

**Exploiting Contextual Information  
for Image Re-Ranking and Rank Aggregation  
in Image Retrieval Tasks**

*Explorando Informações Contextuais para  
Reclassificação de Imagens e Agregação de Listas  
em Tarefas de Recuperação de Imagens*

**Daniel Carlos Guimarães Pedronette**

Este exemplar corresponde à redação final da Tese devidamente corrigida e defendida por Daniel Carlos Guimarães Pedronette e aprovada pela Banca Examinadora.

Campinas, 23 de abril de 2012.

Prof. Dr. Ricardo da Silva Torres (Orientador)

Tese apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA POR  
MARIA FABIANA BEZERRA MULLER - CRB8/6162  
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E  
COMPUTAÇÃO CIENTÍFICA - UNICAMP

P343e Pedronette, Daniel Carlos Guimarães, 1983-  
Explorando informações contextuais para reclassificação de  
imagens e agregação de listas em tarefas de recuperação de  
imagens / Daniel Carlos Guimarães Pedronette. – Campinas, SP :  
[s.n.], 2012.

Orientador: Ricardo da Silva Torres.  
Tese (doutorado) – Universidade Estadual de Campinas,  
Instituto de Computação.

1. Sistemas de recuperação da informação. 2. Recuperação  
da informação. 3. Imagens - Recuperação. 4. Processamento  
de imagens. 5. Reconhecimento de padrões. I. Torres, Ricardo  
da Silva, 1977-. II. Universidade Estadual de Campinas. Instituto  
de Computação. III. Título.

Informações para Biblioteca Digital

**Título em inglês:** Exploiting contextual information for image re-ranking and rank aggregation in image retrieval tasks

**Palavras-chave em inglês:**

Information storage and retrieval systems

Information retrieval

Image - Retrieval

Image processing

Pattern recognition

**Área de concentração:** Ciência da Computação

**Titulação:** Doutor em Ciência da Computação

**Banca examinadora:**

Ricardo da Silva Torres [Orientador]

David Menotti Gomes

José Fernando Rodrigues Júnior

Marco Antonio Garcia de Carvalho

André Santanchè

**Data de defesa:** 23-04-2012

**Programa de Pós-Graduação:** Ciência da Computação

# TERMO DE APROVAÇÃO

Tese Defendida e Aprovada em 23 de Abril de 2012, pela Banca  
examinadora composta pelos Professores Doutores:

*Ricardo Torres*

Prof. Dr. Ricardo da Silva Torres  
IC / UNICAMP

*David Menotti Gomes*

Prof. Dr. David Menotti Gomes  
DC / UFOP

*José Fernando Rodrigues Júnior*

Prof. Dr. José Fernando Rodrigues Júnior  
ICMC / USP

*Marco Antonio G. de Carvalho*

Prof. Dr. Marco Antonio Garcia de Carvalho  
FT / UNICAMP

*André Santanchè*

Prof. Dr. André Santanchè  
IC / UNICAMP



**Exploiting Contextual Information  
for Image Re-Ranking and Rank Aggregation  
in Image Retrieval Tasks**

*Explorando Informações Contextuais para  
Reclassificação de Imagens e Agregação de Listas  
em Tarefas de Recuperação de Imagens*

**Daniel Carlos Guimarães Pedronette**

Abril de 2012

**Banca Examinadora:**

- Prof. Dr. Ricardo da Silva Torres (Orientador)
- Prof. Dr. David Menotti Gomes  
Departamento de Computação - UFOP
- Prof. Dr. José Fernando Rodrigues Júnior  
Instituto de Ciências Matemáticas e de Computação - USP
- Prof. Dr. Marco Antonio Garcia de Carvalho  
Faculdade de Tecnologia - UNICAMP
- Prof. Dr. André Santanchè  
Instituto de Computação - UNICAMP



# Resumo

Sistemas de Recuperação de Imagens baseados no Conteúdo (*Content-Based Image Retrieval* - CBIR) têm como objetivo satisfazer as necessidades dos usuários a partir de especificações de consulta. Dado um padrão de consulta (*e.g.*, uma imagem de consulta) como entrada, um sistema CBIR recupera as imagens mais similares em uma coleção considerando suas propriedades visuais. Como o maior interesse dos usuários diz respeito às primeiras posições da lista de imagens retornadas, a eficácia desses sistemas é extremamente dependente da acurácia da função de distância adotada.

Entretanto, de forma geral, as abordagens de CBIR analisam apenas pares de imagens para a geração das listas de resultados, ignorando importantes informações codificadas nos relacionamentos entre as imagens. Com o objetivo de aumentar a acurácia de sistemas CBIR, algoritmos de reclassificação (*re-ranking*) e agregação de listas (*rank aggregation*) têm sido propostos. Algoritmos de *re-ranking* têm sido usados para explorar informação contextual codificada nos relacionamentos entre as imagens enquanto métodos de *rank aggregation* têm sido propostos para combinar resultados produzidos por diferentes descritores de imagens.

O objetivo desta tese é investigar novas abordagens para modelar e representar informações contextuais, com o objetivo de usá-las em tarefas de *re-ranking* e *rank aggregation* visando aumentar a eficácia de sistemas de CBIR. As principais contribuições desta tese são a criação e implementação de cinco métodos de *re-ranking* e *rank aggregation* (*Distance Optimization Algorithm*, *Pairwise Recommendation*, *Contextual Spaces*, *RL-Sim*, e *Contextual Re-Ranking*) e o uso de computação paralela para execução eficiente de *re-ranking* em GPUs. A avaliação experimental conduzida considerou diferentes descritores de imagens (cor, textura e forma) e várias coleções de imagens. Os resultados dos experimentos demonstram a eficácia dos métodos propostos. Outras contribuições estão relacionadas ao uso dos métodos propostos em recuperação multimodal (considerando aspectos visuais e textuais) e a proposta de novas abordagens para combinar os métodos de *re-ranking* e *rank aggregation* propostos.



# Abstract

Content-Based Image Retrieval (CBIR) systems aims at meeting the user needs expressed in query specifications. Given a query pattern (*e.g.*, query image) as input, a CBIR system retrieves the most similar images in a collection by taking into account image visual properties. Since users are interested in the images placed at the first positions of the returned ranked lists, accurately ranking collection images is of great relevance.

However, in general, CBIR approaches perform only pairwise image analysis for computing the ranked lists. They compute similarity (or distance) measures considering only pairs of images, ignoring the rich information encoded in the relationships among images. Aiming at improving the effectiveness of CBIR systems, *re-ranking* and *rank aggregation* algorithms have been proposed. Re-ranking algorithms have been used to exploit contextual information, encoded in the relationships among collection images, while rank aggregation approaches have been used to combine results produced by different image descriptors.

The objective of this thesis is to investigate new approaches for modeling and representing contextual information, aiming their use in re-ranking and rank aggregation tasks used to improve the effectiveness of CBIR systems. The main contribution of this thesis consists in the creation and implementation of five image re-ranking and rank aggregation methods (Distance Optimization Algorithm, Pairwise Recommendation, Contextual Spaces, RL-Sim, and Contextual Re-Ranking algorithm) and the use of parallel computing for efficient re-ranking computation on GPUs. The experimental evaluation considered different image descriptors (*e.g.*, color, texture, and shape) and several datasets. Experiment results demonstrate the effectiveness of the proposed methods. Other contributions are related to the use of the proposed method in multimodal retrieval tasks, and the proposal of new approaches for combining the proposed re-ranking and rank aggregation methods.



# Agradecimentos

Agradeço a todos pelo apoio no caminho que levou a conclusão desse trabalho.

Agradeço a Deus, pela vida e saúde.

Agradeço à minha família, pela apoio firme e constante. Aos meus pais Carlos e Marilena, que me antecederam, dando os incentivos iniciais, me guiando e auxiliando nos primeiros passos. À minha esposa Daniele, que esteve comigo durante o percurso, caminhando de mãos dadas, com seu amor e carinho, sem os quais muitos obstáculos teriam sido intransponíveis. Aos meus filhos, Guilherme, Giovanni e Laura, que correm a minha frente, me puxando pelos desafios e questionamentos que me proporcionam. Aos meus avós Raul e Genir, que vão sempre comigo como uma lembrança doce a ser guardada. Aos meus tio(a)s e primo(a)s, em especial a minha tia Márcia.

Agradeço ao Prof. Ricardo Torres, pelo seu grande exemplo como orientador. Agradeço pela sua atenção, pela dedicação e pelos inumeráveis ensinamentos. Por ter mais do que me orientado, me guiado durante esses trabalho. Agradeço por ter-me permitido considerá-lo como amigo, dispensando aos seus alunos atenção de inestimável valor humano.

Agradeço à DGA/Unicamp como instituição, aos seus gerentes Marilda e Vladimir e sua coordenadora Edna pelo apoio ao desenvolvimento desse trabalho.

Agradeço ao Prof. Edson Borin e Mauricio Breternitz, que muito me auxiliaram no aprendizado de uma nova área.

Agradeço ao Prof. Marco Antonio Garcia de Carvalho, ao Prof. Neucimar Jerônimo Leite, ao Prof. André Santanchè, ao Prof. David Menotti Gomes, ao Prof. José Fernando Rodrigues Júnior e ao Prof. Marcos André Gonçalves pelas importantes sugestões dadas.

Agradeço aos alunos de doutorado Otávio Penatti, Rodrigo Tripodi e Nádia Kozievitch pela colaboração.

Agradeço a todos os meus amigos que me incentivaram ao longo do caminho. Em especial ao meu amigo Paulo Facco, aos colegas da DGA/Unicamp e ao Arthur Vieira Gomes que acompanharam o desenvolvimento do trabalho.

Agradeço ao apoio financeiro de CAPES, FAEPEX, FAPESP, CNPq, Microsoft e AMD.



# Contents

<b>Resumo</b>	<b>vii</b>
<b>Abstract</b>	<b>ix</b>
<b>Agradecimientos</b>	<b>xi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Research Challenges . . . . .	3
1.3 Thesis Statement . . . . .	4
1.4 Goals and Contributions . . . . .	4
1.5 Organization . . . . .	8
<b>2 Background and Related Work</b>	<b>11</b>
2.1 Content-Based Image Retrieval . . . . .	11
2.1.1 Typical Architecture . . . . .	11
2.1.2 Image Retrieval Model . . . . .	12
2.2 Image Re-Ranking . . . . .	13
2.2.1 Image Re-Ranking Definition . . . . .	13
2.2.2 Re-Ranking Approaches . . . . .	14
2.3 Rank Aggregation . . . . .	21
2.3.1 Rank Aggregation Definition . . . . .	21
2.3.2 Rank Aggregation Approaches . . . . .	22
2.4 Notation . . . . .	23
<b>3 Experimental Protocol</b>	<b>25</b>
3.1 Content-Based Image Retrieval . . . . .	25
3.1.1 Descriptors . . . . .	25
3.1.2 Datasets . . . . .	26
3.1.3 Measures . . . . .	27

3.2	Multimodal Retrieval . . . . .	29
3.2.1	Descriptors . . . . .	29
3.2.2	Datasets . . . . .	30
3.2.3	Measures . . . . .	30
3.3	Summary . . . . .	30
<b>4</b>	<b>Distance Optimization Algorithm</b>	<b>33</b>
4.1	Distance Optimization Algorithm . . . . .	34
4.1.1	Impact of DOA on Distance Distribution . . . . .	35
4.1.2	Cohesion Measure . . . . .	37
4.1.3	Clustering Approach . . . . .	39
4.2	Cluster-Similar Functions . . . . .	41
4.2.1	Similarity of Ranked Lists . . . . .	41
4.2.2	Correlation . . . . .	42
4.3	Updating of Distances . . . . .	43
4.3.1	Update based on Decreasing Distances . . . . .	43
4.3.2	Update based on Correlation of Distances . . . . .	43
4.4	Combination of Approaches . . . . .	45
4.5	Using DOA for Rank Aggregation . . . . .	46
4.6	Aspects of Efficiency . . . . .	47
4.7	Experimental Evaluation . . . . .	47
4.7.1	Experiment 1 - Cluster-Similar Functions Comparison . . . . .	48
4.7.2	Experiment 2 - Correlation Impact . . . . .	49
4.7.3	Experiment 3 - Shape Descriptors . . . . .	49
4.7.4	Experiment 4 - Texture Descriptors . . . . .	51
4.7.5	Experiment 5 - Color Descriptors . . . . .	52
4.7.6	Experiment 6 - General CBIR Tasks . . . . .	52
<b>5</b>	<b>Pairwise Recommendation</b>	<b>55</b>
5.1	The Re-Ranking Algorithm . . . . .	56
5.2	Cohesion Measure . . . . .	58
5.3	Performing Recommendations . . . . .	59
5.4	Clustering Approach . . . . .	61
5.5	Convergence Criterion . . . . .	61
5.6	The Rank Aggregation Algorithm . . . . .	62
5.7	Experimental Evaluation . . . . .	63
5.7.1	Experiment 1 - Impact of Parameters . . . . .	64
5.7.2	Experiment 2 - Shape Descriptors . . . . .	65
5.7.3	Experiment 3 - General CBIR Tasks . . . . .	66

5.7.4	Experiment 4 - Analysis of Convergence . . . . .	68
5.7.5	Experiment 5 - Rank Aggregation . . . . .	70
<b>6</b>	<b>Contextual Spaces</b>	<b>75</b>
6.1	The Contextual Spaces Algorithms . . . . .	76
6.1.1	Contextual Spaces Representation . . . . .	76
6.1.2	The Re-Ranking Algorithms . . . . .	79
6.1.3	Impact of Re-Ranking Algorithms on Distances . . . . .	83
6.1.4	The Rank Aggregation Algorithm . . . . .	87
6.1.5	Aspects of Efficiency . . . . .	87
6.2	Experimental Evaluation . . . . .	88
6.2.1	Experiment 1: Impact of Parameters . . . . .	88
6.2.2	CBIR Tasks Involving Color, Texture, and Shape Descriptors . . . . .	90
6.2.3	Experiment 5: Analysis of Convergence . . . . .	95
6.2.4	Experiment 6: Rank Aggregation . . . . .	96
6.2.5	Experiment 7: Multimodal Retrieval . . . . .	96
<b>7</b>	<b>RL-Sim Re-Ranking</b>	<b>99</b>
7.1	Contextual Distance Measure based on Ranked Lists . . . . .	100
7.2	Comparing Ranked Lists . . . . .	102
7.2.1	Neighborhood Set . . . . .	102
7.2.2	Distance Measures between Top k Lists . . . . .	103
7.3	The RL-Sim Re-Ranking Algorithm . . . . .	105
7.4	Rank Aggregation . . . . .	107
7.5	Experimental Evaluation . . . . .	107
7.5.1	Experiment 1: Impact of Parameters . . . . .	108
7.5.2	Experiment 2: Aspects of Efficiency . . . . .	109
7.5.3	Experiment 3: Re-Ranking Evaluation . . . . .	110
7.5.4	Experiment 4: Rank Aggregation Evaluation . . . . .	113
<b>8</b>	<b>Contextual Re-Ranking</b>	<b>115</b>
8.1	Contextual Information Representation . . . . .	116
8.2	The Contextual Re-Ranking Algorithm . . . . .	117
8.3	The Contextual Rank Aggregation Algorithm . . . . .	122
8.4	Experimental Evaluation . . . . .	122
8.4.1	Experiment 1: Analysis of Contextual Re-Ranking Algorithm . . . . .	125
8.4.2	Experiment 2: Re-Ranking . . . . .	127
8.4.3	Experiment 3: Rank Aggregation . . . . .	130

<b>9</b>	<b>Efficient Image Re-Ranking Computation on GPUs</b>	<b>133</b>
9.1	General Purpose computing on GPUs (GPGPUs)	134
9.2	GPU Acceleration of the Contextual Re-Ranking Algorithm	135
9.2.1	OpenCL	135
9.2.2	Parallel Contextual Re-Ranking	136
9.2.3	Optimizations	138
9.3	Experimental Evaluation	140
9.3.1	Experimental Setup	140
9.3.2	Performance Results	140
9.3.3	Effectiveness Analysis	142
<b>10</b>	<b>Combining Re-Ranking and Rank Aggregation Methods</b>	<b>145</b>
10.1	Cascading Re-Ranking	146
10.2	Re-Ranking with Rank Aggregation Combination	146
10.3	Agglomerative Rank Aggregation	146
10.4	Used Re-Ranking and Rank Aggregation Methods	147
10.4.1	Re-Ranking Methods	147
10.4.2	Rank Aggregation Methods	149
10.5	Experimental Evaluation	149
10.5.1	Descriptors and Datasets	150
10.5.2	Cascading Re-Ranking	150
10.5.3	Combining Re-Ranking methods with Rank Aggregation	151
10.5.4	Agglomerative Rank Aggregation	152
<b>11</b>	<b>Comparing Re-Ranking and Rank Aggregation Methods</b>	<b>155</b>
11.1	Comparison of the proposed Re-Ranking Methods	155
11.2	Comparison of the proposed Rank Aggregation Methods	158
11.3	Comparison with other Re-Ranking Approaches	159
11.4	Comparison with other Rank Aggregation Approaches	160
11.5	Other aspects	160
11.6	Discussion	161
<b>12</b>	<b>Conclusions</b>	<b>163</b>
12.1	Contributions	163
12.2	Future Work	164
	<b>Bibliography</b>	<b>167</b>

# List of Tables

2.1	Notation used along this thesis . . . . .	23
3.1	Summary of experimental protocol: Datasets, Descriptors, and Measures. . . . .	32
4.1	Cluster-similar functions comparison on the MPEG-7 dataset. . . . .	48
4.2	Post-processing methods comparison on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	51
4.3	Distance Optimization Algorithm applied to shape descriptors on MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	51
4.4	Correlation Methods Evaluation on Several Content-Based Image Retrieval Tasks - Mean Average Precision . . . . .	54
5.1	Pairwise Recommendation for Shape Descriptors on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	66
5.2	Pairwise Recommendation Evaluation on Several Content-Based Image Retrieval Tasks ( <i>MAP</i> ). . . . .	70
5.3	Pairwise Recommendation for Descriptors Combination ( <i>MAP</i> ). . . . .	72
6.1	Contextual Spaces evaluation on several content-based image retrieval tasks involving color, texture, and shape descriptors. . . . .	92
6.2	Contextual Spaces for shape descriptors on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	93
6.3	Contextual Spaces for rank aggregation ( <i>MAP</i> ) . . . . .	96
6.4	Descriptors scores ( <i>MAP</i> ) on UW Database . . . . .	98
6.5	Contextual Spaces on multimodal retrieval tasks ( <i>MAP</i> as score) . . . . .	98
7.1	RL-Sim Re-Ranking using Intersection Distance Measure for shape descriptors on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	111
7.2	RL-Sim Re-Ranking using Kendall's Tau Distance Measure for shape descriptors on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	112
7.3	MAP scores for RL-Sim Re-Ranking using Intersection Distance Measure in different CBIR tasks. . . . .	113

7.4	MAP scores for RL-Sim Re-Ranking using Kendall’s Tau Distance Measure in different CBIR tasks. . . . .	114
7.5	MAP scores regarding the use of RL-Sim Algorithm for Rank Aggregation	114
8.1	Contextual Re-Ranking Evaluation in Content-Based Image Retrieval Tasks.	128
8.2	Contextual Re-Ranking for Shape Descriptors on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	128
8.3	Contextual rank aggregation on several content-based image retrieval tasks ( <i>Mean Average Precision</i> ) . . . . .	131
9.1	Performance comparison for different kernel implementations - No optimizations, using list of increments . . . . .	141
9.2	Performance comparison for different kernel implementations - Optimizations and increments direct on Affinity Matrix W . . . . .	141
9.3	Best Combination of Kernels/Devices - Total Run Time . . . . .	142
9.4	Contextual Re-Ranking Effectiveness Evaluation - MAP. . . . .	143
10.1	Cascading Re-Ranking Methods on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	151
10.2	Re-Ranking with Rank Aggregation Combination on CBIR Tasks ( <i>MAP</i> ). . . . .	152
10.3	Re-Ranking and Rank Aggregation Combination for Shape Descriptors on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	152
10.4	Aglomerative Rank Aggregation Combination for CBIR Tasks ( <i>MAP</i> ). . . . .	153
11.1	MAP scores for proposed re-ranking methods in different CBIR tasks. . . . .	157
11.2	Recall@40 scores for proposed re-ranking methods in shape retrieval tasks. . . . .	157
11.3	MAP scores for proposed Rank Aggregation methods in general CBIR tasks. . . . .	158
11.4	Re-Ranking methods comparison on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	159
11.5	Re-Ranking and post-processing methods comparison on the Kimia99 dataset. . . . .	160
11.6	Rank Aggregation methods comparison on the MPEG-7 dataset ( <i>Recall@40</i> ). . . . .	161
11.7	Comparison of re-ranking methods regarding various criteria. . . . .	162

