retrieves the top-ranking similar instances of the query facial image from a large-scale weakly labeled web facial image database by using content-based image retrieval techniques; and thirdly, it names the query image by mining the top-ranking similar images and the corresponding weak label information. In this thesis, we mainly focus on tackling three open challenging tasks: (i) how to effectively enhance the initial weak name labels, (ii) how to effectively exploit a short list of candidate facial images and their weak label information for the face name annotation task, and (iii) how to effectively boost the annotation performance of SBFA problem by adopting sophisticated machine learning techniques in a unified framework.

This thesis investigates a family of effective and efficient algorithms to tackle the aforementioned challenges. In particular, the main contributions of this thesis are listed as follows:

- To enhance the initial weak label information, we propose an Unsupervised Label Refinement (ULR) approach by adopting graph-based learning techniques. An effective optimization algorithm is proposed to solve the large-scale learning task and a Clustering-based Approximation (CBA) algorithm is proposed to improve the system scalability.
- To fully mine the top-ranking similar images and the corresponding weak label information, we propose a Weak Label Regularized Local Coordinate Coding (WLRCC) algorithm, which exploits the principle of local coordinate coding to learn more discriminative facial features for efficient label propagation in a sparse reconstruction scheme.
- By combining two different learning schemes: “transductive learning” and “inductive learning” in the same framework based on the entropy information, we propose a “Unified Learning Framework for Auto Face Annotation” (UTIL), which efficiently improve the annotation performance without introduce much training effort.
- To go beyond the limitation of single facial feature representation and general Euclidean distance space, we propose a multi-model based “Learn to Name Faces” (L2NF) algorithm, which optimize the fusion of multiple annotation modalities in a learning to rank framework.
- Based on search-based face annotation framework, we built a real world demonstration system: FANS, which is available at <http://msm.cais.ntu.edu.sg/fans>. It can be used to predict the celebrity name, or find the most similar celebrity image of the query images. In addition to the website interface, we also deploy an Android APP for the mobile devices, which popularizes the usage of our system.

This thesis has constructed an extensive set of experiments by evaluating the performance of the proposed techniques in comparison to the state-of-the-art methods on large-scale real-world facial image databases. Encouraging results illustrate that the proposed techniques significantly boost the state-of-the-art annotation performance.

Acknowledgments

I am grateful to all those who have helped and supported me throughout my Ph.D. life.

I sincerely thank my supervisor Dr. Chu Hong Hoi, for his persistent guidance during my graduate study. Dr. hoi has taught me much not only on research, but also about presentation and English writing skills, and so on. All his efforts towards my education have significantly improved many aspects of my development. This study would not have been possible without the guidance, patience and support of him. Besides the valuable research skills that I learn-t from him, his personal fascination also affects me to be a kind, happy and self-motivated person.

I would grateful acknowledge my co-supervisor Dr. Ying He, an expert in computer graphics, for his precious advice and unlimited encouragement on various projects. His co-supervision is very important for my Ph.D. study. I also appreciate my research collaborators who have provided help and guidance throughout the research projects in this thesis.

Many thanks to all my friends in Nanyang Technological University, who supported me a lot by providing valuable feedback in many fruitful discussions. This report would not have been possible in this form without the support and collaboration of several friends, in particular Dr. Hao Xia, Dr. Pengcheng Wu, Dr. Peilin Zhao, Dr. Bin Li, Dr. Qian Sun, Dr. Minqi Zhang, Dr. Xiang Ying, Dr. Bo Li, Dr. Xiaoning Wang, Dr. Ming Zhao, Dr. Chi-Fu Lai, Dr. Dao Thi Phuong Quynh, Dr. Xiaoqi Yan, and Dr. Min Meng.

Finally, I thank my parents and my families for always motivating, supporting and for their infinite amount of love, tolerance, and patience, which enabled me to fully concentrate on my research work.

Contents

Abstract	i
Acknowledgments	iii
List of Figures	ix
List of Tables	xiii
1 Introduction	1
1.1 Automatic Face Annotation	1
1.1.1 Model-based Face Annotation	2
1.1.2 Search-based Face Annotation	3
1.2 Main Challenges of Search-based Face Annotation	3
1.3 Contributions	6
1.4 Scope and Organization	8
2 Literature Review	11
2.1 General Face Recognition and Verification	11
2.2 General Image Annotation	14
2.3 Automatic Face Annotation	15
2.3.1 Personal Photo Based Face Annotation	16
2.3.2 Name-based Face Annotation	17
2.3.3 Caption-based Face Annotation	18
2.3.4 Search-based Face Annotation	19
2.4 Machine Learning Techniques	20
2.4.1 Supervised Learning	21
2.4.2 Semi-supervised Learning	24
2.4.3 Unsupervised Learning	25

2.5	Sparse Representation	26
2.5.1	Dictionary Learning	27
2.5.2	Coding Learning	28
3	Search-based Face Annotation: An Overview	33
3.1	Weakly Labeled Faces Database (WLFDB)	34
3.1.1	Facial Image Database Construction	34
3.1.2	Database Evaluation	36
3.2	Facial Images Preprocessing	38
3.2.1	Face Detection (FD)	38
3.2.2	Face Alignment (FA)	39
3.2.3	Feature Extraction (FE)	40
3.2.4	Feature Indexing (FI)	41
3.3	Facial Image Retrieval and Name Annotation	41
3.4	Real-world Demo System: FANS	42
3.4.1	Query Input Methods	42
3.4.2	Presentation of Retrieval Results	44
3.4.3	Mobile Application on Android System	45
3.5	Summary	46
4	Unsupervised Label Refinement	49
4.1	Unsupervised Label Refinement by Learning on Weakly Labeled Data	49
4.1.1	Problem Formulation	49
4.1.2	Algorithms	53
4.1.3	Clustering-based Approximation (CBA)	57
4.2	Experiments	61
4.2.1	Experiment Testbed	62
4.2.2	Evaluation of Automatic Face Annotation	63
4.2.3	Evaluation on Varied Top- n and Top- T Settings	65
4.2.4	Evaluation on Varied Numbers of Images per Person in Database	65
4.2.5	Evaluation on a Larger Database: WDB-600K	67
4.2.6	Evaluation of Parameter Sensitivity	68

4.2.7	Evaluation of Optimization Efficiency	69
4.2.8	Evaluation of Clustering-based Approximation	71
4.2.9	Label Refinement on Artificial Dataset	73
4.3	Summary	73
5	Weak Label Regularized Local Coordinate Coding	75
5.1	Weak Label Regularized Local Coordinate Coding	75
5.1.1	Preliminaries	76
5.1.2	Problem Formulation	78
5.1.3	Optimization	81
5.1.4	Face Name Annotation	84
5.2	Offline Approximation for WLRLCC	85
5.3	Experimental Results	87
5.3.1	Experiment Testbed	87
5.3.2	Impact of the Top- n and Top- T Settings	87
5.3.3	Evaluation of Automatic Face Annotation on “WDB-040K”	90
5.3.4	Evaluation of Approximation for WLRLCC	91
5.3.5	Evaluation of Parameter Sensitivity	94
5.4	Summary	95
6	Fusion of Transductive and Inductive Learning for Face Annotation	97
6.1	Fusion of Transductive and Inductive Learning	98
6.2	Weak Label Laplacian SVM (WL-LapSVM)	99
6.2.1	Preliminaries	99
6.2.2	Problem Formulation of WL-LapSVM	101
6.2.3	Optimization Algorithm	103
6.2.4	Name Annotation	104
6.2.5	Pseudo Set Construction and Confidence Weights Setting	104
6.3	Fusion Strategies for Combination	105
6.3.1	Measurement Level Fusion	106
6.3.2	Rank Level Fusion	106
6.4	Experiments	107

6.4.1	Experiment Testbed	107
6.4.2	Evaluation on Positive Sample Size	108
6.4.3	Evaluation of Auto Face Annotation	109
6.4.4	Evaluation on Different Combinations	111
6.4.5	Evaluation on Parameter Sensitivity	112
6.5	Summary	112
7	Multimodal Learning for Learning to Name Faces	115
7.1	L2NF — Learning to Name Faces	115
7.1.1	Preliminaries	116
7.1.2	Problem Overview	117
7.1.3	Weak Label Refinement	121
7.1.4	Multimodal Representation Construction	122
7.1.5	Optimal Fusion of Multiple Modalities	124
7.1.6	Algorithm for Learning to Name Faces	127
7.2	Experimental Results	129
7.2.1	Experimental Testbed	129
7.2.2	Experiments on “WDB-040K” and “WDB-600K”	130
7.2.3	Evaluation on Training Query Sizes	132
7.2.4	Analysis of the Performance Gains	133
7.3	Summary	134
8	Conclusions and Future Work	135
8.1	Summary	135
8.2	Future Work	137
	Publications	139
	References	141

List of Figures

1.1	Web-based Photo-sharing Websites	2
1.2	An Overview of the proposed Search-based Face Annotation Framework	4
2.1	The System Framework of Easyalbumn	17
2.2	Name-based Face Annotation	18
2.3	Caption-based Face Annotation	18
2.4	Local feature extraction and indexing	20
2.5	Dictionary Learning for image resotation	27
2.6	Online Dictionary Learning Approach	28
3.1	The system flow of Search-based Face Annotation (SBFA) scheme. (a) We collect weakly labeled facial images from WWW using web search engines. (b) We pre-process the crawled web facial images, including face detection, face alignment, and feature extraction for the detected faces; after that, we apply LSH to index the extracted high-dimensional facial features. (c) We search for the query facial image to mine the top-ranking similar images and the corresponding label information for annotation.	34
3.2	Two example sets of weakly labeled facial images	37
3.3	The histogram of the number of images per person in the retrieval database. . .	37
3.4	An Example of Face Detection.	38
3.5	An example for DLK face alignment	40
3.6	The methods of inputting query facial image, including entering the URL, up-loading the local image, and directly dragging the images on the webpage. . . .	43
3.7	The representation of retrieval results in FANS system.	44

3.8	(a) Interface of the mobile application, (b) Annotation result using a captured photo, (c) Edit and crop image, and (d) Access the photo albums on the Facebook.	45
4.1	Illustration of computing class similarity matrix between three classes X_1 , X_2 and X_3 . The symbol $\otimes$ denotes the class center. C_{1*} , C_{2*} and C_{3*} are the similarity vectors of the three classes, which are computed according to Eq. 4.22 with $n = 1$. For example, the second value of vector C_{1*} , i.e., C_{12} , refers to the total number of examples in class X_2 belonging to the top $n = 1$ nearest neighbors of examples from class X_1 .	59
4.2	Annotation performance w.r.t. varied n and T values.	66
4.3	The annotation performance on three different databases, which have different numbers of images per person. Specifically, P050, P075, and P100 denote the databases have the top 50, 75, and 100 retrieval images per person, respectively.	66
4.4	Face annotation performance on Database: WDB-600K. The algorithms that end up with “-wm” denote the improved performances achieved by weighted majority voting method in the name annotation step.	68
4.5	Evaluation of parameter sensitivity with respect to α and β for $T = 1$. The values of the vertex on the color mesh illustrate the hit rate gap between ORI and ULR.	69
4.6	The label refinement experiment over an artificial dataset. (a) The demo dataset with 9 classes (persons), half of them are mislabeled. (b) The original noisy label matrix and the refined ones achieved by various algorithms. The distances of the refined label matrix to the ideal label matrix (Y_{true}) are shown at the bottom of each figure.	74
5.1	Illustration of the proposed WRLCC process. In this example, we show the top $n = 8$ retrieved images with $m = 4$ candidate names for query image $\mathbf{x}_q$: (a) gives the input data; (b) illustrates the two key steps of the WRLCC process; (c) shows the output coding result and final annotation scheme.	77

5.2	An example for AWLRLCC. (a) shows the database images $\bullet(\{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_9\})$ and query image $\otimes(\mathbf{x}_q)$; (b) shows the nearest neighbor sets of $\mathbf{x}_q$, $\mathbf{d}_4$, $\mathbf{d}_3$, and $\mathbf{d}_5$; (c) shows the value mapping rules between the pre-computed coding $\mathbf{c}_i$ and the reconstructed coding $\mathbf{s}_i$ based on the new dictionary $\bar{X}$. As an example, for the facial image $\mathbf{d}_4$ with $j = 1$, the corresponding dictionary item is $\mathbf{d}_5$ and $\Upsilon(\bar{X}, \mathbf{d}_5) = 5$, so that $s_{45} = c_{41}$	86
5.3	(a) The hit rate performance of SMW and WLRCC with different Top- n , (b) The average running time of WLRCC for one query sample.	89
5.4	Comparison between WLRCC and the other algorithms on WDB-040K, WDB-100K, WDB-200K, WDB-400K, and WDB-600K	91
5.5	Comparison between WLRCC and AWLRLCC.	93
5.6	Grid search result of WLRCC on WDB-040K.	95
6.1	Annotation performance of WL-LapSVM with different confidence weighting settings. The x -axis is the confidence weight ε_p of the last pseudo sample. . . .	105
6.2	Annotation performance of inductive learning algorithms with varied numbers (p) of pseudo positive samples.	109
6.3	Comparison of face annotation performance by different algorithms, where T is the number of the annotated names.	110
6.4	Comparison of different last-fusion algorithms in the UTIL framework	112
6.5	Grid search result of WL-LapSVM	113
7.1	Introduction of face naming in “ <i>Search-based Face Annotation</i> ”. A visual explanation of Eq. 7.3. For a query instance $\mathbf{q}_i$, its top- n similar samples are $\{(\mathbf{d}_{ij}, \mathbf{y}_{ij})\}_{j=1,2,\dots,n}$, and the predicted label vector is $\hat{\mathbf{y}}_{\mathbf{q}_i}$. $\mathbf{y}_{ij}$ is the label vector of the j -th nearest sample $\mathbf{d}_{ij}$. $\mathbf{x}_{ij}$ is the feature vector between the query instance $\mathbf{q}_i$ and the similar example $\mathbf{d}_{ij}$. The k -th item of $\mathbf{x}_{ij}$ is constructed with the k -th query-neighbor similarity function $\phi_k(\mathbf{q}_i, \mathbf{d}_{ij})$. The inner product value $\mathbf{x}_{ij}^\top \mathbf{w}$ is the confidence values v_{ij} for the j -th label vector $\mathbf{y}_{ij}$	119
7.2	Three factors that affect the annotation performance of SBFA and the corresponding improvement solution.	120
7.3	Face naming performance (hit rate @ top- T) on the two databases: WDB-600K.	132

7.4	Face Naming Performance (hit rate $\text{top-}T = 1$) of L2NF with varied sizes of training queries.	133
-----	---	-----

List of Tables

3.1	The performance of WRLCC with different features.	40
4.1	List of Key Notations	50
4.2	Evaluation of auto face annotation performance in terms of <i>hit rates</i> at top- T annotated names.	64
4.3	The average running time (seconds) of the four proposed algorithms.	70
4.4	Evaluation of running time used by Clustering-based Approximation (CBA) . .	71
4.5	Evaluation of annotation performance of Clustering-based Approximation (CBA)	72
5.1	List of Key Notations	76
5.2	The hit rate performance of SMW ($n = 10$) and WRLCC ($n = 40$) with different Top- T	88
5.3	The hit rate performance of SMW and WRLCC with different Top- n , and the running time of WRLCC.	88
5.4	The comparison among WRLCC and other algorithms on database WDB-040K.	90
5.5	The running time of different approximation schemes, based on database WDB-040K	92
5.6	The performance of WRLCC and the cascade framework combined with ULR.	94
6.1	List of Key Notations	100
6.2	Comparison of auto face annotation performance by different approaches. . . .	111
7.1	List of Key Notations	116
7.2	Face Naming Performance on Database WDB-040K.	131
7.3	Evaluation and analysis of the performance gains.	134

Chapter 1

Introduction

1.1 Automatic Face Annotation

In the digital era today, people can easily capture a photo by various digital devices, and share it through the internet by web photo sharing portals (e.g., Flickr) or social networks (e.g., Facebook), as shown in Figure 1.1. In order to access and management these images, image annotation (e.g. adding metadata in the form of captions or keywords) is an efficient and valuable solution, which is helpful to exploit the text based search techniques to index and retrieve large collections of web images. However manual image annotation is impractical in many real world applications due to its high expenses. Another salient technique to address this challenge is *automatic image annotation* [29], which aims to adopt machine learning techniques to automatically assign the metadata.

Despite extensive studies [21, 29, 31, 94, 99, 107], the general image annotation technique still cannot satisfy the requirements of real-world applications. For example, it is very difficult to train a general model that works well for a large amount of images containing various kinds of objects. In general, the facial image may be one of the most important kinds of web images, which is closely related to human beings' everyday life. Hence automatic face annotation is so far the most effective sub-topic for general image annotation. In particular, auto face annotation aims to detect human faces from a photo image and to tag the facial image with the corresponding names. It can be beneficial to many real-world applications. For example, with

Figure 1.1: Web-based Photo-sharing Websites

automatic face annotation technique, online photo-sharing sites (e.g., Facebook) can automatically annotate uploaded photos to facilitate online photo management. Face annotation can also be applied in news video domain to detect important people appeared in the videos to facilitate news video retrieval and summarization tasks [78, 84].

1.1.1 Model-based Face Annotation

In general, classical face annotation approaches are treated as an extended face recognition problem, where different classification models are trained from a collection of well-labeled facial images by employing the supervised or semi-supervised machine learning techniques [11, 12, 119]. Based on well aligned facial images captured in controlled environment, the “model-based face annotation” scheme can achieve good performance. However, these techniques are limited in several aspects. Firstly, it is usually time-consuming and expensive to collect a large amount of human-labeled training facial images. Secondly, it is difficult to generalize the models when new training data or new persons are added, in which an intensive re-training process is usually required. Last but not least, the annotation/recognition performance often scales poorly when the number of persons/classes is large.

1.1.2 Search-based Face Annotation

Recently, some emerging studies have attempted to explore a promising search-based annotation paradigm for auto facial image annotation by mining the world wide web (WWW), where a mass number of weakly labeled facial images are freely available. Instead of training explicit classification models in the model-based face annotation approach, the search-based face annotation (SBFA) paradigm aims to tackle the face annotation task by exploiting content-based image retrieval (CBIR) techniques [91] and mining massive weakly labeled facial images on the web. The SBFA framework is data-driven and model-free, and somewhat is inspired by the search-based image annotation technique [103] for generic image annotation. The main objective of SBFA is to provide the correct name label for a given query facial image. In particular, given a novel facial image for annotation, we retrieve its short list of most similar facial images from a weakly labeled facial image database, and then annotate the novel query image by mining the top-ranking similar images and their corresponding weak label information.

Considering the importance of Search-based Face Annotation (SBFA) scheme in real world applications and its immense potential as a research topic, the objective of this thesis is to propose a family of robust and efficient algorithms to tackle the key challenges in SBFA framework by adopting the state-of-the-art techniques in various research areas, including computer vision, content-based image retrieval, and machine learning.

1.2 Main Challenges of Search-based Face Annotation

To the best of our knowledge, the SBFA problem is still a new research topic and little research progress has been reported so far. In general, the main objective of SBFA is to provide the correct name for a given query facial image. The paradigm of SBFA is inspired by the search-based image annotation technique for generic image annotation, since face annotation can be generally viewed as a sub-problem of generic image annotation, which has been extensively

Figure 1.2: An Overview of the proposed Search-based Face Annotation Framework

studied for several years but remains a fairly challenging problem. Since SBFA mainly focuses on a special kind of images: facial images, more extra information can be used to improve its annotation performance.

The Search-based Face Annotation (SBFA) system highly depends on the Content-Based Image Retrieval (CBIR) technique, which aims to retrieve a short list of similar images according to the visual content in the query image ¹. The CBIR problem has been extensively studied for several years [26, 80]. Currently, several commercial image search engines have already been released for general image retrieval. How to improve the image retrieval performance is out of the scope of this thesis. We only adopt existing facial image retrieval techniques. Based on the retrieval results, Search-based Face Annotation (SBFA) framework aims to effectively annotate the query facial image by exploiting the top-ranking similar facial images and the corresponding label information. This task is critical and challenging because the associated labels of web facial images are often noisy. In particular, there are three main challenges in the Search-based Face Annotation framework:

Challenge I: How to effectively enhance the initial weak name labels. Due to the nature of web images, there are lots of noisy images in the retrieval result, and the initial label information is also noisy. It's unreliable to use the information directly. Consequently, it's necessary to refine the initial weak labels at first, while the large image number makes the refinement work very challenging.

Challenge II: How to effectively exploit the short list of candidate facial images and their corresponding weak label information for the face name annotation task. This challenge is related to a series of research problems, for example, transfer learning, image re-ranking, and multi-label embedding. Although most of these related problems have been extensively studied for text based image retrieval. The technique of effectively exploiting the top-ranking similar candidate face images is rarely studied, cause of the special characters of facial images and the ineluctable noisy labels.

¹http://en.wikipedia.org/wiki/List_of_CBIR_engines

Challenge III: How to effectively boost the annotation performance of SBFA problem by adopting sophisticated machine learning techniques. The search-based annotation scheme can easily handle large amount of images, however, it is difficult for the traditional label propagation techniques to fully explore the name information among the top-ranking similar images. Machine learning technique aims to “learn” from the existing data and make correct “prediction” on the unseen data. Hence, given a small set of training samples, a suitable learning scheme could help to improve the system performance significantly. The key problem in this challenge is how to formulate the automatic face annotation problem in a correct form by adopting suitable machine learning techniques.

1.3 Contributions

To tackle the aforementioned challenges, this thesis investigates a family of effective and efficient algorithms. In addition, we also develop a real-world face annotation system “FANS—Face ANnotation by Searching Large-scale Web Facial Images”. The main contributes of this thesis are summarized as follows:

(i) **FANS System**

We develop a real-world face annotation system based on a large-scale web facial image database with a total of 6,025 western celebrities. An Android APP is also developed to popularize the usage of our system in the mobile. This system demonstrates the potential of searching and mining web-scale weakly labeled facial images on the internet to tackle the challenging face annotation problem. The live demo of FANS is available online.²

(ii) **Unsupervised Label Refinement (ULR)**

To tackle the first challenge, we propose an effective *Unsupervised Label Refinement* (ULR) approach for refining the labels of web facial images by adopting unsupervised

²<http://msm.cais.ntu.edu.sg/fans>

machine learning techniques. In particular, we formulate the learning problem as a convex optimization and develop effective optimization algorithms to solve the large-scale learning task efficiently. To further speed up the proposed scheme, we also propose a clustering-based approximation algorithm which can improve the scalability considerably. An extensive set of empirical studies are conducted, in which encouraging results show the proposed ULR algorithms can significantly refine the weak label information and boost the face annotation performance.

(iii) **Weak Label Regularized Local Coordinate Coding (WLRLCC)**

To tackle the second challenge, we propose an effective *Weak Label Regularized Local Coordinate Coding* (WLRLCC) technique, which exploits the principle of local coordinate coding by learning sparse features, and employs the idea of graph-based weak label regularization to enhance the weak labels of the top-ranking similar facial images. Moreover, an effective sparse reconstruction scheme is developed to perform the face annotation task. To further reduce the off-line running time, we propose an offline approximation scheme (AWLRLCC), which generally maintains comparable results but significantly reduces the annotation time.

(iv) **Fusion of Transductive and Inductive Learning**

To tackle the third challenge, we aim to solve the automatic face annotation problem by combining the “transductive learning” and “inductive learning” in a last-fusion scheme. We proposed a *Unifying Transductive and Inductive Learning* (UTIL) algorithm to efficiently combine the two schemes together. In particular, from the “inductive learning” view, we propose a new *Weak Label Laplacian Support Vector Machine* (WL-LapSVM) algorithm for generating effective classification models from weakly labeled web facial images; from the view of “transductive learning”, we apply the *Weak Label Regularized Local Coordinate Coding* (WLRLCC) algorithm; finally, we propose an entropy-based

combination scheme to combine the annotation results from the two different learning schemes to maximize the learning efficacy.

(v) **Multimodel Learning for Learning to Name Faces (L2NF)**

To tackle the third challenge and go beyond the limitation of single facial feature representation, we propose a multimodel based Learn to Name Faces (L2NF) algorithm, which optimize the fusion of multiple annotation modalities in a learning to rank framework and significantly boost the annotation performance. In particular, the proposed L2NF algorithm attempts to learn both the optimal *weight vector* for combining different query-neighbor similarity functions and the *refined labels* for enhancing the initial weak labels simultaneously in a unified learning framework. We propose a multimodal learning scheme that (i) first constructs multiple diverse facial features for representing the faces, (ii) further optimizes the distance measure in each feature space using metric learning, and (iii) finally learns the optimal fusion of the multiple representations by adapting the structural SVM algorithm.

1.4 Scope and Organization

This thesis focuses on solving the automatic face annotation problem in the Search-based Face Annotation scheme. The related works including face recognition, general image annotation, model-based face annotation, machine learning techniques are reviewed. A family of efficient and effective algorithms is proposed to improve the annotation performance and achieve state-of-the-art performance based on several real-world facial image databases. Furthermore, a real-world large-scale face annotation system is released. The rest of this thesis is organized as follows:

- In Chapter 2, this chapter reviews some background knowledge related to the main problem that will be discussed in this thesis.

- In Chapter 3, this chapter presents the framework of “Search-based Face Annotation”, where the three main stages are introduced in details. I also present the real-world demo system “FANS” and the Android APP to popularize mobile access.
- In Chapter 4, this chapter mainly investigates the first challenging problem, and introduces the Unsupervised Label Refinement (ULR) algorithm.
- In Chapter 5, this chapter mainly investigates the second challenging problem, and introduces the Weak Label Regularized Local Coordinate Coding algorithm.
- In Chapter 6, this chapter aims to solve the automatic face annotation problem by unifying transductive learning and inductive learning in a last-fusion scheme.
- In Chapter 7, this chapter aims to solve the automatic face annotation problem by adopting multi-model learning techniques, which optimizes the fusion of multiple modalities in different facial feature spaces, and introduces the proposed “Learning to Name Faces” (L2NF) algorithm.
- In Chapter 8, the last chapter summarizes the achievements of this thesis and addresses some directions to be explored in future work.

Each chapter of the thesis is intended to be self-contained. Thus, in some chapters, some definitions, formulas, or illustrative figures that have already appeared in previous chapters, may be briefly reiterated.

Chapter 2

Literature Review

In this chapter, We mainly review the existing research work related to various aspects of Search-based Face Annotation (SBFA), including “*general face recognition and verification*”, “*general image annotation*”, “*face annotation*”, “*machine learning*”, and “*sparse representation*”. More details are introduced in the following sections.

2.1 General Face Recognition and Verification

The first group of related works is on the topics of face recognition and face verification, which are classical research problems in computer vision and pattern recognition. They have been extensively studied for many years and lots of algorithms have been proposed, for example, the principle component analysis(known as Eigenface) [96], fisher discriminant analysis (known as Fisherface) [9], line edge maps (LEMs) [35], elastic bunch graph matching (EBGM) [104], locality preserving projections(known as Laplacianface) [19,41].

Conventional face annotation approaches usually adapt some existing face recognition techniques by training multi-class face classification models from a collection of human-labeled facial images based on supervised machine learning techniques [11, 12, 110]. Although the “*model-based face annotation*” model is limited in several aspects and not suitable for web-scale annotation task, the face recognition and verification techniques are still important to the face annotation task. The comprehensive discussions on the development of face recognition

and face verification techniques can be found in the survey papers [1, 16–18, 47, 93, 121] and books [25, 48, 53].

In general, the traditional studies are limited for the high-quality databases collected in well-controlled environments. Recent years have observed some emerging benchmark studies of unconstrained face recognition and verification techniques on facial images that are collected from the web, such as the LFW benchmark studies [20, 46, 71]. To handle the real-world facial images, several representative face recognition and verification algorithms are proposed for web environment.

The sparse representation based classification (SRC) has been successfully used in computer vision domain, especially for face recognition task. John Wright et al. [105], proposed to encode the input testing facial image as a sparse linear combination of the training samples via l_1 -norm minimization. Following this work, Yang and Zhang proposed the GSRC algorithm, [112] which adopted the Gabor local features to compress the occlusion dictionary. Comparing with SRC algorithm, the GSRC algorithm needs less time cause of the compression (about 40 : 1 – 50 : 1) from original occlusion dictionary to computed Gabor occlusion dictionary. A similar face recognition scheme was proposed by Shi and Li [88] based on hashing. They argued that their algorithm was up to 150 times faster than the SRC algorithm on the YaleB database. Their experimental results show that the hashing-based sparse representation can be recovered with a high probability because hashing method preserves the restrictive isometry property. In this thesis, we adopt the SRC algorithm in the proposed Weak Label Regularized Local Coordinate Coding (WLRCC) algorithm to learn the sparse representations of the top-ranking similar facial images. For more details, please refer to the Chapter 5.

Facial feature is also critical factor for the performance of face recognition and verification. The Local Binary Patterns (LBP) [4, 20] is an efficient facial feature, which has been widely used in several face recognition systems. For example, it has been implemented in the famous computer vision library OPENCV. In the past years, GIST descriptor has received increasing

attention in the context of sense recognition. [89] In this thesis, according to our experiments, the GIST feature also achieves comparable results with the LBP feature for representing web-scale facial images. More details, please refer to the Chapter 3.2.4.

Unlike the previous manually designed encoding methods (e.g., LBP or GIST), Cao et al. [20] proposed a learning-based facial descriptor to verify whether two facial images belong to the same person. They adopted unsupervised learning techniques to learn an encoder from the training examples, which automatically achieved appropriate tradeoff between discriminative power and invariance. More recently, Yin et al. [113], proposed an “Associate-Predict” model to address the intra-personal variation problems.

Distance metric learning (DML) is another technique to enhance facial feature, in which all feature vectors are projected into a new powerful feature space. For example, in [38], two approaches were proposed for the face identification problem: (1) a logistic discriminant approach which learned the metric from a set of labeled image pairs (LDML) and (2) a nearest neighbor approach which computed the probability of two images belonging to the same class (MkNN). The LDML and MkNN algorithms are both based on the Large Margin Nearest Neighbor (LMNN) algorithm and share nearest neighbor for clustering. In [71], Nguyen and Li proposed a cosine similarity metric learning techniques by generalizing the traditional Euclidean based metric learning. All of the previous works were tested on the state-of-the-art face verification dataset: Labeled Faces in the Wild (LFW) [46], and achieved excellent accuracy comparing with other literatures. By adopting learning to ranking algorithm, Mcfee et al. proposed the “Metric Learning to Rank” (MLR) algorithm, [66] which learned a metric such that rankings of data induced by the learned distance were optimized against a ranking loss measure (e.g. ROC area (AUC) or MAP). In the thesis, we adopt the MLR algorithm in the proposed Multimodal Learning for Learning to Name Faces (L2NF) algorithm for multimodal representation construction.

2.2 General Image Annotation

The second group of related works is general image annotation. In general, the image annotation approaches [21, 29, 31, 40] will apply some existing object recognition techniques to train classification models from human-labeled training images or attempt to infer the correlations probabilities between the images and annotation key words. A comprehensive survey of image annotation could be found in [40].

In recent years, the search-based image annotation paradigm, which relies on Content-based Image Retrieval (CBIR) [85], has attracted more and more attention [82, 83, 98, 103]. The search-based image annotation scheme aims to solve the annotation problem based on the web environment. For example, Russell et al. [83] developed a large collection of web images with ground truth labels to facilitate object recognition research. Wang et al. [98] proposed an annotation result refinement approach which combined both the textural information and visual information. In order to avoid normalization, dimension curse and redundancy, Emre et al. [5] proposed a two-layer ensemble learning system, named Supervised Annotation by Descriptor Ensemble (SADE). The SADE system, firstly, extracted a variety of low-level visual descriptors from the image; then fed these descriptors to a separate learning machine in the first layer; finally, a meta-layer classifier, which was trained on the output of the first layer classifiers, was used to annotation images.

Despite the indexing, searching and feature extraction techniques, label propagation is also an extremely important problem for image annotation task. Recently there is a surge of interests to solve the propagation problem by exploring the sparse coding techniques, which has been demonstrated effective in many computer vision applications. An introduction of sparse coding will be presented in the following section 2.5. In [94], a kNN-sparse graph-based semi-supervised learning (SGSSL) approach was proposed to annotate a large-scale web image, in which a sparse graph was built with the sparse representation technique and a semi-supervised

inference was employed to assign label information to the unlabeled data. In order to handle the noisy training labels, a pseudo ideal label vector was introduced and solved iteratively.

Wang et al. [99] proposed another sparse coding based annotation framework (MSC), where the sparse representation was used for feature extraction and label propagation. Firstly they built a semantic graph based on the ℓ_1 -minimization with the label matrix as input. Then they learned a linear transformation matrix for feature dimension reduction with the label-based graph. Using the obtained linear transformation matrix, all the training and test samples were projected into a low-dimensional feature space, where sparse reconstruction was employed again for each test instance with all the training data as dictionary. At last, these sparse coefficients for test instances were further used to propagate the label information from training set to these test instant. The MSC algorithm depends on a well-defined training set, which may be very expensive in building in real word applications. In [107], the group lasso algorithm was directly employed for feature regression, in which a group sparse coefficient vector was achieved for each class (category) and further used for predicting new instants.

2.3 Automatic Face Annotation

The third group of related works is face annotation techniques. Beside of the convectional “*model-based face annotation*” works, the web scale face annotation has been a hot research pot in recent years. In general, these works can be generally classified into four categories:

- *Personal Photo Based Face Annotation (PBFA)*
- *Name-based Face Annotation (NBFA)*
- *Caption-based Face Annotation (CBFA)*
- *Search-based Face Annotation (SBFA)*

2.3.1 Personal Photo Based Face Annotation

For PBFA problem, the collections of personal photos contain rich context clues, such as times tamps, GPS tags, social context, and so on, which are proven crucial important clues to improve the annotation performance.

For the family photograph management systems, Zhang et al. [119] formulated face similarity measure as maximum posteriori (MAP) estimation in a Bayesian framework. For the face representation they used three level features: contextual features (body information), face appearance features (face information) and facial components features (eye information). The experiment results illustrated that the extra feature information were helpful for the final annotation task.

Choi et al. [23] proposed a novel face annotation method that systematically combined the contextual information (time, space, cloth) of photo with the traditional face recognition technologies. With *situation clustering* based on contextual information and *person clustering* with efficient fusion of person-identity features, an useful “person cluster” of multiple facial images was obtained. Each facial image was annotated after projecting the measurement-level fusion features into a low-dimensional space.

Jingyu Cui [24] implemented an “EasyAlbum” software for personal photos management, as shown in Figure 2.1. Four kinds of features were used in the system: scene feature, time tag, facial feature and cloth feature. With combination of location clustering, face clustering and re-ranking for Ad hoc annotation, their scheme could substantially improve the annotation performance.

As a conclusion, the PBFA algorithm can achieve an impressive result. Hence, several successful commercial applications are also developed, e.g. iPhoto, Picasa.

Figure 2.1: The System Framework of Easyalbumn

2.3.2 Name-based Face Annotation

For NBFA problem, the input query is a person name, and the main aim is to refine the text-based retrieval result by employing visual facial context information, which is fairly close to the automatic image re-ranking problem.

Ozkan and Duygulu [76] proposed a method to associate names and faces for the querying name among a large collection of news photos, with the assumption that a person’s face was likely to appear when the name was mentioned in the corresponding caption. Given a set of images retrieved for a name query, a graph model was used to find the densest sub-graph as the most related result, as shown in Figure 2.2(a). Based on the similar assumption and problem setting, Le and Satoh [55] proposed a new local density score to represent the importance of each returned images for the name query in the first step. In the second step, they proposed a new SVM machine “Rank-By-Bagging-ProbSVM” to re-ranking the ranking step one’s result. Figure 2.2(b) illustrated the system proposed in [55], which was also commonly used in NBFA system.

Figure 2.2: Name-based Face Annotation

2.3.3 Caption-based Face Annotation

Instead of filtering the text-based retrieval result, the main task of CBFA problem is to directly annotate each facial image with the names extracted from its caption information.

In [11, 13], Berg et al. proposed to combine a simple appearance model using kPCA and LDA, with a language model based on context for name assignment. Specially, they used expectation maximization (EM) procedure to link a face appearance model. Phi The Pham et al. [77] generalized the framework of Berg [11, 13], playing with this intrinsic (a) symmetry between names and faces. Different models were systematically proposed depending on the task and focus to assign names to the faces or to assign faces to the names or to identify pairs of matching names and faces.

Figure 2.3: Caption-based Face Annotation

Matthieu Guillaumin et al. [36] formulated the naming people with captions problem into two scenarios: i). finding faces of a single person; ii). Assigning names to all faces. For the

first scenario, they followed the graph-based approach in [76] with a series of modification, which achieved better performance. For the second scenario, they extended the graph-based approach by finding sub graphs of the similarity graph for each name with a graph-flow based scheme for computing acceleration.

The caption based annotation task can also been applied into video processing. In [90], Josef Sivic et al. investigated the problem of automatically labeling faces of characters in TV or movie material with their names, using only weak supervision from automatically aligned subtitle and script text. In order to discriminate different tracks belonging to different persons, they adopted the multi-kernel SVM.

2.3.4 Search-based Face Annotation

The SBFA problem is a fairly new research problem, in which the facial images are used as input queries directly. This thesis mainly focus on the SBFA problem. As discussed in the first chapter, there are mainly three challenging problems in the SBFA scheme.

Neeraj Kumar et al. [54] created an image search engine with about 3.1 million faces which have been automatically labeled on the basis of several facial attributes. Their implemented engine was, however, a “text-query” based facial image search system, which supplied the queries such as “smiling men with blond hair and mustaches”. More Recently, Thao Ngoc Nguyen et al. [72] proposed an efficient method for face retrieval in large video datasets. To make face retrieval in video database robust, a single face track was used for feature representation. To reduce the computation cost they proposed a k -Faces scheme to select representative faces for similarity matching.

Considering the web-scale face retrieval task, Zhong Wu et al. [108] proposed a facial image search system with identity-based quantization and multi-reference re-ranking, which was mainly related to the first challenge problem. They argued that the convectional global high dimension facial features were not suitable for quantization and inverted indexing, while

the bag-of-word representation scheme degraded significantly when applying on face images. As a combination, they proposed an identity- based quantization scheme to quantize locate component-based local features for efficient indexing, as shown in Figure 2.4. They also proposed a multi-reference re-ranking scheme to introduce the global feature information.

Figure 2.4: Local feature extraction and indexing

2.4 Machine Learning Techniques

Machine learning can be broadly defined as computational methods using experience to make accurate predictions. The *experience* refers to the past information available to the learner, which typically takes the form of electronic data collected and made available for analysis. In other words, *machine learning* focuses on the construction of prediction systems that can learn from training data, which is different with *data mining* that aims to discover unknown properties on the data [44]. In the past decade, machine learning techniques have widely used in many applications. In general, *machine learning* can be classified into three categories based on the type of input available during training the model: *supervised learning*, *semi-supervised learning* and *unsupervised learning*.

In this thesis, we aim to solve the web face annotation problem in different aspects. A series of algorithms are proposed by adopting various machine learning techniques. For example, to refine the label information over the whole retrieval database, we proposed the Unsupervised Label Refinement (ULR) algorithm in Chapter 4, which was derived by the graph-based

learning technique. In Chapter 5, we proposed the Weak Label Regularized Local Coordinate Coding (WLRCC) algorithm based on sparse representation technique. In Chapter 6, we proposed a Weak Label Laplacian SVM (WL-LapSVM) algorithm following the traditional semi-supervised learning technique, and adopted an entropy-based last-fusion scheme to unify the “*transductive learning*” and the “*inductive learning*” to achieve the final annotation result. In Chapter 7, we solved the face annotation problem in a supervised multimodel learning scheme by adopting various facial feature that were refined with distance metric learning techniques.

2.4.1 Supervised Learning

Supervised learning considers the problems of estimating certain functions from examples with label information. In supervised learning, each input data instance is associated with some corresponding training label, which is assumed to be the response from a supervisor. Based on the input and output examples, the learner has to adjust the parameters of the system autonomously, so that it can yield the correct output pattern when excited with one of the given input patterns. One important type of supervised learning is “*inductive learning*”. In particular, a broad family of statistical learning theories has been studied to achieve the risk minimization and the generalization maximization in the learning tasks.

Suppose there is a probability distribution P on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \in \mathbb{R}^n$ is the data space where data are generated, and $\mathcal{Y}$ is the label space. In supervised learning scheme, we are given a training set of l independent and identically distributed observations, which are generated according to P :

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_l, y_l)$$

where $\mathbf{x}_i$ is the data vector and y_i is the associated response by a supervisor. In the supervised learning scenario, the goal is to learn a function f to minimize the expected loss called risk

functional [39]:

$$R(f) = \int L(\mathbf{x}, y, f(\mathbf{x})) dP(\mathbf{x}, y) \quad (2.1)$$

where L is the loss function, and a varies of loss functions are proposed in the literature.

In most applications, the probability distribution P is unknown. The problem, therefore, is to minimize the risk function when the probability distribution function $P(\mathbf{x}, y)$ is unknown but the labeled data $\{(\mathbf{x}_i, y_i), 1 \leq i \leq l\}$ is given. Thus, we need consider the empirical estimate of the risk functional [39]:

$$R_{emp}(f) = \frac{1}{l} \sum_{i=1}^l L(\mathbf{x}_i, y_i, f(\mathbf{x}_i)) \quad (2.2)$$

Since directly minimizing the previous empirical risk may lead to numerical instabilities and bad generalization performance. A possible way to avoid this problem is to add a regularization term. A general choice of the stabilization term is the norm in a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_K$ defined over the positive definite kernel function K : $\|f\|_K^2$. As a result, to learn the function f , we can minimize the following regularized risk functional:

$$f^* = \arg \min_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^l L(\mathbf{x}_i, y_i, f(\mathbf{x}_i)) + \lambda \|f\|_K^2 \quad (2.3)$$

λ is the regularization parameter, and the whole penalty norm $\lambda \|f\|_K^2$ imposes smoothness conditions on the solution space. The solution of the previous formulation is determined by the loss function L and the kernel K . By using different loss function, we can generate different algorithms, for example [44]:

(i) Regularized Least Squares Networks (RLS):

$$L(\mathbf{x}, y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$$

(ii) Support Vector Machine Regression (SVMR):

$$L(\mathbf{x}, y, f(\mathbf{x})) = (y - f(\mathbf{x}))_\epsilon$$

(iii) Support Vector Machine Classification (SVMC):

$$L(\mathbf{x}, y, f(\mathbf{x})) = (y - f(\mathbf{x}))_+$$

where $(\cdot)_\epsilon$ is the Vapniks epsilon-insensitive norm and $(\cdot)_+$ is the hinge loss in which $(x)_+ = x$ if x is positive and zero otherwise, and y_i is a real number for both RLS and SVMR, and takes $+1$ or -1 for SVMC. A variety of kernels have been considered in the literature, and three commonly-used kernel functions are listed as follows:

- polynomial kernel: $K(\mathbf{x}, \mathbf{x}_i) = (\langle \mathbf{x}, \mathbf{x}_i \rangle + c)^d$
- Gaussian radial basis kernel: $K(\mathbf{x}, \mathbf{x}_i) = \exp(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2})$
- sigmoid kernel: $K(\mathbf{x}, \mathbf{x}_i) = \tanh(\kappa \langle \mathbf{x}, \mathbf{x}_i \rangle + \theta)$

The famous Support Vector Machine (SVM) algorithm can be easily interpreted from the previous statistical regularization learning framework. More specifically, SVM can be formulated as a similar regularized learning problem:

$$f^* = \arg \min_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^l (1 - y_i * f(\mathbf{x}_i))_+ + \lambda \|f\|_K^2 \quad (2.4)$$

Another general formulation of SVM is to introduce the loss functions as constrains:

$$\begin{aligned} \min_{\mathbf{w}, \xi, b} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^l \xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \Phi(\mathbf{x}_i) - b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, 2, \dots, l. \end{aligned} \quad (2.5)$$

where C is a penalty parameter of the error term ξ_i , which is equivalent to the regularization parameter $\frac{1}{2\lambda}$. λ is the parameter in the regularization framework, $\Phi(\cdot)$ is a kernel mapping function, and the labels y_i are either $+1$ or -1 for a regular binary classification problem. The solution to the above convex optimization problem can be found by introducing the Lagrange functional technique [44].

2.4.2 Semi-supervised Learning

Semi-supervised learning, typically *transductive learning*, deals with classification tasks through utilizing both labeled and unlabeled examples. In the semi-supervised learning scenario, in addition to l labeled data points $\{(\mathbf{x}_i, y_i), 1 \leq i \leq l\}$ we are given, we also have u unlabeled data points $\mathbf{x}_i, l+1 \leq i \leq l+u$, which are drawn according to the marginal distribution P . The decision function is learned from both the labeled data and the unlabeled data. In general, the number of unlabeled examples u is much larger than the labeled ones l . In recent research studies, many methods have been proposed for solving semi-supervised learning problems, such as EM with generative mixture models [73], co-training [14], self-training [81], Transductive Support Vector Machines [49], and graph-based methods [125]. A comprehensive survey can be found in [124].

Following the manifold regularization approach, Laplacian Support Vector Machines (LapSVM) have shown the state of the art performance in semi-supervised classification. We define a kernelized target function $f(\mathbf{x})$, and define $n = l + u$:

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) = \boldsymbol{\alpha}^\top \mathbf{k}(\mathbf{x}) \quad (2.6)$$

where $\kappa(\cdot, \cdot)$ is a kernel function, $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \dots, \alpha_n]$ and $\mathbf{k} = [\kappa(\mathbf{x}_1, \mathbf{x}), \kappa(\mathbf{x}_2, \mathbf{x}), \dots, \kappa(\mathbf{x}_n, \mathbf{x})]$. The Laplacian SVM problem aims to minimize the following objective function with respect to the previous classification function f .

$$\min_{f \in \mathcal{H}_K} g(\boldsymbol{\alpha}) = \sum_{i=1}^l L(\mathbf{x}_i, y_i, f) + \frac{\lambda_1}{2} \Phi(f) + \frac{\lambda_2}{2} \|f\|_K \quad (2.7)$$

where $\|\cdot\|_K$ is the norm in the Reproducing Kernel Hilber Space (RKHS) $\mathcal{H}_K$ of kernel k . L is a loss function on the label data. It can be L_1 hinge loss function

$$L(\mathbf{x}_i, y_i, f) = \frac{1}{2} \sum_{i=1}^l \max(1 - y_i f(\mathbf{x}_i), 0) \quad (2.8)$$

or L_2 hinge loss function:

$$L(\mathbf{x}_i, y_i, f) = \frac{1}{2} \sum_{i=1}^l \max(1 - y_i f(\mathbf{x}_i), 0)^2 \quad (2.9)$$

$\Phi(f)$ is an intrinsic regularizer to employ the geometry information among the label data and unlabeled data by the Laplacian matrix, which is defined as :

$$\Phi(f) = \sum_{i=1}^n \sum_{j=1}^n W_{ij} (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2 = \boldsymbol{\alpha}^\top K L K \boldsymbol{\alpha} \quad (2.10)$$

where K is the kernel matrix of the instances in the training set, L is the graph Laplacian matrix, and $L = D - W$, W is the adjacency matrix of the data graph in training set. D is a diagonal matrix with diagonal elements as $D_{ii} = \sum_{k=1}^n W_{ik}$.

2.4.3 Unsupervised Learning

Unsupervised learning considers the problem of learning from a collection of data instances without training labels. Its goal is to discover “interesting/useful structure” in the data, which is sometimes called *knowledge discovery*. Unsupervised learning is arguably more typical of human and animal learning. It is also more widely applicable than supervised learning or semi-supervised learning, since it does not require a human expert to manually label the data. However, it also contains relatively little information, certainly not enough to reliably estimate the parameters of complex models [69]. One of the most popular areas of study in unsupervised learning is data clustering techniques, which have been widely used for data mining applications. In general, clustering aims to organize a set of data items into various clusters, such that items within a cluster are more “similar” to each other. More details can be found in [15].

The K-Means clustering algorithm is one of the most popular clustering techniques [51]. The main idea of K-Means is to divide data examples so that the within-group scatter is minimized. In particular, assume that we have a set of data samples $\{\mathbf{x}_i\}_{i=1}^N$, the K-Means method contains the following steps:

(i) Initialize a random partition: $\{\mathcal{C}_i\}_{i=1}^k$, where k is the number of clusters

(ii) Update assignments until convergence:

for each $\mathbf{x}_j$, assign $\mathbf{x}_j \rightarrow \mathcal{C}_p$, where $p = \arg \min_i \|\mathbf{x}_j - \mu_i\|_M$ and $\mu_i = \frac{1}{|\mathcal{C}_i|} \sum_{\mathbf{x} \in \mathcal{C}_i} \mathbf{x}$

where M is a distance metric among a family of Mahalanobis Metrics. The Euclidean distance metric is typically used by default for K-Means clustering. The K-Means method is efficient in many applications. However, it is also limited in several aspects. For example, it's sensitive to the initial centroid position. To address this problem, the Bisecting K-means methods is proposed, which is a combination of K-means and hierarchical cluster algorithm.

2.5 Sparse Representation

Recently, along with the advances of the compressive sensing theory and sparse representation (coding) techniques [57, 58, 70], researchers have attempted to apply the emerging sparse coding techniques for many computer vision and pattern recognition tasks, for example, image processing [74], image classification [33, 102, 114], face recognition [87, 92, 105, 112].

Sparse coding (SC) aims to represent each input instance $\mathbf{x} \in \mathbb{R}^d$ using a set of basis vectors $\{\mathbf{b}_j \in \mathbb{R}^d, j = 1, \dots, n\}$ with a sparse coefficient vector $\mathbf{s} \in \mathbb{R}^n$, such that $\mathbf{x} \approx \sum_j \mathbf{b}_j s_j$. Let us denote by matrix $X \in \mathbb{R}^{d \times N}$ for a set of N input data instances, denoted by $B = [B_1, B_2, \dots, B_n] \in \mathbb{R}^{d \times n}$ the basis matrix, and denoted by $S \in \mathbb{R}^{n \times N}$ the sparse coefficient matrix, then the sparse codes problem could be formulated as the following optimization problem [57]:

$$\min_{B, S} \frac{1}{2} \|BS - X\|_F^2 + \lambda \sum_{i,j} \phi(S_{ij}) \quad s.t. \quad \|B_i\|^2 \leq c, \forall i = 1, \dots, n. \quad (2.11)$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix, $\phi(\cdot)$ is a sparsity penalty function (a typical choice is an L_1 penalty function, i.e., $\phi(\mathbf{s}_j) = \|\mathbf{s}_j\|_1$), c is a constant, and λ is a parameter to balance tradeoff between *fitness* and *sparsity*.

The optimization problem in Eq. 2.11 for B and S is non-convex, as a result, generally it is separated into two optimization tasks: (1) Coefficient Learning, i.e., find the solution of S by fixing the dictionary B ; and (2) Dictionary Learning, i.e., find the solution of dictionary B by fixing S . In the past years, a variety of sparse coding algorithms have been proposed to improve sparse coding in different aspects, which mainly consist of two groups, one is related to the “Dictionary Learning” task and another is related to the “Coding Learning” task.

2.5.1 Dictionary Learning

For the dictionary learning with fixed coding S , the problem is a least square problem with quadratic constraints. The learned over-complete dictionary is supposed to be useful in image restoration task [2, 64] and image denoising task [86].

Figure 2.5: Dictionary Learning for image restoration

There are several approaches to solve this problem, such as generic convex optimization solvers (e.g., QCQP solver) as well as gradient descent using iterative projections. Usually the general QCQP solver is very time consuming and gradient algorithm is slow to converge, in [57], the optimal dictionary B is found by solving the Lagrange dual problem and B is achieved with the following equation:

$$B^T = (SS^T + \Lambda)^{-1}(XS^T)^T \quad (2.12)$$

where $\Lambda = \text{diag}(\vec{\lambda})$ is the matrix of dual variables that is found by maximizing the following Lagrange dual:

$$\max_{\lambda} \text{tr}(X^T X - XS^T(SS^T + \Lambda)^{-1}(XS^T)^T - c\Lambda) \quad (2.13)$$

The advantage of solving the dual is that it uses significantly fewer optimization variables than the primal and the above optimization can be simply solved by the Newton's method or the conjugate gradient method.

However, the previous algorithm is unsuitable for very large training dataset where the dimension of B is very huge. Mairal et al. [62] proposed an online sparse coding framework, which iteratively learns the dictionary by including a small set of training data each time. In details, the algorithm is illustrated as Figure 2.6, where input instant x_t is received at time- t , then its sparse coding α_t is archived with the previous dictionary D_{t-1} , at last the dictionary is updated by block-coordinate descent scheme with warm restarts, which is parameter-free and does not require any learning rate tuning. Cause the traditional sparse coding is an unsuper-

```

1: for  $t=1, \dots, T$  do
2:   Draw  $\mathbf{x}_t$ 
3:   Sparse Coding
      
$$\alpha_t \leftarrow \arg \min_{\alpha \in \mathbb{R}^k} \frac{1}{2} \|\mathbf{x}_t - \mathbf{D}_{t-1} \alpha\|_2^2 + \lambda \|\alpha\|_1,$$

4:   Dictionary Learning
      
$$\mathbf{D}_t \leftarrow \arg \min_{\mathbf{D} \in \mathcal{C}} \frac{1}{t} \sum_{i=1}^t \left( \frac{1}{2} \|\mathbf{x}_i - \mathbf{D} \alpha_i\|_2^2 + \lambda \|\alpha_i\|_1 \right)$$

5: end for
    
```

Figure 2.6: Online Dictionary Learning Approach

vised scheme, Julien Mairal et al. [63] extended the unsupervised sparse coding approach to supervised cases by combining the classification models and sparse coding.

2.5.2 Coding Learning

Given a fixed dictionary B , the sparse representation $\mathbf{s}$ for the input instant $\mathbf{x}$ could be achieved with the following optimization problem:

$$\min_{\mathbf{s}} \frac{1}{2} \|\mathbf{x} - B\mathbf{s}\|_2^2 + \lambda \cdot \sigma_1(\mathbf{s}), \text{ where, } \sigma_1(\mathbf{s}) = \|\mathbf{s}\|_1 \quad (2.14)$$

where λ is the weighting parameter for regularizer. Instead of using the original l1-norm regularizer $\sigma_1 = \|\mathbf{s}\|_1$, a series of research works are proposed various regularizers with different assumptions to address the limitations in the original sparse representation scheme.

Suppose the dictionary B could be divided into L groups, Yuan and Lin [115] proposed the group lasso algorithms to select a group of features simultaneously for a given task(s). The regularizer of group lasso is defined as following:

$$\sigma_{\text{GL}}(\mathbf{s}) = \sum_{l=1}^L \sqrt{(w_l)} \|\mathbf{s}_l\|_2 \quad (2.15)$$

where $\mathbf{s} = [\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_L]$, $\mathbf{s}_i$ is the coefficients corresponding to the dictionary items belonging to the i -th group, and w_l is the weighting value for i -th group. The key assumption behind the group lasso regularizer is that if a few features in a group are important, then most of the features in the same group should also be important.

On the contrary, in order to capture the negative correlation among groups, Yang Zhou et al. [122] propose an exclusive lasso regularizer, as in Eq 2.17. Different from the group lasso regularizer, if one feature in a group is given a large weight, the exclusive lasso regularizer tends to assign small or even zero weights to the other features in the same group.

$$\sigma_{\text{EL}}(\mathbf{s}) = \sum_{l=1}^L (\|\mathbf{s}_l\|_1)^2 \quad (2.16)$$

In order to yield sparsity within a group, Jerome Friedman et al. proposed a sparse group lasso regularizer [32], which was defined as following:

$$\sigma_{\text{SGL}}(\mathbf{s}) = \|\mathbf{s}\|_1 + \lambda' \sum_{l=1}^L \|\mathbf{s}_l\|_2 \quad (2.17)$$

In standard sparse coding, the instant $\mathbf{x}$ is described as a combination of dictionary B 's elementary features involving both additive and subtractive interactions. The fact that features can “cancel each other out” using subtraction is contrary to the intuitive notion of combining parts to form a whole [56], Patrik O. Hoyer proposed a non-negative sparse coding scheme,

where an extra positive constraints $\mathbf{s} \geq 0$ were introduced to the traditional sparse coding scheme:

$$\sigma_{\text{NSC}}(\mathbf{s}) = \sum_{i=1}^n s_i, \quad \text{where } s_i \geq 0. \quad (2.18)$$

In [114], Kai Yu et al. proposed a new method for high dimensional nonlinear learning with data distributed on manifolds. A base was learned in an unsupervised basis learning scheme and all the points could be locally approximated by a linear combination of its nearby anchor points. They further showed that a high dimensional nonlinear function could be approximated by a global linear function with respect to this previous coding result. The proposed scheme has an intimate relationship with sparse coding and they regard the local coordinate coding as locally constrained sparse coding, which owns a different regularizer as following:

$$\sigma_{\text{LCC}}(\mathbf{s}) = \sum_{i=1}^n \|B_{\star,i} - \mathbf{x}\|_2 \cdot |s_i| \quad (2.19)$$

where $B_{\star,i}$ is the i -th basis (column) in dictionary B .

Following the observation that locality is more essential than sparsity, as locality must lead to sparsity but not necessary vice versa, in [102] Jinjun Wang et al. proposed a Locality-constrained sparse coding for the coding step in the traditional SPM approach.

Guao et al. [33] pointed out two limitations of general sparse coding scheme: (i), Due to the over-complete codebook/dictionary, even a small variance in local features may result in totally different responses to the basis in the codebook the dependence information among the local features is lost during sparse coding process, (ii), the dependence information among the local features is lost during sparse coding process, in order to address the previous two limitations, they proposed the Laplacian sparse coding (LSC) algorithms as following:

$$\min_{S,B} \frac{1}{2} \|BS - X\|_F^2 + \lambda_1 \sum_{i=1}^N \|\mathbf{s}_i\|_1 + \lambda_2 \frac{1}{2} \sum_{i,j=1}^N W_{ij} \|\mathbf{s}_i - \mathbf{s}_j\|^2 \quad s.t. \quad \|B_i\|^2 \leq c, \forall i = 1, \dots, n. \quad (2.20)$$

where W is the similarity matrix among all the items in X , λ_1 , λ_2 are two regularization parameters, c is positive constraint.

Spare coding algorithm has also been adopted and used into many different tasks, for example, in order to encourage using the same dictionary words for all the items belonging to a class, Samy Bengio et al. proposed group sparse coding (GSC) for bag-of-words document representations. For the items in the same, group, instead of learning the sparse coding for each item separately, GSC uses the following regularizer:

$$\sigma_{\text{GSC}} = \sum_{j=1}^n \|S_{j,\star}\|_p \quad (2.21)$$

where $S = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_N\}$ is the sparse coding matrix of all the input items X . In order to learning the sparse coding for different tasks in the same time, Xiao-Tong Yuan et al. proposed a multi-task joint sparse representation scheme [116]. More information about sparse coding and its application in image analysis could be found in the tutorial given by Francis Bach [7].

Chapter 3

Search-based Face Annotation: An Overview

In this chapter, we briefly introduce the proposed Search-based Face Annotation (SBFA) system, as shown in Figure 3.1. The SBFA framework consists of the following modules:

- (i) The facial image collection step, in which a web image crawler downloads the weakly-labeled facial images with web search engines.
- (ii) The facial image preprocessing step, which consists of *face detection* (FD), *face alignment* (FA), *feature extraction* (FE), *feature indexing* (FI).
- (iii) The content-based facial image retrieval module, which searches the similar facial images for the query facial image.
- (iv) The automated face annotation module for naming the query by mining the top-ranking retrieved similar facial images and their corresponding weakly labels.

In general, the first two steps are applied in the off-line scheme, which can be conducted before the online face annotation task. The last two steps are conducted during the test phase, which thus should be applied very efficiently. The details of each module are described as follows.

Figure 3.1: The system flow of Search-based Face Annotation (SBFA) scheme. (a) We collect weakly labeled facial images from WWW using web search engines. (b) We pre-process the crawled web facial images, including face detection, face alignment, and feature extraction for the detected faces; after that, we apply LSH to index the extracted high-dimensional facial features. (c) We search for the query facial image to mine the top-ranking similar images and the corresponding label information for annotation.

3.1 Weakly Labeled Faces Database (WLFDB)

The first module, as shown in Figure 3.1 (a), is to collect a large-scale web facial images database which are freely available on the World Wild Web (WWW). It is a key step for search-based face annotation system. In literature, some web facial image databases are available from the previous research work, e.g., LFW [46],¹ Yahoo!News [37],² and FAN-Large [75]³. One issue with these three databases is that although the number of persons is quite large as compared to regular face databases, the number of images for each person is quite small, which makes them inappropriate for search-based face annotation tasks. For example, the LFW database has a total of 13,233 images for 5,749 people, in which each person on average has no more than 3 facial images.

3.1.1 Facial Image Database Construction

In order to evaluate the search-based face annotation scheme in even larger web facial image databases, we construct one new database: “WLFDB: Weakly Labeled Faces on the web”,

¹<http://vis-www.cs.umass.edu/lfw/>

²<http://lear.inrialpes.fr/people/guillaumin/data.php>

³<http://www.vision.ee.ethz.ch/~calvin/fan-large/>

which contains 6,025 famous western celebrity and 714,454 web facial images in total. In general, there are two main steps to build a weakly-labeled web facial image database:

- (i) **Step 1:** Constructing a name list of popular celebrities;
- (ii) **Step 2:** Querying the existing web search engines with the celebrity names, and then crawl the web facial images according to the web search results.

In the first step, we collected a name list consisting of 6,025 names downloaded from the website: **IMDb**,⁴ which covered the actors and actresses who were born between 1950 and 1990. Specifically, we collected these names with the billboard: “Most Popular People Born In yyyy”, where yyyy was the year of birth. e.g. the webpage⁵ presents all the actor and actresses who were born in 1975 in the popularity order.

In the second step, we submitted each name in the aforementioned lists as a query to search for the related web images by Google image search engine. **We did not use the face filter provided by Google image search engine.** The top 200 retrieved web images were crawled automatically. **In our framework, we did not crawl the surrounding texts of these web images, hence, the query name was directly assigned to these retrieval results as the initial weak label information.** After that we used the Viola-Jones algorithm to detect the faces and used the Deformable Lucas-Kanade (DLK) algorithm [123] to align facial images into the same well-defined alignment position. Note that **the web images in which no faces were detected were ignored/removed directly.** As a result, we collected 714,454 facial images of 6,025 persons in the “WDB” database.

In order to evaluate the annotation performance of the proposed algorithms in the following chapters, we built the evaluation (test) dataset of “WLFDB” by randomly choosing 119 names from the name list of “WLFDB”. Similar to the construction proceeding of the retrieval

⁴<http://www.imdb.com>

⁵http://www.imdb.com/search/name?birth_year=1975

database, we issued each name as a text query to Google image search engine and crawled the web images with the top 200-th to 400-th search results. Note that **we did not consider the top 200 retrieval results since they had already been used in the retrieval database.** This is helpful to examine the generalization performance of the proposed technique for the unseen facial images. We requested our staff to manually examine the retrieved facial images and remove those irrelevant facial images for each name. As a result, the evaluation dataset for “WLFDB” contained 1,600 facial images. It was a time consuming task to build a large scale test dataset, since every test image should be manually labeled. In the further work, we will continuously extend the test dataset.

In the following chapters, we also denote the “WLFDB” database by “WDB-600K”. In order to evaluate the effect of the number of persons, we build four subsets of the whole “WDB” database “WDB-600K”: “WDB-040K”, “WDB-100K”, “WDB-200K”, and “WDB-400K.” For example, WDB-040K includes only all the facial images of 400 celebrities, and there are 53,448 images in WDB-040K. To facilitate future research by other researchers, we have made our database and their related information publicly available⁶ <http://stevenhoi.org/WLF/>

3.1.2 Database Evaluation

Given the nature of web images and the limited performance of text-based image search engine, these facial images are often noisy, which do not always correspond to the right human name. For examples, figure 3.2 shows two examples in our facial image database, which are crawled with names **Tom Cruise** and **Sean Penn**. As we can see, although the images associated to **Tom Cruise** is good, the images for **Sean Penn** are much noisy. As a result, we call such kind of web facial images with noisy names as weakly labeled facial image data.

For the whole retrieval database, we analysis the number of images per person and the corresponding histogram is shown in Figure 3.3. We can observe that for most persons/classes there are about 100 facial images.

⁶.

(a)Tom Cruise (A good case that most images are correct)

(b)Sean Penn (A bad case where about 50% are wrongly labeled)

Figure 3.2: Two example sets of weakly labeled facial images

Figure 3.3: The histogram of the number of images per person in the retrieval database.

3.2 Facial Images Preprocessing

As shown in Figure 3.1, the second module is facial image preprocessing, which consists of *face detection* (FD), *face alignment* (FA), *facial feature extraction* (FE), *facial feature indexing* (FI).

3.2.1 Face Detection (FD)

For face detection and eye detection, we use the Viola-Jones object detection algorithm [97], which has been implemented in the open source computer vision library OpenCV.⁷ The Viola-Jones object detector is supposed to be the most famous and successful real-time object detector (especially for facial images) in the past decade. It depends on three main ideas : the integral image, classifier learning with AdaBoost, and the attentional cascade structure. [22, 42]

Figure 3.4: An Example of Face Detection.

In our framework, currently we only use the front-view facial detector, as a result, the profile facial images could not be detected. Figure 3.4 presents some face detection examples, including the successful case for front view faces and the failed case for profile view faces.

⁷<http://opencv.willowgarage.com/wiki/>

3.2.2 Face Alignment (FA)

For the facial alignment task, we adopt the unsupervised face alignment technique proposed in [123], which frames the face alignment task into the Deformable Lucas-Kanade (DLK) image registration approach with a non-rigid mapping. Extra threshold function and regularizer are employed to handle the outlier problem and the light variation. In other words, the face alignment task is supposed to iteratively align a single image with respect to another template image by minimizing the sum of squared differences between these two images.