# List of Figures

1.1	Abstract model of image re-ranking algorithms. . . . .	5
1.2	Rank Aggregation Model: Rank Aggregation before Re-Ranking. . . . .	6
1.3	Rank Aggregation Model: Rank Aggregation after Re-Ranking. . . . .	7
1.4	Rank Aggregation Model: Rank Aggregation during Re-Ranking. . . . .	7
1.5	Thesis organization, main concepts, and obtained publications. . . . .	10
2.1	Typical architecture of a content-based image retrieval system [16]. . . . .	12
3.1	Examples of MPEG-7 dataset shapes. . . . .	26
3.2	Examples of Kimia dataset shapes. . . . .	27
3.3	Examples of Brodatz [7] texture images. . . . .	27
3.4	Examples of Soccer dataset [100] images. . . . .	27
3.5	Example of a typical Precision $\times$ Recall curve. . . . .	29
3.6	Examples of UW dataset [22] images and tags. . . . .	31
4.1	Overview of the Distance Optimization Algorithm. . . . .	34
4.2	Bidimensional space representation for two similar images. . . . .	36
4.3	Bidimensional space representation for two non-similar images. . . . .	37
4.4	Impact of the Distance Optimization Algorithm on distances. . . . .	38
4.5	Impact of Distance Optimization Algorithm on ranked lists. . . . .	38
4.6	Example of <i>cluster-similarity</i> between images $img_1$ and $img_2$ with regard to $O_p = \{(1, 8), (2, 6), (3, 5), (4, 4)\}$ . . . . .	42
4.7	Segmentation of ranked lists in the new distance update approach. . . . .	44
4.8	Convergence: number of clusters per iteration. . . . .	49
4.9	Precision vs. Recall: comparing results of distance optimization algorithm variations . . . . .	50
4.10	Distance Optimization applied to Texture Descriptors. . . . .	52
4.11	Distance Optimization applied to Color Descriptors. . . . .	53
5.1	Pairwise Recommendation re-ranking method. . . . .	57
5.2	Computation of the cohesion measure. . . . .	59

5.3	Performing recommendations. . . . .	61
5.4	Impact of parameters $K$ and $T$ for $L=1$ . . . . .	65
5.5	Impact of parameters $K$ and $T$ for $L=2$ . . . . .	66
5.6	Impact of parameter $L$ ( $K=8$ , $T=15$ ). . . . .	67
5.7	Impact of parameters on execution time. . . . .	68
5.8	Impact of Pairwise Recommendation re-ranking on ranked lists. . . . .	68
5.9	Percentage gains in bulls-eye score for each class of MPEG-7 dataset considering CFD [68] shape descriptor. . . . .	69
5.10	Analysis of convergence - Cohesion measure. . . . .	71
5.11	Analysis of convergence - Kendall's tau distance. . . . .	72
5.12	Pairwise Recommendation for Shape Descriptors on the MPEG-7 dataset. . . . .	73
5.13	Distance matrices by shape descriptors on the MPEG-7 dataset: (a) CFD [68]; (b) IDSC [52]; (c) CFD+IDSC+Pairwise Recommendation. . . . .	73
6.1	Similar reference images. . . . .	77
6.2	Bidimensional space representation for two similar images. . . . .	77
6.3	Non-similar reference images. . . . .	78
6.4	Bidimensional space representation for two non-similar images. . . . .	78
6.5	Contextual Spaces based on Mutual-KNN: selection of images for constructing the contextual spaces. . . . .	82
6.6	Similar reference images. . . . .	85
6.7	Bidimensional space representation for two similar images after <i>Contextual Spaces based on Mutual-KNN</i> algorithm execution. . . . .	85
6.8	Non-similar reference images. . . . .	86
6.9	Bidimensional space representation for two non-similar images after <i>Contextual Spaces based on Mutual-KNN</i> algorithm execution. . . . .	86
6.10	Impact of parameters $K_s$ and $K_e$ on precision score for <i>Contextual Spaces based on KNN</i> . . . . .	89
6.11	Impact of parameters $K$ and $T$ on <i>Contextual Spaces based on Mutual-KNN</i> . . . . .	90
6.12	Impact of parameter $\lambda$ on <i>Contextual Spaces based on Mutual-KNN</i> . . . . .	91
6.13	Impact of <i>Contextual Spaces</i> re-ranking algorithm on ranked lists. . . . .	93
6.14	Percentage gain in bulls-eye score for each class of MPEG-7 dataset considering CFD [68] shape descriptor and <i>Contextual Spaces based on KNN</i> algorithm. . . . .	94
6.15	Analysis of convergence - Kendall's tau distance along iterations. . . . .	95
6.16	<i>Contextual Spaces based on KNN</i> for shape descriptors on the MPEG-7 dataset. . . . .	97
7.1	Computation of measure $\psi$ : intersection of ranked lists with different sizes. . . . .	104

7.2	Impact of parameters: $k_s$ and $T$ .	109
7.3	Impact of parameter $\lambda$ on precision.	110
7.4	Evolution of rankings along iterations on the MPEG-7 [48] dataset.	111
8.1	Similar reference images.	117
8.2	Non-similar reference images.	117
8.3	Context image for similar reference images.	117
8.4	Context image for non-similar reference images.	117
8.5	The Contextual Re-Ranking Algorithm.	118
8.6	Example of binary image.	120
8.7	Example of filtered image.	120
8.8	Updates of matrix $W$ .	120
8.9	Relationship among images provided by a single pixel of a context image.	121
8.10	Iteration 1: 91.84%, $K=5$ , $L=50$ .	124
8.11	Iteration 2: 94.41%, $K=7$ , $L=25$ .	124
8.12	Iteration 3: 95.29%, $K=7$ , $L=25$ .	124
8.13	Iteration 4: 95.66%, $K=7$ , $L=25$ .	124
8.14	Impact of iterations on precision.	126
8.15	Contextual re-ranking percent gain for CFD [68] shape descriptor on the MPEG-7 classes.	129
8.16	Evolution of rankings along iterations on the MPEG-7 [48] dataset (first column contains the query image): the first row presents the results of CFD [68] shape descriptor; the remaining rows present the results of the <i>Contextual Re-Ranking Algorithm</i> for each iteration.	130
8.17	Contextual re-ranking and rank aggregation for shape descriptors.	131
9.1	Parallel project of the Contextual Re-Ranking Algorithm.	137
9.2	Update model using list of increments.	138
9.3	Update model with increments direct on Matrix $W$ .	139
10.1	Cascading Re-Ranking.	146
10.2	Re-Ranking with Rank Aggregation Combination.	147
10.3	Agglomerative Rank Aggregation.	148
12.1	Contributions, main concepts, and possible extension of this thesis.	166



# List of Algorithms

4.1	Distance Optimization Algorithm . . . . .	35
4.2	Clustering Algorithm. . . . .	39
4.3	Algorithm evaluateSimilarity. . . . .	40
4.4	Algorithm processImage. . . . .	40
4.5	Rank Aggregation based on Distance Optimization Algorithm . . . . .	46
5.1	Pairwise Recommendation Re-Ranking . . . . .	57
5.2	Recommendations performing . . . . .	60
6.1	Re-Ranking Algorithm: Contextual Spaces KNN . . . . .	81
6.2	Re-Ranking Algorithm: Contextual Spaces Mutual-KNN . . . . .	84
7.1	RL-Sim Re-Ranking Algorithm . . . . .	106
8.1	Contextual Re-Ranking Algorithm . . . . .	119
8.2	Contextual Rank Aggregation Algorithm . . . . .	123



# Chapter 1

## Introduction

### 1.1 Motivation

The huge growth of image collections and multimedia resources available and accessible through various technologies is remarkable. The continuously decrease of storage devices costs and the technological improvements in image acquisition and sharing facilities have enabled the dissemination of very large digital image collections, accessible through various technologies.

In this scenario, there is the need of methods for indexing and retrieving these data. Two common approaches are used to support image searches: the first one is concerned with the proposal of methods for retrieving images based on textual annotation [54, 59]; the second relies on supporting image searches by taking into account image content information, using the so-called Content-Based Image Retrieval systems [21].

Image retrieval approaches based on keywords and textual metadata face serious challenges [28]. Describing the image content with textual descriptions is intrinsically very difficult, and this task has not been made easier by the growth and diversification of image collections. Many applications, especially those dealing with large general image collections face obstacles to obtain textual descriptors, since manual annotation is prohibitively expensive, contextual text is scarce or unreliable, and user needs are impossible to anticipate [28].

Content-Based Image Retrieval (CBIR) can be seen as any technology that helps to search and organize digital picture archives by means of their visual content [21]. For two decades, several CBIR initiatives have proposed the use of image visual properties (such as, shape, color, and texture) in retrieval tasks. Basically, a CBIR system aims at meeting the user needs expressed in a query specification (*e.g.*, by defining a query image as input). The method usually applied to achieve its goal relies on retrieving the most similar images in a collection by taking into account image visual properties. Collection

images are *ranked* in decreasing order of similarity, according to a given *image descriptor*. An image content descriptor is characterized by [16]: (i) an extraction algorithm that encodes image features into feature vectors; and (ii) a similarity measure used to compare two images. The similarity between two images is computed as a function of the distance of their feature vectors.

Several CBIR approaches have been proposed considering applications on different areas, such as facial image retrieval [97], biodiversity information systems [19], medical applications [58], and remote sensing images [24]. Several efforts have been proposed for improving the effectiveness of CBIR approaches. Example of recent initiatives include the use of novel image descriptors, matching algorithms, and new approaches for combining descriptors [29, 51]. A direct way to improve the effectiveness of CBIR systems relies on using more accurate features for describing images. Another possibility is related to the definition of similarity (or distance) functions that would be able to measure the distance between feature vectors in a more effective way.

However, in general, all these approaches perform only pairwise image analysis, that is, they compute similarity (or distance) measures considering only pairs of images, ignoring the rich information encoded in the relationships among images. On the other hand, the user perception usually considers the query specification and the query responses in a given *context*. Context can be broadly defined as all information about the whole situation relevant to an application and its set of users. In interactive applications, the use of context can play an important role [1]. In information retrieval and recommendation systems, context information includes geographic information, user profiles, and relationships among users and objects that can be used for improving the effectiveness of results. In a CBIR scenario, relationships among images, encoded in ranked lists and distances among images, can be used for extracting *contextual information* [42, 77].

Recently, some CBIR approaches [45, 113–115] have been proposed aiming at improving the effectiveness of retrieval tasks replacing pairwise similarities by more global affinities that also consider the relation among all the database objects. In other words, some efforts were put on post-processing the distance/similarity scores, by taking into account the *contextual information* available in relationships among images in a given collection. These methods require no user intervention, training or labeled data, and operate on an absolutely *unsupervised* way. It can be very valuable in a very large number of scenarios, in which training data can be very hard to obtain. Beside that, in a large number of applications the collection of images are static or almost static (personal image collections, for example), what can reduce significantly the need of post-processing efforts.

In practice, given a query image and a ranked list computed by a CBIR descriptor, *contextual information* can be used for *re-ranking* images aiming at improving the effectiveness of the image retrieval task. Contextual information can also be used in *rank*

*aggregation* approaches, in which ranked lists defined by different descriptors are combined aiming at producing more effective results. The main focus of this thesis is to exploit *contextual information* aiming at improving the effectiveness of CBIR tasks, without the need of training data.

## 1.2 Research Challenges

Different resources containing data about relationships among images are available on CBIR tasks. Given an image collection, the distances among all images in the the collection can be computed by a CBIR descriptor, producing a distance matrix  $A$ . Let  $N$  be the size of the collection, we can assume that there will be available  $N^2$  distance values among images. Based on these distances, a ranked list can be obtained for each collection image. A ranked list organizes collection images in decreasing order of similarity and, therefore,  $N$  ranked lists can be obtained (with  $N$  images each one).

In this scenario, the large amount of data available for analysing the relationships among images is remarkable. However, despite the large available data, there are several research challenges related to the transformation of those raw data into useful contextual information, which can be actually used for improving the effectiveness of retrieval tasks. In the following, we discuss some of them:

- **Heterogeneity of data sources:** there are several data sources for analysing relationship among images: distances/similarity scores, ranked lists, nearest neighbors, etc. What kind of data can be used for exploiting useful contextual information?
- **Contextual information encoding and representation:** given a source of information, how can we process available data in order to extract useful contextual information?
- **Efficiency on contextual information representation:** there are a lot of available data about relationships among images. How to process the minimum amount of raw data to extract the most useful contextual information for improving the retrieval effectiveness?
- **Combination:** given various and different ways to exploit contextual information, how can we combine those different approaches aiming at further improving the retrieval effectiveness?
- **Efficiency on implementing image re-ranking and rank aggregation methods:** how can we efficiently compute re-ranking and rank aggregation methods?

- **Applications:** how can we tailor the methods to different retrieval tasks and applications?

The work developed in this thesis addresses those important research challenges.

### 1.3 Thesis Statement

We have discussed the motivations and the great potential of exploiting *contextual information* for CBIR systems. On the other hand, we presented a set of research challenges involved in this task. Now, we aim to state the main related hypothesis that guides this thesis:

*The effectiveness of Content-Based Image Retrieval systems can be improved by exploiting contextual information through different unsupervised re-ranking and rank aggregation methods based on:*

- *Clustering-Based Distance Optimization;*
- *Pairwise Recommendation between Ranked Lists;*
- *Bidimensional Representation related to Context Spaces;*
- *Similarity among Ranked Lists;*
- *Bidimensional Representation related to Context Images.*

The hypothesis is validated by a large experimental evaluation presented along the thesis.

### 1.4 Goals and Contributions

The general objective of this thesis is to investigate new approaches for modelling and representing contextual information, aiming their use in *re-ranking* and *rank aggregation* tasks employed to improve the effectiveness of CBIR systems. The main goals and contributions of this thesis are:

- **Contextual Re-Ranking Methods:** creation and implementation of *five* image re-ranking methods. The methods and main concepts in which they are based on are described in the following:
  - *Distance Optimization Algorithm (DOA):* based on a clustering approach;

- *Pairwise Recommendation*: based on a recommendation process among ranked lists;
- *Contextual Spaces*: based on a bidimensional representation for kNN;
- *RL-Sim*: based on the similarity among ranked lists;
- *Contextual Re-Ranking Algorithm*: based on context images.

Each re-ranking method considers different concepts for exploiting contextual information. The use of different approaches contributes for analysing different, and complementary aspects encoded in relationships among images. However, although the methods represent and process the contextual information on different ways, all the methods are *unsupervised* and *iterative*. The general abstract model that represents all re-ranking methods proposed in this thesis is illustrated in Figure 1.1. The distance among images (represented by the distance matrix) is received as input. After executing the re-ranking algorithm a more effective distance matrix is produced. The process is repeated until a convergence criterion is reached (in the case of the DOA and Pairwise Recommendation approaches) or after a given number of iterations (in the case of Context Spaces, RL-Sim, and Contextual Re-Ranking approaches).

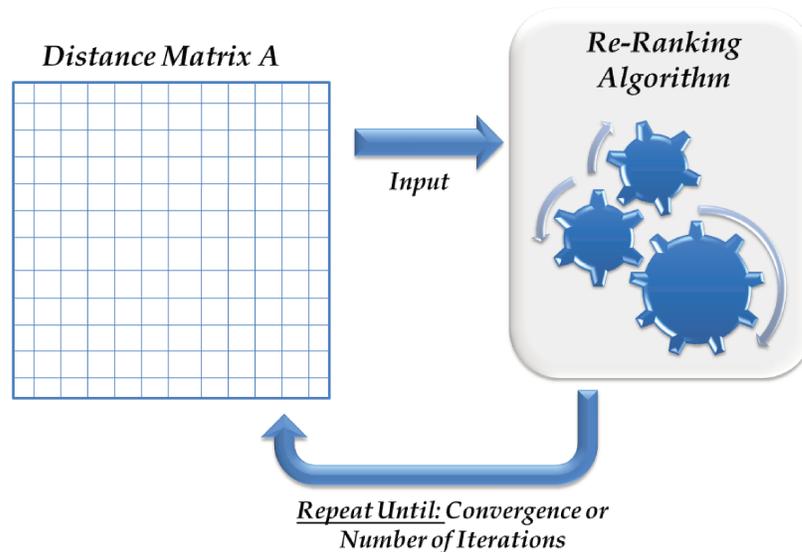


Figure 1.1: Abstract model of image re-ranking algorithms.

- **Contextual Rank Aggregation Methods:** creation and implementation of rank aggregation approaches that use the re-ranking methods to combine CBIR descriptors. Different rank aggregation approaches are proposed for each re-ranking method. We present these approaches based on three general models:

- *Rank Aggregation before Re-Ranking*: in this model, the distance matrix obtained by different CBIR descriptors are first combined using a multiplication approach and, in the following, the combined matrix is used as input for the re-ranking algorithm. Figure 1.2 illustrates this process. This approach is used by the Pairwise Recommendation, the Context Spaces, and the RL-Sim re-ranking algorithms.

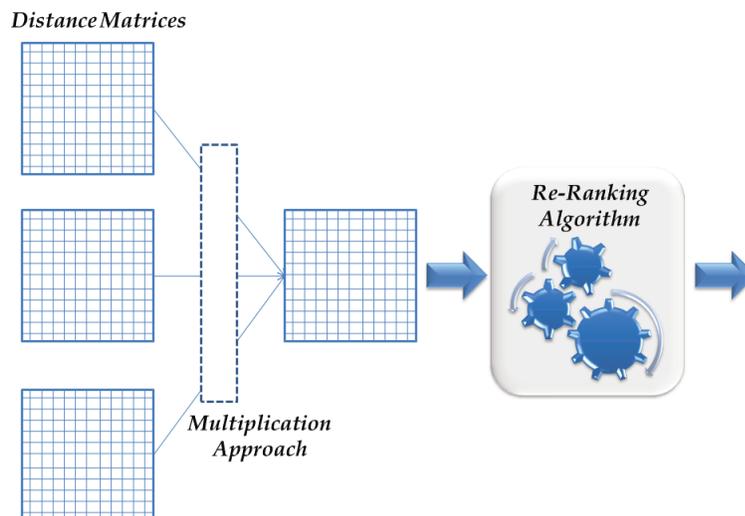


Figure 1.2: Rank Aggregation Model: Rank Aggregation before Re-Ranking.

- *Rank Aggregation after Re-Ranking*: another model consists in executing the re-ranking algorithm for each distance matrix and use the outputs to be combined by a rank aggregation algorithm. This process is illustrated in Figure 1.3. The Distance Optimization Algorithm uses this approach, considering the clusters as the output obtained from the re-ranking method.
  - *Rank Aggregation during Re-Ranking*: the last model is based on a symbiosis between re-ranking and rank aggregation. In fact, the rank aggregation is performed during the re-ranking execution, *e.g.*, the re-ranking algorithm itself is able to combine different distance matrices. Figure 1.4 illustrate this rank aggregation model. This approach is used by the Contextual Re-Ranking algorithm.
- **Multimodal Image Retrieval**: another contribution consists in the use of contextual rank aggregation methods for multimodal image retrieval, *e.g.*, considering textual and visual features.

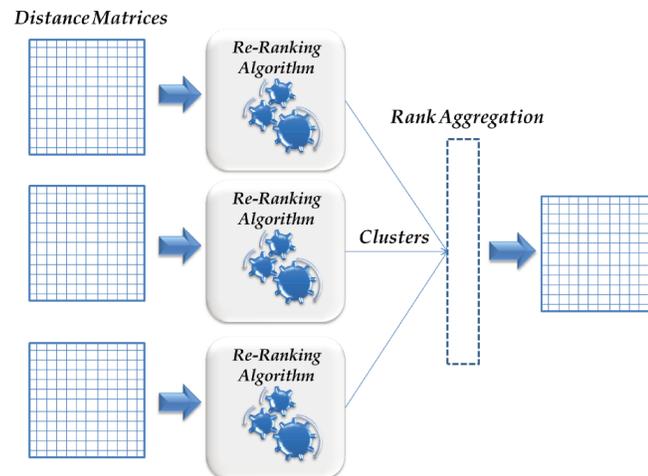


Figure 1.3: Rank Aggregation Model: Rank Aggregation after Re-Ranking.

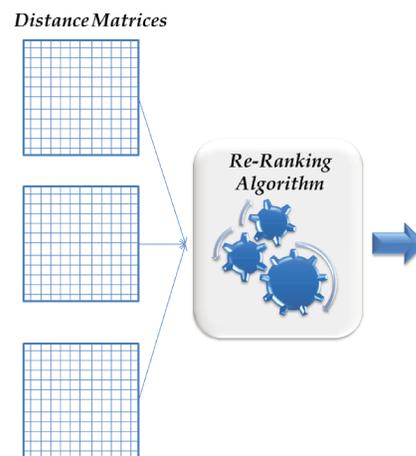


Figure 1.4: Rank Aggregation Model: Rank Aggregation during Re-Ranking.