Let I and T denote an input image and a template image, respectively. With the inverse compositional method, the alignment goal is to find the increment parameter $\Delta \mathbf{s}$ to iteratively update the parameter $\mathbf{s}$ by minimizing the following energy function :

$$E(\mathbf{s}) = \sum_{\mathbf{x}} (T(W(\mathbf{x}; \Delta \mathbf{s})) - I(W(\mathbf{x}; \mathbf{s})))^2 + \lambda_s \mathbf{s}^\top K \mathbf{s} \quad (3.1)$$

Inside of using a global or patch-wise warping function, the DLK algorithm directly parameterize the transformation by the mesh vertex coordinates $\mathbf{s} = [x_1, \dots, x_N, y_1, \dots, y_N]^\top$. $W(\mathbf{x}; \mathbf{s})$ is parametric warping function for the pixel coordinates $\mathbf{x}$. We assume that a point $\mathbf{x}$ lies in a triangle whose three vertices' coordinates are $(x_i, y_i), (x_j, y_j), (x_k, y_k)$, where $i, j, k \in 1, \dots, N$ is the index of each vertex in $\mathbf{s}$. The warping function is defined as following:

$$W(\mathbf{x}; \mathbf{s}) = \begin{bmatrix} x_i & x_j & x_k \\ y_i & y_j & y_k \end{bmatrix} [\epsilon_1, \epsilon_2, \epsilon_3]^\top \quad (3.2)$$

where $(\epsilon_1, \epsilon_2, \epsilon_3)$ are the barycentric coordinates for the point x .

Figure 3.5 shows the alignment proceeding for DLK algorithms. The mesh model is define as Figure 3.5 (a), which has totally 173 vertexes and 300 surfaces. The template image is illustrated in Figure 3.5 (b). The target image is presented in Figure 3.5 (c), where the initial mesh position is achieved according to the face and eyes detection result. The task of alignment is to minimize the square error between the template image in (b) and the aligned image in (d). The Figure 3.5 (e) and (f) show the final alignment result, including the mesh position and

Figure 3.5: An example for DLK face alignment

result face. Similar to current face detection algorithm, the DLK alignment algorithm also limits to the front view facial images.

3.2.3 Feature Extraction (FE)

For feature extraction, we can represent a facial image in various facial feature spaces. To evaluate the performance of different facial feature representations, we conduct one simple experiment based on the proposed WRLCC algorithm in Chapter 5. Table 3.1 shows the annotation performance of different features. All of these features are extracted from the aligned facial images by the DLK algorithm. The features of “GIST” [89], “Edge”, “Color”, and “Gabor” are generated by the toolbox in <http://goo.gl/BzPPx>. For the “LBP” feature, each aligned facial image is divided into 7×7 blocks [4], leading to 2891-dimensional features.

Table 3.1: The performance of WRLCC with different features.

	GIST	Edge	Color	Gabor	LBP ₂₈₉₁	LBP ₅₁₂
Hit Rate	0.7671	0.3724	0.4265	0.4805	0.7928	0.7584
Running Time(s)	1.205	0.681	0.457	0.658	45.830	1.239

From the results, we observe that the original LBP₂₈₉₁ feature achieves the best performance, which is consistent to the existing face recognition studies that show LBP is one of the best facial features. However, the high dimensionality of LBP feature leads to a high running time for online face annotation. For example, based on the original LBP feature, the WRLCC

algorithm takes about 46 seconds on average for each query sample. By reducing the dimensionality of the original LBP feature to the same dimensionality of GIST (512D) with principal component analysis, denoted as “LBP₅₁₂”, the new feature LBP₅₁₂ performs slightly worse than the GIST feature. Therefore, we adopted the GIST (512D) feature to represent the facial image during our experiments.

3.2.4 Feature Indexing (FI)

In order to efficiently achieve the retrieval result, we index the extracted facial features by applying an efficient high-dimensional indexing technique to facilitate the task of similar face retrieval. In our approach we adopt the Multi-probe Locality-Sensitive Hashing (MPLSH) [27], which is a popular and effective high-dimensional indexing technique for approximate nearest neighbor search. Instead of only considering the main bucket, the MPLSH algorithm examines other buckets that are “close to the main bucket to make better use of a smaller number of hash tables.

3.3 Facial Image Retrieval and Name Annotation

The first two modules must be done before a query facial image can be annotated. The next modules are related to online processes of annotating a query facial image on the fly. As shown in Figure 3.1, given a query facial image, we firstly adopt the same pre-processing step as the one for the aforementioned web facial image database, then we employ a similar face retrieval process to find a short list of the most similar faces from the indexed face databases using LSH.

After obtaining the top-ranking most similar faces and the corresponding weak label information, the last key module is to annotate the query image by mining the candidate facial images and their label information, as shown in Figure 3.1 (4). One basic solution is majority voting method (denoted as “ORI” in Chapter 4) and the soft weighted majority voting method (denoted as “SWM” in Chapter 5). To further improve the annotation performance, a serious

machine learning algorithms can be adopted. For example, we can adopt sparse representation technique to annotate the query facial image in Chapter 5 or adopt multimodel learning to explore multi feature space in a learning to rank framework (in Chapter 7). More details will be presented in the following chapters.

3.4 Real-world Demo System: FANS

Based on the search-based face annotation framework, we construct a demonstration system: “**FANS** - Face ANnotation by Searching large-scale web facial images”, which is available online at <http://msm.cais.ntu.edu.sg/fans>. To popularize the usage of our system, we also develop an Android APP in the mobile operation system. The following sections will briefly introduce the basic user interfaces of FANS, including three major parts: *query input methods*, *presentation of retrieval results*, and *user interface on mobile platforms*.

3.4.1 Query Input Methods

FANS provides three types of query input methods for users to specify a query, which is shown in Figure 3.6. Similar to most commercial web image search engines, there is an input query box at the top of the front-page of FANS. A user can easily upload a local query image from his/her computer by clicking the camera icon on the right hand side of the query box. Another input method of FANS is to directly use the web facial images without even downloading them into users’ computers. In particular, the second input method allows a user to specify a query image by simply typing the URL of any facial image on the web in the query box. Sometimes users may want to further explore the retrieval results to pick up some related image for query. Thus, as the third query input method of FANS, users can directly drag any image on the current retrieval page to the input query box. At the same, we allow the user to take pictures with their local web-cameras.

Figure 3.6: The methods of inputting query facial image, including entering the URL, uploading the local image, and directly dragging the images on the webpage.

3.4.2 Presentation of Retrieval Results

In the retrieval result presentation page of FANS, there are four major parts, including the *query image*, the *annotated name*, the *corresponding most similar facial image*, and the *list of top-K similar images*.

Figure 3.7: The representation of retrieval results in FANS system.

As shown in Figure 3.7, we present the query image at top-left corner of the result page, and highlight the face location with red rectangle. Under the query image, we show the alignment result of the detected face. If multiple faces are detected in the query image, we will mark all the facial images with color rectangular, and highlight the current query face in red color. At the top-right corner of the result web page, we present the annotated name and its corresponding image. At the bottom of the result page, we show the top 21 similar facial images and the corresponding names. To browse the details of the retrieval results, users can simply click the thumbnail image to check its original image in a new pop-out window. As a feedback to FANS, users can help to rate the annotation results by using the rating bar under the similarity value, which can collect the information for further improving the similarity estimation of FANS by adopting machine learning techniques.

With the explosion of social network and social media, more and more personalized photos are shared over the social network, e.g. Facebook, Flickr, Google+ and so on. In our FANS system, the users can login to his/her Facebook account to access his/her own album images and the profile images of his/her friends, then the users can directly drag these images as input queries to find out the most similar popular star to himself/herself or to his/her friends. Users can also easily share the retrieval results with his/her friends in the Facebook.

3.4.3 Mobile Application on Android System

In addition to website, we deploy FANS on mobile platforms, which have become an important ports to share and manage photos. We develop an App (a.k.a. “Application software”) on the Android platform, as shown in Fig. 3.8.

Figure 3.8: (a) Interface of the mobile application, (b) Annotation result using a captured photo, (c) Edit and crop image, and (d) Access the photo albums on the Facebook.

The FANS APP has three main functions, as the three buttons shown in Fig. 3.8. By clicking the first “SNAP” button, users can query any photo captured by the mobile cameras.

An annotation result is shown in Fig. 3.8 (b). The left image is the query image, while the right image is the most similar images in the retrieval database. The middle number indicates the similarity value between the query image and the most similar facial image. By clicking the Facebook button below the similar number, users can share the annotation results with their friends via social networks. FANS also allows users to rate the annotation results in FANS APP. In particular, under the query image, a bar with five stars is given for voting the annotation result. For a large size image, users can crop out the facial image region with FANS APP as the query image, as shown in Figure 3.8 (c). By clicking the second button or the third button, users can respectively access the album images in the mobile device or in their Facebook account, so as to explore facial images as queries to interact with FANS, as shown in Figure 3.8 (d).

3.5 Summary

In this chapter, we briefly introduce the basic techniques in Search-based Face Annotation (SBFA) framework. There are still lots of open challenging problems in this emerging research topic. In this thesis, we mainly focus on three key problems:

- (i) How to effectively enhance the initial weak name labels.
- (ii) How to effectively exploit the short list of candidate facial images and their weak label information for the face name annotation task.
- (iii) How to effectively boost the annotation performance of SBFA problem by adopting sophisticated machine learning techniques.

In the following chapters, a family of algorithms is proposed to tackle these problems. In particular, in Chapter 4, we propose an Unsupervised Label Refinement (ULR) algorithm

for the first challenge; in chapter 5, we propose a Weak Label Regularized Local Coordinate Coding (WLRLCC) for the second challenge; and in Chapter 6 & 7, we adopt the semi-supervised learning and multimodel learning techniques to boost the annotation performance in the Search-based Face Annotation framework, where the state-of-the-art results are achieved.

Chapter 4

Unsupervised Label Refinement

Due to the nature of web image, the first challenge of search-based face annotation system is that the labels of the web facial images are noisy. This is critical for automatic face annotation problem as the label quality of the database can affect the final annotation performance. To tackle this challenge, in this chapter, we propose a novel Unsupervised Label Refinement (ULR) scheme by exploiting machine learning techniques to enhance the labels purely from the weakly-labeled images. To improve the efficiency and scalability, we also propose a Clustering-based Approximation (CBA) algorithm.

4.1 Unsupervised Label Refinement by Learning on Weakly Labeled Data

In this section, we present the proposed “Unsupervised Label Refinement” (ULR) algorithm in detail. To clarify the formulation, Table. 4.1 lists the key notations used in this chapter.

4.1.1 Problem Formulation

We denote by $X \in \mathbb{R}^{N \times d}$ the extracted facial image features, where N and d represent the number of facial images and the number of feature dimensions, respectively. We assume there are M names in total in retrieval database. We also denote by $Y \in \{0, 1\}^{N \times M}$ the initial raw label matrix to describe the weak label information, in which the i -th row Y_{i*} represents the

Table 4.1: List of Key Notations

Notation	Description
N	# facial images in the whole retrieval database
M	# names in the whole retrieval database
d	# feature dimensions (e.g. $m = 512$ for GIST feature)
n	# similar images for the query image in the retrieval result
T	# annotated candidate names for each query image
$X \in \mathbb{R}^{N \times d}$	Facial feature matrix of the whole retrieval database
$Y \in \{0, 1\}^{N \times M}$	Initial label matrix to describe the weak label information
$Y_{i*} \in \{0, 1\}^M$	The label vector of the i -th facial image $\mathbf{x}_i \in \mathbb{R}^d$
$W = [W_{ij}] \in \mathbb{R}^{N \times N}$	The similarity matrix, W_{ij} represents the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$
$F \in \mathbb{R}_+^{N \times M}$	The refined label matrix, based on the raw label matrix Y
$Z \in \{0, 1\}^{N \times M}$	An indicator matrix, which marks the non-zero value in Y

label vector of the i -th facial image $\mathbf{x}_i \in \mathbb{R}^d$. In our application, Y is often noisy. In particular, for each weak label value Y_{ij} , $Y_{ij} \neq 0$ indicates that the i -th facial image $\mathbf{x}_i$ has the label name n_j , while $Y_{ij} = 0$ indicates that the relationship between i -th facial image $\mathbf{x}_i$ and j -th name is unknown. Note that we usually have $\|Y_{i*}\|_0 = 1$ since each facial image in our database is uniquely collected by a single query. Following the terminology of graph-based learning methodology, we build a sparse graph by computing the weight matrix $W = [W_{ij}] \in \mathbb{R}^{N \times N}$, where W_{ij} represents the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$.

The goal of the Unsupervised Label Refinement (ULR) problem is to learn a refined label matrix $F_+^* \in \mathbb{R}^{N \times M}$, which is expected to be more accurate than the initial raw label matrix Y . This is a challenging task since we have nothing else but the raw label matrix Y and the data examples X themselves. To tackle this problem, we propose a graph-based learning solution based on a key assumption of “label smoothness”, i.e., the more similar the visual contents of two facial images, the more likely they share the same labels. The label smoothness principle can be formally formulated as an optimization problem of minimizing the following loss function $E_s(F, W)$:

$$E_s(F, W) = \frac{1}{2} \sum_{i,j=1}^n W_{ij} \|F_{i*} - F_{j*}\|_F^2 = \text{tr}(F^\top L F) \quad (4.1)$$

where $\|\cdot\|_F$ denotes the Frobenius norm, W is the weight matrix of a sparse graph constructed from the whole retrieval facial images, $L = D - W$ denotes the Laplacian matrix where D is a diagonal matrix with the diagonal elements as $D_{ii} = \sum_{j=1}^n W_{ij}$, and tr denotes the trace function.

Directly optimizing the above loss function is problematic as it will yield a trivial solution. To overcome this issue, we notice that the initial raw label matrix usually, though being noisy, still contains some correct and useful label information. Thus, when we optimize to search for F , we shall avoid the solution F being deviated too much from Y . To this end, we formulate the following optimization task for the unsupervised label refinement by including a regularization term $E_p(F, Y)$ to reflect this concern:

$$F^* = \arg \min_{F \geq 0} E_s(F, W) + \alpha \cdot E_p(F, Y) \quad (4.2)$$

where α is a regularization parameter and $F \geq 0$ enforces F is nonnegative. Next we discuss how to define an appropriate function for $E_p(F, Y)$.

One possible choice of $E_p(F, Y)$ is to simply set $E_p(F, Y) = \|F - Y\|_F^2$. This is however not appropriate as Y is often very sparse, i.e., many elements of Y are zeros due to the incomplete nature of Y . Thus, the above choice is problematic since it may simply force many elements of F to zeros without considering the label smoothness. A more appropriate choice of the regularization should be applied only to those nonzero elements of Y . To this end, we propose the following choice of $E_p(F, Y)$:

$$E_p(F, Y) = \|(F - Y) \circ Z\|_F^2 \quad (4.3)$$

where Z is an indicator matrix $S = [h(Y_{ij})]$ where $h(x) = 1$ if $x > 0$ and 0 otherwise, and $\circ$ denotes the Hadamard product (i.e., the entry wise product) between two matrices.

Finally, we notice that the solution of the optimization in (3) is generally dense, which is again not desired since the true label matrix is often sparse. To take the sparsity into con-

sideration, we introduce a sparsity regularizer $E_e(F)$ by following the “exclusive lasso” technique [122]:

$$E_e(F) = \sum_{i=1}^n (\|F_{i*}\|_1)^2 \quad (4.4)$$

where we introduce an ℓ_1 norm to combine the label weights for the same person with respect to different names, and an ℓ_2 norm to combine the label weights of different persons together. Combining this regularizer and the previous formulation, we have the final formulation as follows:

$$\begin{aligned} F^* &= \arg \min_{F \geq 0} g(F) \\ g(F) &= E_s(F, W) + \alpha E_p(F, Y) + \beta E_e(F) \end{aligned} \quad (4.5)$$

where $\alpha \geq 0$ and $\beta \geq 0$ are two regularization parameters. The above formulation combines all the terms in the objective function, which we refer to it as “Soft-Regularization Formulation” or “SRF” for short.

Another way to introduce the sparsity is to formulate the optimization by including some convex sparsity constraints, which leads to the following formulation:

$$\begin{aligned} F^* &= \arg \min_{F \geq 0} E_s(F, W) + \alpha E_p(F, Y) \\ \text{s.t. } &\|F_{i*}\|_1 \leq \varepsilon, i = 1, \dots, N \end{aligned} \quad (4.6)$$

where $\alpha \geq 0$ and $\varepsilon > 1$. We refer to this formulation as “Convex-Constraint Formulation” or “CCF” for short.

It is not difficult to see that the above two formulations are convex, which thus can be solved with global optima by applying convex optimization techniques. Next, we discuss efficient algorithms to solve the above optimization tasks.

4.1.2 Algorithms

The above optimization tasks belong to convex optimization or more exactly quadratic programming (QP) problems. It seems to be possible to solve them directly by applying generic QP solvers. However, this would be computationally highly intensive since matrix F can be potentially very large, e.g. for a large 400-person database, the F is a huge matrix that consists of over 10 million variables, which is almost infeasible to be solved by any existing generic QP solver.

4.1.2.1 Algorithm for Soft-Regularization Formulation

We firstly adopt an efficient algorithm to solve the problem in Eq. 4.5, then propose a coordinate descent based approach to improve the scalability. By vectorizing matrix $F \in \mathbb{R}^{N \times M}$ into a column vector $\tilde{\mathbf{f}} = \text{vec}(F) \in \mathbb{R}^{(N \cdot M) \times 1}$, we can reformulate $g(F)$ as follows:

$$\begin{aligned} g(F) &= \text{tr}(F^\top L F) + \alpha \|(F - Y) \circ S\|_F^2 + \beta \|F \cdot \mathbf{1}\|_F^2 \\ &= \tilde{\mathbf{f}}^\top Q \tilde{\mathbf{f}} + \mathbf{c}^\top \tilde{\mathbf{f}} + \mathbf{h} \end{aligned} \quad (4.7)$$

where $\circ$ denotes the Hadamard product, $\otimes$ denotes the Kronecker product, $\tilde{\mathbf{y}} = \text{vec}(Y)$, $\tilde{\mathbf{s}} = \text{vec}(S)$, $\mathbf{1}$ is all one column vector, $U = I_m \otimes L^\top$, $V = (\mathbf{1}^\top \otimes I_n)$, $R = \text{diag}(\tilde{\mathbf{s}})$, $Q = U + \alpha R + \beta V^\top V$, $\mathbf{c} = -2\alpha R^\top \tilde{\mathbf{y}}$, $\mathbf{h} = \alpha \tilde{\mathbf{y}}^\top R \tilde{\mathbf{y}}$ and I_k is an identity matrix with dimension $k \times k$.

As shown in the vectorizing result, the optimization is clearly a QP problem. To efficiently solve this problem, we propose an accelerated multi-step gradient algorithm, which converges at $O(\frac{1}{k^2})$, k is the iteration step.

First of all, we reformulate the QP problem as follows:

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} q(\mathbf{x}|Q, \mathbf{c}) = \mathbf{x}^\top Q \mathbf{x} + \mathbf{c}^\top \mathbf{x} \quad \text{s.t. } \mathbf{x} \geq 0 \quad (4.8)$$

We then define a linear approximation function $p_t(\mathbf{x}, \mathbf{z})$ for the above function q at point $\mathbf{z}$:

$$p_t(\mathbf{x}, \mathbf{z}) = q(\mathbf{z}) + \langle \mathbf{x} - \mathbf{z}, \nabla q(\mathbf{z}) \rangle + \frac{t}{2} \|\mathbf{x} - \mathbf{z}\|_F^2 \quad (4.9)$$

where t is the Lipschitz constant of ∇q . In order to achieve the optimal solution $\mathbf{x}^*$, we will update two sequences $\{\mathbf{x}^{(k)}\}$ and $\{\mathbf{z}^{(k)}\}$, recursively. Commonly at each iteration k , the variance $\mathbf{z}^{(k)}$ is named as *search point* and used for combining the two previous approximate solutions $\mathbf{x}^{(k-1)}$ and $\mathbf{x}^{(k-2)}$. The approximation $\mathbf{x}^{(k)}$ is achieved by solving the following optimization:

$$\mathbf{x}^{(k+1)} = \arg \min_{\mathbf{x}} p_t(\mathbf{x}, \mathbf{z}^{(k)}) \quad \text{s.t. } \mathbf{x} \geq 0 \quad (4.10)$$

After ignoring terms that do not depend on $\mathbf{x}$, the former optimization problem Eq. 4.10 could be equally presented as:

$$\min_{\mathbf{x} \geq 0} \mathbf{g}^\top \mathbf{x} + \frac{t}{2} \|\mathbf{x} - \mathbf{z}^{(k)}\|^2 = t \sum_i \left[\frac{1}{2} (x_i - z_i^{(k)})^2 + \frac{g_i}{t} x_i \right] \quad (4.11)$$

where $\mathbf{g} = 2Q\mathbf{z}^{(k)} + \mathbf{c}$. The solution could be shown directly as follows:

$$x_i = \max(z_i^{(k)} - g_i/t, 0) \quad (4.12)$$

Finally, Algorithm 1 summarizes the optimization progress.

Algorithm 1: Multi-step Gradient Algorithm for ULR

Input: $Q \in \mathbb{R}^{(N \cdot M) \times (N \cdot M)}$, $\mathbf{c} \in \mathbb{R}^{N \cdot M}$, $t \in \mathbb{R}$

Output: $\mathbf{x}^*$

```

1 begin
2    $\alpha_0 = 1; k = 1; \mathbf{z}^{(0)} = \mathbf{x}^{(0)} = \mathbf{x}^{(-1)} = 0;$ 
3   repeat
4     Case SRF : Achieve  $\mathbf{x}^{(k)}$  with Eq. 4.10;
5     Case CCF : Achieve  $\mathbf{x}^{(k)}$  with Eq. 4.15;
6      $\alpha_k = \frac{1 + \sqrt{4\alpha_{k-1}^2 + 1}}{2};$ 
7      $\mathbf{z}^{(k)} = \mathbf{x}^{(k)} + \frac{\alpha_{k-1} - 1}{\alpha_k} (\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)});$ 
8      $k = k + 1;$ 
9   until CONVERGENCE;
```

To further improve the scalability, we propose a coordinate descent approach to solving the optimization iteratively. This can take advantages of the power of parallel computation when solving a very large-scale problem.

For the proposed coordinate descent approach, at each iteration, we optimize only one label vector F_{i*} by leaving the other vectors $\{F_{j*} | j \neq i\}$ intact. Specifically, at the $(t+1)$ -th iteration, we define the following optimization problem for updating $F_{i*}^{(t+1)}$ with $F^{(t)}$:

$$F_{i*}^{(t+1)} = \arg \min_{\mathbf{f}^\top} \Psi(\mathbf{f} | F^{(t)}, i) \quad \text{s.t. } \mathbf{f} \geq 0 \quad (4.13)$$

where the objective function Ψ is defined as follows:

$$\begin{aligned} \Psi(\mathbf{f} | F, i) &= L_{ii} \|\mathbf{f}\|^2 + 2\hat{L}_{i*} \hat{F} \mathbf{f} + \alpha \mathbf{z}^\top R \mathbf{z} + \beta \mathbf{f}^\top T \mathbf{f} \\ &= \mathbf{f}^\top \hat{Q} \mathbf{f} + \hat{\mathbf{c}}^\top \mathbf{f} + \hat{\mathbf{h}} \end{aligned}$$

where $\hat{L}_{i*} \in \mathbb{R}^{1 \times (N-1)}$ is the i -th row of Laplacian matrix L_{i*} by removing the i -th element L_{ii} , $\hat{F} \in \mathbb{R}^{(N-1) \times M}$ is a sub-matrix of F by removing its i -th row F_{i*} , $\mathbf{z} = \mathbf{f} - Y_{i*}^\top$, $R = \text{diag}(S_{i*})$, $T = \mathbf{1} \cdot \mathbf{1}^\top$, $\hat{Q} = L_{ii} I_M + \alpha R + \beta T$, $\hat{\mathbf{c}} = 2(\hat{L}_{i*} \hat{F} - \alpha Y_{i*} R)^\top$ and $\hat{\mathbf{h}} = \alpha Y_{i*} R Y_{i*}^\top$.

The Eq. 4.13 is also a smooth QP problem, but much smaller than the original Eq. 4.7. Similarly, it could be solved efficiently by Algorithm 1. The pseudo-code of the coordinate descent algorithm is summarized in Algorithm 2.

Algorithm 2: Coordinate Descent Algorithm for ULR

Input: $X \in \mathbb{R}^{N \times d}$, $Y \in \{0, 1\}^{N \times M}$
Output: $F^* \in \mathbb{R}^{N \times M}$

```

1 begin
2    $t = 0$  and  $F^{(t)} = Y$ ;
3   repeat
4     for  $i = 1$  to  $n$  do
5       Case SRF: Achieve  $F_{i*}^{(t+1)}$  with Eq. 4.13;
6       Case CCF: Achieve  $F_{i*}^{(t+1)}$  with Eq. 4.21;
7      $t = t + 1$ ;
8   until CONVERGENCE;
```

4.1.2.2 Algorithm for Convex-Constraint Formulation

For the convex-constraint formulation, by doing vectorization, we can reformulate Eq. 4.6 into the following:

$$\min_{\mathbf{x} \geq 0} \mathbf{x}^\top Q^\dagger \mathbf{x} + \mathbf{c}^\top \mathbf{x} \quad \text{s.t.} \quad \sum_{k=0}^{M-1} x_{k \cdot n + i} \leq \varepsilon, i = 1, \dots, n. \quad (4.14)$$

where $Q^\dagger = U + \alpha R$, $\varepsilon \geq 1$, and all the other symbols are the same as Eq. 4.7. We also apply the multi-step gradient scheme to solve Eq. 4.14, however the constraint for the sub-problem is slightly different from Eq. 4.11, which is defined:

$$\min_{\mathbf{x} \geq 0} \frac{t^\dagger}{2} \|\mathbf{x} - \mathbf{v}\|^2 \quad \text{s.t.} \quad \sum_{k=0}^{M-1} x_{k \cdot n + i} \leq \varepsilon, i = 1 \dots, N \quad (4.15)$$

where $\mathbf{v} = \mathbf{z}^{(k)} - \frac{1}{t^\dagger} \mathbf{g}^\dagger$, $\mathbf{g}^\dagger = 2Q^\dagger \mathbf{z}^{(k)} + \mathbf{c}$.

We can split $\mathbf{x}$ into a series of sub-vectors $\bar{\mathbf{x}}^i = [x_i, \dots, x_{(M-1) \cdot N + i}]^\top$, and similarly we can split vector $\mathbf{v}$. Thus, Eq. 4.15 could be reformulated as:

$$\min_{\bar{\mathbf{x}}^0, \bar{\mathbf{x}}^1, \dots, \bar{\mathbf{x}}^N} \frac{t^\dagger}{2} \sum_{i=1}^N \|\bar{\mathbf{x}}^i - \bar{\mathbf{v}}^i\|^2 \quad \text{s.t.} \quad \|\bar{\mathbf{x}}^i\|_1 \leq \varepsilon, \bar{\mathbf{x}}^i \geq 0 \quad (4.16)$$

The above optimization can be decoupled for each sub-vector $\bar{\mathbf{x}}^i$ and solved separately in linear time by following the Euclidean projection algorithm proposed in [61]. Specifically, we can obtain the optimal solution $\bar{\mathbf{x}}^{i*}$ for $\bar{\mathbf{x}}^i$ with the following problem:

$$\bar{\mathbf{x}}^{i*} = \arg \min_{\bar{\mathbf{x}}^i} \|\bar{\mathbf{x}}^i - \bar{\mathbf{v}}^i\|^2 \quad \text{s.t.} \quad \|\bar{\mathbf{x}}^i\| \leq \varepsilon; \bar{\mathbf{x}}^i \geq 0. \quad (4.17)$$

where $\bar{\mathbf{x}}^{i*}$ has a linear relationship with the optimal Lagrangian variable λ^* , which is introduced by the inequality constrain $\|\bar{\mathbf{x}}^i\| \leq \varepsilon$:

$$\bar{x}_j^{i*} = \text{sign}(\bar{v}_j^i) \times \max(|\bar{v}_j^i| - \lambda^*, 0), j = 1, 2, \dots, M. \quad (4.18)$$

where $\text{sign}(\cdot)$ is the previously defined sign function. Suppose $S = \{j | \bar{v}_j^i \geq 0\}$, the optimal λ^* could be obtained as follows:

$$\lambda^* = \begin{cases} 0 & \sum_{k \in S} |\bar{v}_k^i| \leq \varepsilon, \\ \bar{\lambda} & \sum_{k \in S} |\bar{v}_k^i| > \varepsilon. \end{cases} \quad (4.19)$$

where $\bar{\lambda}$ is the unique root of function $f(\lambda)$:

$$f(\lambda) = \sum_{k \in S} \max(|\bar{v}_k^i| - \lambda, 0) - \varepsilon. \quad (4.20)$$

$f(\lambda)$ is a continuous and monotonically decreasing function in $(-\infty, \infty)$. The root $\bar{\lambda}$ could be achieved with a bisection search in linear time. An improved searching scheme is also proposed in [61] by using the characteristic of function $f(\lambda)$, which is out of the scope of this paper.

Similar to the soft-regularization formulation, we can also adopt the coordinate descent scheme to further improve the scalability. In particular, we define a new update function $\Psi^\dagger$ similar to the aforementioned formula in Eq. 4.13:

$$F_{i*}^{(t+1)} = \arg \min_{\mathbf{f}^\top} \Psi^\dagger(\mathbf{f} \mid F^{(t)}, i) \text{ s.t. } \|\mathbf{f}\|_1 \leq \varepsilon, \mathbf{f} \geq 0 \quad (4.21)$$

where $\Psi^\dagger(\mathbf{f} \mid F, i) = \mathbf{f}^\top \hat{Q}^\dagger \mathbf{f} + \hat{\mathbf{c}}^\top \mathbf{f}$ and all symbols are the same as Eq. 4.13 except $\hat{Q}^\dagger = L_{ii}I_M + \alpha R$. Eq. 4.21 is a special case of the optimization problem in Eq. 4.14, and can be solved efficiently by the same algorithm. Finally, the pseudo codes of the algorithm for the convex-constraint formulation are similar to the previous, as shown in Algorithm 1 and Algorithm 2, respectively.

4.1.3 Clustering-based Approximation (CBA)

The number of variables in the previous problem is $n * m$, where n is the number of facial images in the retrieval database and m is the number of distinct names (classes). For a small problem, we can solve it efficiently by the proposed MGA-based algorithms (SRF-MGA or CCF-MGA). For a large problem, we can adopt the proposed CDA-based algorithms (SRF-CDA or CCF-CDA), where the number of variables in each sub-problem is n . However, when n is extremely large, the CDA-based algorithms still can be computationally intensive. One straightforward solution for acceleration is to adopt parallel computation, which can be easily exploited by the proposed SRF-CDA or CCF-CDA algorithms since each of the involved

sub-optimization tasks can be solved independently. However, the speedup of the parallel computation approach quite depends on the hardware capability. To further enhance the scalability and efficiency in algorithms, in this paper, we propose a Clustering-based Approximation (C-BA) solution to speed up the solutions for large-scale problems.

In particular, the clustering strategy could be applied in two different levels: (i) one is on “image-level”, which can be used to directly separate all the n facial images into a set of clusters, and (ii) the other is on “name-level”, which can be used to firstly separate the m names into a set of clusters, then to further split the retrieval database into different subsets according to the name-label clusters. Typically, the number of facial images n is much larger than the number of names m , which means that the clustering on “image-level” would be much more time-consuming than that on “name-level”. Thus, in our approach, we adopt the “name-level” clustering scheme for the sake of scalability and efficiency. After the clustering step, we solve the proposed ULR problem in each subset, and then merge all the learning results into the final enhanced label matrix F .