- **Combining Re-Ranking and Rank Aggregation:** since the proposed methods exploit different and complementary aspects of contextual information, it is intuitive that they can be combined. In this way, another contribution of this thesis is the specification and implementation of approaches for combining the proposed re-ranking and rank aggregation methods.
- **Parallel Computation of Image Re-Ranking:** modelling and implementation of a parallel algorithm for effective image re-ranking computation on heterogeneous computing devices.

The presented work contributes for the use of contextual information considering different aspects of image retrieval systems, such as effectiveness and efficiency.

## 1.5 Organization

This thesis is organized according to the main contributions obtained in our research. Those contributions were published or submitted to conference and journals. In the following, we briefly describe the contents of each chapter indicating associated publication or submission:

- **Chapter 2 - Background and Related Work:** this chapter presents the basic concepts of CBIR, formal definitions for the re-ranking and rank aggregation problems, and discusses related work.
- **Chapter 3 - Experimental Protocol:** this chapter describes the datasets, the CBIR descriptors, and the effectiveness measures considered in the experimental evaluation of the proposed methods.
- **Chapter 4 - Distance Optimization Algorithm:** this chapter presents the re-ranking and rank aggregation approaches based on the Distance Optimization Algorithm (DOA). The content of this chapter can be found in articles published in the proceedings of the *International Conference on Computer Vision Theory and Applications* (VISAPP 2010) [68], in the proceedings of *Conference on Graphics, Patterns and Images* (SIBGRAPI 2010) [66], and in the *Journal of Visual Languages and Computing* [69].
- **Chapter 5 - Pairwise Recommendation:** this chapter presents the Pairwise Recommendation approach for re-ranking and rank aggregation. Its content can be found in an article accepted for publication in the journal *Information Sciences* [64].
- **Chapter 6 - Contextual Spaces:** this chapter discusses the use of Contextual Spaces for re-ranking and rank aggregation tasks. The content of this chapter can be found in papers published in the proceedings of the *International Conference on Multimedia Retrieval* (ICMR 2011) [71] and accepted for publication in the journal *Multimedia Tools and Applications* [75].
- **Chapter 7 - RL-Sim Re-Ranking:** this chapter presents the RL-Sim algorithm for re-ranking and rank aggregation tasks. The content of this chapter can be found in papers published in the proceedings of the *International Conference on Computer Analysis of Images and Patterns* (CAIP 2011) [72] and submitted to journal *Pattern Recognition* [65].
- **Chapter 8 - Contextual Re-Ranking:** this chapter describes the Contextual Re-Ranking and the Contextual Rank Aggregation algorithms. The content of

this chapter can be found in papers published in the proceedings of *Iberoamerican Congress on Pattern Recognition* (CIARP 2010) [67], in the proceedings of the *International Conference on Image Processing* (ICIP 2011) [70] and accepted for publication in the *International Journal of Multimedia Information Retrieval* [63].

- **Chapter 9 - Efficient Image Re-Ranking Computation on GPUs:** this chapter describes the design of a parallel algorithm for efficient image re-ranking computation. Its content can be found in a paper accepted for publication in the *International Symposium on Parallel and Distributed Processing* (ISPA 2012) [74].
- **Chapter 10 - Combining Re-Ranking and Rank Aggregation Methods:** this chapter describes conducted experiments aiming at comparing three different approaches for combining re-ranking and rank aggregation methods. The content of this chapter can be found in a paper accepted for publication in the *Iberoamerican Congress on Pattern Recognition* (CIARP 2012) [73].
- **Chapter 11 - Comparing Re-Ranking and Rank Aggregation Methods:** this chapter presents a comparison among the proposed re-ranking and rank aggregation methods with each other and with state-of-the-art methods proposed in the literature.
- **Chapter 12 - Conclusions:** this chapter discusses conclusions and presents possible extensions to be addressed in future work.

Figure 1.5 illustrates the overall organization of this thesis, considering the main concepts and their relationships. This figure also shows the main contributions and associated publications. The colors of this figure aims at organizing the meanings of each concept: in red, the main subjects related to the research; in green, the main contributions of this thesis; in blue, the related concepts used to address the contributions; and in yellow, the associated publications.

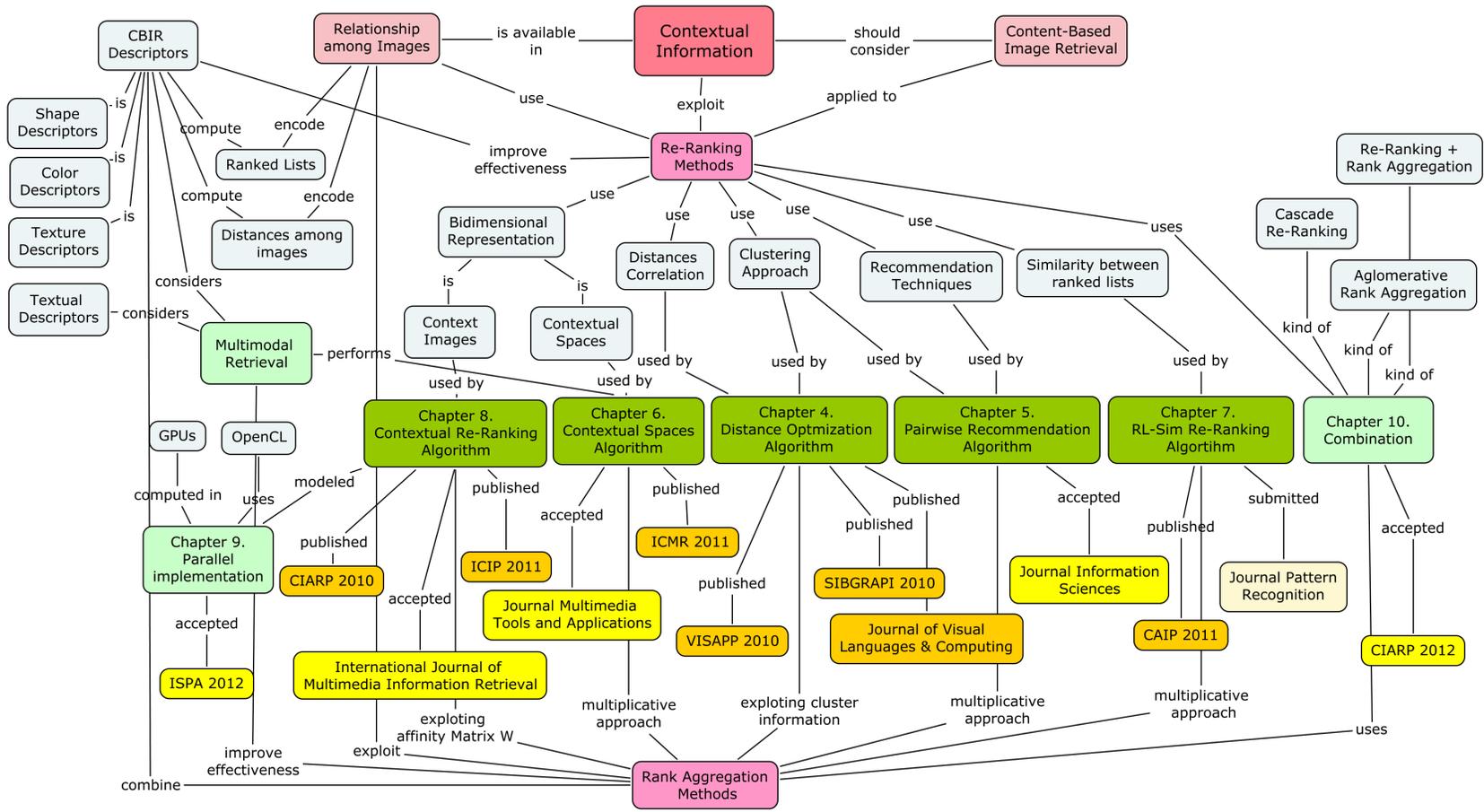


Figure 1.5: Thesis organization, main concepts, and obtained publications.

# Chapter 2

## Background and Related Work

### 2.1 Content-Based Image Retrieval

Content-based image retrieval (CBIR) can be broadly defined as any technology that in principle helps to organize digital picture archives by their visual content. By this definition, anything ranging from an image similarity function to a robust content-based image annotation engine can be considered as a component of a CBIR system. Considering the research opportunities related to the specification and implementation of those systems, researchers from different fields, such as, computer vision, image processing, machine learning, information retrieval, human-computer interaction, database systems among others are contributing and becoming part of the CBIR community [21].

#### 2.1.1 Typical Architecture

Basically, a CBIR system aims at meeting the user needs expressed in a query specification (*e.g.*, by defining a query image as input). Figure 2.1 shows a typical architecture of a content-based image retrieval system [16]. Two main functionalities are supported: data insertion and query processing. The *data insertion* subsystem is responsible for extracting appropriate features from images and storing them into the image database. In general, this process is performed off-line. The query processing is organized as follows: the *interface* allows a user to specify a query by means of a query pattern and to visualize the retrieved most similar images. The *query-processing* module extracts a feature vector from a query pattern and uses a distance function (such as the Euclidean distance) to evaluate the similarity between the query image and the database images. Next, it ranks the database images in a decreasing order of similarity to the query pattern and forwards the most similar images to the interface module. A formal definition of the image retrieval model adopted in this work is presented in next section.

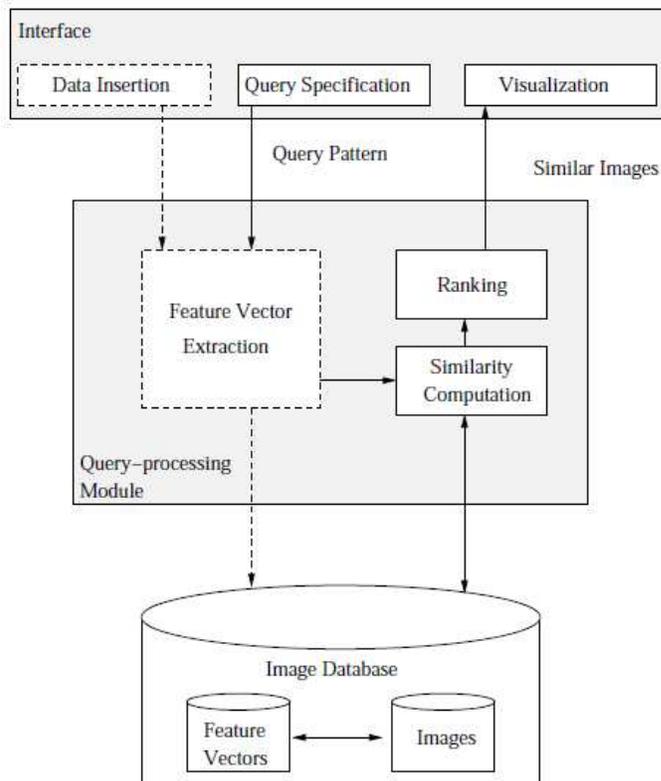


Figure 2.1: Typical architecture of a content-based image retrieval system [16].

### 2.1.2 Image Retrieval Model

Let  $\mathcal{C}=\{img_1, img_2, \dots, img_N\}$  be an *image collection*. Let  $\mathcal{D}$  be an *image descriptor* which can be defined [16] as a tuple  $(\epsilon, \rho)$ , where:

- $\epsilon: \hat{I} \rightarrow \mathbb{R}^n$  is a function, which extracts a feature vector  $v_{\hat{I}} \in \mathbb{R}^n$  from an image  $\hat{I}$ .
- $\rho: \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$  is a distance function that computes the distance between two images as the distance between their corresponding feature vectors.

In order to obtain the distance between two images  $img_i$  and  $img_j$ , it is necessary to compute the value of  $\rho(\epsilon(img_i), \epsilon(img_j))$ . For simplicity and readability purposes, we use the notation  $\rho(img_i, img_j)$  along this thesis.

The distance  $\rho(img_i, img_j)$  among all images  $img_i, img_j \in \mathcal{C}$  can be computed to obtain an  $N \times N$  distance matrix  $A$ , such that  $A[i, j] = \rho(img_i, img_j)$ .

Given a query image  $img_q$ , we can compute a ranked list  $R_q$  in response to the query, based on the distance matrix  $A$ . The ranked list  $R_q=(img_1, img_2, \dots, img_N)$  can be defined as a permutation of the collection  $\mathcal{C}$ . A permutation  $\sigma_q$  is as a bijection from the

collection  $\mathcal{C}$  onto the set  $[N] = \{1, 2, \dots, N\}$ , where  $N$  is the cardinality  $|\mathcal{C}|$  of collection  $\mathcal{C}$ . For a permutation  $\sigma_q$ , we interpret  $\sigma_q(i)$  as the position (or rank) of image  $img_i$  in the ranked list  $R_q$ . Therefore, we can say that, if  $img_x$  is ranked before  $img_y$ , that is  $\sigma_q(x) < \sigma_q(y)$ , then  $\rho(img_q, img_x) \leq \rho(img_q, img_y)$ .

We also can take each image  $img_i \in \mathcal{C}$  as a query image  $img_q$ , in order to obtain a set  $\mathcal{R} = \{R_1, R_2, \dots, R_N\}$  of ranked lists for each image of collection  $\mathcal{C}$ .

## 2.2 Image Re-Ranking

The definition of appropriate distance measures plays a key role in many multimedia applications, including classification, clustering, and retrieval tasks. For example, choosing a good distance measure is often critical to building an effective content-based image retrieval (CBIR) system. In general, aiming at retrieving the most similar images to a query image, CBIR systems compute a predefined distance measure between the query image and an image in database. Traditional distance measures, as the Euclidean distance, consider the pairwise similarity between any two images. In many situations, these approaches fail to return satisfactory results, mainly due to the well-known *semantic gap* problem [39].

Recently, many studies have demonstrated, both empirically and theoretically, that a learned measure can significantly improve the effectiveness in classification, clustering, and retrieval [112]. In special for CBIR systems, there has been considerable research on improving the distance measures [4, 10, 11, 38, 39, 42, 45, 49, 62, 77, 84, 102, 109, 113–115, 120, 121]. Promising results have been obtained considering several approaches and techniques. One of these approaches is called *image re-ranking*, which is one of the main focuses of this thesis.

Section 2.2.1 presents a formal definition for the image re-ranking task. Section 2.2.2 discusses related work, considering the image re-ranking problem, distance metric learning, and correlated areas. For organization purposes, we consider a comprehensive classification, depending on the availability of the training examples. We categorize the approaches into three main categories: supervised, semi-supervised, and unsupervised algorithms.

### 2.2.1 Image Re-Ranking Definition

Re-ranking can be broadly defined as a process of refining the search results: the re-ranking methods take an initial ranking and aggregate some information for improving the effectiveness of the retrieval process. An image re-ranking algorithm takes as input the distance matrix  $A$  and the set of ranked lists  $\mathcal{R}$  for computing a new and more effective

distance matrix  $\hat{A}$ . Therefore, distances among all images contained in the matrix  $A$  are redefined by a more effective distance measure. A re-ranking method that considers relationships among all images in a collection can be represented by the function  $f_r$ :

$$\hat{A} = f_r(A, \mathcal{R}). \quad (2.1)$$

Given the new distance matrix  $\hat{A}$ , a new set  $\hat{\mathcal{R}}$  can be obtained.  $\hat{\mathcal{R}}$  contains the new ranking positions of all collection images, that is, the collection images are re-ranked. Note that the main aspect of  $f_r$  consists in exploiting all relationships encoded in both  $A$  and  $\mathcal{R}$ . The definition of function  $f_r$  applied to CBIR scenario is similar to the concept of *global ranking* [80] used in information retrieval domain. The re-ranking algorithms proposed in this thesis consist in different implementations of the function  $f_r$ .

## 2.2.2 Re-Ranking Approaches

This section discusses related work to the image re-ranking problem. First, concepts and approaches related to distance metric learning are presented. The approaches are grouped into three categories depending on the availability of the training examples: supervised, semi-supervised, and unsupervised algorithms. In the following, we discuss unsupervised approaches used by CBIR systems and re-ranking methods.

### Supervised Approaches

In *supervised learning*, training samples are available for the task of inferring a function. Each training sample is a pair consisting of an input object and a desired output value. As defined in [125], let the domain of object instances be  $\mathcal{X}$ , and the domain of labels be  $\mathcal{Y}$ . Let  $P(x, y)$  be an (*unknown*) joint probability distribution on instances and labels  $\mathcal{X} \times \mathcal{Y}$ . Given a training sample  $\{(x_i, y_i)\}_{i=1}^n$ , supervised learning trains a function  $f : \mathcal{X} \mapsto \mathcal{Y}$ , with the goal that  $f(x)$  predicts the true label  $y$  on future data  $x$ .

*Supervised distance metric learning* approaches attempt to learn metrics that keep all the data points within the same classes close, and place all the data points from different classes far apart. In distance metric learning, the label information is usually specified in the form of pairwise constraints on the data: (1) equivalence constraints, which state that the given pair are semantically similar and therefore should be close in the learned metric; and (2) inequivalence constraints, which indicate that the given points are semantically dissimilar and should not be close in the learned metric [112].

In [109], a supervised distance metric learning with application to clustering is presented. It considers the situation where a user indicates that certain points in an input space are considered to be “*similar*”. The algorithm aims at learning a distance metric

that respects these relationships, *i.e.*, one that computes small distances between similar pairs of objects. In other words, given examples of similar pairs of points in  $\mathbb{R}^n$ , it learns a distance metric over  $\mathbb{R}^n$  that respects these relationships when improving distance measures among points. The Continuous Conditional Random Fields (CRF) has been proposed in [80] for the learning in global ranking tasks. This model is defined as a conditional probability distribution over ranking scores of objects. It represents the content information of objects as well as the relation information between objects, necessary for global ranking.

In the CBIR domain, a guideline to learn a robust distance measure for similarity estimation is presented in [117]. A more effective distance measure is learned by training with different distance measures on each feature element and by selecting iteratively the most important feature elements for estimation of similarity. In [49], a rank-based distance metric learning approach is presented. The goal is to learn a distance metric from a number of training samples with side information, *i.e.*, relevance judgments, based on rankings. The approach compares the distances of pairwise constraints that are generated by the same query, aiming at weighting different features.

An approach based on the “*learning-to-rank*” paradigm is presented in [28]. Different supervised learning algorithms (Support Vector Machines, Genetic Programming, and Association Rules) are used to effectively combine multiple CBIR descriptors in order to improve ranking performance. A set of query images is provided as input to the learning algorithms. In addition, associated with each query image, a set of sample images is provided. That set represents the corresponding similarities to the query image. The relevance of an image to the query image is also informed as input (*e.g.*, an image is relevant if it is truly similar to the query image, otherwise it is irrelevant). This information is used for training, so that the learning algorithms produce a ranking function that maps similarities to the level of relevance of collection images for defined query images. When a new query image is given, the relevance of the returned images is estimated according to the learned function, by using supervised algorithms.

### Semi-Supervised Approaches

As the name suggests, *semi-supervised learning* is somewhere between unsupervised and supervised learning. In fact, most semi-supervised learning approaches are based on extending either unsupervised or supervised learning to include additional information from the other learning paradigm [125]. Semi-supervised learning refers to the use of both labeled and unlabeled data for training. The training data consists of both  $l$  labeled instances  $\{(x_i, y_i)\}_{i=1}^l$  and  $u$  unlabeled instances  $\{x_j\}_{j=l+1}^{l+u}$ . It contrasts supervised learning (data all labeled) or unsupervised learning (data all unlabeled) [122, 125]. While labeled data  $(x, y)$  is difficult to collect, unlabeled data  $x$  are available in large quantity and are

easy to obtain.

The semi-supervised learning problem considers the prior assumption of consistency, which means: (1) nearby points are likely to have the same label; and (2) points on the same structure (usually referred to as a manifold or a cluster) are likely to have the same label [120].

Semi-supervised methods have attracted great attention in the past few years. In [123, 124], labeled and unlabeled data are represented as vertices in a weighted graph, with edge weights encoding information about the similarity between instances. The learning problem is modeled by a Gaussian random field on this graph, where the mean of the field is characterized in terms of harmonic functions. In [123], the semi-supervised “*Label Propagation*” algorithm is formulated as a process of propagation on a graph, where node labels are propagated to neighbor nodes according to their proximity. In this process the labels are fixed on the unlabeled data. Therefore, labeled data act like sources that disseminate labels through unlabeled data. The Label Propagation algorithm has also inspired unsupervised approaches on shape retrieval domain [113].

In [38, 39], a semi-supervised distance metric learning approach is presented with focus on CBIR applications. It aims at learning effective distance metrics by training data and using unlabeled data when log data are limited and noisy. The training data is obtained by exploring historical relevance feedback log data. The systems considers an accumulate feedback information collected in multiple image retrieval sessions possibly conducted by multiple users for different search targets. This paradigm of utilizing CBIR log data in an image retrieval task is referred as “*Collaborative Image Retrieval*” (*CIR*). The learning problem is formulated into a convex optimization task.