According to the name labels, we could divide all the facial images $X \in \mathbb{R}^{N \times d}$ into M classes: $X = [X_1, X_2, \dots, X_M]$. We denote by $C \in \mathbb{R}^{M \times M}$ the class similarity matrix for all the M classes (names). Consider the variety of facial images and the noisy nature of web images, traditional hierarchical clustering algorithms (such as “Single-Link”, “Complete-Link” and “Average-Link”) are not suitable to our problem. In our framework, following the terminology of shared nearest neighbors, we proposed a *co-occurrence likelihood* in Eq. 4.22 to compute the similarity value C_{ij} , which measures the likelihood that instances from the two classes X_i, X_j are co-occurred together in the retrieval results by some particular web search engine (e.g., Google):

$$C_{ij} = \sum_{\forall \mathbf{x}_i \in X_i} \sum_{x_p \in \mathcal{N}_n(\mathbf{x}_i)} \mathbb{I}_{(\mathbf{x}_p \in X_j)} \quad (4.22)$$

Figure 4.1: Illustration of computing class similarity matrix between three classes X_1 , X_2 and X_3 . The symbol $\otimes$ denotes the class center. C_{1*} , C_{2*} and C_{3*} are the similarity vectors of the three classes, which are computed according to Eq. 4.22 with $n = 1$. For example, the second value of vector C_{1*} , i.e., C_{12} , refers to the total number of examples in class X_2 belonging to the top $n = 1$ nearest neighbors of examples from class X_1 .

where $\mathcal{N}_n(\mathbf{x}_i)$ is the set of top- n nearest facial images w.r.t. $\mathbf{x}_i$ in the whole retrieval database (we use the nearest facial set $\mathcal{N}_n(\mathbf{x}_j)$ with $n = 50$ in our experiments), $\mathbb{I}_{(\mathbf{x}_p \in X_j)}$ is an indicator function which outputs 1 if $x_p \in X_j$ and 0 otherwise. According to this definition, a large value of C_{ij} means that the instances in class X_i are more likely to be similar to the instances in class X_j . In other words, the instances in X_i and X_j should be put together for joint class label refinement in our proposed label enhancement step. In order to normalize the elements in the matrix C , we divide each column C_{*j} by its maximum value except C_{jj} :

$$C_{ij} = \begin{cases} \frac{C_{ij}}{\max_{k \neq i} C_{kj}} & \text{if } j \neq i, \\ v_{\max} & \text{if } j = i. \end{cases} \quad (4.23)$$

where $v_{\max}$ is a constant value and set as $v_{\max} = 1$ in our experiment. Fig. 4.1 shows an example to demonstrate the calculation of matrix C among three classes X_1 , X_2 and X_3 with $n = 1$. After the normalization, the co-occurrence likelihood vectors for the three classes are $[110]$, $[110]$ and $[001]$, which are consistent to our observation that instances from class C_1 and C_2 are more likely to be mixed together.

For the proposed solution, there is an important practical assumption for the clustering step, i.e., the sizes of different clusters should be similar, which aims to avoid the undesired

case where one cluster significantly dominates the others. In our CBA framework, we propose two kinds of solutions: one is the Bisecting K-means clustering based algorithm referred to as “BCBA” for short, and the other is the Divisive Clustering based algorithm referred to as “DCBA” for short.

Algorithm 3: Bisecting K-means Clustering-based Approximation (BCBA)

Input: $C \in \mathbb{R}^{M \times M}$, $q_c \in \mathbb{N}$, $I_{\text{loop}} \in \mathbb{N}$.
Output: Clustering result list L_{list}

- 1 Add $\mathcal{M}_0$ to L_{list} ; /* $\mathcal{M}_0$ contains all the points*/
- 2 **repeat**
- 3 Remove the largest cluster $\mathcal{M}_l$ from L_{list} ;
- 4 **for** $i = 1$ **to** t **do**
- 5 Bisect $\mathcal{M}_l$ to $\mathcal{M}_1^{(i)}$ and $\mathcal{M}_2^{(i)}$;
- 6 Compute Sum of Squared Error(SSE $_i$);
- 7 Select the result with the lowest SSE $_i$ value;
- 8 Add $\mathcal{M}_1^{(i)}$, $\mathcal{M}_2^{(i)}$ to L_{list} ;
- 9 **until** $|L_{\text{list}}| = q_c$;

In the BCBA scheme, i -th row C_{i*} is used as the feature vector for class X_i . In each step, the largest cluster is bisected for I_{loop} times and the clustering result with the lowest sum-of-square-error (SSE) value is used to update the clustering lists. In our framework, we set I_{loop} to 10. The details of the BCBA scheme are illustrated in Algorithm 3, where q_c is the cluster number. In the DCBA scheme, the symmetrical matrix $\hat{C} = \frac{C+C'}{2}$ is used for building a minimum spanning tree (MST). Instead of performing the complete hierarchical clustering, in our framework, we directly separate the classes into the q_c clusters. In order to balance the cluster sizes, the bisection scheme is also employed. Specifically, in each iteration step, we partition the largest cluster into two parts by cutting its largest MST edge to ensure the size of the smaller cluster in the cutting result is larger than a predefined threshold value $T_{\text{threshold}}$. We set $T_{\text{threshold}} = \frac{M}{2*q_c}$ in our framework. The details of the DCBA scheme are shown in Algorithm 4.

Algorithm 4: Divisive Clustering-based Approximation (DCBA)

Input: $\hat{C} \in \mathbb{R}^{M \times M}$, $q_c \in \mathbb{N}$, $T_{\text{threshold}} \in \mathbb{N}$.
Output: Clustering result list L_{list}

- 1 Build MST T_{tree} with $\hat{C}$ and add all edges to E_{edge} ;
- 2 Add $\mathcal{M}_0$ to L_{list} ; /* $\mathcal{M}_0$ contains all the points*/
- 3 **while** $|L_{\text{list}}| < q_c$ **do**
- 4 Remove the largest cluster $\mathcal{M}_l$ from L_{list} ;
- 5 Extract sub-MST $T_{\text{tree}}^\dagger$ and edge set $E_{\text{edge}}^\dagger$ of $\mathcal{M}_l$;
- 6 **repeat**
- 7 **if** $|E_{\text{edge}}^\dagger| == 0$ **then**
- 8 Rebuild edge list $E_{\text{edge}}^\dagger$ with $T_{\text{tree}}^\dagger$;
- 9 $T_{\text{threshold}} = T_{\text{threshold}} - 1$;
- 10 Remove the largest edge e^s from $E_{\text{edge}}^\dagger$;
- 11 Cutting e^s in $T_{\text{tree}}^\dagger$ to split $\mathcal{M}_l$ into $\mathcal{M}_1, \mathcal{M}_2$
- 12 **until** $\min(|\mathcal{M}_1|, |\mathcal{M}_2|) \geq T_{\text{threshold}}$;
- 13 Add $\mathcal{M}_1, \mathcal{M}_2$ to L_{list} ;

We denote by $L_{\text{list}} = \{\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_{q_c}\}$ the clustering result. Using the clustering result, we first split the whole retrieval database X into q_c subsets $\{\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_{q_c}\}$, where $\mathcal{S}_i = \bigcup_{j \in \mathcal{M}_i} X_j$. Then the proposed ULR problem is conquered on each subset individually. For each subset $\mathcal{S}_i$, the number of classes is around $\frac{M}{q_c}$ on average and the number of facial images is around $\frac{N}{q_c}$ on average, which means that the number of variables to be optimized by ULR on each subset $\mathcal{S}_i$ has been reduced to $\frac{N \times M}{q_c^2}$, which is much smaller than the original number of variables on the entire database. As a result, each small sub-problem could be solved efficiently. Besides, as the sub-problems on different subsets are independent, a parallel computation framework could also be adopted for further acceleration.

4.2 Experiments

We implement all the algorithms described previously for solving the proposed ULR task. We finally adopt the soft-regularization formulation of the proposed ULR technique in our evaluation since it is empirically faster than the convex-constraint formulation according to our

implementations. The proposed ULR algorithm aims to refine the weak label information over the whole retrieval database. In our experiments, to evaluate the label refinement performance, we adopt the basic *majority voting* annotation scheme. In particular, given a novel facial image for annotation, we retrieve its top- n most similar facial images from a weakly labeled facial image database, and then annotate the facial image by performing majority voting on the labels associated with the top- n similar facial images.

4.2.1 Experiment Testbed

The baseline result is directly achieved based on the original label matrix Y , which is denoted as “ORI” in our experiments. To better examine the efficacy of our technique, we also implement other existing label refinement algorithms for comparisons. Specifically, the compared methods include the following:

- “ORI”: a baseline method that simply adopts the original label information for the search-based annotation scheme, denoted as “ORI” for short.
- “CL”: a consistency learning algorithm [120] proposed to enhance the weakly labeled facial image database, denoted as “CL” for short.
- “MKM”: a modified K-means clustering algorithm [11] proposed to cluster web facial images associated with the extracted names from the surrounding captions, denoted as “MKM” for short. We note that the original MKM algorithm was proposed to address a similar noisy label enhancement problem, but slightly different from our setting in that the number of raw noisy labels of each facial image in their problem setting can be more than 1, which is however exactly equal to 1 in our problem setting.
- “LPSN”: a label propagation through sparse neighborhood algorithm [118] proposed to propagate label information among the neighborhoods achieved by sparse coding, denoted as “LPSN” for short.

- “ULR _{$\beta=0$} ”: the proposed ULR algorithm (Soft-Regularization Formulation in Eq. 4.5) without the sparsity regularizer $E_e(F)$.
- “ULR”: the proposed unsupervised label refinement method, denoted as “ULR” for short.

To evaluate the annotation performance, we adopt the *hit rate* at top- T annotated results as the performance metric, which measures the likelihood of having the true label among the top- T annotated names. Specifically, for $T = 1$, the *hit rate* is the same with the *accuracy*. This annotation metric is also used in the following chapters. In our experiments, we adopt the “WDB”-series database, which is introduced in Chapter 3. It consists of four retrieval databases in different sizes and one query database with about 1,600 images. In particular, we adopt the “WDB-040K” and “WDB-600K” databases in the experiments of this chapter. In the following parts, we also denote these two databases by “WDB-040K” and “WDB-600K” for short.

Further, we discuss parameter settings. For the ULR implementation, we construct the sparse graph W by setting the number of nearest neighbors to 5 for all cases. In addition, for the two key regularization parameters α and β in the proposed ULR algorithm, we set their values via cross validation. In particular, we randomly divide the test dataset into two equally-sized parts, in which one part is used as validation to find the optimal parameters by grid search, and the other part is used for testing the performance. This procedure is repeated 10 times, and their average performances are reported in our experiments.

4.2.2 Evaluation of Automatic Face Annotation

In this experiment, we aim to evaluate the face annotation performance based on the search-based face annotation scheme. We evaluate the proposed ULR algorithm from different aspects on database “WDB-040K” with top 100 retrieval facial images per person, and then verify its

performance on the large-scale database “WDB-600K”. Table 4.2 shows the average annotation performance (hit rates), in which both mean and standard deviation are reported. Several observations can be drawn from the results.

First of all, it is clear that ULR which employs unsupervised learning to refine labels consistently performs better than the ORI baseline using the original weak label, the existing CL algorithm, MKM algorithm, and the LPSN algorithm. The promising result validates that the proposed ULR algorithm can effectively exploit the underlying data distribution of all data examples to refine the label matrix and improve the performance of the search-based face annotation approach. Second, we note that the ULR algorithm outperforms its special case “ $URL_{\beta=0}$ ” without the sparsity regularizer in the SRF formulation, which validates the importance of the sparsity regularizer. Finally, when T is small, the hit rate gap, i.e., the hit rate difference between ORI and ULR is more significant, and the annotation performance increases slowly when T is large. In practice, we usually focus on the small T value since users typically would not be interested in a long list of annotated names.

Table 4.2: Evaluation of auto face annotation performance in terms of *hit rates* at top- T annotated names.

	$T = 1$	$T = 2$	$T = 3$	$T = 4$	$T = 5$
ORI	0.548 ± 0.013	0.661 ± 0.011	0.766 ± 0.009	0.837 ± 0.010	0.872 ± 0.010
CL	0.634 ± 0.012	0.752 ± 0.010	0.829 ± 0.010	0.858 ± 0.010	0.883 ± 0.009
MKM	0.578 ± 0.011	0.717 ± 0.012	0.797 ± 0.012	0.856 ± 0.008	0.882 ± 0.010
LPSN	0.609 ± 0.015	0.758 ± 0.016	0.805 ± 0.015	0.845 ± 0.016	0.879 ± 0.014
$URL_{\beta=0}$	0.675 ± 0.015	0.773 ± 0.017	0.842 ± 0.016	0.875 ± 0.015	0.907 ± 0.015
ULR	0.715 ± 0.008	0.809 ± 0.005	0.859 ± 0.009	0.892 ± 0.007	0.916 ± 0.010

4.2.3 Evaluation on Varied Top- n and Top- T Settings

This experiment aims to examine the relationship between the annotation performance of varied values of n and T respectively for top- n retrieved images and top- T annotated names. To ease our discussion, we only show the results of the ULR algorithm. The face annotation performance of varied n and T values are illustrated in Fig. 4.2.

Some observations can be drawn from the experimental results. First of all, when fixing n , we find that increasing T value generally leads to better hit rate results. This is not surprising since generating more annotation results certainly gets a better chance to hit the relevant name. Second, when fixing T , we find that the impact of the n value to the annotation performance fairly depends on the specific T value. In particular, when T is small (e.g. $T = 1$), increasing the n value leads to the decline of the annotation performance; but when T is large (e.g. $T > 5$), increasing the n value often boosts the performance of top- T annotation results. Such results can be explained as follows. When T is very small, e.g. $T = 1$, we prefer a small n value such that only the most relevant images will be retrieved, which thus could lead to more precise results at top-1 annotated results. However, when T is very large, we prefer a relatively large n value since it can potentially retrieve more relevant images and thus can improve the hit rate at top- T annotated results.

4.2.4 Evaluation on Varied Numbers of Images per Person in Database

This experiment aims to further examine the relationship between the annotation performance and the number of facial images per person in building the facial image database. Unlike the previous experiment with top 100 retrieval facial images per person in the database, we created three variables of varied-size databases, which consist of top 50, 75, and 100 retrieval facial images per person, respectively. We denote these three databases as P050, P075, and P100, respectively.

Figure 4.2: Annotation performance w.r.t. varied n and T values.

Figure 4.3: The annotation performance on three different databases, which have different numbers of images per person. Specifically, P050, P075, and P100 denote the databases have the top 50, 75, and 100 retrieval images per person, respectively.

Fig. 4.3 shows the experiment results of average annotation performance. It is clear that the larger the number of facial images per person collected in our database, the better the average annotation performance can be achieved. This observation is trivial since more potential images are included into the retrieval database, which is beneficial to the annotation task. We also noticed that enlarging the number of facial images per person in general leads to the increases of computational costs, including time and space costs for indexing and retrieval as well as the ULR learning costs.

4.2.5 Evaluation on a Larger Database: WDB-600K

This experiment aims to verify the annotation performance of the proposed SBFA framework over a larger retrieval database: “WDB-600K”. As the test database is unchanged, the extra facial images in the retrieval database are definitely harmful to the nearest facial retrieval result for each query image. A similar result can also be observed in [108], where the mean average precision become smaller for a larger retrieval database. As a result, the final annotation performance of WDB-600K will be worse than the one over WDB-040K. More details of the experiment are presented in Fig. 4.4.

Some observations can be drawn from the experimental results. First of all, the proposed ULR algorithm can efficiently enhance the initial noisy label and achieve the best performance over the other algorithms. Secondly, all the algorithms perform slightly worse on the larger retrieval database. In detail, the ULR annotation performance on WDB-600K is about 0.83% of the one on WDB-040K.

To further improve the system performance, we adopt a weighted majority voting scheme for face annotation. Specifically, we assign a weighting value to each facial image in the short list of similar faces according to its ranking position : $w(n; q, \sigma) = e^{-\frac{(n-1)^q}{\sigma}}$, where n is the ranking position and $q > 0$, $\sigma > 0$ are two positive parameters. Obviously the larger n is, the smaller the weighting $w(n; q, \sigma)$ is, which means less contribution is introduced for

Figure 4.4: Face annotation performance on Database: WDB-600K. The algorithms that end up with “-wm” denote the improved performances achieved by weighted majority voting method in the name annotation step.

the label annotation. In our experiment, we set $q = 0.5$ and $\sigma = 9.0096$. The improved performance is also presented in Fig. 4.4. The main observation is the annotation performance can be significantly improved, for example, the performance of ULR is boosted to 65.2% from 59.3%. This experiment also illustrates that the performance of our current SBFA system can be further improved by adopting other more sophisticated techniques in different stages of the proposed solution, which is out of the scope of our focus in this paper.

4.2.6 Evaluation of Parameter Sensitivity

There are two key parameters α and β in the ULR algorithm. In the previous experiments, we find the best values by cross validation. In this experiment, we further examine their sensitivity to the annotation performance. Figure 4.5 shows an evaluation of the annotation performance by a grid search on varied values of parameter $\alpha \in [0, 0.6]$ and $\beta \in [0, 0.7]$ in one of cross validation experiments. The value of each vertex on the color mesh represents the hit rate gap between ULR and ORI.

We can draw some observations from the results. First of all, we find that for all α and β values, the proposed ULR scheme always has a positive improvement over ORI with the

Figure 4.5: Evaluation of parameter sensitivity with respect to α and β for $T = 1$. The values of the vertex on the color mesh illustrate the hit rate gap between ORI and ULR.

original labels. Second, the best α and β values are about 0.3 and 0.1, respectively. Further, we notice that it is not difficult to find such good α and β values to obtain comparable results. In general, ULR can attain fairly good results when choosing $\alpha \in [0.2, 0.4]$ and $\beta \in [0.01, 0.4]$. All these results show that ULR is quite robust to the parameter settings and is always effective to enhance the annotation performance.

4.2.7 Evaluation of Optimization Efficiency

This section aims to conduct extensive evaluations on the running time cost by the four different algorithms. We refer the four algorithms with the following abbreviations:

- SRF-MGA: Soft-Regularization Formulation solved by the Multi-step Gradient.
- SRF-CDA: Soft-Regularization Formulation solved by the Coordinate Decent.
- CCF-MGA: Convex-Constraint Formulation solved by the Multi-step Gradient.
- CCF-CDA: Convex-Constraint Formulation solved by the Coordinate Decent.

We first compare two algorithms: SRF-MGA and CCF-MGA, which adopt the same gradient-based optimization scheme for two different formulations, as shown in Algorithm 1. We use

an artificial dataset with varied numbers of classes $m = 20, 40, 60, 80, 100$ where each class corresponds to a unique Gaussian distribution. We set the number of examples generated from each class as $P = 100$, and the total number of examples $n = 2K, 4K, 6K, 8K, 10K$. The goal of our ULR optimization task is to optimize the refined label matrix $F \in \mathbb{R}^{N \times M}$, which has the total number of unknown variables V would be 40K, 160K, 360K, 640K, 1M, respectively for each of the above cases. For the iteration number, we set it to 50 for both algorithms.

We randomly generate the artificial dataset and run the algorithms over the dataset. This procedure was repeated five times. The first two columns of Table 4.3 show the average running time cost obtained by both SRF-MGA and CCF-MGA algorithms, respectively. We observe that the time cost growth rate of SRF-MGA is always slower than that of CCF-MGA, which indicates that SRF-MGA runs always more efficiently than CCF-MGA. To further compare the difference of their growth rates, we try to fit the running time costs T with respect to the number of variables V by a function $T = a \times V^b$, where $a, b \in \mathbb{R}$ are two parameters. By fitting the functions, we obtain $a = 9.04E - 7$ and $b = 1.45$ for SRF-MGA, and $a = 3.70E - 8$ and $b = 1.74$ for CCF-MGA.

Table 4.3: The average running time (seconds) of the four proposed algorithms.

V	SRF-MGA	CCF-MGA	SRF-CDA	CCF-CDA
40000	4.80	6.76	76.63	136.50
160000	29.04	49.05	197.92	330.30
360000	95.43	178.45	399.11	607.20
640000	228.44	494.29	670.23	950.40
1000000	428.46	1076.04	1022.70	1457.40

Next, we compare running time cost of RF-CDA and CCF-CDA by adopting the similar settings as the previous experiment. For the iteration number, we set the outer-loop iteration number for CDA to 30 and fix the inner iteration number with respect to their subproblems to 30. The average running time cost is illustrated in the last two columns of Table 4.3.

First, the SRF-based algorithm SRF-CDA spend less time cost than the CCF-based algorithm CCF-CDA. Second, the running time cost grows almost linearly with respect to the number of variables for both CDA based algorithms. More specifically, by fitting the time cost function $T = a \times V^b$ with respect to the number of variables V , we have $a = 3.93E - 3$ and $b = 0.90$ for SRF-CDA, and $a = 1.50E - 2$ and $b = 0.83$ for CCF-CDA, which shows that the time cost growth rates of both algorithms are empirically sub linear. This encouraging result indicates that both CDA based algorithms are efficient and scalable for large-scale dataset.

Table 4.4: Evaluation of running time used by Clustering-based Approximation (CBA)

	ULR $q_c = 01$	Build C	$q_c = 02$		$q_c = 04$		$q_c = 08$		$q_c = 16$	
			BCBA	DCBA	BCBA	DCBA	BCBA	DCBA	BCBA	DCBA
T_C	—	60.70 ± 0.70	1.82 ± 0.17	0.85 ± 0.01	3.80 ± 0.18	0.96 ± 0.01	5.43 ± 0.21	1.19 ± 0.04	7.06 ± 0.21	1.36 ± 0.01
T_U	26628.94 ± 768.06	—	7068.59 ± 259.19		3357.14 ± 62.25		2933.77 ± 75.41		2654.69 ± 67.44	
T_T	26628.94	—	7131.12	7130.14	3421.64	3418.81	2999.90	2995.67	2722.46	2716.76
%	—	—	26.78%	26.78%	12.85%	12.84%	11.27%	11.25%	10.22%	10.20%

4.2.8 Evaluation of Clustering-based Approximation

In this experiment, we aim to evaluate the acceleration performance of the two proposed Clustering-based Approximation (CBA) schemes (BCBA and DCBA) on the large database WDB-600K. A good approximation is expected to achieve a high reduction in running time with a small loss in annotation performance. Thus, this experiment evaluates both running time and annotation performance. The running time of CBA scheme mainly consists of three parts: (i) the time of constructing the similarity matrix C ; (ii) the time of clustering; (iii) the total time of running ULR algorithm in each subset. The running time costs of different clustering algorithms with different cluster numbers ($q_c = 02, 04, 08, 16$) are illustrated in Table 4.4. As a comparison, the running time of directly adopting ULR algorithm on the whole retrieval database is also presented in the second column of Table 4.4, denoted as “URL ($q_c = 01$)”. Some observations can be drawn from these results.

First of all, the proposed CBA scheme can significantly decrease the running time for the label refinement task. For example, for BCBA and DCBA schemes with $q_c = 02$, the total running time can reduce from about 26,629 seconds to 7,131(27%) seconds and 7,130(27%) seconds respectively. Secondly, increasing the value of cluster number q_c generally leads to less running time, however, the reduction becomes marginal where q_c is larger than some threshold(e.g. $q_c = 08$). Thirdly, the running time of the division clustering algorithm is a bit smaller than the one of bisecting K-mean algorithm. The reasons leading to this phenomenon are twofold: one is there is no need for multi loops in each bisection step of DCBA, another is the similarity matrix $\hat{C}$ is directly used for MST building without extra computation.

For the annotation performance, the weighted majority annotation result of the two CBA schemes (BCBA and DCBA) with different cluster number q_c are presented in Table 4.5. Two observations can be drawn from the results.

Table 4.5: Evaluation of annotation performance of Clustering-based Approximation (CBA)

top- T	ORI	ULR $q_c = 01$	$q_c = 02$		$q_c = 04$		$q_c = 08$		$q_c = 16$	
			BCBA	DCBA	BCBA	DCBA	BCBA	DCBA	BCBA	DCBA
$T = 1$	0.574 ± 0.017	0.652 ± 0.019	0.650 ± 0.017	0.644 ± 0.015	0.641 ± 0.016	0.630 ± 0.015	0.638 ± 0.018	0.627 ± 0.015	0.631 ± 0.016	0.624 ± 0.018
$T = 2$	0.692 ± 0.015	0.761 ± 0.014	0.758 ± 0.013	0.763 ± 0.015	0.758 ± 0.014	0.765 ± 0.014	0.763 ± 0.014	0.759 ± 0.015	0.767 ± 0.014	0.753 ± 0.013
$T = 3$	0.778 ± 0.014	0.820 ± 0.013	0.825 ± 0.013	0.820 ± 0.013	0.828 ± 0.013	0.817 ± 0.012	0.823 ± 0.013	0.819 ± 0.012	0.823 ± 0.011	0.810 ± 0.014
$T = 4$	0.813 ± 0.013	0.862 ± 0.013	0.857 ± 0.011	0.855 ± 0.010	0.866 ± 0.011	0.855 ± 0.010	0.855 ± 0.010	0.859 ± 0.011	0.860 ± 0.009	0.852 ± 0.010
$T = 5$	0.844 ± 0.012	0.874 ± 0.013	0.873 ± 0.010	0.869 ± 0.009	0.877 ± 0.010	0.874 ± 0.010	0.868 ± 0.010	0.873 ± 0.010	0.876 ± 0.008	0.872 ± 0.010

Firstly, although the approximation algorithms (BCBA, DCBA) slightly degrade the final annotation performance, their performances are still much better than the other compared algorithms for small T value. Considering the reduction in running time, the proposed clustering-based approximation scheme is a good approximation for the ULR algorithm, which could significantly improve the scalability of search-based face annotation framework. Secondly, the performance difference between BCBA and DCBA are statistically marginal, but the average performance of BCBA is a bit better than DCBA.

4.2.9 Label Refinement on Artificial Dataset

In this experiment, we aim to evaluate the label refinement performance of different algorithms. We build an artificial dataset that consists of 9 classes (persons) in 2-dimensional space with 20 samples for each class. To introduce noise into the label matrix, we randomly mislabel half of the whole dataset. All the data points are illustrated in Fig. 4.6(a), and the original noisy label matrix is shown as the leftmost one in Fig. 4.6(b). Given the dataset and the noisy label matrix, we compute the enhanced label matrixes using the four algorithms used in the previous experiments, as shown in Fig. 4.6(b).

Several observations can be drawn from the above results: first, the MKL and CL algorithms work well for the classes with less noise (e.g. Person 1 and Person 9), but they fail for the classes where more samples are mislabeled and widely distributed (e.g. Person 4 and Person 5). Second, by adopting the graph information, both LPSN and ULR can handle all the classes better. Obviously, by finding the maximum value in each label vector, we can recover the ideal label matrix from the refined label matrix F_{ULR} . Third, for the proposed ULR algorithm, we also consider the distortion with the original label matrix ($E_p(F, Y)$ in Eq.4.5) and the sparsity of each label vector ($E_e(F)$ in Eq.4.5). As a result, ULR can achieve more stable and sparse refined label matrix that is more suitable for our face annotation problem.

4.3 Summary

In this chapter, we focus on tackling the first critical challenge and propose an Unsupervised Label Refinement (ULR) algorithm. To reduce the computing cost and improve the scalability, we propose a Clustering-based Approximation (CBA) solution, which successfully accelerates the optimization task without introducing much performance degradation. From an extensive set of experiments, we find that the proposed technique achieved promising results under a variety of settings. In the next chapter, we will tackle the second challenge by adopting sparse representation techniques based on the top-ranking similar facial images.

Figure 4.6: The label refinement experiment over an artificial dataset. (a) The demo dataset with 9 classes (persons), half of them are mislabeled. (b) The original noisy label matrix and the refined ones achieved by various algorithms. The distances of the refined label matrix to the ideal label matrix (Y_{true}) are shown at the bottom of each figure.

Chapter 5

Weak Label Regularized Local Coordinate Coding

The second challenge in the SBFA framework is how to effectively exploit the short list of candidate facial images and their weak label information. To address this critical problem, we propose a novel Weak Label Regularized Local Coordinate Coding (WLRCC) algorithm in a unified learning scheme, which exploits the local coordinate coding principle for learning more discriminative features and makes use of the graph-based regularization to enhance the weak labels simultaneously. In particular, given a query facial image for annotation, we firstly search for a short list of top- n most similar facial images from a weakly labeled web facial image database. Next, we apply the proposed WLRCC algorithm to obtain a more discriminative local coordinate coding representation and an enhanced label matrix as well. Finally, we apply an effective sparse reconstruction scheme to produce final facial name annotation. In the following sections, I will present the proposed WLRCC in detail.

5.1 Weak Label Regularized Local Coordinate Coding

In this section, we present the proposed “Weak Label Regularized Local Coordinate Coding” (WLRCC) algorithm in detail. To clarify the formulation, Table. 5.1 lists the key notations used in this chapter.

Table 5.1: List of Key Notations

Notation	Description
$\mathbf{x}_q \in \mathbb{R}^d$	The query facial image feature vector (the input of the SBFA framework)
$\mathbf{y}_q \in \{0, 1\}^m$	The unknown name vector of $\mathbf{x}_q$ (the output of the SBFA framework)
d	# feature dimensions (e.g. $m = 512$ for GIST feature)
n	# similar images for the query image $\mathbf{x}_q$ in the retrieval result
m	# names among all the similar facial images
$\mathbf{x}_i \in \mathbb{R}^d$	The i -th similar image of the query image $\mathbf{x}_q$, $i = 1, 2, \dots, n$
$\mathbf{y}_i \in \{0, 1\}^m$	The corresponding name vector of the i -th similar image $\mathbf{x}_i$
$X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$	The feature matrix of the top- n retrieval results
$\tilde{Y} \in \{0, 1\}^{n \times m}$	The initial name information matrix, where $\tilde{Y}_{i*} = \mathbf{y}_i$
$B = [X, I] \in \mathbb{R}^{d \times (n+d)}$	The sparse coding dictionary constructed based on the retrieval results
$Z \in \{0, 1\}^{n \times m}$	The indicator matrix, which marks the non-zero value in $\tilde{Y}$

5.1.1 Preliminaries

Consider a query facial image $\mathbf{x}_q \in \mathbb{R}^d$ in a d -dimensional feature space, which is associated with an unknown class denoted by a class label vector $\mathbf{y}_q$. Assume the n retrieval results of the query image $\mathbf{x}_q$ are $\{(\mathbf{x}_i, \mathbf{y}_i) | i = 1, 2, \dots, n\}$, where $\mathbf{y}_i \in \{0, 1\}^m$ is the name label vector of its corresponding facial image $\mathbf{x}_i$ and $\|\mathbf{y}_i\|_0 = 1$, and m is the total number of classes (names) among all the similar facial images. Let $X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$ denote the feature matrix of the retrieval results. We represent the initial name information with a class-membership-indicator matrix $\tilde{Y} \in \mathbb{R}^{n \times m}$, where $\tilde{Y}_{i*} = \mathbf{y}_i$, the i -th row of matrix, denotes the class label vector for $\mathbf{x}_i$.

The proposed WLRCC algorithm aims to learn a more discriminative representation for the candidate similar facial images and enhances the initial weak label information in a unified framework. We will present the formulation of coding learning, label enhancement, and WLRCC, respectively in the following sections.

Figure 5.1: Illustration of the proposed WRLCC process. In this example, we show the top $n = 8$ retrieved images with $m = 4$ candidate names for query image $\mathbf{x}_q$: (a) gives the input data; (b) illustrates the two key steps of the WRLCC process; (c) shows the output coding result and final annotation scheme.

5.1.2 Problem Formulation

5.1.2.1 Sparse Features via Local Coordinate Coding

Sparse representation has been successfully applied in many applications. Recently, it has been proved that it is important to exploit the locality information for the linear embedding of high dimensional data on the manifold in Local Coordinate Coding (LCC) [114]. In contrast to traditional sparse coding, the intuition of LCC is to assign a larger coefficient to the dictionary item that is closer to the encoded item. In details, the goal of LCC is to develop a new representation of feature vectors in which each feature vector is described as a linear combination of several nearby items among the basis vectors. The sparse representation of each vector is adapted, named as Local Coordinate Coding.

In our framework, we adopted the LCC algorithm for the coding step of WLRLLCC. Specifically, we reconstruct the locality coding $\mathbf{s}_i$ of the i -th facial image $\mathbf{x}_i$ in the retrieval results with the dictionary $B = [X, I] \in \mathbb{R}^{d \times (n+d)}$ as follows:

$$\min_{\hat{\mathbf{s}}_i} \frac{1}{2} \|\mathbf{x}_i - B\hat{\mathbf{s}}_i\|^2 + \lambda \sum_{k=1}^{n+d} |\hat{s}_{ik}| \|B_{*k} - \mathbf{x}_i\|^2 \text{ s.t. } \hat{s}_{ii} = 0; \quad (5.1)$$

where $\mathbf{s}_i$ is a sub-vector of $\hat{\mathbf{s}}_i$ with its first n elements: $\hat{\mathbf{s}}_i = [\mathbf{s}_i, \xi_i]$, ξ_i is related to the noise information which is useless in our framework, I is the identity matrix, and B_{*k} is the k -th column of dictionary B . A similar leave-one-out representation model is adopted in sparse subspace clustering [30]. For simplicity, we also denote the coding problem for $\mathbf{x}_i$ by $e(\hat{\mathbf{s}}_i; \mathbf{x}_i)$.

In order to make sure the neighbor samples of each $\mathbf{x}_i$ are included in the retrieval results X , we apply query expansion for the query image $\mathbf{x}_q$ by including the top n' nearest samples of each query result into the final retrieval database. In our experiment we simply fix n' to 3 for all cases. One difference between our coding algorithm and the original LCC algorithm is that we directly construct the dictionary B instead of learning it by optimization, which is proposed in [105] and achieves excellent experiment results on the face recognition task. This method is also well known as “*cross-and-bouquet*” model for dense error correction [106].

According to our formulation, the j -th element of $\mathbf{s}_i$ tries to measure the similarity between the facial images $\mathbf{x}_i$ and $\mathbf{x}_j$, so a negative value is contrary to this intuitive notion and doesn't make sense in our scenario. As a result, we introduce an extra non-negative constraint to the previous formulation following Non-Negative Sparse Coding (NNSC) [45], in which all the elements in $\hat{\mathbf{s}}_i$ are forced to be non-negative: $\hat{s}_{ij} \geq 0, j = 1, 2, \dots, n$.

Finally, we can give the formulation for all the n facial images in the retrieval results:

$$E_1(\hat{S}; X) = \sum_{i=1}^n e(\hat{\mathbf{s}}_i; \mathbf{x}_i). \quad (5.2)$$

where $\hat{S} \in \mathbb{R}^{(n+d) \times n} = [S; \Xi]$, $S \in \mathbb{R}^{n \times n}$ is the non-negative local coordinate coding of X , and $\Xi \in \mathbb{R}^{d \times n}$ is the noise matrix.

5.1.2.2 Weak Label Enhancement

The previous formulation shows that the j -th local coefficient s_{ij} of facial image $\mathbf{x}_i$ essentially encodes the locality information between $\mathbf{x}_i$ and $\mathbf{x}_j$, where $j \neq i$. Specifically, a larger value of s_{ij} indicates that $\mathbf{x}_j$ is more representative of $\mathbf{x}_i$. In addition, from a view of graph-based semi-supervised learning, for any two facial images, the smaller their local distance, the more likely they belong to the same person. As a result, a larger value of s_{ij} implies that the name labels of $\mathbf{x}_i$ and $\mathbf{x}_j$ are more likely to be the same.

Based on the above motivation, we can give the following formulation to enhance the initial weak label matrix $\tilde{Y}$ as follows:

$$\min_{Y \geq 0} \frac{1}{2} \sum_{i,j} s_{ij} \|Y_{i*} - Y_{j*}\|^2 + \lambda \|(Y - \tilde{Y}) \circ Z\|_F^2 \quad (5.3)$$

where $Z = [h(\tilde{Y}_{ij})]$ is an indicator matrix: $h(x) = 1$ if $x > 0$ and otherwise $h(x) = 0$, and $\circ$ denotes the Hadamard product of two matrices. In the above objective function, the first term enforces that the class labels of two facial images $\mathbf{x}_i$ and $\mathbf{x}_j$ to be similar if the local sparse coefficient s_{ij} is large, and the second term is a regularization term that prevents the refined

label matrix being deviated too much from the initial weak matrix. Since the initial label matrix is noisy, we apply the regularization of the second term on only these nonzero elements in $\tilde{Y}$.

In general, the optimal solution to the problem in Eq. 5.3 is dense; however, the ideal true label matrix is often very sparse. We introduce some convex sparsity constraints to take into the consideration of sparsity, i.e., $\|Y_{i*}\|_1 \leq \varepsilon, \varepsilon \geq 1$, where $i = 1, 2, \dots, n$ (we choose $\varepsilon = 1$ in this work). These constraints are included to limit the number of name labels assigned to each facial image.

5.1.2.3 Weak Label Regularized Local Coordinate Coding

The above two optimization tasks of “sparse feature learning” and “label enhancement” are performed separately. Specifically, the sparse features S are learned from the optimization in Eq. 5.1. Then it is used by the optimization in Eq. 5.3 to refine the label matrix Y . To better exploit the potential of the two learning approaches, we propose the Weak Label Regularized Local Coordinate Coding (WRLCC) scheme, which aims to reinforce the two learning tasks via a unified optimization framework. Specifically, the optimization of WRLCC can be formulated as follows:

$$\begin{aligned} \min_{\hat{S}, Y} E_1(\hat{S}; X) + E_2(Y, S) &= \min_{\hat{S}, Y} \frac{1}{2} \|B\hat{S} - X\|_F^2 + \\ &\lambda_1 \text{tr}(\mathbf{1} \cdot (\hat{S} \circ V)) + \lambda_2 \text{tr}(Y^\top LY) + \lambda_3 \|(Y - \tilde{Y}) \circ M\|_F^2 \\ \text{s.t. } \hat{S}_{ii} &= 0, \|Y_{i*}\|_1 \leq 1, i = 1, 2, \dots, n, \hat{S} \geq 0, Y \geq 0 \end{aligned} \quad (5.4)$$

where $V \in \mathbb{R}^{(n+d) \times n}$, $V_{ij} = \|B_{*i} - X_{*j}\|^2$, $L = D - S$, D is a diagonal matrix, with $D_{ii} = \frac{\sum S_{i*} + \sum S_{*i}}{2}$, $Y \in \mathbb{R}^{n \times m}$, $\mathbf{1}$ is all-one-element matrix with dimension $n \times (n + d)$, and $\text{tr}(\cdot)$ denotes a trace function. In the above, $\lambda_2 \text{tr}(Y^\top LY)$ is a label smoothness regularizer which connects the label matrix and the sparse features. In Figure 5.1, we show the whole process of the proposed WRLCC algorithm with a simple example by using the top $n = 8$ retrieval results and $m = 4$ candidate names.

Remark. Although WRLCC is similar to some existing sparse representation annotation schemes, it differs from the existing approaches in two major aspects. First, instead of using the traditional sparse representation for semantic graph construction, a non-negative local coordinate coding algorithm is adopted, which exploits the advantages of both NNSC [45] and LCC [114]. Second, WRLCC employs a unified optimization scheme to solve both sparse feature learning and label enhancement in an iterative approach; as a result, the enhanced label information is also exploited by the sparse coding step, which in some sense becomes a weakly supervised sparse learning approach while traditional sparse coding is often unsupervised.

5.1.3 Optimization

The optimization problem in Eq. 5.4 is generally non-convex. To solve this challenging optimization, we propose to solve $\hat{S}$ and Y alternatively by iteratively solving two optimization steps: (1) *Code Learning*, and (2) *Label Learning*. The step of updating $\hat{S}$ can be transformed into a weighted non-negative sparse coding problem, and the step of updating Y is a quadratic programming problem.

5.1.3.1 Code Learning

By first fixing the label matrix Y and ignoring the constant terms, the optimization problem in Eq. 5.4 can be reformulated as follows:

$$\begin{aligned} Q_Y(\hat{S}) = \min_{\hat{S}} & \frac{1}{2} \|B\hat{S} - X\|_F^2 + \text{tr}(\mathbf{1} \cdot (\hat{S} \circ Z)) \\ \text{s.t. } & \hat{S}_{ii} = 0, i = 1, 2, \dots, n, \text{ and } \hat{S} \geq 0 \end{aligned} \quad (5.5)$$

where $Z = \lambda_1 V + \lambda_2 U$, $U \in \mathbb{R}^{(n+d) \times n}$, for all $j = 1, 2, \dots, n$, if $i \leq n$, $U_{ij} = \frac{1}{2} \|Y_{i\star} - Y_{j\star}\|^2$; otherwise $U_{ij} = 0$. The other variables follow the same definitions in Eq. 5.4.

The optimization problem in Eq. 5.5 can be further separated into a series of sub-problems for each coding coefficient $\hat{S}_{\star i}$ of facial image $X_{\star i}$. Each sub-problem is a weighted non-

negative sparse coding problem, which can be written into the following optimization:

$$\min_{\hat{\mathbf{s}} \geq 0} \frac{1}{2} \|B\hat{\mathbf{s}} - X_{*i}\|^2 + \sum_{k=1}^{n+d} \hat{s}_k Z_{ki} \quad \text{s.t.} \quad \hat{s}_i = 0. \quad (5.6)$$

In our approach, we adopt the Fast Iterative Shrinkage and Thresholding Algorithm (FISTA) [8], a popular and efficient algorithm for the linear inverse problem that has been already implemented for sparse learning in [60]. Since we aim to solve the problem related to only the top-ranking images in the retrieval results, where n is usually small, FISTA is efficient enough for our application.

5.1.3.2 Label Learning

Similarly, by fixing $\hat{S}$ and ignoring the constant terms, the optimization problem in Eq. 5.4 can be reformulated as follows:

$$\begin{aligned} Q_{\hat{S}}(Y) = \min_Y \text{tr}(Y^\top L Y) + \lambda \|(Y - \hat{Y}) \circ Z\|_F^2 \\ \text{s.t. } Y \geq 0, \|Y_{i*}\|_1 \leq 1, i = 1, 2, \dots, n. \end{aligned} \quad (5.7)$$

where $\lambda = \frac{\lambda_3}{\lambda_2}$, and the other variables follow the same definitions in Eq. 5.4. The optimization problem is a quadratic program with a series of closed and convex constraints (ℓ_1 ball). In order to efficiently solve this problem, we first vectorize the label matrix $Y \in \mathbb{R}^{n \times m}$ into a column vector $\bar{\mathbf{y}} = \text{vec}(Y) \in \mathbb{R}^{(n \cdot m) \times 1}$, and then rewrite the previous optimization problem as follows:

$$Q(\bar{\mathbf{y}}) = \min_{\bar{\mathbf{y}} \geq 0} \bar{\mathbf{y}}^\top \Psi \bar{\mathbf{y}} + \mathbf{c}^\top \bar{\mathbf{y}}, \quad \text{s.t.} \quad \forall i, \sum_{k=0}^{m-1} \bar{y}_{k \cdot n + i} \leq 1. \quad (5.8)$$

where $\Psi = I_m \otimes L^\top + \lambda R$, I_m is an identity matrix with dimension $m \times m$, $R = \text{diag}(\text{vec}(M))$, $\mathbf{c} = -2\lambda R^\top \cdot \text{vec}(\tilde{Y})$, and $\bar{y}_j$ is the j -th element in $\bar{\mathbf{y}}$. In order to solve this problem, we employ the multi-step gradient scheme [8] and the efficient Euclidean projection algorithm [61].

Specifically, in order to achieve the optimal solution $\bar{\mathbf{y}}^*$, we recursively update two sequences $\{\bar{\mathbf{y}}^{(k)}\}$ and $\{\mathbf{z}^{(k)}\}$ in the multi-step gradient scheme, where k is the iteration step.

Commonly at each iteration k , the variance $\mathbf{z}^{(k)}$ is named as the search point and used to construct the combination of the two previous approximate solutions $\bar{\mathbf{y}}^{(k-1)}$ and $\bar{\mathbf{y}}^{(k-2)}$. In our problem, the sub-block problem is defined as:

$$\bar{\mathbf{y}}^{(k+1)} = \arg \min_{\bar{\mathbf{y}} \geq 0} \frac{t}{2} \|\bar{\mathbf{y}} - \mathbf{v}\|^2 \text{ s.t. } \forall i, \sum_{k=0}^{m-1} \bar{y}_{k \cdot n + i} \leq 1. \quad (5.9)$$

where $\mathbf{v} = \mathbf{z}^{(k)} - \frac{1}{t}\mathbf{g}$ and $\mathbf{g} = 2\Psi\mathbf{z}^{(k)} + \mathbf{c}$, t could be a fixed positive constant or searched in a backtracking step [8]. The key problem is to solve Eq. 5.9 efficiently. In our work, we employ the Euclidian projection algorithm [61], which has been shown as an efficient solution to such kind of ℓ_1 ball constrained problem with linear time complexity of $O(n)$ as compared with other algorithms of $O(n \log(n))$.

Further, in order to apply the Euclidian projection algorithm, we split $\bar{\mathbf{y}}$ into a series of m -dimensional sub-vectors with $\bar{\mathbf{y}}^i = [\bar{y}_{k \cdot n + i}]_{k=0}^{m-1}$, where $i = 1, 2, \dots, n$, and similarly we can split the vector $\mathbf{v}$. Specifically, each optimal solution $\bar{\mathbf{y}}^{i*}$ for sub-vector $\bar{\mathbf{y}}^i$ can be achieved by solving the following problem:

$$\bar{\mathbf{y}}^{i*} = \arg \min_{\bar{\mathbf{y}}^i \geq 0} \frac{t}{2} \|\bar{\mathbf{y}}^i - \mathbf{v}^i\|^2 \text{ s.t. } \|\bar{\mathbf{y}}^i\|_1 \leq 1. \quad (5.10)$$

The problem in Eq. 5.10 has a non-negative constraint, which is different from the one in [61]. The optimal solution to this problem is given as:

$$\bar{y}_j^{i*} = h(v_j^i) \max(|v_j^i| - \lambda^*, 0), j = 1, 2, \dots, m. \quad (5.11)$$

where $h(\cdot)$ is the indicator function in Eq. 5.3, $\bar{y}_j^i$ and v_j^i are the j -th elements in $\bar{\mathbf{y}}^i$ and $\mathbf{v}^i$, and λ^* is the optimal solution for the dual form of Eq. (5.10). Suppose $S = \{j | v_j^i \geq 0\}$, the value of λ^* can be computed as follows:

$$\lambda^* = \begin{cases} 0 & \sum_{k \in S} v_k^i \leq 1, \\ \bar{\lambda} & \sum_{k \in S} v_k^i > 1. \end{cases} \quad (5.12)$$

where $\bar{\lambda}$ is the unique root of function $f(\lambda) = \sum_{k \in S} \max(|v_k^i| - \lambda, 0) - 1$, which is continuous and monotonically decreasing in $(-\infty, \infty)$. The root $\bar{\lambda}$ can be achieved by a bisection search

in linear time complexity of $O(\dim(\bar{\mathbf{y}}^i))$. It is well-known that the multi-step gradient scheme converges at $O(\frac{1}{k^2})$, where k is the iteration step, indicating our optimization can be solved efficiently.

5.1.4 Face Name Annotation

After obtaining the sparse coding matrix S and the enhanced label matrix Y , the last key step of our framework is to perform the face name annotation. First, the query facial image $\mathbf{x}_q$ is also projected into the local coordinate coding space. Specifically, the new coding $\mathbf{s}_q$ w.r.t. $\mathbf{x}_q$ can be achieved by solving the following optimization problem:

$$[\mathbf{s}_q; \xi_q] = \arg \min_{\hat{\mathbf{s}} \geq 0} \frac{1}{2} \|B\hat{\mathbf{s}} - \mathbf{x}_q\|^2 + \lambda \sum_k \hat{s}_k \|B_{*k} - \mathbf{x}_q\|^2 \quad (5.13)$$

where the parameter setting is the same as the previous WRLCC algorithm.

Although the manifold structure is supposed to be well-fitted in the new local coordinate space, it is still unsuitable to directly use a one-vs-one similarity measure for face name annotation. For the second step, we employ a sparse reconstruction scheme in the local coordinate coding space to recover the potential weighting vector $\mathbf{w}_q$ for name annotation in the label space. The optimization problem is given as follows:

$$\min \|\mathbf{w}_q\|_1 \quad \text{s.t.} \quad \mathbf{s}_q = S_{I_n} \cdot \mathbf{w}_q \quad (5.14)$$

where $S_{I_n} = S + I_n$, S is the local coordinate codes obtained from the previous step and I_n is a $n \times n$ identity matrix.

Finally, the label vector $\mathbf{y}_q$ can be directly computed by:

$$\mathbf{y}_q = Y^\top \cdot \mathbf{w}_q$$

The value y_{qk} , $k \in \{1, 2, \dots, m\}$ measures the confidence of the k -th name being assigned to the query facial image $\mathbf{x}_q$. Thus, a name with a large confidence value will be ranked on the top position for the final annotation result.

5.2 Offline Approximation for WRLCC

The proposed WRLCC algorithm requires a small set of similar facial images for each query image, which can be efficiently solved in a short time, however, its computation cost may be still too heavy for the time-critical applications. To reduce the computational cost time of the proposed WRLCC algorithm, one straight-forward solution is to adopt the PCA dimension reduction techniques over the original high dimensional feature space. The smaller the new dimension space is, the less time it takes for WRLCC algorithm. The key limitation for the PCA-based approximation is the information loss during dimension reduction may affect the final annotation performance.

In this section, we propose an offline approximation scheme for WRLCC (AWRLCC for short), which could both significantly reduce the running-time and achieve comparable results. In details, we first pre-compute the local coordinate coding for each facial image in the retrieval database with its own n neighborhoods by Eq. 5.13 and save all the coding results. Then, in the annotation step, for each query image we could directly reconstruct the sparse features of its n nearest instances based on the offline coding results without extra computational costs.

We denote by $\mathcal{D} = \{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N\}$ the whole retrieval database, where N is the total number of facial images. For each facial image $\mathbf{d}_i$, we denote by $\mathfrak{N}(\mathbf{d}_i)$ its n similar images in the retrieval database except itself. Then, we can achieve its encoding result $\mathbf{c}_i$ with Eq. 5.13, where the dictionary is constructed as $[\mathfrak{N}(\mathbf{d}_i), I]$. For the query image $\mathbf{x}_q$, its n most similar facial images $X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n] = [\mathbf{d}_{\mathcal{J}_1}, \mathbf{d}_{\mathcal{J}_2}, \dots, \mathbf{d}_{\mathcal{J}_n}]$ is also a subset of the whole retrieval database $\mathcal{D}$, where $\mathcal{J}$ is an index vector for the whole retrieval database. In the offline computation step, the sparse coding of $\mathbf{x}_i$ (or $\mathbf{d}_{\mathcal{J}_i}$) is $\mathbf{c}_{\mathcal{J}_i}$ and its j -th element $c_{\mathcal{J}_i j}$ measures the weight of the j -th item in the nearest neighbor set $\mathfrak{N}(\mathbf{d}_{\mathcal{J}_i})$. As a result, the key problem in our approximation scheme is to construct $\mathbf{x}_i$'s (or $\mathbf{d}_{\mathcal{J}_i}$'s) new coding result $\mathbf{s}_i$ for WRLCC based on the previous coding results. The new coding results should share the same dictionary $\bar{X}$, which is

 (a) Database and Query (b) Point Relationship (c) Original Features $\rightarrow$ New Features

Figure 5.2: An example for AWLRLCC. (a) shows the database images $\bullet$ ($\{d_1, d_2, \dots, d_9\}$) and query image $\otimes$ (x_q); (b) shows the nearest neighbor sets of x_q , d_4 , d_3 , and d_5 ; (c) shows the value mapping rules between the pre-computed coding c_i and the reconstructed coding s_i based on the new dictionary $\bar{X}$. As an example, for the facial image d_4 with $j = 1$, the corresponding dictionary item is d_5 and $\Upsilon(\bar{X}, d_5) = 5$, so that $s_{45} = c_{41}$.

constructed with all the neighbor sets $\mathfrak{N}(d_{j_i})$ of d_{j_i} , $i = 1, 2, \dots, n$:

$$\bar{X} = \bigcup_{i=1}^n \mathfrak{N}(d_{j_i}) \bigcup X \quad (5.15)$$

where the number of dictionary items in $\bar{X}$ is $\bar{n}$, $\bar{n} \geq n$. Define a function $\Upsilon(\bar{X}, d)$ which could return the index value of the vector d in the dictionary $\bar{X}$. As a result, the new coding result $s_i \in \mathbb{R}^{\bar{n}}$ of x_i could be constructed based on $\bar{X}$ with the following rules: i) For $j = 1, 2, \dots, n$, $s_{i\Upsilon(\bar{X}, \mathfrak{N}(d_{j_i})_j)} = c_{j_i j}$. ii) The left items in s_i are set as zero. Similar to the general WLRCC algorithm, the local coordinate coding result s_q of x_q could be achieved by solving the problem in Eq. 5.13, where $B = [\bar{X}, I]$.

Figure 5.2 shows an example for AWLRLCC, where $\otimes$ denotes the query image x_q , $\bullet$ denotes the database images, and $n = 3$. As shown in Figure 5.2(b), the nearest neighbor set of x_q is $\mathfrak{N}(x_q) = \{x_1, x_2, x_3\} = \{d_4, d_3, d_5\}$, and the index vector is $\mathfrak{J} = [4, 3, 5]$. Based on the reconstructed dictionary $\bar{X} = \mathfrak{N}(d_4) \cup \mathfrak{N}(d_3) \cup \mathfrak{N}(d_5) \cup \mathfrak{N}(x_q) = [d_1, d_2, \dots, d_9]$, the value mapping rules between the pre-computed coding c_i and the reconstructed coding s_i are illustrated in Figure 5.2(c). E.g., for the facial image d_4 with $j = 1$, the corresponding dictionary item is d_5 and $\Upsilon(\bar{X}, d_5) = 5$, so that $s_{45} = c_{41}$.

5.3 Experimental Results

5.3.1 Experiment Testbed

We compare the proposed WRLCC against a soft-max weighted majority voting based baseline algorithm “SMW” and five state-of-the-art algorithms for face/image annotation, including “S-Recon” [105], “SGSSL” [94], “LCC” [114], “MtBGS” [107], and “MRR” [108]. Most of these algorithms were proposed in recent years, as briefly introduced in Section 2.

To evaluate the annotation performance, we adopt the *hit rate* at top- T annotated results as the performance metric, which measures the likelihood of having the true label among the top- T annotated names. Specifically, for $T = 1$, the *hit rate* is the same with the *accuracy*. In our experiments, we adopt the “WDB”-series database, which is introduced in Chapter 3. It consists of four retrieval databases in different sizes and one query database with about 1,600 images. In particular, we adopt the “WDB-040K”, “WDB-100K”, “WDB-200K”, and “WDB-600K” databases in the experiments of this chapter

Furthermore, we discuss the cross validation issue. For all the algorithms, we conduct it in the experiments on the WDB-040K; as a result, the parameters with the best average hit rate results are directly used in the following experiments based on the other databases. Specifically, we randomly divide the query set into two parts of equal size, in which one part is used as the validation set to find the optimal parameters by a grid search, and the other part is used for performance evaluation. Such a procedure is repeated 10 times and the average performance is computed over the 10 trials.

5.3.2 Impact of the Top- n and Top- T Settings

This experiment aims to evaluate the impact on the final annotation performance by varied settings of n and T values for Top- n retrieved facial images and Top- T annotated names.

5.3.2.1 Impact of the Top- T Settings

Table 5.2 shows the hit rate performance of the baseline SMW algorithm ($n = 10$) and the WRLCC algorithm ($n = 40$) with different T values based on the WDB-040K database.

Table 5.2: The hit rate performance of SMW ($n = 10$) and WRLCC ($n = 40$) with different Top- T

	$T = 1$	$T = 2$	$T = 3$	$T = 4$	$T = 5$
SMW	0.6090	0.7200	0.7631	0.7881	0.8006
WRLCC	0.7671	0.8100	0.8356	0.8481	0.8575

We observe that for a fixed n value increasing the value of T generally leads to a better annotation performance, but the improvement becomes marginal when T is large. This observation is not surprising as more names get a better chance to hit the relative name. Specially, for $T = 1$ the hit rate measures the precision of the annotation system, and for a large T value, the hit rate is very similar to the recall metric. In practice, we mainly focus on a small T value since users usually would not be interested in a long name list.

5.3.2.2 Impact of the Top- n Settings

Figure 5.3 (a) and Table 5.3 show the hit rate performance of the baseline SMW algorithm and the WRLCC algorithm with different n values based on the WDB-040K database. For both of the two algorithms we fix T to 1. The average running time of WRLCC algorithm for one query sample is presented in Figure 5.3 (b) and the last row of Table 5.3.

Table 5.3: The hit rate performance of SMW and WRLCC with different Top- n , and the running time of WRLCC.

	$n = 10$	$n = 20$	$n = 40$	$n = 60$	$n = 80$
SMW	0.6090	0.5738	0.5513	0.5300	0.5119
WRLCC	0.7131	0.7413	0.7671	0.7775	0.7863
Time(s)	0.295	0.579	1.205	2.077	3.337

Figure 5.3: (a) The hit rate performance of SMW and WRLCC with different Top- n , (b) The average running time of WRLCC for one query sample.

Several observations can be made from the experiment results. First of all, for the baseline SMW algorithm, when increasing the value of n , the annotation performance decreases. However, for the proposed WRLCC algorithm, increasing n leads to the improvement of the annotation performance. Generally, for two facial images, the smaller their local distance is, the more likely they belong to the same person. As a result, for the top n similar instances, a small n value results in a high precision and a low recall, while a large n value often produces a high recall and a low precision. Since SMW is a simple weighted majority voting algorithm without any re-ranking, it favors the retrieval result with a high precision; thus a small parameter n gets a better performance. On the other hand, the WRLCC algorithm carries a re-ranking scheme and typically prefers a high recall of the retrieved results such that a potentially larger pool of facial images can be exploited for re-ranking. Second, the running time of the WRLCC algorithm increases for a large n value, as more instances will be encoded and the dictionary is enlarged.

These previous observations are beneficial for determining the appropriate parameters in our following experiments. For SWM, its parameter n for Top- n retrieved images is set to 10. For the other algorithms that usually carry a re-ranking or sample selection step, we set the parameter n to 40, which is a trade-off between the annotation performance and running time.

5.3.3 Evaluation of Automatic Face Annotation on “WDB-040K”

In this experiment, we compare the proposed WRLCC algorithm with the other algorithms on the WDB-040K database. Table 5.4 show the average annotation performance at different T values. We can make several observations from the results.

Table 5.4: The comparison among WRLCC and other algorithms on database WDB-040K.

	$T = 1$	$T = 2$	$T = 3$	$T = 4$	$T = 5$
WRLCC	0.7671 ± 0.010	0.8009 ± 0.010	0.8263 ± 0.010	0.8361 ± 0.009	0.8496 ± 0.007
S-Recon	0.7369 ± 0.011	0.7830 ± 0.012	0.8080 ± 0.010	0.8273 ± 0.009	0.8400 ± 0.011
SGSSL	0.7310 ± 0.011	0.7770 ± 0.010	0.8079 ± 0.011	0.8231 ± 0.013	0.8338 ± 0.012
MRR	0.6593 ± 0.015	0.7448 ± 0.010	0.7829 ± 0.008	0.7973 ± 0.009	0.8051 ± 0.010
MtBGS	0.7246 ± 0.012	0.7796 ± 0.011	0.8019 ± 0.009	0.8169 ± 0.010	0.8311 ± 0.009
MSC	0.6546 ± 0.012	0.6938 ± 0.012	0.7194 ± 0.010	0.7336 ± 0.009	0.7446 ± 0.009
SMW	0.6090 ± 0.019	0.7150 ± 0.015	0.7599 ± 0.012	0.7848 ± 0.012	0.7969 ± 0.012

First, it is clear that the proposed WRLCC algorithm consistently performs better than the other algorithms, especially for small T values. Considering $t = 1$, the performance of the baseline SMW algorithm is about 61.3%. The proposed WRLCC can boost the performance to 77.1%. The increased accuracy validates the effectiveness of the proposed WRLCC algorithm in improving retrieval-based face annotation. Second, greater performance improvement is achieved by the WRLCC algorithm for a small T value, which is more critical for real applications. In practice, we mainly focus on a small T value since users usually would not be interested in a long list of annotated names.

5.3.3.1 Evaluation on “WDB” with Varied Number of Persons

We examine the impact of the number of persons over the following four databases: WDB-040K, WDB-100K, WDB-200K, WDB-400K, and WDB-600K, where the numbers of persons are increased from 400 to 6,025. The experiment results are presented in Figure 5.4. For clarity, only the annotation result with $\text{Top-}T=1$ are shown.

Figure 5.4: Comparison between WLRCC and the other algorithms on WDB-040K, WDB-100K, WDB-200K, WDB-400K, and WDB-600K

We can make a few observations from the results. First, similar to the previous observations, the WLRCC algorithm consistently achieves the best annotation performance among all the compared algorithms for different database sizes. Second, it is clear that the annotation performance will decrease when the number of persons increases, similar to the observation on facial image retrieval in [108]. With the increase in the number of persons, more noisy images are collected into the retrieval database for the same query image, making the retrieval problem more challenging.

5.3.4 Evaluation of Approximation for WLRCC

In this section, we aim to evaluate the running time of the proposed WLRCC algorithm, the PCA-based approximation scheme, and the offline approximation scheme (AWLRCC). For

PCA-based approximation, we randomly select 1,000 images from the retrieval database to learn the eigenvectors, then both the query image $\mathbf{x}_q$ and the top- n nearest neighbors X are projected into a new k -dimensional feature space, where the general WRLCC algorithm is applied for annotation. For the offline approximation, all the facial images in the retrieval database are first encoded based on its top- n neighbors, then the query image x_q is annotated with its top-rank candidate images based on the proposed offline approximation scheme for WRLCC (AWRLCC). All the experiment results are presented in Table 5.5.

Table 5.5: The running time of different approximation schemes, based on database WDB-040K .

	WRLCC	PCA-300	PCA-200	PCA-100	AWRLCC
Time	1.205	0.773	0.630	0.524	0.216
(s)	± 0.041	± 0.028	± 0.020	± 0.030	± 0.017
Hit	0.7671	0.7589	0.7538	0.7453	0.7602
Rate	± 0.010	± 0.051	± 0.011	± 0.024	± 0.020

Several observations can be made from the experiment results. First, by using the PCA dimension reduction, the running time can be considerably reduced. For example, for the new feature “PCA-200”, the PCA-based approximation scheme needs about 0.63 second, however the annotation performance also decreases to 75.9%. Second, the proposed offline approximation algorithm (AWRLCC) takes less time than the PCA-based scheme, but achieves a better annotation result. For example, it achieves the best approximation result 76.0% with only about 0.22 second. Although the proposed “AWRLCC” algorithm achieves a good approximation result with less running time, its drawback is that it takes extra storage (memory) space because all the pre-computed results should be saved for the on-the-fly reconstruction, and pre-computing would also take a long time. In order to further evaluate the approximation result of AWRLCC, we compare it with the general WRLCC over all the databases, as shown in Figure 5.5. Apparently, for different databases, the proposed AWRLCC can always

achieve good approximation results, which is only about 1% worse than the general WRLCC algorithm.

Figure 5.5: Comparison between WRLCC and AWRLCC.

5.3.4.1 Comparison between ULR and WRLCC

The ULR algorithms in the previous chapter aims to refine the weak labels over the whole database, and its simple majority voting scheme is not effective. In contrast, the WRLCC algorithm carries a re-ranking scheme by fully exploiting the short list of top similar images via a unified optimization scheme. In this experiment, we adopted the WRLCC algorithm over the same data with the ULR algorithm, as shown in the fourth column of Table 5.6. The experiment result shows that the WRLCC algorithm can archive a better result by applying a more sophisticated annotation scheme.

Since the WRLCC algorithm and ULR algorithm focus on different part in the SBFA framework. The ULR algorithm can be adopted as a preprocessing step over the whole retrieval database for the WRLCC algorithm. In other words, the WRLCC algorithm could be subsequently used based on the refined label matrix achieved by ULR algorithm, where

the ULR refined label matrix is used to construct the initial matrix $\tilde{Y}$ for each query image in Eq. 5.4. Obviously, this can further improve the annotation performance. The improved result is shown in the fifth column of Table 5.6.

Table 5.6: The performance of WRLCC and the cascade framework combined with ULR.

	ORI	ULR	WRLCC	ULR+WRLCC
Hit Rate	0.548	0.715	0.766	0.784
	± 0.013	± 0.008	± 0.016	± 0.013

As a summary, for the search-based face annotation task, the proposed WRLCC algorithm could achieve promising results based on different database sets. It also can be combined with the previous ULR algorithm subsequently to further improve the annotation performance. Furthermore, for the other related face annotation problems, the WRLCC algorithm can be easily adapted and achieve competitive results with the state-of-the-art algorithms on the public web facial image database.

5.3.5 Evaluation of Parameter Sensitivity

For the proposed WRLCC algorithm, the best parameters λ_1 and λ_3 are found by a grid search on the WDB-040K database, and are also adopted in the experiments on other databases. Figure 6.5 shows the grid search results, where the ranges for λ_1, λ_3 are $\{0.001, 0.005, 0.01, 0.1, 1, 3, 5\}$. From the figure, we notice that the performance of WRLCC tends to be stable in some region $\lambda_1 \in [0.005, 0.1]$ and $\lambda_3 \in [0.1, 3]$. We thus choose $\lambda_1 = 0.01$ and $\lambda_3 = 1$ in our experiments, which are also adopted in the other experiments on databases of varied scales. As shown in Figure 5.4, the encouraging performance of WRLCC indicates that it is generally robust in terms of the parameter setting for similar experiments.

Figure 5.6: Grid search result of WRLCC on WDB-040K.

5.4 Summary

In this chapter, we tackle the second challenge in the SBFA framework, which aims to exploit the short list of candidate facial images and their weak label information for the face name annotation. We propose a Weak Label Regularized Local Coordinate Coding (WRLCC) algorithm, which effectively exploits the principles of both local coordinate coding and graph-based weak label regularization. Moreover, a sparse reconstruction scheme is developed to perform face annotation task. We have conducted extensive empirical studies on several web facial image databases, which validate the efficacy of WRLCC. To further improve the efficiency and scalability, we then propose an offline approximation scheme (AWRLCC) to speed up the original WRLCC algorithm, saving a significant amount of computational time while maintaining highly comparable performance. In the following two chapters, we will tackle the third challenge by adopting sophisticated machine learning techniques.