The semi-supervised approach proposed in [37] aims at learning distance functions by training binary classifiers with margins, where the classifiers are defined over the product space of pairs of images. The classifiers are first trained and, in the following, they can distinguish between pairs in which the images are from the same class and pairs which contain images from different classes. The distance learning method combines boosting hypotheses over the product space with a weak learner based on partitioning the original feature space. It allows incorporate unlabeled data into the training process.

## Transductive Learning

There are two variations of semi-supervised learning approaches, namely inductive and transductive semi-supervised learning. In supervised classification, the training sample is fully labeled, so the interest is concerned with the performance of developed algorithms on future test data. However, in semi-supervised classification the training sample contains some unlabeled data. Therefore, there are two distinct goals: one is to predict the labels on future test data; and other is to predict the labels on the unlabeled instances in the

training sample. We may call the first one as inductive semi-supervised learning, and the latter transductive learning [122]. Note that, although transductive learning is more commonly used for semi-supervised learning approaches, it designates a behavior opposed to traditional inductive learning.

The learning by transduction approach was first proposed in [32]. In this problem, the interest relies on the classification of a particular example rather than a general rule (function) for classifying future examples, as in supervised inductive learning. An example of a transductive learning task is relevance feedback in information retrieval. In relevance feedback, users can provide positive and negative examples for the kinds of objects in which they are interested. These objects are the training examples, while the rest of the collection is the test set. The goal is to generalize from the training examples and find remaining documents in the collection that match the users information need [18, 20, 30, 44]. The main difference between the transductive setting and the regular inductive setting consists in that the learner can observe the examples in the test set and potentially exploit data structure in their distribution. In this context, the term has been used for designating unsupervised approaches [11, 113].

In [89], an approach that aims at turning transductive and standard supervised learning algorithms into semi-supervised learners is presented. It constructs a family of data-dependent norms that allow to capture the structure and reflect the underlying geometry of the data.

## Unsupervised Approaches

In *unsupervised learning* approach, the “*learning*” method receives only the domain of object instances  $\mathcal{X}$ , that is no training labeled data is available. Since labeled data usually requires very expensive human labor, whereas unlabeled data is far easier to obtain, unsupervised learning represents a very attractive solution in many scenarios.

As pointed by [33], although it may seem somewhat paradoxical to imagine what a method could possibly learn, given that it does not get any feedback from its environment, it is possible to develop frameworks for unsupervised learning based on the notion that the machines goal is to build representations of the input space that can be used for decision making and predicting future inputs. In a sense, “unsupervised learning can be thought of as finding patterns in the data above and beyond what would be considered pure unstructured noise” [33]. Two classic examples of unsupervised learning are *clustering* and *dimensionality reduction*.

When dealing with data in high dimensions, a challenging problem is how to reduce the complexity of a data set preserving information that is important for understanding the data structure itself. That is also valid for performing tasks such as clustering, classification, and regression [47]. The *dimensionality reduction* term designates methods that

aim at finding meaningful low-dimensional structures hidden in their high-dimensional observations [96].

The classical techniques for dimensionality reduction, principal component analysis (PCA) and multidimensional scaling (MDS), aim at discovering the true structure of data lying on or near a linear subspace of the high-dimensional input space. PCA aims at finding a low-dimensional embedding of the data points that best preserves their variance as measured in the high-dimensional input space. Classical MDS finds an embedding that preserves the interpoint distances, equivalent to PCA, when those distances are Euclidean [96].

Some approaches rely on modifying not the measure itself, but the space in which the measure is computed [77]. This includes ISOMAP [96] and Local Linear Embedding (LLE) [82]. In ISOMAP [96], the geodesic distance between faraway points is estimated given only input space distances. For neighboring points, *input-space* distance gives a good approximation to geodesic distance. Considering faraway points, geodesic distance can be approximated by adding up a sequence of “short hops” between neighboring points. These approximations are computed efficiently by finding shortest paths in a graph with edges connecting neighboring data points [96]. In Locally Linear Embedding (LLE) [82], an unsupervised learning algorithm computes low-dimensional, neighborhood-preserving embeddings of high-dimensional inputs. LLE maps the inputs into a single global coordinate system of lower dimensionality. By exploiting the local symmetries of linear reconstructions, LLE can learn the global structure of non-linear manifolds.

There are evidences [47] that nonlinear dimensionality reduction, clustering, and data set parameterization can be solved within the same framework. The main idea is to define a system of coordinates with an explicit metric that reflects the connectivity of a given data set, using Markov random walk on the data. Intrinsic geometry means a set of rules that describe the relationship between the objects in the data set without reference to structures outside of them. In this case, intrinsic geometry is defined by the connectivity of the data points in a diffusion process.

Non-linear dimensionality reduction methods that consider geometrically structures of datasets [47, 82, 96] are also commonly referred as *manifold learning*. In [119], an extensive applications of these methods are presented.

Clustering is another classic example of unsupervised learning. Basically, clustering is the task of assigning a set of objects into groups (*clusters*), so that the objects in the same cluster are more similar to each other than to those objects in other clusters. In this way, it also depends on the accuracy of the distance measure used to assess the similarity between two objects. In [36], an unsupervised approach for metric learning in the context of clustering was proposed. It performs transformations of data which give clean and well-separated clusters, where clean clusters are those for which membership scores can

be predicted. In [11], an unsupervised clustering approach was proposed, based on graph transduction. Following the concepts of semi-supervised methods, the algorithm focuses on the problem that uses unlabeled data for learning and improving the clustering task.

Unsupervised approaches were proposed for improving effectiveness of information retrieval tasks. A definition for the term “*global ranking*” was proposed in [80]. Basically, a global ranking approach considers that relations always exist between objects and it is better to define the ranking model as a function of all the objects to be ranked. In [43], an approach that explores information of users clicks was proposed for re-ranking in the web search scenario. Inter-documents similarity are considered in [23] and a clustering approach is applied for regularizing retrieval scores. In [111], a semi-supervised label propagation algorithm [123] was used for re-ranking documents in information retrieval applications.

### Unsupervised Learning in CBIR systems

In general, traditional CBIR systems perform only pairwise image analysis, that is, they compute similarity (or distance) measures considering only pairs of images, ignoring the rich information encoded in the relations of several images. However, in recent years, several CBIR approaches [4, 42, 45, 62, 77, 84, 102, 113–115, 121] have been proposed aiming at improving the effectiveness of retrieval tasks replacing pairwise similarities by more global affinities that also consider the relation among the database objects [115]. Although using a very diverse taxonomy (re-ranking [62, 84], graph transduction [4, 113], diffusion process [114], affinity learning [115], contextual similarity/dissimilarity measures [42, 77, 102]), these *post-processing* methods have in common the fact of all approaches propose to improve the effectiveness of retrieval tasks by exploiting the information about the relationships among database objects on an *unsupervised* way (with no training data). Another important common point consists in the iterative behaviour adopted by various methods [42, 113, 114].

In [113], a *graph-based transductive learning algorithm* is proposed for shape retrieval tasks. It learns a better metric through graph transduction by propagating the model through existing shapes, in a way similar to computing geodesics in dataset manifold. The method does not require learning the shape manifold explicitly and it does not require knowing class labels of existing shapes. The better metric is learned by collectively propagating the similarity measures to the query shape and between the existing shapes through graph transduction. Although inspired by label propagation algorithm [123], which is semisupervised, the shape retrieval is treated as an unsupervised problem [113].

In [114], a *locally constrained diffusion process* is proposed for shape retrieval systems. The work observes that, since differences between shapes in the same class can be very

large and differences between shapes in different classes can be very small, no pairwise shape comparison can describe shape dissimilarity correctly. The distance between two shapes can be correctly described only if it is considered in the context of other shapes similar to them. The influence of other shapes is propagated as a diffusion process on a graph formed by a given set of shapes. The weights of graph edges are defined by applying a Gaussian to the shape distance. A reversible Markov chain based on the graph is constructed and used to propagate the influence of shapes. Another approach based on propagating the similarity information in a weighted graph is proposed in [115] as *affinity learning*. Instead of propagating the similarity information in the original graph, it uses a tensor product graph (TPG) obtained by the tensor product of the original graph with itself.

Graphs are also used by other approaches. In [45], the underlying structure of the shape manifold is estimated from the shape similarity scores between all the shapes within a database. A modified mutual kNN graph is proposed as the underlying representation used for shape retrieval. A shortest path propagation algorithm is proposed in [102], which is a graph-based algorithm for shape/object retrieval. Given a query object and a target database object, it explicitly finds the shortest path between them in the distance manifold of the database objects. Then a new distance measure is learned based on the shortest path to replace the original distance measure.

Beside graph methods, *context* is a term frequently used for designating post-processing methods that consider relationships among images. In general interactive applications, the use of *context* can play an important role. Context can be broadly defined as all information about the whole situation relevant to an application and its set of users [1]. In CBIR systems, it means that, when humans have to judge the similarity between two images, they always do so in a given context, *i.e.*, they do not only consider the two objects to be compared [77].

In [42], a *contextual dissimilarity measure* is introduced, aiming at improving the accuracy of image searches based on bag-of-features. The proposed measure takes into account the local distribution of the vectors and estimates distance updates by modifying the neighborhood structure. The dissimilarity measure improves the symmetry of the  $k$ -neighborhood relationship by iteratively regularizing the average distance of each vector to its neighborhood. The method performs a global analysis of properties in small overlapping neighborhoods, resembling methods for non-linear dimensionality reduction, inspired by ISOMAP [96] and LLE [82].

In [77], a *family of contextual measures* is proposed. The similarity between two distributions is measured in the context of a third distribution. These contextual measures are then used to the image retrieval problem. The context is estimated from the neighbors of a query. Using different contexts, and especially contexts at multiple scales (*i.e.*, broad

and narrow contexts), provides different views on the same problem and combining the different views can improve retrieval accuracy.

Clustering approaches are also used by re-ranking methods that exploit contextual information in CBIR domain. A re-ranking framework for CBIR systems based on contextual dissimilarity measures is proposed in [84]. The contexts are modeled using a clustering algorithm to group similar images from the ranked list. In [62], a re-ranking algorithm using post-retrieval clustering for CBIR is proposed. In the first step, images are retrieved using visual features such as color histogram. Next, the retrieved images are analyzed using hierarchical agglomerative clustering methods and the rank of the results is adjusted according to the distance of a cluster to a query.

Although there are a significant number of approaches aiming at exploiting the relationships among images for improving the effectiveness of CBIR systems, this area is relatively new and many research challenges are still open. Several methods [102, 113, 114] are based on graph or matrices multiplication approaches, which require high computational efforts. In other cases [45, 62, 113, 114] experimental evaluation considered only one type of visual property (*e.g.*, shape or color). Efficiency issues are not addressed in the majority of works and results of different methods are not combined. In this thesis, we addressed the image re-ranking problem in a general way: proposing different re-ranking approaches and techniques for combining them, conducting a large experimental evaluation, and using parallel computing for efficient re-ranking computation.

## 2.3 Rank Aggregation

This section presents a discussion about *rank aggregation* methods, presenting a formal definition for the rank aggregation problem and discussing related work.

### 2.3.1 Rank Aggregation Definition

Basically, rank aggregation approaches aim at combining different rankings in order to obtain a more accurate one. Let  $\mathcal{C}$  be an image collection and let  $\mathcal{D} = \{D_1, D_2, \dots, D_m\}$  be a set of  $m$  image descriptors. The set of descriptors  $\mathcal{D}$  can be used for computing a set of distances matrices  $\mathcal{A} = \{A_1, A_2, \dots, A_m\}$ . As discussed in previous subsection, for each distance matrix  $A_i \in \mathcal{A}$ , a set of ranked lists  $\mathcal{R}_i = \{R_1, R_2, \dots, R_N\}$  can be computed. Let  $\mathcal{R}_{\mathcal{A}} = \{\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_m\}$  be a set of sets of ranked lists (one set  $\mathcal{R}_i$  for each matrix  $A_i$ ), the objective of rank aggregation methods that consider relationships among images is to use the sets  $\mathcal{A}$  and  $\mathcal{R}_{\mathcal{A}}$  as input for computing a new distance matrix  $\hat{A}_{\mathcal{C}}$ :

$$\hat{A}_c = f_a(\mathcal{A}, \mathcal{R}_A). \quad (2.2)$$

Based on the combined distance matrix  $\hat{A}_c$ , a new set of ranked lists can be computed. The rank aggregation algorithms proposed in this thesis consist in different implementations of the function  $f_a$ .

### 2.3.2 Rank Aggregation Approaches

Another approach for improving CBIR systems consists in using *rank aggregation* techniques. Different CBIR descriptors produce different rankings. Therefore, it is intuitive that different descriptors may provide different but complementary information about images. The main goal of rank aggregation approaches is to combine different rankings in order to obtain a more accurate one. Recently, rank aggregation is being employed in many new applications [8, 55], such as document filtering, spam webpage detection, meta-search, word association finding, multiple search, and similarity search.

More precisely, rank aggregation can be seen as the task of finding a permutation that minimizes the Kendall-tau distance to the input rankings, where the Kendall-tau distance is defined as the sum over all input rankings of the number of pairs of elements that are in a different order in the input ranking than in the output ranking [83]. If the input rankings are permutations, this problem is known as the Kemeny rank aggregation problem [83]. The best combinations occur when both systems being combined have good performance, although it is possible to get improvement when only one of the systems has good performance [15]. This observation is consistent with the statement that the combination with the lowest error occurs when the classifiers are independent and non-correlated [15].

Unsupervised approaches considering *retrieval scores* [31] or *rank positions* [13] have been widely used in information retrieval tasks. In the CBIR domain, the use of machine learning techniques have been employed aiming at accurately ranking the returned images. One common approach is to use learning methods to combine information coming from different descriptors. These approaches include techniques like Support Vector Machines [28], Genetic Programming [28] and others.

Recently, learning-to-rank approaches are being considered [28]. Their objective is to use machine learning techniques to combine different CBIR descriptors. Rank aggregation also can be thought as an unsupervised regression, in which the goal is to find an aggregate ranking that minimizes the distance to each of the given ranked lists [86]. It also can be seen as the problem of finding a ranking of a set of elements that is “closest to” a given set of input rankings of the elements [25, 26, 83].

Rank aggregation techniques can also be exploited for multimedia retrieval, in special

in multimedia objects that are composed by different media such as text and image. There has been an explosion of such type of digital content in the last years [8, 12]. Most of the techniques developed in that context fall in three different categories: early fusion, late fusion, and transmedia fusion. The early fusion approach consists in representing the multimedia objects in a multimodal feature space designed via a joint model that attempts to map image-based features with text-based features. On the contrary, late fusion and transmedia fusion strategies consists in running the visual and textual experts independently. Late fusion techniques mainly consist in merging the monomedia similarity information by means of aggregation functions. In transmedia approaches, the main idea is to first use one of the modalities (say image) to gather relevant objects and then to switch to the other modality (text representations) with the aim at aggregating their results [12].

Different approaches and applications, whether supervised or unsupervised, consider only scores or positions for producing new rankings. In this work, we exploit not only information computed by different descriptors, but also the relationship among images. We propose different unsupervised approaches for combining information from different descriptors and our re-ranking algorithm for exploiting contextual information.

## 2.4 Notation

This section summarizes the notation of the main symbols (*concepts*) used along this thesis. Table 2.1 presents the symbols used along the thesis and their respective meanings.

Table 2.1: Notation used along this thesis

Symbol	Meaning
$\mathcal{C}$	Image collection.
$\mathcal{D}$	Image descriptor.
$\rho$	Image descriptor distance function.
$N$	Cardinality of collection.
$A$	Initial distance matrix.
$\mathcal{R}$	Initial set of ranked lists.
$\hat{A}$	Distance matrix after re-ranking.
$\hat{\mathcal{R}}$	Set of ranked lists after re-ranking.



# Chapter 3

## Experimental Protocol

This chapter aims at describing the *experimental protocol* used to evaluate the re-ranking and rank aggregation methods proposed in our work. Section 3.1 discusses the datasets, descriptors, and measures used for evaluation. Section 3.2 addresses the same issues, but now considering the multimodal retrieval problem. All experiments were conducted considering all images in the collections as query images. Results presented in this thesis represent the average scores.

### 3.1 Content-Based Image Retrieval

We considered four different datasets, and twelve image descriptors, considering different visual properties, involving shape, color, and texture descriptors. In general, post-processing methods [45, 62, 113, 114] have been evaluated considering only one type of visual property (usually, either color or shape). Methods proposed in [45, 113, 114] used shape descriptors, while the method proposed in [62] used a color descriptor.

#### 3.1.1 Descriptors

- **Shape:** We evaluate the use of our methods with six shape descriptors: Segment Saliences (SS) [17], Beam Angle Statistics (BAS) [2], Inner Distance Shape Context (IDSC) [52], Contour Features Descriptor (CFD) [68], Aspect Shape Context (ASC) [53], and Articulation-Invariant Representation (AIR) [35].
- **Color:** We evaluate our methods for three color descriptors: Border/Interior Pixel Classification (BIC) [90], Auto Color Correlograms (ACC) [41], and Global Color Histogram (GCH) [93].

- **Texture:** The experiments consider three well-known texture descriptors: Local Binary Patterns (LBP) [60], Color Co-Occurrence Matrix (CCOM) [46], and Local Activity Spectrum (LAS) [94].

Further details on used image descriptors can be found in [76].

### 3.1.2 Datasets

- **MPEG-7:** The MPEG-7 dataset [48] is a well-known shape collection, commonly used for shape descriptors and post-processing methods evaluation and comparison. It is composed by 1400 shapes divided into 70 classes of 20 images each. The size of images range from  $(50 \times 48)$  to  $(526 \times 408)$  pixels. Figure 3.1 presents some examples of images of the MPEG-7 dataset.

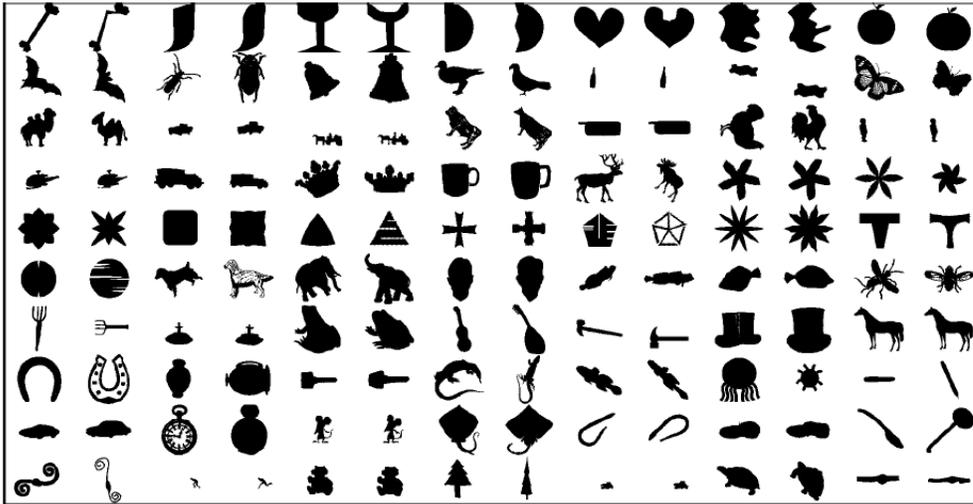


Figure 3.1: Examples of MPEG-7 dataset shapes.

- **Kimia:** We also present experimental results on the Kimia dataset [87]. This dataset contains 99 shapes grouped into nine classes. The retrieval results are summarized as the number of shapes from the same class among the first top 1 to 10 shapes, where the best possible result for each of them is 99 (this score is referenced along the thesis as Kimia score). Shapes of Kimia dataset shapes are illustrated in Figure 3.2.
- **Brodatz:** We used the Brodatz [7] dataset, a popular dataset for texture descriptors evaluation. The Brodatz dataset is composed of 111 different textures of size  $(512$

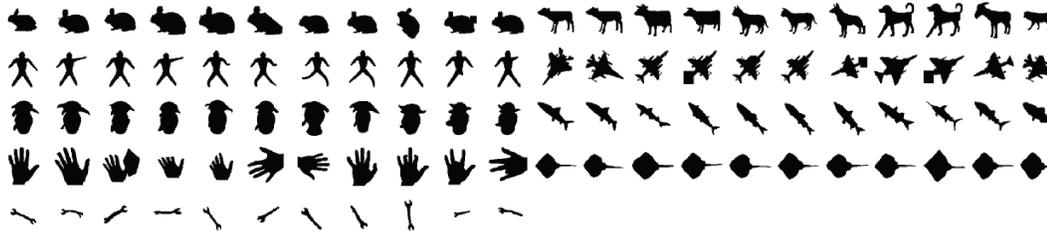


Figure 3.2: Examples of Kimia dataset shapes.

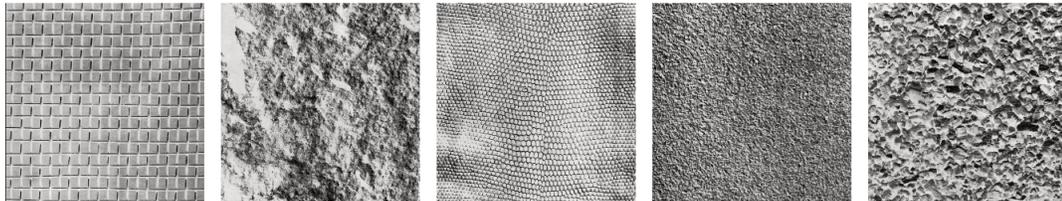


Figure 3.3: Examples of Brodatz [7] texture images.

$\times 512$ ) pixels. Each texture is divided into 16 blocks ( $128 \times 128$ ) pixels of non-overlapping sub images, such that 1776 images are considered. Some examples of textures are presented in Figure 3.3.

- **Soccer Dataset:** that dataset was used in [100] and composed by images from 7 soccer teams, containing 40 images per class. The size of images range from ( $198 \times 148$ ) to ( $537 \times 672$ ) pixels. Some samples of this dataset are illustrated in Figure 3.4.

### 3.1.3 Measures

- **Mean Average Precision (MAP):** The measure of effectiveness most commonly used in recent years has been *Average Precision (AP)*. The average precision measure combines recall and precision to give a single-value measure. Average precision is calculated by taking the set of ranks at which the relevant items occur, comput-



Figure 3.4: Examples of Soccer dataset [100] images.

ing the precision at those positions, and then averaging the set of precision values obtained [57]. More formally, let  $q$  be a query item and let  $N_r$  be the number of relevant items in a collection for a given query  $q$ . Let  $\langle r_i | i = 1, 2, \dots, d \rangle$  be a ranked relevance vector to depth  $d$ , where  $r_i$  indicates the relevance of the  $i$ th ranked document scored as either 0 (not relevant) or 1 (relevant), the AP is defined as follows:

$$AP = \frac{1}{N_r} \sum_{i=1}^d \left( \frac{r_i}{i} \sum_{j=1}^i r_j \right). \quad (3.1)$$

The average precision across a series of queries can be averaged, which defines the *Mean Average Precision* (MAP) measure. Let  $Q$  be the number of queries, the MAP is defined as follows:

$$MAP = \frac{\sum_{l=1}^Q AP(q_l)}{Q}. \quad (3.2)$$

Furthermore MAP also approximates the average area under the precision  $\times$  recall curve (see definition below) for a set of queries. In our experiments, the MAP computation considers all the images in the datasets as query images.

- **Recall@40:** for the MPEG-7 dataset [48], we use the *bullseye score*, a measure broadly used for that collection. It counts all matching objects within the 40 most similar candidates. Since each class consists of 20 objects, the retrieved score is normalized with the highest possible number of hits.

This score is also referenced along the thesis as *Recall@40* (recall at 40th image). The recall measure can be basically defined as the fraction of relevant instances that are retrieved, at a given ranking position.

- **Precision  $\times$  Recall Curve:** the *precision* and *recall* measures can be used together for evaluating the effectiveness of retrieval systems. While *precision* considers the number of relevant items divided by the total number of retrieved items, the *recall* measure can be defined as the number of relevant retrieved divided by the total number of existing relevant items.

In order to evaluate ranked lists, precision can be plotted against recall after each retrieved item. This graph is commonly called as *precision  $\times$  recall curve*, which has a classical concave shape. The graph shows the trade-off between precision and recall. For example, trying to increase recall, typically introduces more non-relevant items into the results, thereby reducing precision (*i.e.*, moving to the right along

the curve). Trying to increase precision typically reduces recall by removing some relevant objects from the ranked lists (*i.e.*, moving to left along the curve) [56]. Figure 3.5 illustrates an example of a typical precision  $\times$  recall curve.

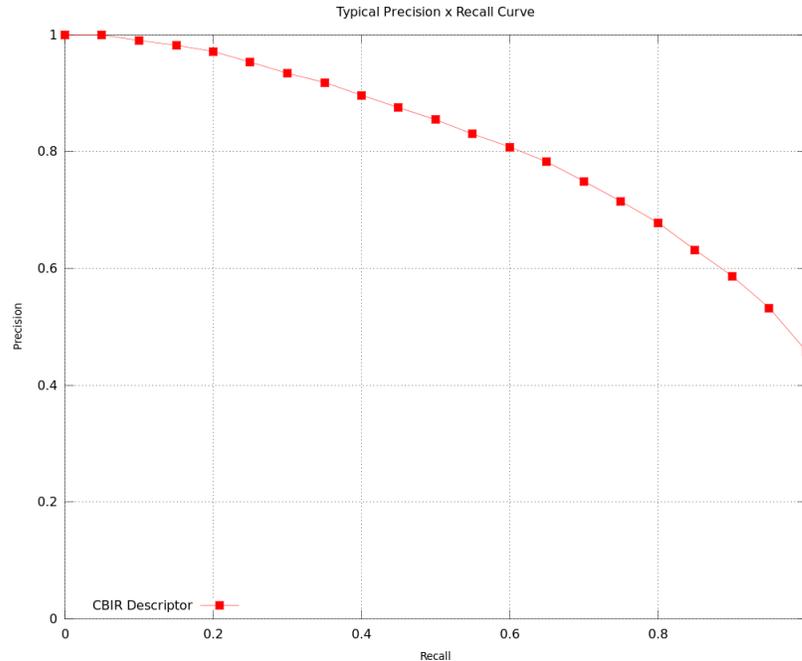


Figure 3.5: Example of a typical Precision  $\times$  Recall curve.

For MAP and  $Recall@40$  values, we computed the gain obtained by the re-ranking algorithm. Let  $M_b$  be the value of the measure before the use of the re-ranking algorithm and let  $M_a$  be the value after its use, the gain is computed as follows:

$$Gain = \frac{M_a - M_b}{M_b}. \quad (3.3)$$

## 3.2 Multimodal Retrieval

This section presents the image and textual descriptors used for multimodal retrieval. The multimodal UW dataset [22] is also described.

### 3.2.1 Descriptors

- *Visual Color Descriptors*: we consider three color descriptors on experiments: Border/Interior Pixel Classification (BIC) [90], Global Color Histogram (GCH) [93] (both already mentioned on Section 3.1.1), and the Joint Correlogram (JAC) [104].

- *Visual Texture Descriptors*: for texture we use the Homogeneous Texture Descriptor (HTD) [105], Quantized Compound Change Histogram (QCCH) [40], and Local Activity Spectrum (LAS) [94] (the last also considered in Section 3.1.1).
- *Textual Descriptors*: six well-known textual similarity measures are considered for textual retrieval: Cosine similarity measure (COS) [3], Term Frequency - Inverse Term Frequency (TF-IDF) [3], Dice coefficient (DICE) [50], Jaccard coefficient (JACCARD) [50], Bag of Words representation (BOW) [9], and OKAPI [81].

### 3.2.2 Datasets

- **UW dataset**: the UW dataset [22] was created at the University of Washington and consists of a roughly categorized collection of 1,109 images. The images are of various sizes and mainly include vacation pictures from various locations. These images are partly annotated using keywords. On average, for each image the annotation contains 6 words (tags). The maximum number of words per image is 22 and the minimum is 1. There are 18 categories: the smallest category contains 22 images and the largest contains 255 images. The average category size is 55. All dataset images are considered as query images in our experiments. Figure 3.6 shows some examples of UW dataset [22].

### 3.2.3 Measures

- **Mean Average Precision (MAP)**: we consider the MAP score, already discussed in Section 3.1.3.

## 3.3 Summary

This section summarizes the experimental protocol used along the thesis. Table 3.1 presents, for each dataset, the descriptors and measures used in the experiments.



buildings clouds  
mountain people sand  
sky



bench, car, house,  
lantern, trees,  
window



trees, bushes,  
overcast sky,  
building, post



buildings, fountain,  
grass, lantern, sky



overcast sky, house,  
car, sidewalk, struct,  
bushes, flowers,



mosque, tiles, people,  
sky, car



partially cloudy sky,  
hills, trees, grasses,  
ground, houses



Husky Stadium,  
north stands, people,  
football, field,...



sailboats, ice, water,  
buildings

Figure 3.6: Examples of UW dataset [22] images and tags.

Table 3.1: Summary of experimental protocol: Datasets, Descriptors, and Measures.

<b>Dataset</b>	<b>Descriptors</b>	<b>Type</b>	<b>Measures</b>
MPEG-7 [48]	SS [17], BAS [2], IDSC [52], CFD [68], ASC [53], AIR [35]	Shape	MAP, Recall@40, Precision $\times$ Recall
Kimia [87]	CFD [68]	Shape	Kimia score
Brodatz [7]	LBP [60], CCOM [46], LAS [94]	Texture	MAP, Precision $\times$ Recall
Soccer [100]	BIC [90], ACC [41], GCH [93]	Color	MAP, Precision $\times$ Recall
UW Dataset [22]	BIC [90], GCH [93] JAC [104], HTD [105], QCCH [40], LAS [94], COS [3], TF-IDF [3], DICE [50], JACCARD [50], BOW [9], OKAPI [81]	Color, Texture, Textual	MAP

# Chapter 4

## Distance Optimization Algorithm

In this chapter, we present the *Distance Optimization Algorithm (DOA)*, a post-processing method that exploits a clustering approach for performing image re-ranking in CBIR tasks. The algorithm explores the fact that if two images are similar, their distances to other images and therefore their ranked lists should be similar as well. The main idea of the algorithm consists in clustering images and then using the created clusters for updating distances and performing image re-ranking. These steps are repeated in an iterative manner until a convergence criterion is reached (*cohesion* measure). Figure 4.1 illustrates the main steps of the algorithm, that is detailed in Section 4.1. Clusters are created according to the similarity of images. These similarity scores are computed by special functions called *cluster-similar functions*. Two approaches based on the similarity of ranked lists and distances correlations are proposed to implement cluster-similar functions. We also demonstrate how the algorithm can be applied to the problem of combining ranked lists defined by different CBIR descriptors.

We evaluated the proposed method on shape, color, and texture descriptors. Experimental results demonstrate that the proposed method can be applied to several CBIR tasks and yields better results in terms of effectiveness than various post-processing algorithms recently proposed in the literature.

This chapter is organized as follows. Section 4.1 presents the Distance Optimization Algorithm. Sections 4.2 and 4.3 detail two important steps of the algorithm: the use of cluster-similar functions and approaches for updating distances. In Sections 4.4 and 4.5, we discuss variations and applications of the algorithm. Section 4.6 discusses aspects of efficiency. Section 4.7 presents the experimental evaluation.

The *Distance Optimization Algorithm* was first proposed in [68] and extended in [66, 69], exploiting a clustering approach for image re-ranking. An important advantage of *Distance Optimization Algorithm* consists in its flexibility. It can be easily tailored to different CBIR tasks, considering shape, color and texture descriptors.

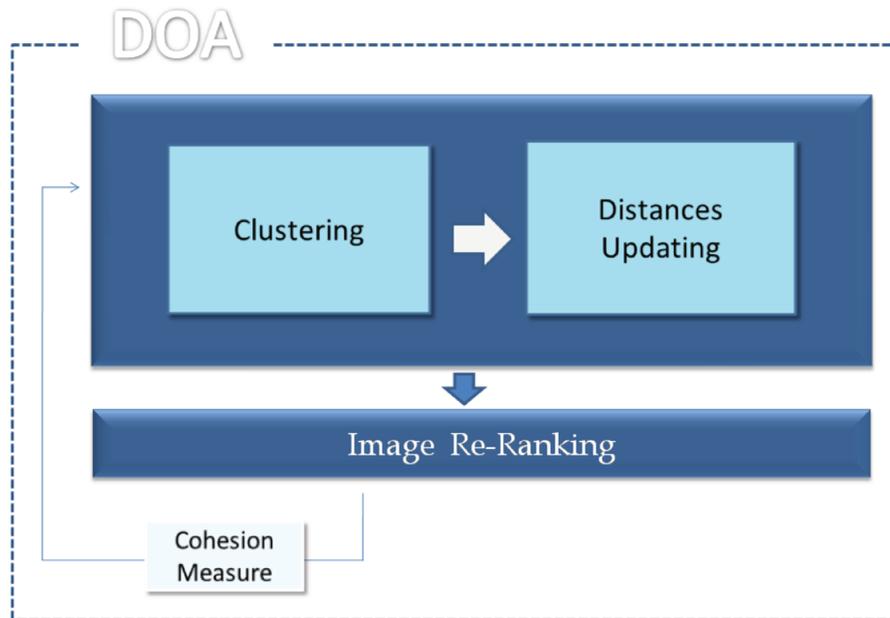


Figure 4.1: Overview of the Distance Optimization Algorithm.

## 4.1 Distance Optimization Algorithm

The *Distance Optimization Algorithm* aims at improving the effectiveness rate of a given descriptor  $\mathcal{D}$  for an image collection  $\mathcal{C}$  based on the set of ranked lists  $\mathcal{R}$  defined by  $\mathcal{D}$ .

The algorithm explores the fact that if two images are similar, their distances to other images and therefore their ranked lists should be similar as well. Basically, the algorithm redefines the distance among images, given the similarity of their ranked lists or the correlation of their distances. A clustering approach is used for that. The algorithm performs two main steps in an iterative way until a convergence criterion is reached:

1. Create clusters, by analyzing information of ranked lists;
2. Update (decrease) distances among images of a same cluster.

Images are assigned to the same cluster if they are *cluster-similar*. In this work we present two approaches to assess how cluster-similar two images are: one considers the similarity of ranked lists, while other takes into account the distances among images. Next, distances among images belonging to the same cluster are updated (decreased). Redefining distances leads to performing a re-ranking of the set  $\mathcal{R}$ . This process is repeated until the “quality” of the formed clusters does not improve and, therefore, “good” ranked lists

are created. A *cohesion* measure is proposed for evaluating the clusters and it is used as a convergence criterion. Algorithm 4.1 presents the Distance Optimization Algorithm, illustrated in Figure 4.1.

---

**Algorithm 4.1** Distance Optimization Algorithm
 

---

**Require:** Distance matrix  $A$   
**Ensure:** Optimized distance matrix  $A_o$

- 1:  $lastCohesion \leftarrow 0$
- 2:  $currentCohesion \leftarrow computeCohesion(A)$
- 3: **while**  $currentCohesion > lastCohesion$  **do**
- 4:    $Cls \leftarrow createClusters(A)$
- 5:    $W \leftarrow updateDistances(A, Cls)$
- 6:    $lastCohesion \leftarrow currentCohesion$
- 7:    $currentCohesion \leftarrow computeCohesion(A)$
- 8: **end while**
- 9:  $A_o \leftarrow A$

---

The most relevant steps of Algorithm 4.1 are those related to Lines 4 and 5. In these steps, the algorithm creates clusters and updates distances among images based on created clusters. The remaining of the algorithm computes and evaluates the convergence criterion - *cohesion* measure.

### 4.1.1 Impact of DOA on Distance Distribution

Consider the bidimensional space constructed by taking into account pairwise image distances. Consider the image space  $\mathbb{R}^2$  defined by the image collection  $\mathcal{C} = \{img_1, img_2, \dots, img_N\}$  and a distance function  $\rho : \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$ , where  $\mathbb{R}$  denotes real numbers.

We can use this space for analyzing the similarity of collection images with regard to two arbitrary images  $img_i, img_j \in \mathcal{C}$  (these images are used as reference). Consider a graphic representation of the image collection  $\mathcal{C}$  on a Cartesian coordinate system. Let  $img_l \in \mathcal{C}$  be an image. We can plot a point representing  $img_l$  on the plane, considering its distances to the images  $img_i$  and  $img_j$ .

Given two reference images  $img_i$  and  $img_j$ , we can consider a plane where the  $x$  axis represents the values of distances of collection images with regard to image  $img_i$  and the  $y$  axis represents the values of distances of collection images with regard to  $img_j$ . The position of an image  $img_l \in \mathcal{C}$  is given by the ordered pair  $(\rho(img_i, img_l), \rho(img_j, img_l))$ , where  $\rho(img_i, img_l)$  and  $\rho(img_j, img_l)$  are the distances of  $img_l$  to the reference images  $img_i$  and  $img_j$ , respectively. We can use this same approach to determine the position of all collection images.

Figure 4.2 shows the graphic representation of an image collection (MPEG-7 dataset [48]) by taking into account two reference images which are very similar. In this example we have used the CFD descriptor [68] to compute distances among images. Note that the distribution of images follows a linear behavior. As the reference images are similar, their distance to other collection images are similar as well.

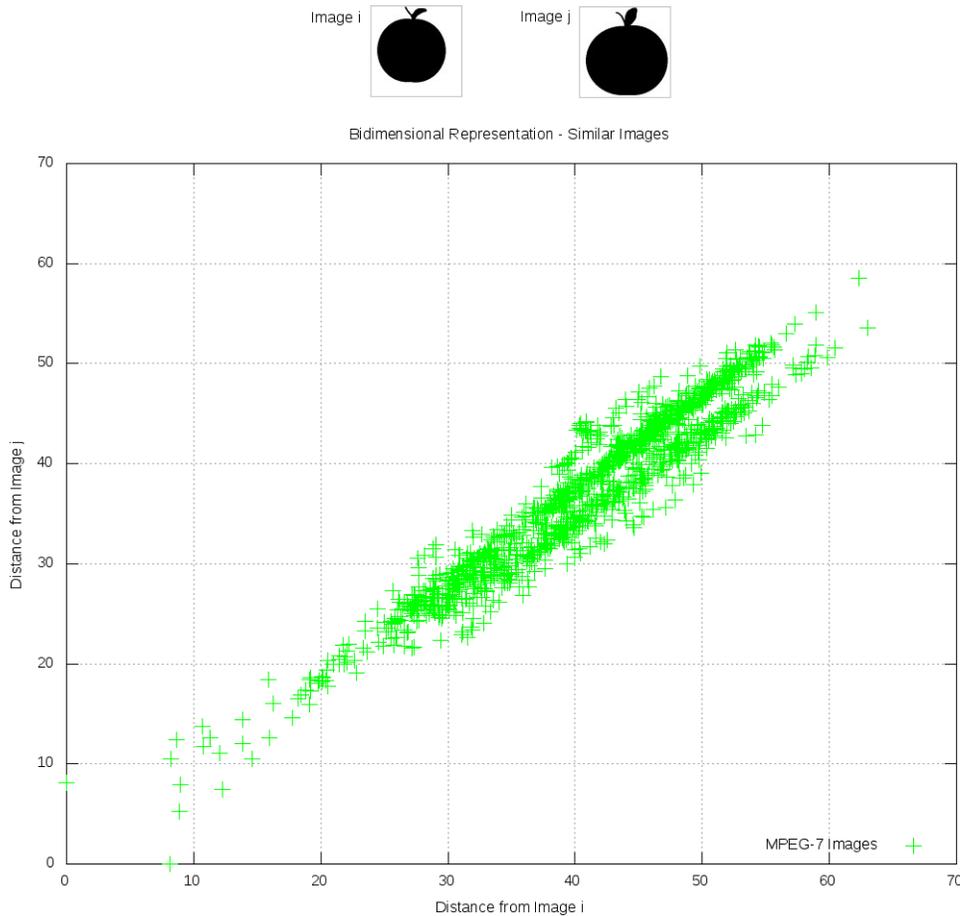


Figure 4.2: Bidimensional space representation for two similar images.

Figure 4.3 shows the same representation, now considering two reference images which are not similar, according to the same descriptor. Note that the two graphic representations present very distinct characteristics. Our goal is to use this information for image re-ranking.