Chapter 6

Fusion of Transductive and Inductive Learning for Face Annotation

In general, the automatic face annotation problem can be solved by two types of methodologies: the “*transductive learning*” and the “*inductive learning*”. To solve face annotation problem from the view of “*inductive learning*”, one can apply some classical inductive (or model-based) face recognition/verification algorithms, which have been extensively studied for many years. To solve the face annotation problem from the view of “*transductive learning*”, one can apply the search-based face annotation (SBFA) scheme, which is a data-driven approach.

In this chapter, we propose a Unifying Transductive and Inductive Learning (UTIL) by combining these two learning techniques in last-classification fusion scheme. From the “*inductive learning*” view, we propose a new Weak Label Laplacian Support Vector Machine (WL-LapSVM) algorithm for generating effective classification models from weakly labeled web facial images; from “*transductive learning*”, we apply the proposed Weak Label Regularized Local Coordinate Coding (WLRLCC) algorithm in Chapter 5; finally, we propose an entropy-based fusion scheme for combining the annotation results from the two different learning schemes to maximize the learning efficacy. In the following sections, the proposed UTIL algorithm will be introduced in detail.

6.1 Fusion of Transductive and Inductive Learning

In this section, we briefly introduce the proposed framework of Unifying (Fusion) Transductive and Inductive Learning (UTIL) for auto face annotation. It combines both transductive and inductive learning techniques in a systematic approach.

For the “*transductive learning*” step, we apply the state-of-the-art Weak Label Regularized Local Coordinate Coding (WLRCC) algorithm in a SBFA paradigm, which aims to annotate the query image by fully exploring the top- n similar images and their corresponding labels. Two key factors affect its final annotation performance: (1) Generating more represented feature for re-ranking as all the top ranking images are close to each other in the original feature space; (2) Enhancing the initial weak labels. In WLRCC algorithm, these two problems are tackled simultaneously in one optimization problem.

For the “*inductive learning*” step, we have to address the problem of insufficient labeled data for training effective classifiers. Since the number of images and the number of persons are both large in the web facial image database, it is impossible and impractical to label all the facial images due to the expensive human labeling costs. In our framework, we assume that only one facial image can be manually labeled for each person. On the other hand, all the facial images are weakly labeled during the crawling step. As a result, the core problem is how to effectively train classifiers based on a small number of well labeled data and a large amount of weakly labeled data. To tackle issue, a natural choice is to explore semi-supervised learning techniques, e.g., semi-supervised support vector machines. However, the conventional semi-supervised learning techniques cannot deal with weakly labeled data properly. In this paper, we propose the Weak Label Laplacian Support Vector Machines (WL-LapSVM) algorithm to overcome the challenge.

To achieve the final annotation result, we combine the annotation results from both “*transductive learning*” and “*inductive learning*.” In general, the annotation problem can also be formulated as a multi-class classification problem. Hence, the previous combination task is a

typical classifier ensemble problem, which is one of the widely explored methods in pattern recognition. Combining classifiers can reduce the error rate in classification tasks in opposite to single classifiers. In addition, the combination of different techniques can make the performance of the system more robust against the difficulties that each individual classifier may have on each particular data set [68]. As stated in [28], the base classifiers should be different, but they should be comparable as well. In our framework, the “*transductive learning*” and “*inductive learning*” schemes are derived from different schemes, and achieve comparable annotation performance. Hence, the combination is helpful to improve the annotation performance. In our experiments, we evaluate several last-fusion scheme to merge the two annotation results. We propose an entropy based weighting combination scheme, which achieves fairly good fusion result with less computation effort.

6.2 Weak Label Laplacian SVM (WL-LapSVM)

In this section, we present the proposed Weak Label Laplacian SVM (WL-LapSVM) algorithm for solving the “*inductive learning*” task in the proposed Unifying Transductive and Inductive Learning (UTIL) scheme. To clarify the formulation, Table. 6.1 lists the key notations used in this chapter.

6.2.1 Preliminaries

We denote the whole retrieval database and the corresponding name labels with $D = \{(\mathbf{x}_i, \mathbf{y}_i) | i = 1, 2, \dots, N\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is a d -dimensional facial feature vector, $\mathbf{y}_i \in \{0, 1\}^M$ is the corresponding name label vector with only one non-zero value: $\|\mathbf{y}_i\|_0 = 1$, N denotes the total number of facial images in the whole database, and M is the total number of unique human names. For simplicity, in the following sections, we denote the label vector $\mathbf{y}_i$ by y_i , which equals the index value of the non-zero item in $\mathbf{y}_i$. Further, we denote by $D^j = \{(\mathbf{x}_k, y_k) \in D | y_k = j\}$ the subset of the crawled images belonging to the j -th name(person).

Table 6.1: List of Key Notations

Notation	Description
N	# facial images in the whole retrieval database
M	# names in the whole retrieval database
d	# feature dimensions (e.g. $m = 512$ for GIST feature)
$D = \{(\mathbf{x}_i, \mathbf{y}_i)\}$	The whole retrieval database, $i = 1, 2, \dots, N$
$\mathbf{x}_i \in \mathbb{R}^d$	The i -th facial image in the retrieval database
$\mathbf{y}_i \in \{0, 1\}^M$	The corresponding label vector of $\mathbf{x}_i$, $\ \mathbf{y}_i\ _0 = 1$
y_i	Class label of $\mathbf{x}_i$, $y_i = j$ if $\mathbf{y}_i[j] \neq 0$
$D^j = \{(\mathbf{x}_k, y_k)_{y_k=j}\}$	The subset of the crawled images belonging to the j -th name(person)
L^j, U^j	The <i>label</i> set and <i>unlabel</i> set of the image set D^j , $D^j = L^j \cup U^j$
$\mathbf{f}^j$	The classification model of j -th class
T^j	The training set of j -th class for model training
l	# label samples in training set T^j
u	# unlabel samples in training set T^j
$z \in \{-1, 1\}$	The class label of training sample

In order to train inductive classifiers for each person, we manually label a small number of facial images as the preliminary set of labeled images. As a result, for the j -th name(person) in the retrieval database, its image set D^j can be further divided into two subsets: the *label set* L^j and the *unlabel set* U^j , where $D^j = L^j \cup U^j$. In particular, in our experiment only one image is labeled for each name (person), which means $|L^j| = 1, j = 1, 2, \dots, M$. This kind of setting is reasonable, since in real-world application the number of names is very large and it is time-consuming and impractical to label a large amount of labeled data.

As there is only one positive sample for each class (person), general semi-supervised learning technique, e.g. the Laplacian SVM, can be used to solve the small-sample problem. However, it does not work well in our problem due to the limited number of positive samples. We also notice all the facial images in the database are assigned with weak labels, which can be employed for prior information for classifier learning. To address the previous problem, we propose a variant of Laplacian SVM algorithm, the Weak Label Laplacian Support Vector Machine (WL-LapSVM), to handle the noisy web images in our weakly labeled facial image database.

6.2.2 Problem Formulation of WL-LapSVM

For each class (person), we will train a separate classification model $\mathbf{f}^j, j = 1, 2, \dots, M$, which determines whether a facial image $\mathbf{x}$ belongs to class j or not. In particular, if $\text{sign}(\mathbf{f}^j(\mathbf{x})) > 0$, then the facial image $\mathbf{x}$ is supposed to be in the j -th class and be labeled with the j -th name.

Generally, for the j -class(person), we can construct its training set T^j by introducing the labeled subset L^j from the j -th class as the labeled positive samples, introducing the labeled subsets $\{L^k | k \neq j\}$ from the other classes as the labeled negative samples, and the unlabeled subset U^j from the j -th class as the unlabeled samples. However, in our application, the size of labeled positive training set L^j is extremely small (only 1), as a result, we only collect a subset of the labeled negative samples into the training sets T^j . For simplicity, we can represent the training set T^j of the j -th class as follows:

$$T^j = \{(\mathbf{x}_1, z_1), (\mathbf{x}_2, z_2), \dots, (\mathbf{x}_l, z_l), \mathbf{x}_{l+1}, \mathbf{x}_{l+2}, \dots, \mathbf{x}_{l+u}\}$$

where $z \in \{-1, 1\}$ is the class labels, l is the number of labeled samples, and u is the number of unlabeled samples ($l + u = |T^j|$). Following the traditional Laplacian SVM algorithm (LapSVM) [67], we define a kernelized target function $\mathbf{f}^j(\mathbf{x})$ for the j -class, which is also denoted as $\mathbf{f}(\mathbf{x})$ for short in the rest parts of this section:

$$\mathbf{f}(\mathbf{x}) = \sum_{k=1}^{l+u} \alpha_k \kappa(\mathbf{x}_k, \mathbf{x}) = \boldsymbol{\alpha}^\top \mathbf{k}(\mathbf{x}), \mathbf{x}_k \in T^j \quad (6.1)$$

where $\kappa(\cdot, \cdot)$ is a kernel function, $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \dots, \alpha_{l+u}]$ and $\mathbf{k}(\mathbf{x}) = [\kappa(\mathbf{x}_1, \mathbf{x}), \kappa(\mathbf{x}_2, \mathbf{x}), \dots, \kappa(\mathbf{x}_{l+u}, \mathbf{x})]$

The traditional Laplacian SVM problem is to minimize the following objective function with respect to the previous classification function $\mathbf{f}$.

$$\min_{\mathbf{f}(\mathbf{x})} g(\boldsymbol{\alpha}) = \sum_{k=1}^l V(\mathbf{x}_k, z_k, \mathbf{f}) + \frac{\lambda_1}{2} \Phi(\mathbf{f}) + \frac{\lambda_2}{2} \|\mathbf{f}\|_A \quad (6.2)$$

where $\|\cdot\|_A$ is the norm in the Reproducing Kernel Hilber Space (RKHS) $\mathcal{H}_k$ of kernel κ .

V is a loss function on the label data, we choose the L_2 hinge loss function in our experiment :

$$V(\mathbf{x}_k, z_k, \mathbf{f}) = \frac{1}{2} \sum_{k=1}^l \max(1 - z_k \mathbf{f}(\mathbf{x}_k), 0)^2 \quad (6.3)$$

$\Phi(\mathbf{f})$ is an intrinsic regularizer to employ the geometry information among the label data and unlabeled data by the Laplacian matrix, which is defined as :

$$\Phi(\mathbf{f}) = \sum_{\mathbf{x}_i, \mathbf{x}_j \in T^j} W_{ij}(\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j))^2 = \boldsymbol{\alpha}^\top K L K \boldsymbol{\alpha} \quad (6.4)$$

where K is the kernel matrix of the instances in the training set T^j , L is the graph Laplacian matrix, and $L = D - W$, W is the adjacency matrix of the data graph in training set ($W_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{x}_i, \mathbf{x}_j \in T^j$, we choose κ as a RBF kernel function in our framework). D is a diagonal matrix with diagonal elements as $D_{ii} = \sum_j W_{ij}$.

In our application, the traditional Laplacian SVM algorithm does not work well, because the positive labeled images are very limited (only one positive reference image) and the facial images are in a high dimensional space and cannot be separated linearly. In order to overcome this challenge, we propose to employ $p \leq u$ samples from the unlabeled samples of T^j as the pseudo-positive labeled samples, which means that they are not definitely positive samples. To reduce the risk of using these unlabeled samples, we propose to assign each unlabeled instance with a confidence weighting value, which could be achieved with extra information (e.g. the ranking position in the searching result).

As a result, the previous unlabeled samples $\{\mathbf{x}_k | k = l + 1, l + 2, \dots, l + u\}$ in T^j can be represented as $\{(\mathbf{x}_k, z_k, \varepsilon_k) | k = l + 1, l + 2, \dots, l + u\}$, where the confidence weighting value $\varepsilon_k \in [0, 1]$. The label z_k is set as 1 for the collected pseudo samples, and for the uncollected samples, we can just set its confidence value as $z_k = 0$. Correspondingly, for the labeled instances in T^j , we can also assign a fixed confidence weights $\varepsilon_k = 1$ for $k = 1, 2, \dots, l$. By employing the weighted unlabeled instances into the traditional Laplacian SVM as an extra term, we achieve a new confidence value weighted Laplacian SVM formulation as follows:

$$\begin{aligned} \min_{\mathbf{f}(\mathbf{x})} g(\alpha) = & \sum_{k=1}^l \varepsilon_k V(\mathbf{x}_k, z_k, \mathbf{f}) + \frac{\lambda_1}{2} \Phi(\mathbf{f}) + \\ & \frac{\lambda_2}{2} \|\mathbf{f}\|_A + \lambda_3 \sum_{k=l+1}^{l+u} \varepsilon_k V(\mathbf{x}_k, z_k, \mathbf{f}) \end{aligned} \quad (6.5)$$

where $\varepsilon_k = 1$ for $k = 1, 2, \dots, l$, and $\varepsilon_k \in [0, 1]$ for $k = l + 1, l + 2, \dots, l + u$. We refer to the proposed modified Laplacian SVM based on weak label as “**WL-LapSVM**” for short. If $\lambda_3 = 0$, the formulation reduces to a standard Laplacian SVM algorithm.

6.2.3 Optimization Algorithm

In this section, we briefly introduce the optimization algorithm for the proposed WL-LapSVM, which could be reformulated as follows:

$$\min_{\mathbf{f}(\mathbf{x})} g(\boldsymbol{\alpha}) = \frac{1}{2} \sum_{i=1}^{l+u} \varepsilon'_i \max(1 - z_i \mathbf{k}_i^\top \boldsymbol{\alpha}, 0)^2 + \frac{\lambda_1}{2} \boldsymbol{\alpha}^\top K L K \boldsymbol{\alpha} + \frac{\lambda_2}{2} \boldsymbol{\alpha}^\top K \boldsymbol{\alpha} \quad (6.6)$$

where $\mathbf{k}_i^\top = \mathbf{k}(\mathbf{x}_i)$, $\varepsilon'_i = 1$ with $i = 1, 2, \dots, l$, and $\varepsilon'_i = \lambda_3 \varepsilon_i \in [0, \lambda_3]$ with $i = l + 1, l + 2, \dots, l + u$. In order to solve this problem, we follow the Newton Method proposed in [67].

In each step, we update $\boldsymbol{\alpha}$ with the following rule:

$$\boldsymbol{\alpha}^{(t)} = \boldsymbol{\alpha}^{(t-1)} - s H^{-1} \nabla_{\boldsymbol{\alpha}} \quad (6.7)$$

where t is the iteration number, s is the step size, and $\nabla_{\boldsymbol{\alpha}}$ and H are the gradient vector and Hessian matrix for $g(\boldsymbol{\alpha})$ in Equation 7. For ∇ , we have:

$$\begin{aligned} \nabla_{\boldsymbol{\alpha}} &= \sum_i \varepsilon'_i \mathbf{k}_i z_i (z_i \mathbf{k}_i^\top \boldsymbol{\alpha} - 1) + \lambda_1 K L K \boldsymbol{\alpha} + \lambda_2 K \boldsymbol{\alpha} \\ &= K S K \boldsymbol{\alpha} - K S \mathbf{z} + \lambda_1 K L K \boldsymbol{\alpha} + \lambda_2 K \boldsymbol{\alpha} \end{aligned} \quad (6.8)$$

where $S \in \mathbf{R}^{n \times n}$ is a diagonal matrix and its i -th element in the main diagonal is ε'_i . K is the kernel matrix of the instances in the corresponding training set. The Hessian matrix H could be achieved as follows:

$$H = \nabla_{\boldsymbol{\alpha}}^2 = K S K + \lambda_1 K L K + \lambda_2 K \quad (6.9)$$

The step size s could be fixed to 1 or optimized by line searching. The iterative update is guaranteed to converge when the error vector does not change for two consecutive iterations.

6.2.4 Name Annotation

For each name (person) in the retrieval database, we can build a WL-LapSVM model $\mathbf{f}^j, j = 1, 2, \dots, M$, respectively. To annotate the query image $\mathbf{x}_q$, we firstly compute its prediction value $y_{qj} = \mathbf{f}^j(\mathbf{x}_q), j = 1, 2, \dots, M$; then convert these prediction value into the probability scale: $p_{qj} = \frac{1}{1+\exp(-y_{qj})}$ by fitting a sigmoid function following the technique in [79]. Finally, the annotated name list is obtained by sorting the previous probability value $\{p_{q1}, p_{q1}, \dots, p_{qM}\}$.

6.2.5 Pseudo Set Construction and Confidence Weights Setting

For the proposed WL-LapSVM algorithm, there are mainly two critical problems that highly affect its annotation performance: 1) one problem is how to collect the p pseudo positive samples from the unlabeled training samples; 2) another problem is how to set the confidence value for each pseudo positive sample. In the following, we will take the j -th class as an example, where the image set is $D^j = L^j \cup U^j$ and the training set is T^j .

For the first problem, in our framework there are two ways to introduce the pseudo samples from the unlabeled set U^j : one way is to collect the images that are close to the true positive sample in L^j ; another way is to collect the images with high Google ranking values, which means that these images are at the top-ranking position in Google retrieval result. According to our experiments, we notice that the performance of the second scheme is much better than the first scheme. For example, supposed $p = 80$ extra pseudo samples are collected, the annotation performance of the second scheme is 68%, compared with 26% of the first method.

For the second problem, it aims to introduce the prior knowledge into WL-LapSVM by setting the confidence weighting value ε in Eq. 6.5. Suppose the p pseudo positive samples are $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_p\}$, sorted by the Google ranking value. In our framework, we set the confidence

Figure 6.1: Annotation performance of WL-LapSVM with different confidence weighting settings. The x -axis is the confidence weight ε_p of the last pseudo sample.

weight of $\mathbf{x}_i$ according to its index value i by using a monotonic decreasing function: $\varepsilon_i = \exp(\frac{i-1}{\theta})$, where θ is a parameter. By choosing different θ value, we can control the contribution of the pseudo positive sample set. In figure 6.1, for different number of pseudo positive samples ($p = 60$ and $p = 80$), we evaluate different confidence weights setting by making $\varepsilon_p = \exp(\frac{p-1}{\theta})$ equal 0.1, 0.2, $\dots$, 1. Obviously, when $\varepsilon_p \in [0.3, 0.6]$, the annotation performances are consistently better. As a result, we set $\varepsilon_p = 0.5$ for all the following experiments.

6.3 Fusion Strategies for Combination

The last step for the proposed UTIL framework is to combine the annotation results from both “*transductive learning*” and “*inductive learning*.” Generally, the annotation problem can also be formulated as a multi-class classification problem, and the combination is a typical late-fusion (post-classification fusion) problem. There are numerous score late fusion methods in the literature [6, 52], which can be classified into three levels: *Abstract Level*, *Rank Level*, and *Measurement level*. As there are only two “classifiers” in the UTIL framework, the voting-based *abstract level* fusion is unsuitable for our experiments. For the *rank level* fusion, each model (the “*transductive learning*” model and the “*inductive learning*” model) outputs a list of possible names for the query image, sorted in decreasing order of confidence. For the *measurement level* fusion, each model outputs the possibility (confidence) values of assigning different name to the query image.

In practical, for a query image, the annotation results of the “transductive” and “inductive” learning steps can be represented as two possibility (measurement) score vectors: $\mathbf{p}_T, \mathbf{p}_I \in [0, 1]^M$, respectively. For example, p_{Ti} illustrates the possibility of assigning i -th name to the query image according for the “*transductive learning*” model. We can easily generate the ranking results for the two models by sorting the confidence score vector $\mathbf{p}_T, \mathbf{p}_I \in [0, 1]^M$, respectively.

6.3.1 Measurement Level Fusion

For the *measurement level* fusion, following the normalization scheme in [6], we use *sum rule* for measurement combination and adopt two kinds of normalization methods: the min-max normalization which is refereed as “MF-MinMax” in the following experiments and the Z-score normalization which is referred as “MF-Zscore” for short.

6.3.2 Rank Level Fusion

For the *rank level* fusion scheme, Ho et al. [43] describe three methods to combine the ranks assigned by the different models: *highest rank* method, *Borda count* method, and *Logistic Regression* method. In our experiments, we only adopt the borda count scheme for fusion as there are just two models in the UTIL framework. Instead of directly using the rank position as the rank value, we set the rank value according to a monotonic decreasing function. For the combination weights of different ranks, we propose two kinds of methods: one is based on confidence value regression (“RF-Reg”), and another is based on the confidence value entropy information (“RF-Entropy”).

For “RF-Reg”, given a set of query images as the training set, we set the weight value w_T for the rank of the “transductive” model as 1 if the “transductive” model achieves a better annotation performance than the “inductive” model, otherwise, $w_T = 0$. Then we adopt the SVM algorithm to train a regression model for the weight value w_T by using the possibility

(measurement) scores as the feature vector. Finally, for a test query image, we use the learned regression model to predict the weighting value w_T for the rank result of the “transductive” model and generate the weighting value w_I for the rank result of the “inductive” model by $1 - w_T$.

For “RF-Entropy” method, we aim to avoid the computation cost in the previous regression scheme and estimate the weighting value w_T according to the entropy information of the confidence (measurement) vector. In particular, we define the entropy of the confidence vector $\mathbf{p}_T$ as:

$$\pi_T = - \sum_i \frac{p_{Ti}}{\|\mathbf{p}_T\|_1} \log \frac{p_{Ti}}{\|\mathbf{p}_T\|_1}$$

Similarly, we can achieve the entropy value π_I for $\mathbf{p}_I$. It is not difficult to find that when the entropy value π_T is large, the difference of the measurement scores among different candidate names is small, which indicates that the corresponding “transductive” model is less confidence, so that we should set a small weighting value for the rank result of the “transductive” model. As a result, we set the weight w_T for the “transductive” model as:

$$w_T = 1 - \frac{\exp(\pi_T)}{\exp(\pi_T) + \exp(\pi_I)} = 1 - w_I$$

where w_I is the weight value for the rank result from the “inductive” model.

6.4 Experiments

6.4.1 Experiment Testbed

To evaluate the annotation performance, we adopt the *hit rate* at top- T annotated results as the performance metric, which measures the likelihood of having the true label among the top- T annotated names. Specifically, for $T = 1$, the *hit rate* is the same with the *accuracy*.

In our experiments, we adopt the “WDB”-series database, which is introduced in Chapter 3. It consists of four retrieval databases in different sizes and one query database with about 1,600

images. In order to train the proposed WL-LapSVM model in the “*inductive learning*” step, for each person/class in the retrieval database, we manually label one front-view facial image as the reference image. Notice that we do not make extra collection for the reference image, which aims to examine the generalization performance of the proposed Unifying Transductive and Inductive Learning (UTIL) scheme. Due to the manual labeling effort, in our experiments, we only label the retrieval database “WDB-040K” which contains 400 persons and more than 40,000 facial images.

Further, we discuss parameter settings. For the “*transductive learning*” step, we retrieve 40 most similar images for each query image from the retrieval database. For the “*inductive learning*” step, as a fair comparison, we adopt the **RBF** kernel for all the compared algorithms with $\sigma = 0.2$. For the other parameters, we randomly divide the query(test) database into two parts of equal size, and randomly collect one part for tuning the optimal parameters by a grid search scheme.

6.4.2 Evaluation on Positive Sample Size

In this experiment, we evaluate how the number of pseudo positive samples affects the performance of different algorithms. We compare the proposed WL-LapSVM algorithm with three baseline algorithms: the SVM algorithm using the only one positive sample, the LapSVM algorithm using the only one positive sample, and another SVM that employs p weakly labeled samples as positive samples, denoted as “WL-SVM” for short. The experimental result is presented in Figure 6.2.

We can draw several observations from the results. First, for both SVM and LapSVM, the annotation performances are rather poor by using only one positive reference image, and the hit rate of SVM and LapSVM at the top-1 position are only 7.6% and 10.3%, respectively. By introducing the manifold information, the LapSVM algorithm is slightly better than SVM. Second, by using the pseudo positive samples, the annotation performance of SVM can be

Figure 6.2: Annotation performance of inductive learning algorithms with varied numbers (p) of pseudo positive samples.

improved significantly. In particular, when the number of pseudo samples is increased from 10 to 150, the annotation performance will also be boosted from 23.0% to 62.0%. The additional pseudo positive training samples take credit for the performance improvement, which indicates it is important to employ extra samples in the “*inductive learning*” step. Third, the annotation performance can be further improved by adopting the proposed WL-LapSVM algorithm, which indicates that the proposed WL-LapSVM algorithm can effectively use the pseudo samples by assigning different pseudo positive samples with different confidence values. For example, by using only 20 pseudo positive samples, the proposed WL-LapSVM algorithm can achieve a better annotation performance than WL-SVM with 80 pseudo positive samples. This is very important in a large-scale problem, where the number of persons is huge and more training samples will take more storage space and computational costs.

6.4.3 Evaluation of Auto Face Annotation

In this experiment, we evaluate the annotation performance of the “transductive learning” step and the “*inductive learning*” step, respectively. For the “*transductive learning*” scheme, we adopt two algorithms, including a majority-voting based algorithm “SMW” and the state-of-

the-art “WLRCC” algorithm [100]. For the “*inductive learning*” scheme, we adopt the WL-SVM algorithm in the previous experiment and the proposed WL-LapSVM algorithm. For both WL-SVM and WL-LapSVM algorithms, we use $p = 150$ pseudo positive samples in our experiment. In Figure 6.3 and Table 6.2, both the mean and standard deviation of the annotation performance (*hit rate*) are reported with different T values, where T is the number of annotated names.

Figure 6.3: Comparison of face annotation performance by different algorithms, where T is the number of the annotated names.

Several observations can be drawn from the above experimental results. First of all, for the “*transductive learning*” step, the WLRCC algorithm significantly outperforms the simple baseline algorithm “SMW”, which is similar to the observations reported in [100]. Second, for the “*inductive learning*” step, the proposed WL-LapSVM algorithm achieves comparable results with the state-of-the-art WLRCC algorithm. In particular, the WL-LapSVM algorithm performs slightly worse than the WLRCC algorithm when only one name is annotated ($T = 1$), however, its performance is better for large T values. It indicates that the “*inductive learning*” algorithm WL-LapSVM has a better recall performance than the “*transductive learning*” algorithm WLRCC.

6.4.4 Evaluation on Different Combinations

In this experiment, we evaluate the annotation performance of the proposed Unifying Transductive and Inductive Learning (UTIL) scheme, by combining the two annotation models with different last-fusion algorithms, including the *measurement level* fusion (“MF-MinMax”, “MF-Zscore”) and the *rank level* fusion (“RF-Reg”, “RF-Entropy”). The average annotation performance are reported in Figure 6.4 and Table 6.2, respectively.

Table 6.2: Comparison of auto face annotation performance by different approaches.

T	SMW	WL-SVM	WLRLCC	WL-LapSVM	MF-MinMax	MF-Zscore	RF-Reg	RF-Entropy
01	0.6110	0.6190	0.7665	0.7624	0.7674	0.7838	0.8025	0.7988
	± 0.008	± 0.011	± 0.013	± 0.014	± 0.013	± 0.013	± 0.012	± 0.012
02	0.7168	0.7198	0.8076	0.8268	0.8188	0.8354	0.8491	0.8449
	± 0.009	± 0.010	± 0.012	± 0.012	± 0.012	± 0.011	± 0.008	± 0.008
03	0.7608	0.7739	0.8279	0.8473	0.8393	0.8561	0.8693	0.8634
	± 0.008	± 0.008	± 0.009	± 0.011	± 0.009	± 0.010	± 0.006	± 0.007
04	0.7855	0.8015	0.8433	0.8675	0.8563	0.8700	0.8805	0.8783
	± 0.009	± 0.008	± 0.009	± 0.011	± 0.008	± 0.007	± 0.007	± 0.005
05	0.7988	0.8223	0.8566	0.8808	0.8695	0.8824	0.8901	0.8850
	± 0.008	± 0.007	± 0.008	± 0.010	± 0.007	± 0.007	± 0.010	± 0.005

Several observations can be drawn from the above experimental results. First, by adopting proper fusion algorithm, the proposed UTIL scheme can significantly boost the annotation performance. In particular, the annotation results of WLRLCC and WL-LapSVM are 0.7665 and 0.7624. By using the last-fusion in UTIL, the performance can be boosted to 0.8025 by “RF-Reg” fusion and 0.7988 by “RF-Entropy” fusion. Second, in our experiments, the *rank level* fusion algorithms are more suitable for the proposed UTIL framework than the *measurement level* fusion algorithms. In particular, both “RF-Reg” and “RF-Entropy” consistently outperform the *measurement level* fusion (“MF-MinMax” and “MF-Zscore”). Furthermore, the “MLR-MinMax” fusion scheme even performs worse than the individual “WL-LapSVM” algorithm for large T values. Third, for the *rank level* fusion scheme, although “RF-Reg” is slightly better, it is a supervised scheme that needs extra labeled samples and training efforts.

The proposed entropy based fusion method “RF-Entropy” can achieve a very close combination result without extra efforts.

Figure 6.4: Comparison of different last-fusion algorithms in the UTIL framework

6.4.5 Evaluation on Parameter Sensitivity

For the proposed WL-LapSVM algorithm, the parameter λ_3 in Equation 6.5 is fixed as 1, and the parameters λ_1 and λ_2 are found by a grid search scheme. Figure 6.5 shows one the grid search result, where the ranges for λ_1, λ_2 are $\{0.01, 0.05, 0.1, 0.5, 1\}$. We notice that the performance of WL-LapSVM tends to be stable in the region $\lambda_1 \in [0.01, 0.2]$ and $\lambda_2 \in [0.08, 0.7]$. For this grid searching result, we choose $\lambda_1 = 0.1$ and $\lambda_2 = 0.5$ for the further experiments. In our experiments, we also found that generally the WL-LapSVM algorithm performs well with the parameters located in the previous range, which indicates that the WL-LapSVM algorithm is robust in terms of the parameter setting.

6.5 Summary

In this chapter, we propose a Unifying Transductive and Inductive Learning (UTIL) algorithm by combining the “*inductive learning*” algorithm with the Search-based Face Annotation (SBFA)

Figure 6.5: Grid search result of WL-LapSVM

framework in a last-classification fusion scheme. In particular, to address the small positive sample problem in the “*inductive learning*” scheme, we propose a Weakly Label Laplacian Support Vector Machines (WL-LapSVM) algorithm to train classifiers based on weakly labeled data. For the “*transductive learning*” scheme, we adopt the proposed WRLCC algorithm in Chapter 5. In the last-fusion stage, we propose an entropy-based *rank level* fusion algorithm, which performs as well as the supervised regression-based fusion algorithm without extra training efforts. Our empirical results show that the proposed UTIL scheme can significantly outperform both the transductive and inductive annotation approaches. In general, the last-fusion algorithm is limited for multi facial feature representations. In the next chapter, we aim to adopt the multimodel learning into the Search-based Face Annotation framework.

Chapter 7

Multimodel Learning for Learning to Name Faces

In this chapter, We propose a multimodel learning algorithm: “*Learning to Name Faces*” (L2NF) for search-based face annotation, which attempts to learn both the *optimal fusion weight vector* of the multimodel facial feature representation and the *refined labels* for enhancing the initial weak labels simultaneously. In particular, the key challenge for naming the query facial image is to effectively measure the similarity between the query and its nearest instances by combining diverse facial feature representations and their proper distance measurements. To tackle this problem, we propose a multimodal learning scheme that (i) first constructs multiple diverse facial features for representing the faces, (ii) further optimizes the distance measure on each feature space (modality) using distance metric learning, and (iii) finally learns the optimal fusion of the multiple representations by adapting the structural SVM algorithm.

7.1 L2NF — Learning to Name Faces

In this section, we present the proposed “Learning to Name Faces” (L2NF) framework for search-based face annotation in detail. To clarify the formulation, Table. 7.1 lists the key notations used in this chapter.