We also aim at assessing the impact of Distance Optimization Algorithm on the distances among images. For this analysis we construct the same graphic of Figure 4.3 (where an image collection is represented in terms of distances of two non-similar images) after the execution of Distance Optimization Algorithm. This graphic is presented in Figure 4.4. We can observe very distinct sets of points: ( $i$ ) points next to  $x$  axis (images

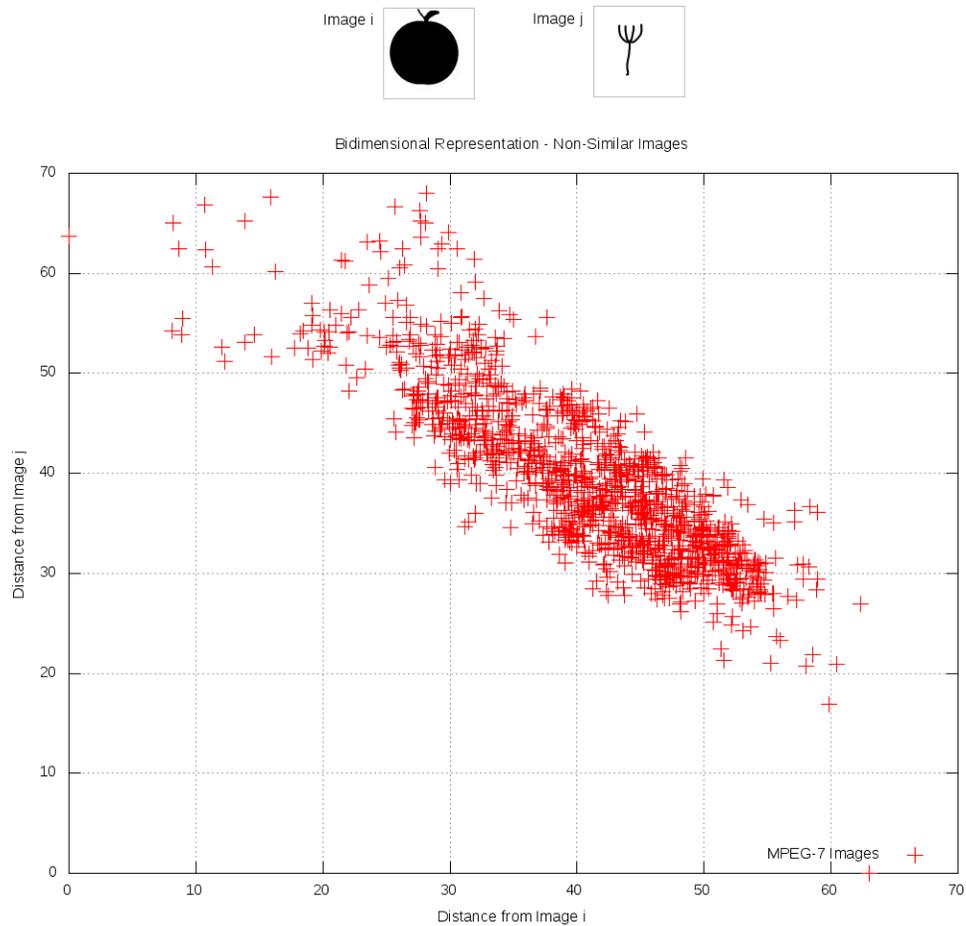


Figure 4.3: Bidimensional space representation for two non-similar images.

similar to image  $i$ );  $(ii)$  points next to  $y$  axis (images similar to image  $j$ );  $(iii)$  central points (remaining images). If two images belong to the same cluster, their distance are updated (decreased).

Figure 4.5 illustrates an example of the use of the Distance Optimization Algorithm with the CFD [68] shape descriptor. The first row presents the retrieval results for the CFD [68] shape descriptor (first image as a query). The second row presents retrieval results for the same shape descriptor after using the Distance Optimization Algorithm. We can observe that the wrong results are removed from the top positions of the ranked lists and significant improvements are obtained after the execution of DOA.

### 4.1.2 Cohesion Measure

The main goal of the *cohesion* measure is to estimate the quality of clusters. This measure is used as a convergence criterion of DOA. Its definition is based on the conjecture that,

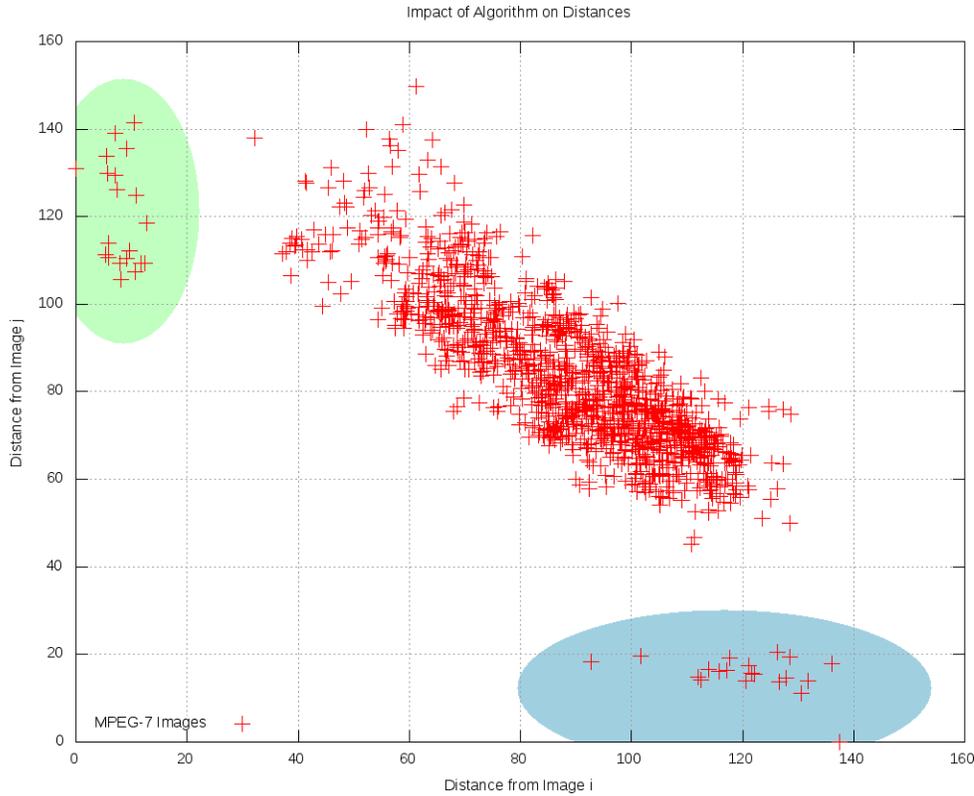


Figure 4.4: Impact of the Distance Optimization Algorithm on distances.



Figure 4.5: Impact of Distance Optimization Algorithm on ranked lists.

if a cluster is “good” (cluster of similar images), then images should refer to each other at first positions of their ranked lists (high cohesion). This conjecture is somehow close to the cluster hypothesis [101], which states that “closely associated documents tend to be relevant to the same requests”.

Let  $Cl = \{img_1, img_2, \dots, img_m\}$  be a set (or a cluster) of  $m$  images. Let  $R_q$  be the ranked list of query images  $img_q \in Cl$  with its  $m$  top images. The *cohesion* of  $Cl$  is computed based on its  $2 \times K$  nearest neighbors, considering the ranked lists  $R_q$ . It is defined as follows:

$$cohesion(Cl) = \frac{\sum_{j=0}^m \sum_{i=0}^{2 \times K} (2 \times K - i) \times (2 \times K / p) \times S(img_i, img_j, Cl)}{m^2}. \quad (4.1)$$

The constant  $p$  (we use  $p=10$  in our experiments) defines variation of a weight for each position in the ranked list. The goal is to give high weights to the first positions of the ranked lists.  $S$  function assumes value 1, if  $Cl$  contains the image ranked at position  $i$  of the ranked list defined by query image  $img_j \in Cl$ , or assumes value 0, otherwise.

### 4.1.3 Clustering Approach

#### Clustering Algorithm

Our clustering approach can be divided into two main modules: (i) a graph-based algorithm for clustering; (ii) and steps for checking and improving clusters based on the cohesion measure. Algorithm 4.2 shows the steps for clustering images. The main step of the algorithm is the function *evaluateSimilarity*. This function is in charge of creating an initial set of clusters. We can observe that this step is executed two times, in Lines 2 and 5, with different parameters. The main idea consists in creating clusters based on a initial parameter  $P_1$  and then assessing the quality of created clusters using the cohesion measure. Clusters that present a low quality according to the cohesion measure are split and their images are re-processed with a more restrictive parameter  $P_2$  (Line 5).

---

**Algorithm 4.2** Clustering Algorithm.

---

**Require:** Graph  $G$ , Parameters  $P_1$  and  $P_2$ .

**Ensure:** A set of clusters  $Cls$ .

```

1:  $Cls \leftarrow \{\}$ 
2:  $Cls \leftarrow evaluateSimilarity(G(V, E), P_1, Cls)$ 
3:  $Cls \leftarrow mergeClusters(Cls)$ 
4:  $Cls \leftarrow divideClusters(Cls)$ 
5:  $Cls \leftarrow evaluateSimilarity(G(V, E), P_2, Cls)$ 
6:  $Cls \leftarrow mergeClusters(Cls)$ 

```

---

The *evaluateSimilarity* step uses a graph-based approach for making initial clusters. Let  $G(V, E)$  be a directed and weighted graph, where a vertex  $v_i \in V$  represents an image  $img_i \in \mathcal{C}$ . The weight  $w_e$  of edge  $e = (v_i, v_j) \in E$  is defined by the ranking position of image  $img_j$  ( $v_j$ ) at the ranked list of  $img_i$  ( $v_i$ ). Algorithms 4.3 and 4.4 show the main steps for creating clusters by the *evaluateSimilarity* algorithm. In Algorithm 4.3, an empty cluster is created and all images are submitted to *processImage* procedure, detailed in Algorithm 4.4.

The *processImage* procedure consists in a recursive algorithm that adds images to the *current cluster*. This procedure aims at traversing the graph  $G$  that represents a cluster. As it can be observed in Algorithm 4.4, step 5 considers the top  $2 \times K$  nearest neighbors of an image. Two images are assigned to the same cluster (one more vertex

is visited in the graph) only if they are *cluster-similar* according to a *cluster-similar function*, considering a parameter  $P_x$  (Step 7 of Algorithm 4.4). In this way, the most relevant decision of the algorithm is related to the cluster-similar function, which must analyze distance correlation and ranked list similarities for deciding if two images should be assigned to the same cluster. The parameter  $P_x$  defines the confidence of the cluster-similar function results: restrictive  $P_x$  values lead to small but more precise clusters. Sections 4.2.1 and 4.2.2 discuss the cluster-similar functions in details.

---

**Algorithm 4.3** Algorithm evaluateSimilarity.

---

**Require:** Graph  $G = (V, E)$  and parameter  $P$ .

**Ensure:** Set of clusters  $Cls$ .

```

1:  $Cls = \{ \}$ 
2: for all  $i$  such that  $0 \leq i < |V|$  do
3:    $currentCluster = \{ \}$ 
4:   processImage ( $img_i, G, P_x$ )
5:    $Cls \leftarrow Cls \cup currentCluster$ 
6: end for

```

---



---

**Algorithm 4.4** Algorithm processImage.

---

**Require:** Image  $img_i$ , Graph  $G = (V, E)$ , and  $P_x$ .

```

1: if alreadyProcessed( $img_i$ ) then
2:   return
3: end if
4:  $currentCluster = currentCluster \cup img_i$ 
5: for all  $j$  such that  $0 \leq j < |2 \times K|$  do
6:    $img_j \leftarrow \sigma_i(j)$ 
7:   if clusterSimilar( $img_i, img_j, P_x$ ) then
8:     if not alreadyProcessed( $img_j$ ) then
9:       processImage( $img_j, G, P_x$ )
10:    else
11:       $currentCluster \leftarrow currentCluster \cup clusterOf(img_j)$ 
12:    end if
13:  end if
14: end for

```

---

## Improving Clusters

After the execution of *evaluateSimilarity* procedure (Step 2 of Algorithm 4.2), a set of initial clusters is produced. In subsequent steps, the distance optimization algorithm

aims at improving the quality of those clusters. The main idea consists in identifying situations in which clusters can be improved: (i) images that were not assigned to any clusters; (ii) clusters that could be merged and; (iii) “wrong” clusters that should be split and re-processed with a more restrictive cluster-similar function.

Function  $mergeClusters(Clusters)$  (Step 3 of Algorithm 4.2) checks if there are clusters that could be merged. This step first deals with clusters with only one image. Let  $R_i$  be the ranked list of the image of such a cluster. If the *cohesion* of the  $top_c$  images in  $R_i$  are greater than a threshold ( $th_{cohesion}$ ), then  $img_j \in R_i$  is added to the cluster that has more images of  $R_i$ .

This function is also in charge of merging small-size clusters. A small-size cluster is added to a larger cluster, if the *cohesion* of this new group is greater than a threshold  $th_{cohesion}$ . Furthermore, the new group should have a *cohesion* greater than the average cohesion of the initial clusters. Weights are defined by the size of the initial clusters.

If the *cohesion* of a formed cluster is less than threshold  $th_{cohesion}$ , this cluster is split and the status of its images is set to “non-processed” in the  $divideClusters(Clusters)$  function. These images are processed in steps 5 and 6 of Algorithm 4.2. In this case, a more restrictive parameter  $P_2$  is used in the cluster-similar functions.

## 4.2 Cluster-Similar Functions

The cluster-similar functions represents the main procedure of our clustering approach. It decides whether two images should be assigned to the same cluster. We present in next subsections two different approaches for implementing the cluster-similar functions.

### 4.2.1 Similarity of Ranked Lists

**Definition 1.** Let  $(k, l)$  be an ordered pair. Two images  $img_i$  and  $img_j$  are  $(k, l)$ -**similar**, if  $w_{e_{i,j}} \leq k$  and  $w_{e_{j,i}} \leq l$ , where  $e_{i,j} = (img_i, img_j)$  is the edge between images  $img_i$  and  $img_j$ .

**Definition 2.** Let  $O_p = \{(k_0, l_0), (k_1, l_1), \dots, (k_m, l_m)\}$  be a set of ordered pairs. Two images,  $img_i$  and  $img_j$ , are **cluster-similar** according to  $O_p$ , if  $\exists (k_a, l_a) \in O_p | img_i$  and  $img_j$  are  $(k_a, l_a)$ -similar.

Figure 4.6 illustrates how to determine if two images are *cluster-similar*. In this example,  $O_p = \{(1, 8), (2, 6), (3, 5), (4, 4)\}$ . First, it is checked if  $img_1$  and  $img_2$  are (1, 8)-similar. In this case, if the image ranked at the first position of ranked list  $R_{img_1}$  is at one of the eight first positions of ranked list  $R_{img_2}$ , images  $img_1$  and  $img_2$  are (1, 8)-similar. If not, the second pair of  $O_p$  is used, and so on.

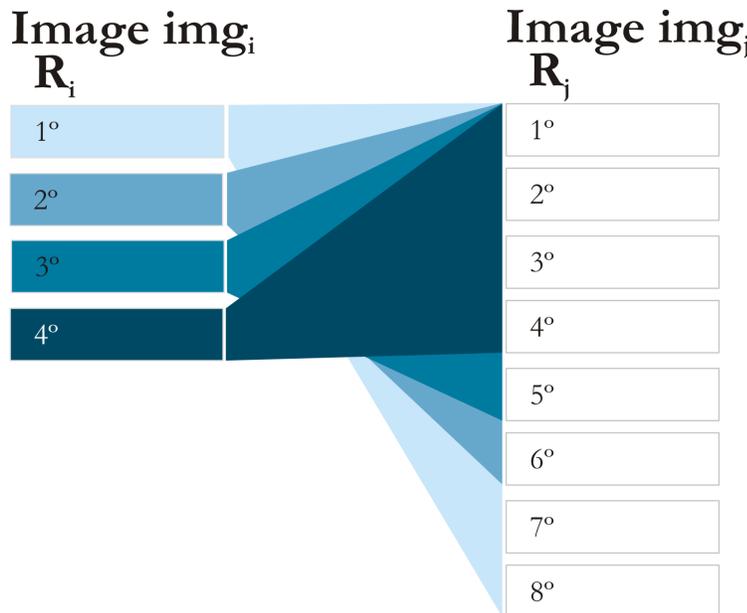


Figure 4.6: Example of *cluster-similarity* between images  $img_1$  and  $img_2$  with regard to  $O_p = \{(1, 8), (2, 6), (3, 5), (4, 4)\}$ .

In our experiments, we used the first parameter  $P_1$  as  $O_{p_1} = \{(1, 8), (2, 6), (3, 5), (4, 4)\}$  and the second  $P_2$  (and more restrictive) as  $O_{p_2} = \{(1, 6), (2, 4), (3, 3)\}$ .

## 4.2.2 Correlation

Besides the graphic representation discussed in Section 4.1.1, we can use statistical measures to characterize the distribution of distances among images. Our goal is to measure the similarity between images  $img_i$  and  $img_j$  using distances from images  $img_i$  and  $img_j$  to other images. In statistics, a measure of association is a numerical index which describes the strength or magnitude of a relationship among variables [110]. We analyze this relationship by using Pearson's Correlation Coefficient:

$$r = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2} \sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2}}. \quad (4.2)$$

Pearson's correlation coefficient  $r$  for continuous data ranges from -1 to +1, where  $r = -1$  data lie on a perfect straight line with a negative slope;  $r = 1$  data lie on a perfect straight line with a positive slope.

As discussed in [113], if the database is large, the computation of post-processing methods with all  $N$  objects may become impractical. A solution is proposed considering only the distances of  $K$ -nearest neighbors given in the ranked lists of images  $img_i$  and

$img_j$ . Note that we consider  $KNNs$  of image  $img_i$  and  $KNNs$  of image  $img_j$  for composition of vectors  $X$  and  $Y$  (used for Pearson correlation computation). Thus, the size of these vectors may range from  $K$  (when  $KNNs$  of  $img_i$  and  $img_j$  have the same elements) to  $2 \times K$  (when all elements of  $KNNs$  of  $img_i$  and  $img_j$  are different).

Given the correlation measure  $r$  and a threshold value  $\Theta$ , we can define a *cluster-similar* function based on correlation  $s : \mathcal{C} \times \mathcal{C} \rightarrow \{0, 1\}$ , where  $img_i, img_j \in \mathcal{C}$ :

$$s(img_i, img_j) = \begin{cases} 1, & \text{if } r \geq \Theta \\ 0, & \text{otherwise} \end{cases}.$$

The cluster-similar function based on distance correlation presents the important advantage of easy customization of threshold  $\Theta$ . In this way, for collections with very different sizes, only the parameter  $K$  needs to be changed (for cluster-similar function based on the similarity of ranked lists all the ordered pairs should be changed). In our experiments, we used parameters  $P_1, P_2$  respectively as  $\Theta_1 = 0.5$  and  $\Theta_2 = 0.75$ .

## 4.3 Updating of Distances

Given a set of clusters, we aim at exploiting this information for updating the distance among images. In the next subsections, we present two different methods for updating distances values.

### 4.3.1 Update based on Decreasing Distances

A cluster represents a set of similar images. In this way, the simplest strategy for distance updating consists in decreasing the distances among all images in a cluster. Let  $\lambda$  be a constant, such that  $\lambda < 1$ . Let  $Cl = \{img_1, img_2, \dots, img_m\}$  be a cluster. Let  $img_i, img_j \in Cl$  be images in the cluster  $Cl$  and let  $\rho(img_i, img_j)$  be the distance between images  $img_i$  and  $img_j$ . The distance updating based only on cluster information consists in computing a new distance  $\hat{\rho}(img_i, img_j) = \lambda \times \rho(img_i, img_j)$ .

### 4.3.2 Update based on Correlation of Distances

The updating approach that uses only the  $\lambda$  constant value to compute new distances ignores any other information encoded in the relations among images. We aim at combining information of both clusters and correlation to update distances. Our strategy for that considers the set of ranked lists  $\mathcal{R} = \{R_1, R_2, \dots, R_N\}$ . Let  $R_i \in \mathcal{R}$  be the ranked list produced by matrix  $A$  for image  $img_i$ . Let  $Cl_i$  be the cluster to which image  $img_i$  was assigned. The update approach is performed by dividing the ranked list  $R_i$  in three segments as follows:

- $Seg_1$ : an image  $img_j \in Seg_1$ , if  $img_j \in Cl_i$ , i.e., if images  $img_i$  and  $img_j$  belong to the same cluster  $Cl_i$ ;
- $Seg_2$ : an image  $img_k \in c \times KNN$  of  $R_i$  and  $k \notin Seg_1$ , i.e., if the index  $i_k$  of the image  $k$  in ranked list  $R_i$  is such that  $i_k < c \times K$  and  $img_k$  does not belong to the same cluster of  $img_i$ ;
- $Seg_3$ : an image  $l \notin Seg_1$  and  $l \notin Seg_2$ .

Figure 4.7 illustrates the three segments of a given ranked list  $R_i$  according to these criteria. For each segment of the ranked list, a different update method is performed. Note that the magnitude of constant  $c$  defines the size ( $c \times K$ ) of segment  $Seg_2$ , which is updated according to the correlation coefficient.

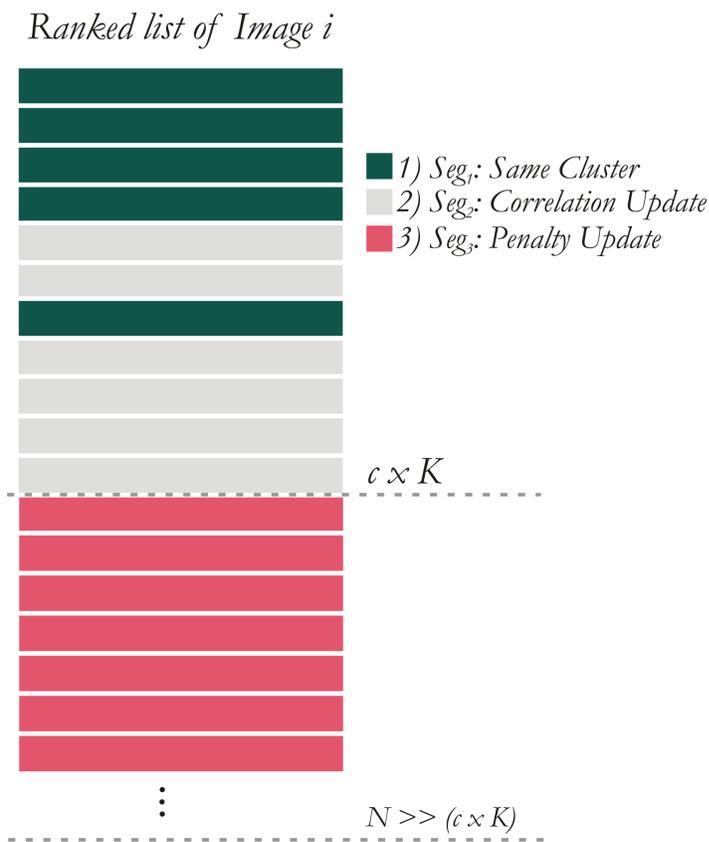


Figure 4.7: Segmentation of ranked lists in the new distance update approach.

Let  $\rho(img_i, img_j)$  be the current distance between the images  $img_i$  and  $img_j$ , and  $\hat{\rho}(img_i, img_j)$  the distance after the update, the value of an updated distance is computed for each segment as follows:

- $Seg_1$ :  $\hat{\rho}(img_i, img_j) = \rho(img_i, img_j) \times \lambda$
- $Seg_2$ :  $\hat{\rho}(img_i, img_j) = \rho(img_i, img_j) \times (1 + [(1 - \lambda) \times (1 - \bar{r})])$
- $Seg_3$ :  $\hat{\rho}(img_i, img_j) = \rho(img_i, img_j) \times [1 + (1 - \lambda)]$

where  $\bar{r}$  value represents the value of correlation normalized in the interval  $[0,1]$ .

The central idea behind this approach is to explore the correlation information for updating distances. When images are in the set  $Seg_1$  (same cluster) the distance are multiplied by constant  $\lambda < 1$ , as initially proposed in Section 4.3.1. However, when images are in the set  $Seg_2$  (fuzzy region on ranked list), an adaptive update is performed: the value for multiplying distance ranges in the interval  $[1, 1 + (1 - \lambda)]$ , depending on the correlation between images. The remaining images (the set  $Seg_3$ ) are multiplied by a fixed value greater than 1:  $1 + (1 - \lambda)$ , which we name as *penalty update*.

Note that images in the same cluster have their distances reduced. Otherwise, all remaining images in the ranked list have their distances increased. The reasons for only images in the  $Seg_2$  set suffer influence of correlation are the same of previous discussed for the choice of  $KNN$  images for correlation computation: the computation of correlation for updating distances for all images may become impractical for large databases. Remind that  $Seg_2$  considers only  $c \times K$  neighbors of image  $img_i$ .

## 4.4 Combination of Approaches

The distance optimization algorithm is flexible for combining different approaches for *cluster-similar functions* and *distance updating* process. In this work, we present two cluster-similar functions and two methods for distances updating. It is possible, for example, to combine the cluster-similar function based on ranked list similarities with the approach for updating distances using correlation.

Since the method based on correlation for updating distances includes the Decreasing Updating approach, we focused our experiments on using the correlation method. In this way, in Section 4.7, we present the experimental results of Decreasing Updating only for shape descriptors. We identified the variations of Distance Optimization Algorithm (DOA) as follows: (i) **DOA-RL-DU**: it uses the cluster-similar function based on similarity of Ranked Lists (RL) and the Decreasing Updating (DU) method for updating distances; (ii) **DOA-RL-Cor**: it uses the cluster-similar function based on similarity of Ranked Lists (RL) and the Correlation approach (Cor) for updating distances; and (iii) **DOA-Cor-Cor**: it uses the cluster-similar function based on Correlation (Cor) and the Correlation approach (Cor) for updating distance.

## 4.5 Using DOA for Rank Aggregation

*Rank Aggregation* consists in combining many ranked lists from multiple ranking algorithms, in order to obtain a “better” ordering. It can be applied to CBIR context for combining different descriptors results. In this section, we described our approach based on the distance optimization algorithm for performing rank aggregation tasks.

The distance optimization algorithm exploits information encoded in distances matrix and ranked lists for creating clusters. The clusters somehow summarizes similarity information among images. However, different descriptors may lead to different distances scores among images, and therefore different ranked lists for a given image. Consequently, it leads to the creation of different clusters when the distance optimization algorithm is used. We aim at combining cluster information of different descriptors for computing a unique and more effective distances matrix for all descriptors.

Let  $\mathcal{D} = \{D_1, D_2, \dots, D_d\}$  be a set of image descriptors that can be applied to CBIR tasks on an image collection  $\mathcal{C}$ . Let  $\mathcal{S}_i = \{Cl_1, Cl_2, \dots, Cl_c\}$  be a set of clusters obtained from distance optimization algorithm applied for descriptor  $D_i$ . Let  $Cl_j = \{img_1, img_2, \dots, img_l\}$  be a cluster such that  $Cl_j \in \mathcal{S}_i$ . We aim at combining the information of all clusters for computing an unique distance matrix  $A_c$ . For that, we use an affinity matrix  $W_c$ , that is computed as follows. If two images  $img_x$  and  $img_y$  are assigned to the same cluster for a given descriptor, the *affinity* between them  $W_c[x, y]$  receives an increment. Algorithm 4.5 outlines the steps for rank aggregation algorithm.

---

### Algorithm 4.5 Rank Aggregation based on Distance Optimization Algorithm

---

**Require:** Set of clusters  $\mathcal{S}_i$  for each Descriptor  $D_i$ , Image collection  $\mathcal{C}$

**Ensure:** Distance Matrix  $A_c$

```

1: for all  $(img_x, img_y) \in \mathcal{C}$  do
2:    $W_c[x, y] \leftarrow 1$ 
3: end for
4: for all  $\mathcal{S}_i \in \mathcal{S}$  do
5:   for all  $Cl_j \in \mathcal{S}_i$  do
6:     for all  $img_x \in Cl_j$  do
7:       for all  $img_y \in Cl_j$  do
8:          $W_c[x, y] \leftarrow W_c[x, y] + 1$ 
9:       end for
10:    end for
11:   end for
12: end for
13: for all  $(img_x, img_y) \in \mathcal{C}$  do
14:    $A_c[x, y] \leftarrow 1/W_c[x, y]$ 
15: end for

```

---

Note that, after all increments, Step 12 performs the final computation of matrix  $A_c$  as an inverse matrix  $W_c$ .

## 4.6 Aspects of Efficiency

The Distance Optimization Algorithm, as other post-processing methods, was originally designed for an off-line execution. The overall complexity of the DOA algorithm is  $O(N^2)$  and the constant associated with the asymptotic notation is very variable, depending on properties of used datasets, size of clusters, and convergence criterion.

However, the algorithm can be easily extended considering different performance optimizations. For example, the number of iterations can be reduced by changing the convergence criterion. Since the algorithm is executed while the cohesion measure is increasing, the convergence criterion can be tailored to considering a trade-off between effectiveness and efficiency. For example, the algorithm could stop when the difference between cohesion presented in current and previous iteration is smaller than a given parameter. Once the focus of this work is the proposal of the Distance Optimization Algorithm and its effectiveness evaluation, these optimizations are left for future work.

## 4.7 Experimental Evaluation

In this section, we present a set of experiments aiming at demonstrating the effectiveness of the proposed method. We analyzed and compared our method under several aspects in different experiments:

- **Experiment 1 - Cluster-Similar Functions Comparison:** Section 4.7.1 presents a comparison considering the two presented cluster-similar functions.
- **Experiment 2 - Correlation Impact:** Section 4.7.2 presents an analysis of the effects of correlation on the distance optimization algorithm.
- **Experiment 3 - Shape Descriptors:** Section 4.7.3 discusses the results of applying our method for several shape descriptors, considering the well-known MPEG-7 dataset [48]. We also evaluated the algorithm considering rank aggregation tasks.
- **Experiment 4 - Texture Descriptors:** Section 4.7.4 describes conducted experiments involving texture descriptors.
- **Experiment 5 - Color Descriptors:** Section 4.7.5 presents the experiment results for color descriptors.

- **Experiment 6 - General CBIR Tasks:** Section 4.7.6 presents the experiment results for general CBIR tasks involving shape, texture, and color descriptors.

### 4.7.1 Experiment 1 - Cluster-Similar Functions Comparison

For evaluating the effectiveness of correlation as a cluster-similar function, we conducted experiments comparing a cluster-similar function based on correlation with a cluster-similar function based on similarity of ranked lists. Since that cluster-similar functions determine when an image should be assigned to a cluster, the effectiveness of overall algorithm depends directly on the accuracy of the results of these functions.

We use the MPEG-7 dataset [48] in the experiments. Since the MPEG-7 dataset consists of 1400 images divided into 70 shape classes of 20 images each, considering an image as a query, a perfect cluster-similar function should return “true” for 20 shapes of its ranked list and “false” for remaining images. Considering each collection image as a query, a perfect cluster-similar function should return a total of 28,000 ( $1400 \times 20$ ) “true” values.

We compare cluster-similar functions based on correlation and ranked list similarities, using the CFD descriptor [68]. We use the following parameters for thresholds: ordered pairs  $O_p = \{(1, 6), (2, 4), (3, 3)\}$  for ranked lists similarities and  $\Theta > 0.25$  for distances correlation function.

The results of cluster-similar function comparison are presented in Table 4.1. As we can observe, the cluster-similar function based on correlation presents a number of “true” values much greater than that one observed for the function based on similarity of ranked lists: for both correct and wrong judgements. When we compare the relative rate *Wrongs/Corrects*, the two functions presented similar results, but for the rate *Corrects/Expected*, a better performance can be observed for the cluster-similar function based on correlation.

Table 4.1: Cluster-similar functions comparison on the MPEG-7 dataset.

<b>“True” results for Cluster-Similar function</b>	<b><i>Ranked Lists Similarities</i></b>	<b><i>Distances Correlation</i></b>
<i>Expected</i>	28,000	28,000
<i>Corrects</i>	2,532	8,294
<i>Wrongs</i>	24	76
<i>Rate Wrongs/Corrects</i>	<b>0.95%</b>	<b>0.91%</b>
<i>Rate Corrects/Expected</i>	<b>9.04%</b>	<b>29.62%</b>

### 4.7.2 Experiment 2 - Correlation Impact

We analyzed the impact of using correlation information on updating distances and how this use affects the distance optimization algorithm. Besides retrieval results (presented in next section) the convergence of algorithm is a good indicator of this behavior.

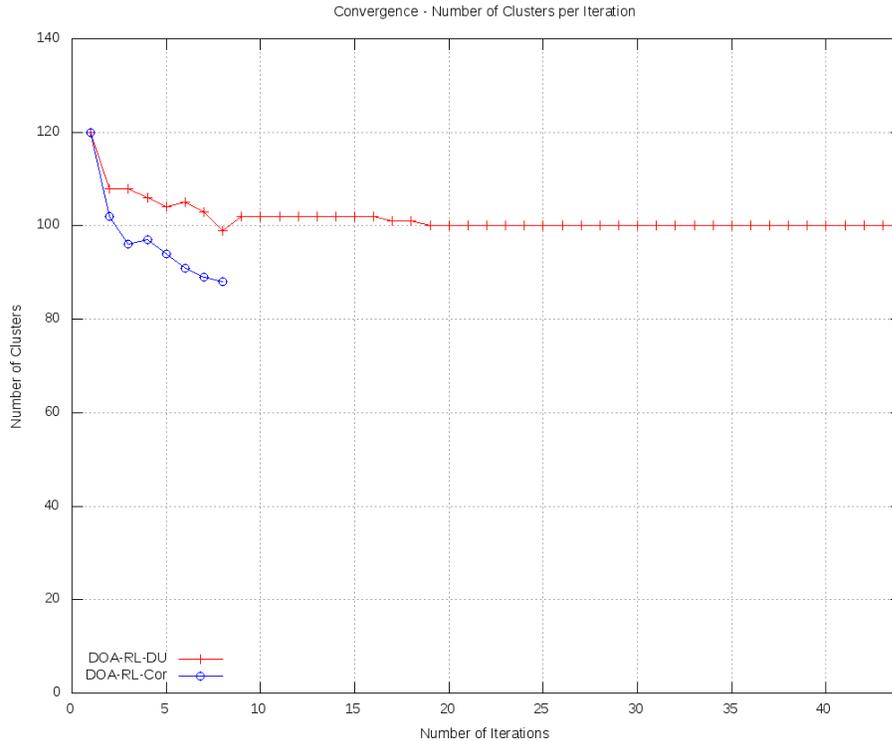


Figure 4.8: Convergence: number of clusters per iteration.

Figure 4.8 illustrates the evolution of distance optimization algorithm in terms of number of clusters by iteration. We consider the two distance updating approaches for comparison: (i) decreasing distances and; (ii) correlation. We use the cluster-similar function based on similarity of ranked lists for both distance updating approaches. As we can observe, the correlation affects the algorithm positively by decreasing the number of cluster (next to expected number of clusters - 70) and the number of iterations necessary to reach this value.

### 4.7.3 Experiment 3 - Shape Descriptors

In this section, we aim at evaluating our method with regard to two different aspects: (i) comparing the different approaches of cluster-similar functions and distances updating approaches, and (ii) evaluating the use of our method with several shape descriptors.

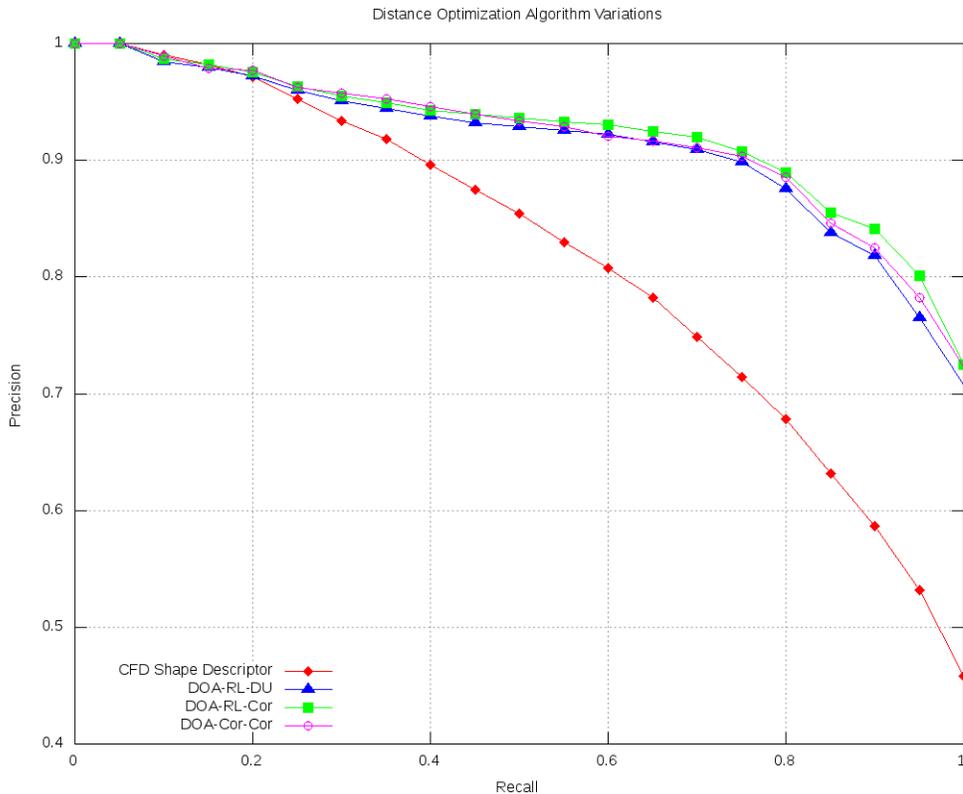


Figure 4.9: Precision vs. Recall: comparing results of distance optimization algorithm variations

For the experiments, we use the following parameter values for distances updating:  $\lambda = 0.95$ ,  $K = 20$ , and  $c = 6$ . Regarding other parameters, we use  $top_n = 40$ ,  $th_{cohesion} = 70$ , and  $top_c = 10$ .

We compared three variations of the distance optimization algorithm on the MPEG-7 dataset. Figure 4.9 presents the Precision vs. Recall curves considering the CFD [68] shape descriptor and variations of distance optimization algorithm. As we can observe, the approach that uses cluster-similar function based on the similarity of ranked lists and distances updating based on correlation presents better precision values.

We also evaluate our method considering the MPEG-7 dataset with the so-called bullseye score. In Table 4.2, we present results of distance optimization algorithm, the CFD [68] and IDSC [52] shape descriptors, that have been used as inputs.

Finally, we evaluate the use of our methods when applied to other shape descriptors. The MPEG-7 database is again used in this experiment. Results are presented in Table 4.3. Note that the effectiveness gains (shown between square brackets) are always positive and ranges from +5.40% to +21.00%.

Table 4.2: Post-processing methods comparison on the MPEG-7 dataset (*Recall@40*).

Algorithm	Descriptor	Score	Gain
CFD [68]	-	84.43%	-
IDSC [52]	-	85.40%	-
DOA-RL-DU	CFD [68]	92.56%	+9.63%
DOA-Cor-Cor	CFD [68]	92.90%	+9.12%
DOA-RL-Cor	CFD [68]	93.62%	+10.88%
DOA-RL-Cor - Rank Aggregation	CFD [68]+IDSC [52]	96.46%	-
DOA-Cor-Cor - Rank Aggregation	CFD [68]+IDSC [52]	97.00%	-

Table 4.3: Distance Optimization Algorithm applied to shape descriptors on MPEG-7 dataset (*Recall@40*).

Shape Descriptor	Score	DOA-RL-Cor	DOA-Cor-Cor
SS [17]	43.99%	50.93% [+15.78%]	53.23% [+21.00%]
BAS [2]	75.20%	85.11% [+13.18%]	84.15% [+11.90%]
IDSC [52]	85.40%	90.02% [+5.40%]	90.39% [+5.84%]
CFD [68]	84.43%	93.62% [+10.88%]	92.90% [+10.03%]
ASC [53]	88.39%	90.66% [+2.57%]	93.61% [+5.91%]
AIR [35]	93.67%	97.68% [+4.28%]	98.81% [+5.49%]

#### 4.7.4 Experiment 4 - Texture Descriptors

Our goal is to evaluate the application of our method for several CBIR tasks considering different visual properties (shape, color, and texture). The evaluation uses three well-known texture descriptors and compared the effectiveness of retrieval before and after the execution of the two variations of the distance optimization algorithm proposed in this work: (*i*) distance optimization + correlation (for update); (*ii*) distance optimization + correlation (for update + cluster-similar function).

We used the Brodatz [7] dataset. Since the Brodatz dataset presents different categorization characteristics from the MPEG-7 dataset, we changed some parameters of the distance optimization algorithm: we used  $th_{cohesion} = 55$ ,  $top_c = 8$ , and  $K = 15$ . For other parameters, we use the same values as used for the MPEG-7 collection.

Figure 4.10 presents the Precision vs. Recall curve for descriptors CCOM [46] and LAS [94] before and after execution of the two variations of distance optimization algorithm. We can observe that for LAS [94] descriptor, the distance optimization algorithm improved approximately 15% on recall at 1.

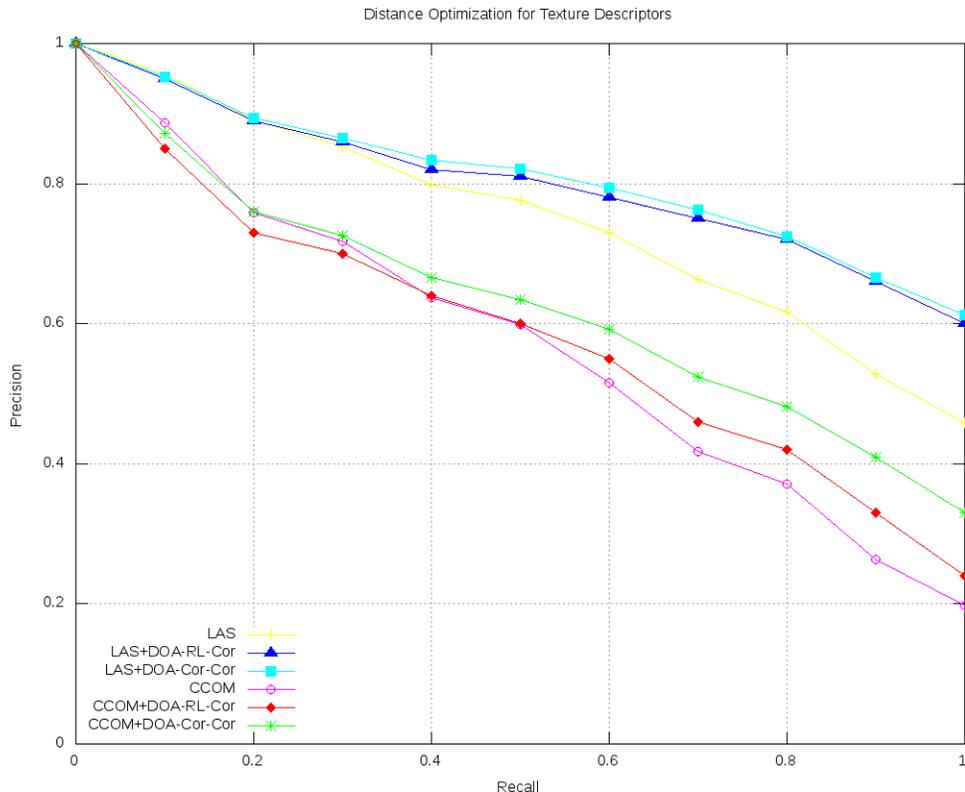


Figure 4.10: Distance Optimization applied to Texture Descriptors.