Table 7.1: List of Key Notations

Notation	Description
n	# retrieved similar instances for each query
m	# unique names in the whole retrieval database
$\{c_i\}_{i=1,\dots,m}$	m names set C in the whole retrieval database
N_t	# training samples
N_f	# query-neighbor similarity function $\phi_k(\cdot, \cdot)$
$\mathbf{q}_i$	i -th query sample
$\mathbf{y}_{\mathbf{q}_i}, \hat{\mathbf{y}}_{\mathbf{q}_i}$	correct label vector and predicted one w.r.t. $\mathbf{q}_i$
$(\mathbf{d}_{ij}, \mathbf{y}_{ij})$	top- n nearest samples and label vectors w.r.t. $\mathbf{q}_i$
$\mathbf{x}_{ij} = \Phi(\mathbf{q}_i, \mathbf{d}_{ij})$	query-neighbor similarity feature vector (QNSF)
$Y_i \in \mathbb{R}^{m \times n}$	label matrix, w.r.t. $\{\mathbf{y}_{ij}\}$
$X_i \in \mathbb{R}^{N_f \times n}$	query-neighbor similarity matrix w.r.t. $\{\mathbf{x}_{ij}\}$
$\pi_{\mathbf{q}_i} \in \Pi_i^*$	correct ranked name list set and one instance $\pi_{\mathbf{q}_i}$
$\hat{\pi}_{\mathbf{q}_i}$	predicted ranked name list w.r.t. $\mathbf{q}_i$
$\phi_k(\cdot, \cdot)$	k -th query-neighbor similarity function

7.1.1 Preliminaries

Let us denote by $Q = \{\mathbf{q}_i | i = 1, 2, \dots, N_t\}$ the set of query facial images, and assume there are a total of m names (classes) in the whole retrieval database, denoted by $C = \{c_1, c_2, \dots, c_m\}$. Each query facial image $\mathbf{q}_i \in Q$ is associated with one name (class label) $c_{\mathbf{q}_i} \in C$. Notice that we assume that there is only one name for each person. We denote by $\mathbf{y}_{\mathbf{q}_i} \in \{0, 1\}^m$ the label vector for the query instance $\mathbf{q}_i$, which contains only one non-zero item: $\|\mathbf{y}_{\mathbf{q}_i}\|_0 = 1$. If the query instance $\mathbf{q}_i$ is annotated with the k -th name ($c_{\mathbf{q}_i} = c_k$), then $\mathbf{y}_{\mathbf{q}_i}[k] = 1$. In the SBFA framework, the name label of the query face is predicted based on its nearest facial images. Assume the top- n retrieval results of the query image $\mathbf{q}_i$ are $\{(\mathbf{d}_{ij}, \mathbf{y}_{ij}) | j = 1, 2, \dots, n\}$, where $\mathbf{d}_{ij}$ is the j -th similar image in the retrieval result and $\mathbf{y}_{ij} \in \{0, 1\}^m$ is its corresponding label vector. We denote by $Y_i = [\mathbf{y}_{i1}, \mathbf{y}_{i2}, \dots, \mathbf{y}_{in}] \in \mathbb{R}^{m \times n}$ the label matrix for the i -th query $\mathbf{q}_i$.

For each query-neighbor pair $(\mathbf{q}_i, \mathbf{d}_{ij})$, we can create one query-neighbor similarity based feature vector:

$$\mathbf{x}_{ij} = \Phi(\mathbf{q}_i, \mathbf{d}_{ij}) = [\phi_k(\mathbf{q}_i, \mathbf{d}_{ij})]_{k=1}^{N_f}$$

where $\phi_k(\cdot, \cdot)$ represents the k -th query-neighbor similarity function and N_f is the number of the query-neighbor similarity functions. Typically, the query-neighbor similarity function is related to three factors: (1) the facial feature representation, (2) the distance metric, and (3) the mapping function between the distance value and the similarity value. For example, we can extract the Local binary patterns (LBP) as the facial feature, apply the L2-norm (Euclidean) distance as the distance metric, and the radial basis function with $\gamma = 0.1$ as the similarity-mapping function:

$$\exp\left(-\frac{1}{\gamma^2}\|\mathbf{q}_i^{(\text{lbp})} - \mathbf{d}_{ij}^{(\text{lbp})}\|^2\right).$$

To estimate the similarity more accurately by exploring more information, we can leverage multiple diverse query-neighbor similarity functions. More details about query-neighbor similarity function construction will be presented in Section 7.1.4. Based on the predefined query-neighbor similarity function and the achieved query-neighbor similarity based feature vector, for the i -th query instance $\mathbf{q}_i$, we denote its query-neighbor similarity matrix by $X_i = [\mathbf{x}_{ik}]$ with $k = 1, 2, \dots, n$ and $X_i \in \mathbb{R}^{N_f \times n}$.

7.1.2 Problem Overview

The basic idea of the SBFA paradigm is to exploit the weak labels of top-ranked similar facial images for naming the query face. The crux for the face naming task lies in how to effectively estimate the confidence values for the weak label vectors of the top-ranked similar instances. Given a query image $\mathbf{q}_i$ and its top- n retrieval results $\{(\mathbf{d}_{ij}, \mathbf{y}_{ij}) | j = 1, 2, \dots, n\}$, we denote by v_{ij} the confidence value for its j -th similar image $\mathbf{d}_{ij}$. Then, the estimated label vector of $\mathbf{q}_i$, denoted as $\hat{\mathbf{y}}_{\mathbf{q}_i}$, can be generated as:

$$\hat{\mathbf{y}}_{\mathbf{q}_i} = \sum_j \mathbf{y}_{ij} * v_{ij} = Y_i \mathbf{v}_i \quad (7.1)$$

where $\mathbf{v}_i = [v_{i1}, v_{i2}, \dots, v_{in}]^\top$. Obviously, the confidence value v_{ij} is related to both query $\mathbf{q}_i$ and the j -th similar instance $\mathbf{d}_{ij}$. In our problem, we assume it linearly depends on the query-neighbor similarity based feature vector $\mathbf{x}_{ij}$, that is, $v_{ij} = \mathbf{x}_{ij}^\top \mathbf{w}$. Hence, the confidence vector

$\mathbf{v}_i$ can be achieved as follows:

$$\mathbf{v}_i = X_i^\top \mathbf{w} \quad (7.2)$$

where $\mathbf{w} \in \mathbb{R}^{N_f}$ is a weight vector for multimodal fusion, which aims to combine different features of X generated by the N_f diverse similarity functions. In other words, each confidence value v_{ij} is a weighted linear combination of the corresponding query-neighbor similarity based feature vector $\mathbf{x}_{ij}$.

Remark. This aforementioned assumption is not difficult to understand as follows: each item in $\mathbf{x}_{ij}$ (e.g. the k -th item $\mathbf{x}_{ij}[k]$) is only related to the corresponding similarity function (e.g. $\phi_k(\cdot, \cdot)$). A large $\mathbf{x}_{ij}[k]$ indicates that the j -th retrieved instance is more similar to the query instance based on the k -th similarity function. Hence, it is more possible that the query instance has the same label vector with the j -th retrieved instance. As a result, the similarity value $\mathbf{x}_{ij}[k]$ is correlated with the confidence value v_{ij} . Typically, different similarity functions can perform very differently in practice, hence they should be combined appropriately by a proper weight vector $\mathbf{w}$.

By combining Eq. 7.1 and Eq. 7.2, the estimated label vector $\hat{\mathbf{y}}_{\mathbf{q}_i}$ for the query image $\mathbf{q}_i$ can be computed as follows:

$$\hat{\mathbf{y}}_{\mathbf{q}_i} = Y_i \mathbf{v}_i = Y_i X_i^\top \mathbf{w} \quad (7.3)$$

where Y_i and X_i vary for different query instances, and $\mathbf{w}$ is independent of the query instances. We show a visual example in Figure 7.1 to help understand this formula.

To generate the final annotation results (a sorted candidate name list), we can rank all the m names by sorting the predicted label vector $\hat{\mathbf{y}}_{\mathbf{q}_i}$ in descending order, as shown in Figure 7.2. We denote by $\hat{\pi}_{\mathbf{q}_i}$ the ranked name list, in which the item $\hat{\pi}_{\mathbf{q}_i}[j] \in C$ is the j -th annotated name. Given the correct name of the query instance $\mathbf{q}_i$ is $c_{\mathbf{q}_i}$, a good annotation system should ensure $c_{\mathbf{q}_i}$ appears at the top-ranked position or ideally at the first position. Hence, our problem aims to minimize the ranking position of the correct name $c_{\mathbf{q}_i}$, which can be formulated as follows:

$$\min_{\mathbf{w}} \sum_{i=1}^{N_t} \text{loss}(c_{\mathbf{q}_i}, \hat{\pi}_{\mathbf{q}_i}) \text{ where } \hat{\pi}_{\mathbf{q}_i} = \text{rank}(\hat{\mathbf{y}}_{\mathbf{q}_i}) \text{ and } \hat{\mathbf{y}}_{\mathbf{q}_i} = Y_i X_i^\top \mathbf{w}$$

Figure 7.1: Introduction of face naming in “*Search-based Face Annotation*”. A visual explanation of Eq. 7.3. For a query instance $\mathbf{q}_i$, its top- n similar samples are $\{(\mathbf{d}_{ij}, \mathbf{y}_{ij})\}_{j=1,2,\dots,n}$, and the predicted label vector is $\hat{\mathbf{y}}_{q_i}$. $\mathbf{y}_{ij}$ is the label vector of the j -th nearest sample $\mathbf{d}_{ij}$. $\mathbf{x}_{ij}$ is the feature vector between the query instance $\mathbf{q}_i$ and the similar example $\mathbf{d}_{ij}$. The k -th item of $\mathbf{x}_{ij}$ is constructed with the k -th query-neighbor similarity function $\phi_k(\mathbf{q}_i, \mathbf{d}_{ij})$. The inner product value $\mathbf{x}_{ij}^T \mathbf{w}$ is the confidence values v_{ij} for the j -th label vector $\mathbf{y}_{ij}$.

In general, the loss value should be zero if the correct name c_{q_i} is at the first position of $\hat{\pi}_{q_i}$, and the loss value of c_{q_i} at the top-ranked position should be smaller than the one of c_{q_i} at a lower-ranked position. The goal of the whole learning to name faces scheme is to minimize the loss values over all the query instances by addressing the following three key factors: (i) the noisy label matrix Y_i , (ii) the query-neighbor similarity matrix X_i , and (iii) the combination weight vector w , as shown in Figure 7.2.

Figure 7.2: Three factors that affect the annotation performance of SBFA and the corresponding improvement solution.

In particular, we attempt to address each of them respectively in the following approach:

- To address the noisy nature of web images, we propose to refine the initial weak label information Y_i by a graph-based refinement scheme for exploiting the “label smoothness” assumption;
- To address the variances of web facial images captured under various conditions (illumination, position, age, and gender, etc.), we can construct multiple diverse query-neighbor similarity functions and further improve the similarity measurements by employing distance metric learning techniques;

- To find the optimal multimodal fusion, we propose a supervised learning to rank scheme to optimize the weight vector $\mathbf{w}$ by applying the structural SVM algorithms on a set of training query samples (N_t query images and their corresponding top- n retrieval results).

7.1.3 Weak Label Refinement

In this section, we aim to refine the initial weak label matrix for each query independently. In particular, for a query $\mathbf{q}$ (the subscript of query index value is omitted), its top- n similar samples are $\{\mathbf{d}_1, \dots, \mathbf{d}_n\}$ and the corresponding noisy label matrix is denoted by $\tilde{Y}$. We enhance the initial label matrix $\tilde{Y}$ in a manifold learning scheme based on the key assumption of “*label smoothness*”, which means that the more similar the visual contents of two facial images are, the more likely they share the same labels [101].

In particular, for two images $\mathbf{d}_i$ and $\mathbf{d}_j$ in the top- n nearest samples, we can compute their similarity value vector based on the query-neighbor similarity function: $\Phi(\mathbf{d}_i, \mathbf{d}_j) \in \mathbb{R}^{N_f}$. By using the weight vector $\mathbf{w}$ learned in Section 7.1.5, we can get the similarity value between $\mathbf{d}_i$ and $\mathbf{d}_j$ as $S_{ij} = \mathbf{w}^\top \Phi(\mathbf{d}_i, \mathbf{d}_j)$. A large value of S_{ij} indicates that $\mathbf{d}_i$ is more similar to $\mathbf{d}_j$. Hence, a larger value of S_{ij} implies that the label vectors of $\mathbf{d}_i$ and $\mathbf{d}_j$ are more likely to be the same. Based on the above motivation, we can achieve the following formulation to enhance the initial weak label matrix $\tilde{Y}$:

$$\min_{Y \geq 0} \sum_{i,j} S_{ij} * \|Y_{:,i} - Y_{:,j}\|^2 + \beta \|(Y - \tilde{Y}) \circ M\|_F^2 \quad (7.4)$$

where $\circ$ denotes the Hadamard product of two matrices, and M is the mask matrix indicating the non-zero values in $\tilde{Y}$. In Eq. 7.4, the first term enforces the “*label smoothness*” assumption. Following the previous work [100], the second term is a regularization term that prevents the refined label matrix being deviated too much from the initial weak matrix $\tilde{Y}$. Notice that the label refinement problem in Eq. 7.4 depends on the weight vector $\mathbf{w}$ achieved by Eq. 7.8, while learning the weight vector $\mathbf{w}$ depends on the input data Y , as shown in Eq. 7.8. In our problem, we update the label matrices $Y_i, i = 1, \dots, n$ and the weight vector $\mathbf{w}$ iteratively.

7.1.4 Multimodal Representation Construction

In this section, we aim to construct the multimodal representation between the query instances and their corresponding top-ranked similar samples, which is based on the query-neighbor similarity function: $\Phi = \{\phi_k\}_{k=1,2,\dots,N_f}$. Generally, we can represent one facial image in different feature space, e.g. LBP feature, GIST feature, and Gabor feature. Suppose there are $\mathcal{K}$ kinds of features in total, we can represent by $(\mathbf{q}^{(k)}, \mathbf{d}_i^{(k)})$ the query-neighbor similarity based feature pair between the query image $\mathbf{q}$ and its i -th nearest sample $\mathbf{d}_i$ in the k -th feature space. Following the existing works on distance metric learning [111], we can define a distance metric $M^{(k)}$ in the k -th feature space, hence, the distance between $\mathbf{q}^{(k)}$ and $\mathbf{d}_i^{(k)}$ can be expressed by

$$d_{M^{(k)}}(\mathbf{q}^{(k)}, \mathbf{d}_i^{(k)}) = \sqrt{(\mathbf{q}^{(k)} - \mathbf{d}_i^{(k)})^\top M^{(k)} (\mathbf{q}^{(k)} - \mathbf{d}_i^{(k)})}$$

and the inner product between $\mathbf{q}^{(k)}$ and $\mathbf{d}_i^{(k)}$ can be expressed by

$$\langle \mathbf{q}^{(k)}, \mathbf{d}_i^{(k)} \rangle_{M^{(k)}} = (\mathbf{q}^{(k)})^\top M^{(k)} (\mathbf{d}_i^{(k)})$$

Based on the k -th feature space and distance matrix $M^{(k)}$, there are two ways to compute the similarity values between two instances: one way is using the heat kernel to transform the distance value into a similarity value which is widely used in semi-supervised learning [10]. In detail, the similarity value between $\mathbf{q}^{(k)}$ and $\mathbf{d}_i^{(k)}$ can be computed as follows:

$$\phi(\mathbf{q}, \mathbf{d}_i; k, \gamma) = \exp(-d_{M^{(k)}}^2(\mathbf{q}^{(k)}, \mathbf{d}_i^{(k)})/\gamma^2), \quad \gamma \in \Gamma$$

where the query-neighbor similarity function ϕ depends on the feature type k and the parameter γ . As a result, we can achieve $k * |\Gamma|$ query-neighbor similarity functions, where Γ is the set of all possible parameters γ during the experiments.

Another way to compute the similarity value is using the sparse representation technique which have been adopted to construct adjacency matrix in some recent works [94]. In detail, in the k -th feature space with $M^{(k)}$ as the distance metric, we can achieve the sparse representation

$\mathbf{s}^{(k)}$ for $\mathbf{q}^{(k)}$ based on the dictionary $D^{(k)} = [\mathbf{d}_1^{(k)}, \dots, \mathbf{d}_n^{(k)}]$ with the kernelized sparse coding algorithm [34], which can be formulated as follows:

$$\begin{aligned} \mathbf{s}_\lambda^{(k)} = \arg \min_{\mathbf{s}^{(k)} \geq 0} & \langle \mathbf{q}^{(k)}, \mathbf{q}^{(k)} \rangle_{M^{(k)}} + (\mathbf{s}^{(k)})^\top K_{DD}^{(k)} (\mathbf{s}^{(k)}) \\ & - 2(\mathbf{s}^{(k)})^\top K_{Dq}^{(k)} + \lambda \|\mathbf{s}^{(k)}\|_1 \end{aligned}$$

where $\mathbf{s}_\lambda^{(k)}$ is the achieved sparse representation with parameter λ , $K_{DD}^{(k)}$ is a $n \times n$ matrix with $\{K_{DD}^{(k)}\}_{ij} = \langle \mathbf{d}_i^{(k)}, \mathbf{d}_j^{(k)} \rangle_{M^{(k)}}$, and $K_{Dq}^{(k)}$ is a $n \times 1$ vector with $\{K_{Dq}^{(k)}\}_i = \langle \mathbf{q}^{(k)}, \mathbf{d}_i^{(k)} \rangle_{M^{(k)}}$. The i -th item in the sparse represent $\mathbf{s}_\lambda^{(k)}$ presents the representative ability of the i -th dictionary instance $\mathbf{d}_i^{(k)}$ for the encoding instance $\mathbf{q}^{(k)}$. Hence, the similarity value between $\mathbf{q}^{(k)}$ and $\mathbf{d}_i^{(k)}$ is computed as follows:

$$\phi(\mathbf{q}, \mathbf{d}_i; k, \lambda) = \mathbf{s}_\lambda^{(k)}[i], \quad \lambda \in \Lambda$$

where the query-neighbor similarity function ϕ depends on the feature type k and the parameter λ . As a result, we can achieve $k * |\Lambda|$ query-neighbor similarity functions, where Λ is the set of all possible parameters Λ during the experiments.

Finally, for each feature space, we have to choose a distance metric $M^{(k)}$. We can use the original feature space by setting $M^{(k)}$ with the identify matrix. To keep all the data points within the same classes close and separate all the data points from different classes far apart, it is better to adopt distance metric learning (DML) techniques to learn a better distance metric for each feature space respectively. Generally, any supervised DML algorithms can be used since the query-neighbor similarity function ϕ is independent of the DML algorithms. In our problem, we adopt ‘‘Metric Learning to Rank’’ (MLR) algorithm [66] that learns a metric such that rankings of data induced by the learned distance are optimized against a ranking loss measure (e.g. ROC area (AUC) or MAP). In this setting, the ‘‘relevant’’ results (in the same class) should lie close in space to the query, and ‘‘irrelevant’’ results should be pushed far away.

7.1.5 Optimal Fusion of Multiple Modalities

In this section, we aim to find the optimal weight vector $\mathbf{w}$ for optimizing the multimodal fusion in combining multiple types of facial feature representations. In particular, given a label matrix Y_i and a query-neighbor similarity matrix X_i , we can directly achieve the annotation result of $\mathbf{q}_i$ with the fusion vector $\mathbf{w}$ according to Eq. 7.3. Hence, finding the optimal fusion vector $\mathbf{w}$ that achieves the best ranked name list in Eq. 7.4 is equivalent to learning a multimodal annotation function with parameter $\mathbf{w}$ as follows:

$$f(\mathbf{w}) : \mathcal{Y} \times \mathcal{X} \rightarrow \Pi$$

based on a set of training samples $\{(\mathbf{q}_i, \mathbf{y}_{\mathbf{q}_i}, L_i, X_i)\}$ with $i = 1, 2, \dots, N_t$ by minimizing the annotation errors. The input space contains all the multiplication results between label matrix Y_i and query-neighbor similarity matrix X_i . The output space Π contains all the possible annotation results (the ranked name list). Obviously, the result of the function f is a structural output instead of a scalar value. Hence, it could be formulated as a structural SVM problem [50, 95], which has been solidly studied in several research works and has been used for ranking problems in [66, 109, 117]. To specialize a general structure SVM algorithm for a particular problem, we have to define two functions: the “*loss function*” Δ and the “*feature combination function*” Ψ .

7.1.5.1 Loss Function

The loss function is denoted as $\Delta(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}})$ in our problem, where $\boldsymbol{\pi}$ is the ground-truth ranked name list generated by the ground-truth label vector $\mathbf{y}$, while $\hat{\boldsymbol{\pi}}$ is the predicted ranked name list generated by the predicted label vector $\hat{\mathbf{y}}$. Notice that we omit the subscript for the query index for clarity. The “*hit rate*” at the top- T annotated names is used as the performance metric, which measures the likelihood of having the correct name among the top- T annotated names. In real world application, we prefer a high hit rate value with a small T value, that is, prefer the correct names to be at the top-ranked position in the ranked name list $\hat{\boldsymbol{\pi}}$.

For one query facial image, suppose it has t_1 correct names ($t_1 = 1$ in our problem since we assume there is only one name for each person), all the correct names are at the top positions of the ground-truth name list π , followed by all the incorrect names. For the predicted name list $\hat{\pi}$, if we only consider its top t_2 names, the loss function can be formulated as follows:

$$\Delta(\pi, \hat{\pi}) = 1 - \sum_{i=1}^{t_1} \sum_{j=1}^{t_2} h_1(\pi_i, \hat{\pi}_j) * \frac{1}{j} \quad (7.5)$$

where $h_1(\cdot, \cdot)$ is a judgment function that equals 1 if the i -th name π_i in π is the same with the j -th name $\hat{\pi}_j$ in $\hat{\pi}$, and 0 otherwise. As we can see, in Eq. 7.5 we set the loss value as 1 for all the annotation results in which the position of the correct name is larger than t_2 . For example, if $t_2 = 1$, we focus on only the first annotated name which means that if the first name in $\hat{\pi}$ is correct, then the loss value is 0, otherwise, the loss value is 1. If $t_2 = m$, the loss function in Eq. 7.5 becomes a special case of MAP loss.

7.1.5.2 Structural-based Feature Combination

Typically, the feature combination function aims to combine a set of feature vectors based on a ranking result. In our problem, the ranking result is denoted by π , which contains all the m names in the name set $C = \{c_1, \dots, c_m\}$. For a query facial image $\mathbf{q}$, its name vector is $\mathbf{y} \in \{0, 1\}^m$ with $\|\mathbf{y}\|_0 = 1$ since each facial image has only one correct name. Hence, $y_k = 1$ indicates that the k -th name c_k in C is the correct name for the query image $\mathbf{q}$. We denote by I_1 the index set of all the correct names, which contains only one item (k) according the previous case. Similarly, we denote by I_2 the index set of all the incorrect names, which contains $m - 1$ items $\{1, \dots, k - 1, k + 1, \dots, m\}$. Following the previous work [117], we define the feature combination function Ψ to combine the input label matrix Y and the input query-neighbor similarity matrix X based on a ranked name list π , as shown in Eq. 7.6:

$$\Psi(Y, X, \pi) = \frac{1}{|I_1| * |I_2|} \sum_{i \in I_1} \sum_{j \in I_2} h_2(c_i, c_j, \pi) [XY_{i:}^\top - XY_{j:}^\top] \quad (7.6)$$

where $Y_{k:}$ is the k -th row of the label matrix L , and $h_2(\cdot, \cdot, \cdot)$ is a ranking judgment function. If the name c_i is ranked before c_j in the ranked name list π , $h_2(c_i, c_j, \pi) = 1$; otherwise, $h_2(c_i, c_j, \pi) = -1$ if c_j is ranked before c_i .

Remark. For one group of input data $(Y, X, \mathbf{w})$, we can compute the label vector with Eq. 7.3: $\mathbf{y} = YX^\top \mathbf{w}$, which is subsequently used to generate the ranked name list π subsequently following the previous discussion. It is shown that, based on the feature combination function in Eq. 7.6, we can obtain the same ranked name list by solving the following problem [109]:

$$\tilde{\pi} = \arg \max_{\pi \in \Pi} F(\mathbf{w}, Y, X, \pi) = \mathbf{w}^\top \Psi(Y, X, \pi) \quad (7.7)$$

where $F(\mathbf{w}, Y, X, \pi)$ is the discriminant function. It indicates that we can learn the weight vector $\mathbf{w}$ by maximizing the discriminant function $F(\mathbf{w}, L, X, \pi)$ over the a set of correct ranked label lists, and predict the new label vector of the unseen query with Eq. 7.3.

Using the *loss function* in Eq. 7.5 and the *feature combination function* in Eq. 7.6, we can achieve the objective function to learn the weight vector $\mathbf{w}$ based on the structural SVM, which is shown as follows:

$$\begin{aligned} \min_{\mathbf{w}, \Xi=[\xi_1, \dots, \xi_{N_t}]} & \frac{1}{2} \mathbf{w}^\top \mathbf{w} + \frac{C}{N_t} \sum_{i=1}^{N_t} \xi_i \\ \text{s.t. } & \forall i, \xi_i \geq 0 \quad \text{and} \quad \forall i, \forall \pi_{\mathbf{q}_i} \in \Pi_i^*, \forall \pi \in \Pi \setminus \Pi_i^* : \\ & \mathbf{w}^\top \Psi(Y_i, X_i, \pi_{\mathbf{q}_i}) - \mathbf{w}^\top \Psi(Y_i, X_i, \pi) \geq \Delta(\pi_{\mathbf{q}_i}, \pi) - \xi_i \end{aligned} \quad (7.8)$$

In the above formulation, the objective function of Eq. 7.8 is similar to that of the general SVM algorithm, where C is a regularization parameter to tradeoff between the training error and the model complexity. For the constraints, if the value of discriminant function F in Eq. 7.7 for an incorrect ranking $\pi \in \Pi \setminus \Pi_i^*$ is greater than that for one true ranking $\pi_{\mathbf{q}_i} \in \Pi_i^*$, the slack variable ξ_i must be at least $\Delta(\pi_{\mathbf{q}_i}, \pi)$, which indicates the sum of slacks $\sum_i \xi_i$ upper bounds the empirical risk for the training samples based on the loss function defined in Eq. 7.5. Since the number of constraints in Eq. 7.8 is extremely large, we adopt the cutting plane algorithm [50,

Algorithm 5: Cutting plane algorithm for Eq. 7.8

Input: $(\mathbf{q}_i, \mathbf{y}_{\mathbf{q}_i}, X_i, Y_i), i = 1, 2, \dots, N_t, C, \epsilon$
Output: weight vector $\mathbf{w}$

```

1  $\mathcal{W}_i \leftarrow \emptyset$ , for all  $i = 1, \dots, N_t$ 
2 repeat
3   for  $i = 1$  to  $N_t$  do
4      $E(\boldsymbol{\pi}; \mathbf{w}) \equiv \Delta(\boldsymbol{\pi}_{\mathbf{q}_i}, \boldsymbol{\pi}) + \mathbf{w}^\top \Psi(Y_i, X_i, \boldsymbol{\pi})$ 
5      $\tilde{\boldsymbol{\pi}} = \arg \max_{\boldsymbol{\pi} \in \Pi} E(\boldsymbol{\pi}; \mathbf{w})$ 
6     if  $E(\tilde{\boldsymbol{\pi}}, \mathbf{w}) > \xi_i + \epsilon$  then
7        $\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{\tilde{\boldsymbol{\pi}}\}$ 
8       Get  $(\mathbf{w}, \Xi)$  by solve Eq. 7.8 over  $\mathcal{W} = \bigcup_i \mathcal{W}_i$ 
9 until no  $\mathcal{W}_i$  has changed during iteration;
```

117] to efficiently solve the optimization in Eq. 7.8, as shown in Algorithm 5. More details about the cutting plane algorithm can be found in [50].

7.1.6 Algorithm for Learning to Name Faces

In the above, we separately discuss the three key factors that affect the final annotation result of the proposed SBFA framework, including the label matrix Y , the query-neighbor similarity matrix X and the weight vector $\mathbf{w}$, which collectively determine the annotation result as $\mathbf{y} = YX^\top \mathbf{w}$. In this section, we will present the overall training for unifying all these three factors in a unified learning framework, and how to apply the models learned by L2NF for on-the-fly face annotation of a novel query facial image.

First of all, Algorithm 6 shows the overall algorithmic framework for training the models by L2NF. At the beginning, we attempt to optimize each of the distance metrics for each facial feature space using the distance metric learning technique as discussed in Section 7.1.4. After obtaining the set of optimal distance metrics $M^{(k)}$, we can then construct the set of query-neighbor similarity functions Φ based on the set of multiple diverse facial feature representations and their distance measures. Using the query-neighbor similarity functions Φ , we can generate the query-neighbor feature matrices X_i for each query q_i in the training query

Algorithm 6: L2NF—Algorithm for training the models

Input: Training set $(\mathbf{q}_i, \mathbf{y}_{\mathbf{q}_i}, \mathbf{d}_{ij}, \mathbf{y}_{ij})$ in $\mathcal{K}$ feature spaces with $i = 1, \dots, N_t$ and $j = 1, \dots, n$, name sets C with m names, parameters β, Λ and Γ

Output: weight vector $\mathbf{w}$ and similarity function set Φ

- 1 **for** $k = 1$ **to** $\mathcal{K}$ **do**
- 2 Learn the optimal distance metric $M^{(k)}$ in Section 7.1.4
- 3 Build query-neighbor similarity functions Φ with varied combinations of $\lambda \in \Lambda, \gamma \in \Gamma$, and $M^{(k)}, k = 1, 2, \dots, \mathcal{K}$
- 4 Construct query-neighbor feature matrix $X_i, i = 1, \dots, N_t$
- 5 **repeat**
- 6 Get the weight vector $\mathbf{w}$ by solving Eq. 7.8
- 7 **for** $i = 1$ **to** N_t **do**
- 8 Refine the label matrix Y_i by solving Eq. 7.4
- 9 **until** *CONVERGENCE*;

set. Finally, we optimize both the optimal weight vector $\mathbf{w}$ and the refined label matrices Y by an iterative scheme. At the end of the whole training scheme, we obtain the final model that consists of the set of query-neighbor similarity functions Φ and the optimal weight vector $\mathbf{w}$ for multimodal fusion.

Algorithm 7: Algorithm of on-the-fly face annotation by L2NF

Input: Novel query $\mathbf{q}$ in $\mathcal{K}$ feature spaces, query-neighbor similarity function Φ , optimal weight vector $\mathbf{w}^*$

Output: Annotation result π

- 1 Retrieval the top- n similar images the $\{(\mathbf{d}_i, \mathbf{y}_i)\}_{i=1,2,\dots,n}$
- 2 Construct the query-neighbor similarity matrix X based on the query-neighbor similarity function Φ
- 3 Obtain Y by refining the initial label matrix with Eq. 7.4
- 4 $\mathbf{y} = YX^\top \mathbf{w}^*$
- 5 Get the ranked names list π by sorting $\mathbf{y}$ in descending order

The above training framework as shown in Algorithm 6 can be done in off-line learning manner. After completing the training, we can apply the model for online face annotation for naming a novel query facial image on-the-fly. Algorithm 7 summarizes the proposed algorithm for on-the-fly annotation of an unseen query facial image. Specifically, given a new query facial

image, we first conduct a similar face retrieval step to find a short list of most similar faces. After that, we construct the query-neighbor similarity matrix X based on the query-neighbor similarity function Φ . We then further refine the initial label matrix Y of the current query using Eq. 7.4. Finally, we compute the label vector by Eq. 7.3 and obtain the final annotation result π by sorting the label vector y in descending order.

7.2 Experimental Results

7.2.1 Experimental Testbed

We compare the proposed “L2NF” framework with several existing works that are proposed for web-scale face annotation or general image annotation, including “WRLCC” [100], “S-GSSL” [94], “MtBGS” [107], “MRR” [108], and a simple baseline algorithm that adopts the weighted majority voting “WMV”. We also extend the WRLCC algorithm and the WMV algorithm into a multimodal scheme, by equally combining the face naming results from different facial feature spaces, denoted as “WRLCC_{mm}” and “WMV_{mm}”.

To evaluate the annotation performance, we adopt the *hit rate* at top- T annotated results as the performance metric, which measures the likelihood of having the true label among the top- T annotated names. Specifically, for $T = 1$, the *hit rate* is the same with the *accuracy*.

In our experiments, we adopt the “WDB”-series database, which is introduced in Chapter 3. It consists of four retrieval databases in different sizes and one query database with about 1,600 images. In particular, we adopt the “WDB-040K” and “WDB-600K” databases in the experiments of this chapter. To construct the query-neighbor similarity functions, we adopt three kinds of features as the facial descriptors: the LBP feature [3, 4], the GIST feature [89, 100], and the Gabor feature [59]. In particular, the 2891-dimensional LBP feature is extracted by dividing the face images into 7×7 blocks, which is further projected into a lower 500-dimensional feature space using Principal Component Analysis (PCA). Both GIST features and Gabor features are extracted over the whole aligned facial image.

Further, we discuss parameter settings. The parameter set for heat kernel is $\Gamma = \{-1, 0, 1, 2, 3, 4\}$, while the parameter set for sparse representation is $\Lambda = \{0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$. Moreover, the parameters for the structural SVM and distance metric learning are tuned according to the training sets. To evaluate the annotation performance, we adopt the *hit rate* at top- T annotated results as the evaluation metric, which measures the likelihood of having the true label among the top- T annotated names for a query facial image.

7.2.2 Experiments on “WDB-040K” and “WDB-600K”

This experiment aims to evaluate the face naming performance of the proposed “L2NF” framework on the database WDB-040K by comparing with the aforementioned seven existing algorithms. For the facial image retrieval task in L2NF, we adopt the JEC algorithm to combine distances from different face descriptors [65], which allows each individual distance to contribute equally. The same retrieval scheme is used to find the similar nearest samples for the multimodal extensions: “WLRCC_{mm}” and “WMV_{mm}”. For the single model solution, we use the GIST feature as the facial descriptor, which is similar to the experiment setting in [100]. For the 1,600 query facial images, we randomly select half of them to learn the distance metrics of different facial features and the multimodal representation combination w . Such a procedure is repeated 10 times and the average performance is computed over the 10 trials, as shown in Table.7.2.

Several observations can be drawn from the results. First, it is clear that the proposed L2NF algorithm significantly outperforms the other algorithms. First, the WLRCC algorithms achieves the best performance among the single model solution by using only one type of facial feature (GIST). For example, the simple baseline WMV is about 60.9% with $T = 1$, which is boosted to 76.7% by WLRCC. Second, if multiple facial features are available, the performance of the multimodal WLRCC_{mm} increases to 80.9%, and 65.6% for the multimodal WMV_{mm}. It indicates that using multiple facial representations is helpful for the face

Table 7.2: Face Naming Performance on Database WDB-040K.

	T01	T02	T03	T04	T05
WMV	0.6090 ± 0.012	0.7150 ± 0.009	0.7599 ± 0.009	0.7848 ± 0.008	0.7969 ± 0.008
SGSSL	0.7310 ± 0.011	0.7770 ± 0.010	0.8079 ± 0.011	0.8231 ± 0.013	0.8338 ± 0.012
MtBGS	0.7023 ± 0.015	0.7654 ± 0.012	0.7896 ± 0.011	0.8058 ± 0.009	0.8215 ± 0.010
MRR	0.6640 ± 0.010	0.7560 ± 0.007	0.7875 ± 0.008	0.8005 ± 0.008	0.8155 ± 0.009
WRLCC	0.7671 ± 0.010	0.8009 ± 0.010	0.8263 ± 0.010	0.8361 ± 0.009	0.8496 ± 0.007
WMV _{mm}	0.6560 ± 0.014	0.7491 ± 0.013	0.8010 ± 0.012	0.8144 ± 0.013	0.8244 ± 0.014
WRLCC _{mm}	0.8088 ± 0.011	0.8568 ± 0.014	0.8714 ± 0.015	0.8799 ± 0.012	0.8859 ± 0.011
L2NF	0.8663 ± 0.011	0.8918 ± 0.011	0.8983 ± 0.010	0.9025 ± 0.010	0.9054 ± 0.009

naming task which again validates the importance of this study. More specifically, the improvements of these two algorithms (WRLCC_{mm} and WMV_{mm}) in the multimodal scheme are mainly due to two aspects: (i) the retrieval result becomes better when combining multiple features and distance measures is combined by JEC [65]. For example, for the WMV algorithm, if we use the multiple features for retrieval but only GIST feature for annotation, its performance is 64.2%, which is higher than the one that use only GIST feature for both retrieval and annotation (60.9%). (ii) the combination step enlarges the possibility that the correct name is chosen. Both of the two aspects are also beneficial for the L2NF framework. Last but not least, the proposed L2NF framework can further improve the face naming performance to 86.6%, which indicates the constructed multimodal representations are more discriminative and the learned combination can efficiently combine various query-neighbor similarity function in different facial feature space. The performance improvement is mainly gained from three aspects as discussed in Section 7.1.2: the refined label matrix, the constructed multimodal represen-

tation based on distance metric learning techniques, and the learned optimal combination of various query-neighbor similarity functions. More details are discussed in Section 7.2.4.

Further, we evaluate the face naming performance of the proposed L2NF framework on a larger facial image databases: WDB-600K. For clarity, we mainly focus on the evaluation of the algorithms using multimodal representations. The experimental results are shown in Figure 7.3.

Figure 7.3: Face naming performance (hit rate @ top-T) on the two databases: WDB-600K.

We can make several observations from the results. First of all, similar to the previous observations, the proposed L2NF framework consistently achieves the best annotation performance. Secondly, the performance on WDB-600K is lower than the one on WDB-400K which is consistent with the previous observation in [100], since increasing the persons number leads to a larger database of more images for the same query, which makes the retrieval task more challenging.

7.2.3 Evaluation on Training Query Sizes

This experiment aims to evaluate the impact of the training query sizes in the L2NF framework based on the WDB-040K database. In the previous experiments, we adopt half of the 1,600

query images as the training set. In this experiment, we evaluate the annotation performance under varied number of queries. Specifically, instead of using all the training samples (totally 800), we build three small training sets by randomly collecting 200, 400 and 600 query images, respectively. The experimental results of hit-rate @ top-1 performance are shown in Figure 7.4. From the results, it is obvious that the face naming performance increases when more training samples are available, and the final performance tends to become saturated when the training query size is above 600. Finally, even with a small number of queries for training, e.g., only 200 training samples, the L2NF algorithm can achieve a good performance (0.8338), which remains much better than the state-of-the-art “WRLCC_{mm}” scheme.

Figure 7.4: Face Naming Performance (hit rate top- $T = 1$) of L2NF with varied sizes of training queries.

7.2.4 Analysis of the Performance Gains

This experiment aims to analyze how different factors affect the face naming performance by the proposed L2NF scheme as shown in Figure 7.2. In particular, there are key factors: the refined label matrix, the constructed multiple representations, and the optimal weight vector for multimodal fusion.

Table 7.3: Evaluation and analysis of the performance gains.

	$\text{L2NF}_{M=I}^{\mathbf{w}=1}$	$\text{L2NF}_{M=I}^{\mathbf{w}=\mathbf{w}^*}$	$\text{L2NF}_{M=M^*}^{\mathbf{w}=1}$	$\text{L2NF}_{M=M^*}^{\mathbf{w}=\mathbf{w}^*}$
Hit Rate	0.7941	0.8120	0.8403	0.8663

First of all, to examine the efficacy of the refined label matrix Y , we compare it with the initial raw label matrix $\tilde{Y}$ using the simplest baseline algorithm WMV by excluding other factors in affecting annotation performance. Our result indicates that the refined label matrix can boost the performance from 60.9% (without refinement) to 62.0% (after refinement). Further, we examine the efficacy of another two factors as shown in Table 7.3. We denote by M^* the learned distance metric and $\mathbf{w}^*$ the optimal multimodal fusion vector. When the weight vector $\mathbf{w}$ is fixed to 1 and the distance metrics are based on Euclidean distance (M is set to an identity matrix), the hit-rate @ top-1 performance of the resulting L2NF algorithm (denoted as $\text{L2NF}_{M=I}^{\mathbf{w}=1}$) is 79.4%. This value can be boosted to 84.3% if we adopt the optimized metric M^* , and further boosted to 86.6% if we further use the optimal weight vector for multimodal fusion. As a conclusion, the proposed L2NF framework is able to leverage all the three factors for achieving the state-the-art performance in a systematic and synergic scheme.

7.3 Summary

In this chapter, we aim to further improve the annotation performance of search-based face annotation problem by adopting multimodel learning techniques. We propose the “Learning to Name Faces” (L2NF) algorithm by exploring multimodal learning scheme on weakly labeled facial image data. We conduct a set of extensive empirical studies on two benchmark real-world facial image databases, in which encouraging results show that the proposed L2NF algorithm significantly boosts the annotation performance of the search-based face annotation task and achieve the state-of-the-art annotation performance.

Chapter 8

Conclusions and Future Work

8.1 Summary

This thesis focuses on solving the auto face annotation problem in the search-based face annotation scheme and further improve the annotation performance by adopting various state-of-the-art machine learning techniques. In particular, we proposed a family of effective and efficient algorithms (ULR, WLRCC, UTIL, and L2NF) to tackle the three challenges in Chapter 1. In addition, we develop a real-world face annotation system “FANS–Face ANnotation by Searching Large-scale Web Facial Images”,¹ which was built upon a large-scale real-world web facial image database with a total of 6,025 persons and about 1 million facial images. An android APP is developed to easy the usage of our system.

The main contributions of this thesis can be further described as follows: first, to tackle the first challenge, we propose an effective *Unsupervised Label Refinement* (ULR) approach for refining the labels of web facial images by adopting unsupervised machine learning techniques. In particular, we formulate the learning problem as a convex optimization and develop effective optimization algorithms to solve the large-scale learning task efficiently. To further speed up the proposed scheme, we also propose a clustering-based approximation algorithm which can improve the scalability considerably. An extensive set of empirical studies are conducted on a large-scale web facial image testbed, in which encouraging results showed that

¹<http://msm.cais.ntu.edu.sg/fans>

the proposed ULR algorithms can significantly refine the weak label information and boost the face annotation performance.

Second, to tackle the second challenge, we propose an effective *Weak Label Regularized Local Coordinate Coding* (WLRLCC) technique, which exploits the principle of local coordinate coding by learning sparse features, and employs the idea of graph-based weak label regularization to enhance the weak labels of the similar facial images. Moreover, an effective sparse reconstruction scheme is developed to perform the face annotation task. To further reduce the off-line running time, we also propose an offline approximation scheme (AWLRLCC), which generally maintains comparable results but significantly reduces the annotation time.

Third, to tackle the third challenge with both transductive and inductive learning techniques, we proposed the Unifying Transductive and Inductive Learning (UTIL) scheme in a last-classification fusion scheme. In the “inductive learning” view, to tackle the small sample problems, we propose a *Weak Label Laplacian Support Vector Machine* (WL-LapSVM) algorithm for generating effective classification models from weakly labeled web facial images. In the last-fusion stage, we propose an entropy-based combination scheme to combine the annotation results from the two different learning schemes to maximize the learning efficacy.

Fourth, to tackle the third challenge and go beyond the limitation of single facial feature representation, we proposed a multimodel based Learn to Name Faces (L2NF) algorithm, which optimize the fusion of multiple annotation modalities in a learning to rank framework and significantly boost the annotation performance. In particular, the proposed L2NF algorithm attempts to learn both the optimal *weight vector* for combining different query-neighbor similarity functions and the *refined labels* for enhancing the initial weak labels simultaneously in a unified learning scheme.

Finally, we develop a real-world face annotation system based on a large-scale web facial image database with a total of 6,025 western celebrities and about 1 million facial images. An android APP is also developed to easy the usage of our system in the mobile operation

system. This system demonstrates the potential of searching and mining web-scale weakly labeled facial images on the internet to tackle the challenging face annotation problem. The live demo of FANS is available online.

In a summary, this thesis aims to tackle the automatic face annotation problem in the Search-based Face Annotation (SBFA) framework. Three key challenges have been tackled by a series of proposed algorithms and the state-of-the-art face annotation performance is achieved based on several real-world large-scale web facial image databases.

8.2 Future Work

Although a substantial number of promising achievements have been reported for the Search-based Face Annotation algorithm, there are still numerous open issues that need to be further explored in future work.

An important issue is to develop efficient facial image retrieval techniques. In the search-based face annotation framework, the retrieval result directly affects the final annotation results. In particular, suppose we could achieve a perfect facial image retrieval result, the basic majority voting algorithm could also achieve a good annotation result. This challenge may be solved in a variety of ways by adopting different computer vision and machine learning techniques. For example, we can build a better hashing map function in a supervised way.

Our framework is also limited in several aspects, which also give the direction for further works. We assume each name corresponds to a unique single person, which makes our problem more clearly. However, this is not always true for real-life scenarios. For example, it is possible that two persons have the same name or one person may have multiple names. Such kind of practical duplicate name issues may be partially solved by extending our algorithms, e.g., via learning the similarity between any two names both in the name space and visual space.

We assume the top retrieved web facial images are related to the query name. This is clearly true for celebrities who have many photos on the internet. However, when the query

facial image is not a well-known person, there may not exist many relevant facial images on the WWW. This is a common limitation of all existing data-driven annotation techniques.

Publications

Related Publications

- **Dayong Wang**, Steven C.H. Hoi, Ying He, Jianke Zhu, Tao Mei, Jiebo Luo. Retrieval-based Face Annotation by Weak Label Regularized Local Coordinate Coding. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol.36, no.3, p-p.550,563, March 2014.
- **Dayong Wang**, Steven C.H. Hoi, Ying He, Jianke Zhu. Mining Weakly Labeled Web Facial Images for Search-based Face Annotation. In *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol.26, no.1, pp.166,179, Jan. 2014
- **Dayong Wang**, Steven C.H. Hoi, Ying He, C. Miao. Learning to Name Faces: A Multimodal Learning Scheme for Search-Based Face Annotation. In *Proceedings of ACM International Conference on Information Retrieval (SIGIR'13)*. ACM, Dublin, Ireland, 2013.
- S.C.H. Hoi, **Dayong Wang**, I.Y. Cheng, E. Lin, J. Zhu, Y. He, and C. Miao FANS: Face ANnotation by Searching large-scale web facial images. In *Proceedings of the 22nd International World Wide Web Conference (WWW'13) Demos Track*.
- **Dayong Wang**, Steven C.H. Hoi, Ying He. A Unified Learning Framework for Auto Face Annotation by Mining Web Facial Images. In *Proceedings of ACM Conference on Information and Knowledge Management (CIKM'12)*. ACM, Maui Hawaii, USA, 2012.

- **Dayong Wang**, Steven C.H. Hoi, Ying He, Jianke Zhu. Retrieval-based Face Annotation by Weak Label Regularized Local Coordinate Coding. In *Proceedings of ACM International Conference on Multimedia (MM'11)*. ACM, Scottsdale, Arizona, USA, 2012.
- **Dayong Wang**, Steven C.H. Hoi, Ying He. Mining Weakly Labeled Web Facial Images for Search-based Face Annotation. In *Proceedings of ACM International Conference on Information Retrieval (SIGIR'11)*. ACM, Beijing, China, 2011.

Other Publications

- B.S. Chew, L.P. Chau, Y. He, **Dayong Wang**, C. H. Hoi. Spectral Geometry Image: Image based 3D models for digital broadcasting applications. In *IEEE Transactions on Broadcasting*, Volume 57, No. 3, pp.636-635, September 2011.
- **Dayong Wang**, Steven C.H. Hoi, Ying He. An Effective Approach to Pose Invariant 3D Face Recognition. In *Proceedings of International Conference on Multimedia Modeling (MMM'11)*. Taipei, Taiwan, 2011.
- Ying He, B.S. Chew, **Dayong Wang**, Steven C.H. Hoi, L.P. Chau. Streaming 3D Meshes Using Spectral Geometry Images. In *Proceedings of ACM International Conference on Multimedia (MM'09)*. ACM, Beijing, China, 2009.

References

- [1] Michele Nappi A.F. Abatea. 2d and 3d face recognition a survey. *Pattern Recognition Letters*, 28(14):1885–1906, October 2007.
- [2] M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Trans. on [see also Acoustics, Speech, and Signal Processing, IEEE Trans. on]*, 54(11):4311–4322, 2006.
- [3] T. Ahonen, A. Hadid, and M. Pietikainen. Face recognition with local binary patterns. In *ECCV'04*, volume 1, pages 469–481, 2004.
- [4] Timo Ahonen, Abdenour Hadid, and Matti Pietikäinen. Face description with local binary patterns: Application to face recognition. *IEEE TPMAI*, 28(12):2037–2041, 2006.
- [5] Emre Akbas and Fatos T. Yarman-vural. Automatic image annotation by ensemble of visual descriptors. In *Computer Vision and Pattern Recognition*, 2007.
- [6] Arun Ross Anil Jain, Karthik Nandakumar. Score normalization in multimodal biometric systems. *Pattern Recognition*, 38(12):2270 – 2285, 2005.
- [7] Francis Bach, Julien Mairal, Jean Ponce, and Guillermo Sapiro. Tutorial on sparse coding and dictionary learning for image analysis. Technical report, CVPR, 2010.
- [8] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIIMS*, 2(1):183–202, 2009.
- [9] Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE TPMAI*, 19(7):711–720, 1997.

REFERENCES

- [10] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.
- [11] T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, Erik G. Learned-Miller, and D. A. Forsyth. Names and faces in the news. In *IEEE CVPR’04*, 2004.
- [12] T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, Erik G. Learned-Miller, and D. A. Forsyth. Names and faces. Technical report, Technical Report, University of California, Berkeley, 2007.
- [13] Tamara L. Berg, Alexander C. Berg, Jaety Edwards, and David Forsyth. Who’s in the picture. In *NIPS’05*, Cambridge, MA, 2005. MIT Press.
- [14] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory, COLT’ 98*, pages 92–100, New York, NY, USA, 1998. ACM.
- [15] Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, editors. *Advanced Lectures on Machine Learning, ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*, volume 3176 of *Lecture Notes in Computer Science*. Springer, 2004.
- [16] Kevin W. Bowyer, Kyong Chang, and Patrick Flynn. A survey of approaches and challenges in 3d and multi-modal 3d+2d face recognition. *Comput. Vis. Image Underst.*, 101(1):1–15, 2006.
- [17] Kevin W. Bowyer, Kyong Chang, and Patrick Flynn. A survey of approaches and challenges in 3d and multi-modal 3d+2d face recognition. *Computer Vision and Image Understanding*, 101, 2006.
- [18] K.W. Bowyer, K. Chang, and P. Flynn. A survey of approaches to three-dimensional face recognition. In *Pattern Recognition*, volume 1, pages 358 – 361 Vol.1, 23-26 2004.
- [19] Deng Cai, Xiaofei He, Jiawei Han, and HongJiang Zhang. Orthogonal laplacianfaces for face recognition. *IEEE Trans. on Image Processing*, 15(11):3608–3614, 2006.

REFERENCES

- [20] Zhimin Cao, Qi Yin, Xiaoou Tang, and Jian Sun. Face recognition with learning-based descriptor. In *IEEE CVPR'10*, 2010.
- [21] G Carneiro, A B Chan, P Moreno, and N Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *IEEE TPMAI*, pages 394–410, 2006.
- [22] Zhengyou Zhang Cha Zhang. A survey of recent advances in face detection. Technical report, Microsoft, 2010.
- [23] Jae Young Choi, Seungji Yang, Yong Man Ro, and Konstantinos N. Plataniotis. Face annotation for personal photos using context-assisted face recognition. In *Proceeding of the 1st ACM international conference on Multimedia information retrieval, MIR '08*, pages 44–51, New York, NY, USA, 2008. ACM.
- [24] Jingyu Cui, Fang Wen, Rong Xiao, Yuandong Tian, and Xiaoou Tang. Easyalbum: an interactive photo annotation system based on face clustering and re-ranking. In *ACM SIGCHI'07*, pages 367–376, 2007.
- [25] Kresimir Delac and Mislav Grgic. *Face Recognition*. IN-TECH, 2007.
- [26] T. Dharani and I.L. Aroquiaraj. A survey on content based image retrieval. In *Pattern Recognition, Informatics and Medical Engineering (PRIME), 2013 International Conference on*, pages 485–490, 2013.
- [27] Wei Dong, Zhe Wang, William Josephson, Moses Charikar, and Kai Li. Modeling lsh for performance tuning. In *CIKM'08*, pages 669–678, 2008.
- [28] R. P W Duin. The combining classifier: to train or not to train? In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 2, pages 765–770 vol.2, 2002.
- [29] P. Duygulu, K. Barnard, J. de Freitas, and D. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV'02*, 2002.
- [30] Ehsan Elhamifar and René Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *CoRR*, abs/1203.1005, 2012.

REFERENCES

- [31] Jianping Fan, Yuli Gao, and Hangzai Luo. Multi-level annotation of natural scenes using dominant image components and semantic concepts. In *ACM SIGMM'04*, pages 540–547, 2004.
- [32] Jerome Friedman, Trevor Hastie, and Robert Tibshiraniz. A note on the group lasso and a sparse group lasso. Technical report, Statistics, Stanford University, 2010.
- [33] Shenghua Gao, Ivor W. Tsang, Liang-Tien Chia, and Peilin Zhao. Local features are not lonely - laplacian sparse coding for image classification. In *IEEE CVPR'10*, 2010.
- [34] Shenghua Gao, Ivor Wai-Hung Tsang, and Liang-Tien Chia. Kernel sparse representation for image classification and face recognition. In *ECCV'10*, ECCV'10, pages 1–14, Berlin, Heidelberg, 2010. Springer-Verlag.
- [35] Yongsheng Gao and M.K.H. Leung. Face recognition using line edge map. *IEEE TPMAI*, 24(6):764–779, jun 2002.
- [36] Matthieu Guillaumin, Thomas Mensink, Jakob Verbeek, and Cordelia Schmid. Automatic face naming with caption-based supervision. In *IEEE CVPR'08*, pages 1–8, 2008.
- [37] Matthieu Guillaumin, Thomas Mensink, Jakob Verbeek, and Cordelia Schmid. Face recognition from caption-based supervision. In *IJCV'11*. Springer, 2011.
- [38] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Is that you? metric learning approaches for face identification. In *ICCV'09*, Kyoto, Japan, September 2009.
- [39] Zhen Guo, Zhongfei (Mark) Zhang, Eric P. Xing, and Christos Faloutsos. Semi-supervised learning based on semiparametric regularization. In *SDM*, pages 132–142. SIAM, 2008.
- [40] Allan Hanbury. A survey of methods for image annotation. *J. Vis. Lang. Comput.*, 19:617–627, October 2008.
- [41] Xiaofei He, Shuicheng Yan, Yuxiao Hu, and Partha Niyogi. Face recognition using laplacianfaces. *IEEE TPMAI*, 27(3):328–340, 2005.

REFERENCES

- [42] Erik Hjelmås and Boon Kee Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83:236–274, 2001.
- [43] Tin Kam Ho, J.J. Hull, and S.N. Srihari. Decision combination in multiple classifier systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(1):66–75, 1994.
- [44] Steven C.H. Hoi. *Statistical Machine Learning for Data Mining and Collaborative Multimedia Retrieval*. PhD thesis, The Chinese University of Hong Kong, 2006.
- [45] Patrik O. Hoyer. Non-negative sparse coding. *CoRR*, cs.NE/0202009, 2002.
- [46] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.
- [47] Rabia Jafri and Hamid R. Arabnia. A survey of face recognition techniques. *Journal of Information Processing Systems*, 5:41–68, 2009.
- [48] Anil K. Jain and Stan Z. Li. *Handbook of Face Recognition*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.
- [49] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.
- [50] Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, October 2009.
- [51] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):881–892, July 2002.
- [52] J. Kittler, M. Hatef, R. P W Duin, and J. Matas. On combining classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(3):226–239, 1998.

REFERENCES

- [53] Mislav Grgic Kresimir Delac and Marian Stewart Bartlett. *Recent Advances in Face Recognition*. I-Tech Education and Publishing, 2008.
- [54] Neeraj Kumar, Peter N. Belhumeur, and Shree K. Nayar. Facetracer: A search engine for large collections of images with faces. In *European Conference on Computer Vision*, pages 340–353, 2008.
- [55] Duy-Dinh Le and Shin’ichi Satoh. Unsupervised face annotation by mining the web. In *ICDM’08*, pages 383–392, 2008.
- [56] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 1999.
- [57] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *NIPS’07*, pages 801–808. NIPS, 2007.
- [58] Honglak Lee, Rajat Raina, Alex Teichman, and Andrew Y. Ng. Exponential family sparse coding with application to self-taught learning. In *IJCAI’09*, pages 1113–1119, 2009.
- [59] Chengjun Liu and H. Wechsler. Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. *IEEE Trans. on Image Processing*, 11(4):467–476, apr 2002.
- [60] J. Liu, S. Ji, and J. Ye. *SLEP: Sparse Learning with Efficient Projections*. Arizona State University, 2009.
- [61] Jun Liu and Jieping Ye. Efficient euclidean projections in linear time. In *ICML’09*, pages 657–664, Montreal, Quebec, Canada, 2009.
- [62] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *ICML’09*, pages 689–696, New York, NY, USA, 2009. ACM.
- [63] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Supervised dictionary learning. In *NIPS*, pages 1033–1040, 2008.

REFERENCES

- [64] Julien Mairal, Michael Elad, and Guillermo Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, 17:53–69, 2007.
- [65] Ameesh Makadia, Vladimir Pavlovic, and Sanjiv Kumar. A new baseline for image annotation. In *ECCV'08, ECCV'08*, pages 316–329, Berlin, Heidelberg, 2008. Springer-Verlag.
- [66] Brian Mcfee and Gert Lanckriet. Metric learning to rank. In *ICML'10*, 2010.
- [67] Stefano Melacci and Mikhail Belkin. Laplacian support vector machines trained in the primal. *J. Mach. Learn. Res.*, 12:1149–1184, July 2011.
- [68] Francisco Moreno-Seco, José M. Iñesta, Pedro J. Ponce de León, and Luisa Micó. Comparison of classifier fusion methods for classification in pattern recognition tasks. In *Proceedings of the 2006 Joint IAPR International Conference on Structural, Syntactic, and Statistical Pattern Recognition, SSPR'06/SPR'06*, pages 705–713, Berlin, Heidelberg, 2006. Springer-Verlag.
- [69] Kevin P Murphy. *Machine learning: a probabilistic perspective*. Cambridge, MA, 2012.
- [70] P. Nagesh and Baoxin Li. A compressive sensing approach for expression-invariant face recognition. *IEEE CVPR'09*, pages 1518–1525, june 2009.
- [71] Hieu V. Nguyen and Li Bai. Cosine similarity metric learning for face verification. In *ACCV'10*, June 2008.
- [72] Thao Ngoc Nguyen, Thanh Duc Ngo, Duy-Dinh Le, Shin'ichi Satoh, Bac Hoai Le, and Duc Anh Duong. An efficient method for face retrieval from large video datasets. In *Conference on Image and Video Retrieval*, pages 382–389, 2010.
- [73] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Mach. Learn.*, 39(2-3):103–134, May 2000.
- [74] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: a strategy employed by v1? *Vision Res*, 37(23):3311–3325, December 1997.

REFERENCES

- [75] Mert Ozcan, Jie Luo, Vittorio Ferrari, and Barbara Caputo. A large-scale database of images and captions for automatic face naming. *Idiap-RR Idiap-RR-26-2011*, Idiap, 8 2011.
- [76] Derya Ozkan and Pynar Duygulu. A graph based approach for naming faces in news photos. In *IEEE CVPR'06*, pages 1477–1482, 2006.
- [77] Phi The Pham, Marie-Francine Moens, and Tinne Tuytelaars. Naming persons in news video with label propagation. In *ICME*, pages 1528–1533, 2010.
- [78] Phi The Pham, T. Tuytelaars, and M. Moens. Naming people in news videos with label propagation. *MultiMedia, IEEE*, 18(3):44–55, 2011.
- [79] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press, 1999.
- [80] G. Rafiee, S.S. Dlay, and W.L. Woo. A review of content-based image retrieval. In *Communication Systems Networks and Digital Signal Processing (CSNDSP), 2010 7th International Symposium on*, pages 775–779, 2010.
- [81] C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. In *Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on*, volume 1, pages 29–36, 2005.
- [82] Xiaoguang Rui, Mingjing Li, Zhiwei Li, Wei-Ying Ma, and Nenghai Yu. Bipartite graph reinforcement model for web image annotation. In *ACM SIGMM'07*, pages 585–594, Augsburg, Germany, 2007.
- [83] Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Labelme: A database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008.
- [84] Shin'ichi Satoh, Yuichi Nakamura, and Takeo Kanade. Name-it: Naming and detecting faces in news videos. *IEEE Trans. on MultiMedia*, 6(1), 1999.

REFERENCES

- [85] Shishir K. Shandilya and Nidhi Singhai. A survey on: Content based image retrieval systems. *International Journal of Computer Applications*, 4(2):22–26, July 2010. Published By Foundation of Computer Science.
- [86] Li Shang and Deshuang Huang. Image denoising using non-negative sparse coding shrinkage algorithm. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 1017 – 1022 vol. 1, june 2005.
- [87] Bhavin J. Shastri and Martin D. Levine. Face recognition using localized features based on non-negative sparse coding. *Mach. Vision Appl.*, 18(2):107–122, 2007.
- [88] Qinfeng Shi, Hanxi Li, and Chunhua Shen. Rapid face recognition using hashing. In *Computer Vision and Pattern Recognition*, pages 2753–2760, 2010.
- [89] Christian Siagian and Laurent Itti. Rapid biologically-inspired scene classification using features shared with visual attention. *IEEE TPMAI*, 29:300–312, February 2007.
- [90] J. Sivic, M. Everingham, and A. Zisserman. “Who are you?” – learning person specific classifiers from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.
- [91] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE TPMAI*, 22(12):1349–1380, 2000.
- [92] Jun Sun, Qing Zhuo, Chengyuan Ma, and Wenyan Wang. Sparse image coding with clustering property and its application to face recognition. *Pattern Recognition*, 34(9):1883–1884, 2001.
- [93] Xiaoyang Tan, Songcan Chen, Zhi-Hua Zhou, and Fuyan Zhang. Face recognition from a single image per person: A survey. *Pattern Recognition*, 39(9):1725–1745, 2006.
- [94] Jinhui Tang, Richang Hong, Shuicheng Yan, Tat-Seng Chua, Guo-Jun Qi, and Ramesh Jain. Image annotation by knn-sparse graph-based label propagation over noisily tagged web images. *ACM TIST*, 2:14:1–14:15, 2011.

REFERENCES

- [95] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, September 2005.
- [96] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *IEEE CVPR'91*, pages 586–591, 1991.
- [97] Paul Viola and Michael J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.
- [98] Changhu Wang, Feng Jing, Lei Zhang, and Hong-Jiang Zhang. Image annotation refinement using random walk with restarts. In *ACM SIGMM'06*, pages 647–650, 2006.
- [99] Changhu Wang, Shuicheng Yan, Lei Zhang, and Hong-Jiang Zhang. Multi-label sparse coding for automatic image annotation. *IEEE CVPR'09*, 0:1643–1650, 2009.
- [100] Dayong Wang, Steven C. H. Hoi, Ying He, and Jianke Zhu. Retrieval-based face annotation by weak label regularized local coordinate coding. In *ACM SIGMM'11*, pages 353–362, 2011.
- [101] Dayong Wang, Steven C.H. Hoi, and Ying He. Mining weakly labeled web facial images for search-based face annotation. In *ACM SIGIR'11*, 2011.
- [102] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *IEEE CONFERENCE ON COMPUTER VISION AND PATTERN CLASSIFICATION*, 2010.
- [103] Xin-Jing Wang, Lei Zhang, Feng Jing, and Wei-Ying Ma. Annosearch: Image auto-annotation by search. In *IEEE CVPR'06*, 2006.
- [104] Laurenz Wiskott, Jean marc Fellous, Norbert Krer, and Christoph Von Der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:456–463, 1997.
- [105] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE TPMAI*, 31(2):210–227, April 2008.

REFERENCES

- [106] John Wright and Yi Ma. Dense error correction via l_1 -minimization. *IEEE Trans. Information Theory*, 56(7):3540–3560, July 2010.
- [107] Fei Wu, Yahong Han, Qi Tian, and Yueting Zhuang. Multi-label boosting for image annotation by structural grouping sparsity. In *ACM SIGMM'10*, pages 15–24, 2010.
- [108] Zhong Wu, Qifa Ke, Jian Sun, and Heung-Yeung Shum. Scalable face image retrieval with identity-based quantization and multi-reference re-ranking. In *IEEE CVPR'10*, pages 3469–3476, 2010.
- [109] Jun Xu, Tie-Yan Liu, Min Lu, Hang Li, and Wei-Ying Ma. Directly optimizing evaluation measures in learning to rank. In *ACM SIGIR'08*, pages 107–114, 2008.
- [110] Jun Yang and Alexander G. Hauptmann. Naming every individual in news video monologues. In *ACM SIGMM'04*, pages 580–587, New York, NY, USA, 2004.
- [111] Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. Technical report, Department of Computer Science and Engineering, Michigan State University, 2006.
- [112] Meng Yang and Lei Zhang. Gabor feature based sparse representation for face recognition with gabor occlusion dictionary. In *ECCV'10*, pages 448–461, 2010.
- [113] Qi Yin, Xiaoou Tang, and Jian Sun. An associate-predict model for face recognition. In *Computer Vision and Pattern Recognition*, 2011.
- [114] Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In *NIPS'09*, pages 2259–2267, 2009.
- [115] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of The Royal Statistical Society Series B-statistical Methodology*, 68:49–67, 2006.
- [116] Xiaotong Yuan and Shuicheng Yan. Visual classification with multi-task joint sparse representation. In *CVPR*, pages 3493–3500, 2010.

REFERENCES

- [117] Yisong Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *ACM SIGIR'07*, pages 271–278, 2007.
- [118] Fei Zang and Jiang-She Zhang. Label propagation through sparse neighborhood and its applications. *Neurocomput.*, 97:267–277, November 2012.
- [119] L. Zhang, L. Chen, M. Li, and H. Zhang. Automated annotation of human faces in family albums. In *ACM SIGMM'03*, 2003.
- [120] Ming Zhao, Jay Yagnik, Hartwig Adam, and David Bau. Large scale learning and recognition of faces in web videos. In *FG'08*, pages 1–7, 2008.
- [121] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35(4):399–458, 2003.
- [122] Yang Zhou, Rong Jin, and Steven Chu-Hong Hoi. Exclusive lasso for multi-task feature selection. In *AISTATS'10*, pages 988–995, Oct 2010.
- [123] Jianke Zhu, Steven C.H. Hoi, and Luc Van Gool. Unsupervised face alignment by robust nonrigid mapping. In *ICCV'09*, 2009.
- [124] Xiaojin Zhu. Semi-supervised learning literature survey. Technical report, 2006.
- [125] Xiaojin Zhu and John Lafferty. Harmonic mixtures: combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 1052–1059, New York, NY, USA, 2005. ACM.