### 4.7.5 Experiment 5 - Color Descriptors

We evaluate our method for three color descriptors: BIC [90], ACC [41], and GCH [93]. The experiments were conducted on the Soccer dataset [100].

The parameters of the distance optimization algorithm were the same used on the MPEG-7 dataset. Only the correlation thresholds were changed, since the color descriptors present precision very lower when compared with shape descriptors. Thus it requires higher correlation thresholds for cluster-similar function. We used  $\Theta_1 = 0.25$  and  $\Theta_2 = 0.50$ .

Figure 4.11 presents the Precision vs. Recall curves for descriptors BIC [90] and ACC [41] before and after the use of the distance optimization algorithm. As we can observe, except by BIC+UpCorrelation, all curves presented positive gains.

### 4.7.6 Experiment 6 - General CBIR Tasks

Finally, we evaluate our method in a general way, comparing results for several descriptors (shape, color, and texture) in different datasets. The measure adopted is *Mean Average*

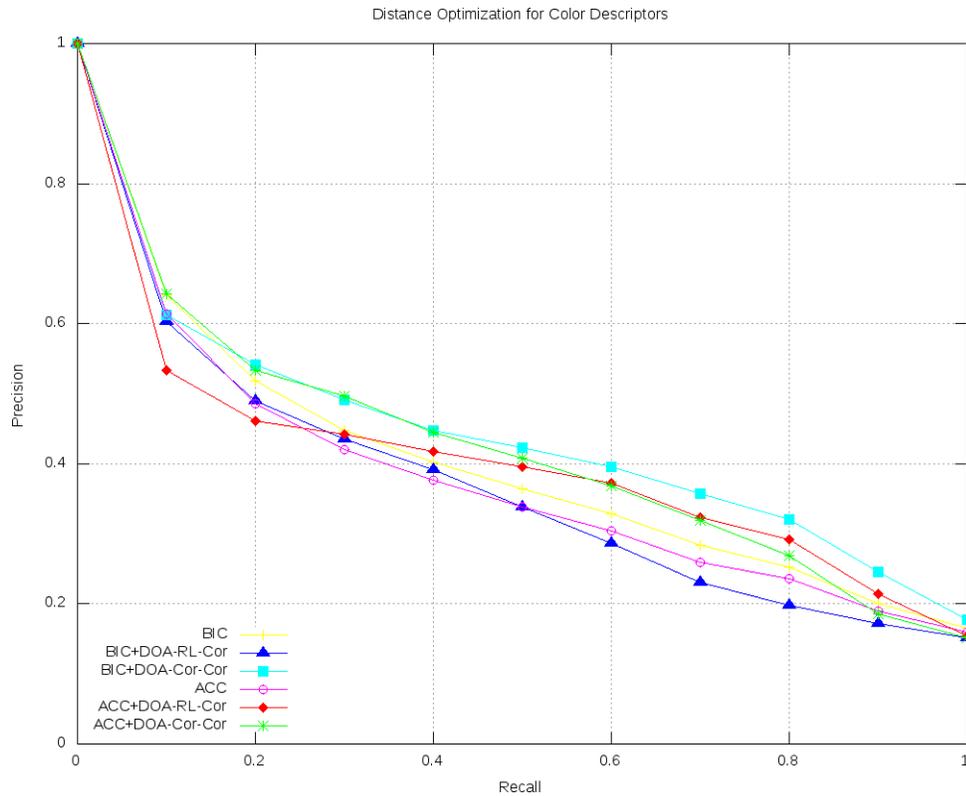


Figure 4.11: Distance Optimization applied to Color Descriptors.

### *Precision (MAP).*

Results are presented in Table 4.4. Except for one result of BIC and GCH color descriptors, the Distance Optimization Algorithm presented positive effectiveness gains for all descriptors, ranging from +2.95% to +29.44%. The approach that uses the cluster-similar function based on correlation presented only positive gains.

We also conducted a paired t-test aiming at evaluating the chance of difference between the means (before and after executing the proposed re-ranking method considering all descriptors) being statistical significant. We conclude that there is a 99.9% of chance of difference being statistical significant considering the DOA-RL-Cor approach and 99% considering the DOA-Cor-Cor approach.

Table 4.4: Correlation Methods Evaluation on Several Content-Based Image Retrieval Tasks - Mean Average Precision

Descriptor	Type	Dataset	Score [%] (MAP)	DOA-RL-Cor	DOA-Cor-Cor
SS [17]	Shape	MPEG-7	37.67%	46.53% [+23.52%]	48.76% [+29.44%]
BAS [2]	Shape	MPEG-7	71.52%	81.05% [+13.32%]	80.84% [+13.03%]
IDSC [52]	Shape	MPEG-7	81.70%	86.94% [+6.41%]	87.77% [+7.43%]
CFD [68]	Shape	MPEG-7	80.71%	91.79% [+13.73%]	91.40% [+13.24%]
ASC [53]	Shape	MPEG-7	85.28%	88.41% [+3.67%]	91.60% [+7.41%]
AIR [35]	Shape	MPEG-7	89.39%	93.54% [+4.64%]	95.77% [+7.14%]
GCH [93]	Color	Soccer	32.24%	30.78% [-4.53%]	33.13% [+2.76%]
ACC [41]	Color	Soccer	37.23%	42.46% [+14.05%]	45.24% [+21.51%]
BIC [90]	Color	Soccer	39.26%	38.16% [-2.80%]	44.23% [+12.66%]
LBP [60]	Texture	Brodatz	48.40%	52.31% [+8.08%]	49.34% [+1.94%]
CCOM [46]	Texture	Brodatz	57.57%	59.27% [+2.95%]	64.60% [+12.21%]
LAS [94]	Texture	Brodatz	75.15%	80.36% [+6.93%]	81.17% [+8.01%]

# Chapter 5

## Pairwise Recommendation

In this chapter we present a new re-ranking method that takes into account relationships among images for improving the effectiveness of CBIR descriptors. We propose a measure for analyzing the quality of ranked lists and use the concept of *recommendation* for modeling and handling relationships among images and then for establishing new relationships among images. The proposed approach based on pairwise recommendation is the main novelty of the re-ranking algorithm, which is conceptually very different from previous works [42, 45, 114, 115]. Recommender systems attempt to reduce information overload by selecting automatically items that match the personal preferences of each user [6, 88]. More formally, “given a collection and an actor, and a set of ratings for objects in that collection produced by others or the same actor, recommends (produces a subset of that collection) for that particular actor [34]”.

Our pairwise recommendation approach is inspired by the concept of recommendation, originally created to consider users ratings. However, our method does not require any user interaction. The recommendations are simulated based on information encoded in ranked lists computed by CBIR descriptors. The relationships among images encoded in ranked lists are used for composing *image profiles* and then for recommending images, that is, an image recommends images (that are possibly relevant) to another image. In this context, a recommendation means that the distance between two images should be decreased and an image should be *moved up* in the ranked list of the image that received the recommendation. Our method also incorporates a simple clustering step for further improving distances among images that belong to a same cluster. Furthermore, our approach also can be used for combining different CBIR descriptors (rank aggregation tasks).

Our strategy opens a new area of investigation, related to the use of recommendation techniques in re-ranking tasks. Our method is detailed in the next section.

This chapter is organized as follows. Section 5.1 describes the image re-ranking al-

gorithm based on pairwise recommendation. Section 5.2 discusses the cohesion measure. Section 5.3 describes how recommendations are performed. Section 5.4 presents our clustering approach and Section 5.5 the convergence criterion. The rank aggregation approach is described Section 5.6. Finally, Section 5.7 presents the experimental evaluation.

## 5.1 The Re-Ranking Algorithm

The main idea of the Pairwise Recommendation re-ranking algorithm relies on the conjecture that images can *recommend* images found at the first positions of their ranked lists (that is, their *K-nearest-neighbors*). In this scenario, *recommendation* means decreasing the distance between images: when an image  $img_i$  recommends an  $img_k$  to an image  $img_j$ , it means that image  $img_j$  should have its distance to  $img_k$  decreased.

Each recommendation is associated with a different *weight* (how much the distance should be decreased). For computing the recommendation weight, we consider the position of images in ranked lists and the *quality* of the ranked lists. We use a *cohesion* measure for estimating the quality of ranked lists and then sorting the ranked lists. We consider, in this way, that images with better ranked lists (higher cohesion) have more authority for making recommendations. After performing all recommendations, ranked lists are considered for clustering images and additional recommendations are made, given the obtained clusters.

Once all distances have been updated by recommendations, a re-ranking can be performed based on the new distance matrix  $A_{t+1}$  (where  $t$  indicates the current iteration) for generating a new set of ranked lists  $\mathcal{R}_{t+1}$ . These steps are repeated in an iterative manner until a convergence criterion is reached. The employed convergence criterion is based on the variation of cohesion measure. At each iteration we increment the number  $K$  of neighbors considered for recommendations. Note that after one iteration, more relevant images are found at first positions of the ranked lists. Non-relevant images are moved out from the first positions of the ranked lists and therefore  $K$  can be increased for considering more images. In the next iteration, more images (larger  $K$ ) are considered in the recommendation process. Finally, when the convergence criterion is reached, a re-ranking is performed based on the final distance matrix  $\hat{A}$ . Figure 5.1 illustrates the main steps of our approach. Algorithm 5.1 outlines our re-ranking method.

The main steps of Algorithm 5.1 are presented in Lines 6, 10, and 13, which refer, respectively, to computing *cohesion*, to making *recommendations*, and to *clustering* images. These steps are detailed in next sub-sections. Note that, in Line 8,  $C = \{c_1, c_2, \dots, c_N\}$  is a set of cohesion scores  $c_i$  computed for each ranked list  $R_i$ . Based on  $C$ , a set  $\mathcal{R}_c$  is computed, where ranked lists are sorted in decreasing order of cohesion. In Line 16, a re-ranking is performed. Once the distance matrix  $A_{t+1}$  is updated, the ranked lists can

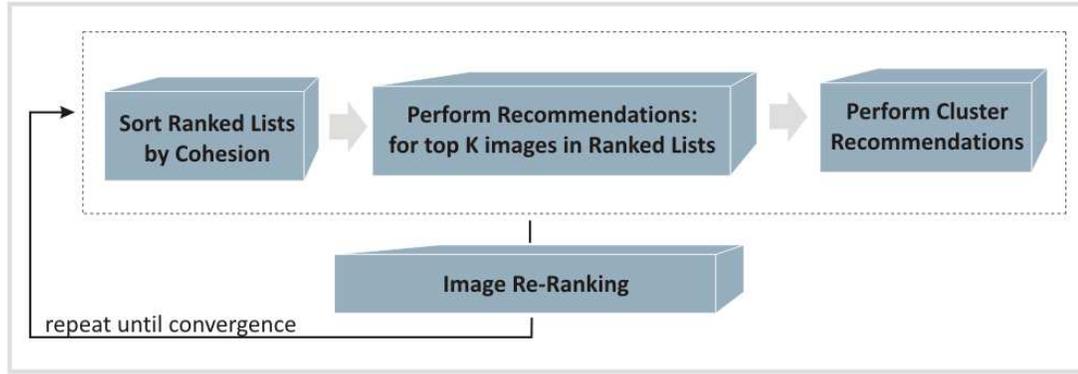
**Pairwise Recommendation**

Figure 5.1: Pairwise Recommendation re-ranking method.

**Algorithm 5.1** Pairwise Recommendation Re-Ranking**Require:** Distance matrix  $A$  and set of ranked lists  $\mathcal{R}$ ,  $K$ ,  $\lambda$ ,  $\epsilon_{cohesion}$ **Ensure:** New distance matrix  $\hat{A}$  and new set  $\hat{\mathcal{R}}$ 

```

1:  $t \leftarrow 0$ 
2:  $A_t \leftarrow A$ 
3:  $currentCohesion \leftarrow 0$ 
4:  $\mathcal{R}_t \leftarrow \mathcal{R}$ 
5: repeat
6:   for all  $R_i \in \mathcal{R}$  do
7:      $c_i \leftarrow computeCohesion(R_i, \mathcal{R}_t)$ 
8:   end for
9:    $\mathcal{R}_c = sortRankedListsByCohesion(\mathcal{R}_t, C)$ 
10:  for all  $R_i \in \mathcal{R}_c$  do
11:     $A_t \leftarrow performRecommendations(A_t, R_i, c_i)$ 
12:  end for
13:  for all  $R_i \in \mathcal{R}_c$  do
14:     $A_t \leftarrow performClusterRecommendations(A_t, R_i)$ 
15:  end for
16:   $A_{t+1} \leftarrow A_t$ 
17:   $\mathcal{R}_{t+1} \leftarrow performReRanking(A_{t+1})$ 
18:   $lastCohesion \leftarrow currentCohesion$ 
19:   $currentCohesion \leftarrow computeAvgCohesion(\mathcal{R}_{t+1})$ 
20:   $t = t + 1$ 
21:   $K = K + 1$ 
22: until  $(currentCohesion - lastCohesion) < (currentCohesion \times \epsilon_{cohesion})$ 
23:  $\hat{A} = A_t$ 
24:  $\hat{\mathcal{R}} = \mathcal{R}_t$ 
  
```

be computed again, that is, images are re-ranked.

## 5.2 Cohesion Measure

In this section, we use a normalized *cohesion* measure for estimating the quality of *ranked lists*, similar to that presented in Chapter 4. The objective of this measure is to assess how “good” a ranked list is. A ranked list is considered “good” when images placed at the top positions refer to each other at the top positions of their ranked lists. It is somehow close to the cluster hypothesis [101], which states that “*closely associated documents tend to be relevant to the same requests*”.

Our method considers that “*high quality*” ranked lists are able to make more accurate recommendations. In this sense, these ranked lists have more authority (defined by the cohesion measure) to make recommendations. This approach is analogous to the PageRank algorithm [61]. Although having different objectives, both PageRank and our cohesion measure exploit the link structure (hyperlinks references in ranked lists) for obtaining information about items (pages, images). Basically, the PageRank algorithm assess the importance of a page by taking into account link structures. In our approach, the cohesion measure aims at assessing the quality of ranked lists by analyzing how images refer to each other in their ranked lists.

The computation of cohesion is as follows: Let  $R_i$  be a ranked list of an image  $img_i$ . Let  $R_{ki} = \{img_1, img_2, \dots, img_K\}$  be a subset of a ranked list  $R_i$  that considers the  $K$  nearest neighbors of  $img_i$ . Let  $img_j \in R_{ki}$  be an image of this subset (one of  $K$ -neighbors of image  $img_i$ ), and let  $R_{kj}$  be a subset of the ranked list of  $img_j$ . Finally, let  $img_p \in R_{kj}$  be an image in the ranked list  $R_{kj}$ . We define the cohesion as follows:

$$cohesion(R_i, K) = \frac{\sum_{img_j \in R_{ki}} \sum_{img_p \in R_{kj}} s(R_{ki}, img_p) \times w(R_{kj}, img_p)}{\sum_{img_j \in R_{ki}} \sum_{img_p \in R_{kj}} w(R_{kj}, img_p)}. \quad (5.1)$$

The terms  $s$  and  $w$  are functions. The objective of the function  $s$  is to determine if image  $img_p$  (that belongs to subset  $R_{kj}$ ) also belongs to subset  $R_{ki}$ . The function  $s$  is defined as follows:

$$s(R_{ki}, img_p) = \begin{cases} 1, & \text{if } img_p \in R_{ki} \\ 0, & \text{otherwise} \end{cases}. \quad (5.2)$$

The function  $w$  takes as input a position of an image in a ranked list. The goal is to give high weights to images at the first positions of the ranked lists. In our algorithm we define  $w$  as  $w(R_{kj}, p) = 1/\sigma_j(p)$ , where  $\sigma_j(p)$  represents the position of image  $img_p$  in the ranked list  $R_{kj}$ . Note that, if all referenced images are in the subset  $R_{ki}$ , the function  $s$  will assume value 1 for all images and therefore cohesion (Equation 5.1) is set to 1. It

indicates a perfect cohesion, where all considered images refer to each other at the first positions of their ranked lists.

Figure 5.2 illustrates the computation of the cohesion measure for the ranked list  $R_i$ . Observe, on the left, the ranked list  $R_i$  and its subset  $R_{ki}$ . On the right, for a given image  $img_j \in R_{ki}$ , it illustrates the ranked list  $R_j$  (and its subset  $R_{kj}$ ). The function  $s$  verifies if an  $img_p$  belongs to both subsets  $R_{ki}$  and  $R_{kj}$ . Function  $w$ , illustrated on the right, computes a weight given the position of image  $img_p$  in the ranked list  $R_j$ .

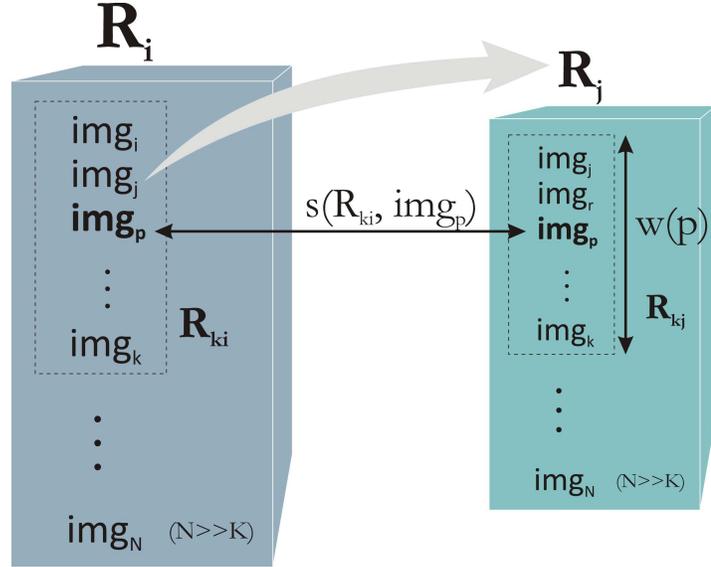


Figure 5.2: Computation of the cohesion measure.

### 5.3 Performing Recommendations

The basic idea of our recommendation method is: “an image  $img_i$  recommends the  $img_y$  to  $img_x$ , if  $img_x$  and  $img_y$  are on the top- $K$  positions of the ranked list of  $img_i$ ”. In this context, the *recommendation* is associated with a *decrease of the distances* between two images ( $img_x$  and  $img_y$ ). The recommendations are performed in the same context where cohesion is computed: considering a subset  $R_{ki}$  with the  $K$ -nearest neighbors of a ranked list  $R_i$ . Observe that, before recommendations, the cohesion of all ranked lists are computed and the ranked lists are sorted in a decreasing order of cohesion. In this way, the recommendations, which represent updates for distance matrix  $A$ , are performed first for ranked lists with higher cohesion. Algorithm 5.2 presents our method for performing recommendations for a given ranked list  $R_i$ .

Variables  $w_x$  and  $w_y$  represent the weight given to images  $img_x$  and  $img_y$  in the

---

**Algorithm 5.2** Recommendations performing
 

---

**Require:** Matrix  $A$ , Ranked list  $R_i$  and Cohesion  $c_i$ 
**Ensure:** Updated matrix  $A$ 

```

1:  $R_{ki} \leftarrow KNN(R_i)$ 
2:  $x \leftarrow 1$ 
3: for all  $img_x \in R_{ki}$  do
4:    $w_x \leftarrow 1 - (\sigma_i(y)/K)$ 
5:    $y \leftarrow 1$ 
6:   for all  $img_y \in R_{ki}$  do
7:      $w_y \leftarrow 1 - (\sigma_i(y)/K)$ 
8:      $w \leftarrow c_i \times w_x \times w_y$ 
9:      $\lambda \leftarrow 1 - \min(1, L \times w)$ 
10:     $A[x, y] \leftarrow \min(\lambda A[x, y], A[y, x])$ 
11:     $y \leftarrow y + 1$ 
12:   end for
13:   $x \leftarrow x + 1$ 
14: end for

```

---

recommendation. The weights are computed based on the position of those images in the ranked lists: for images at first positions of the ranked list, a higher weight is assigned. The weights associated with the first positions indicate where it is more likely to find the most similar (relevant) images, that is, positions that represent more reliable recommendations. These variables are computed in Lines 4 and 7 of Algorithm 5.2, both in the interval  $[0,1]$ . In Line 8, the weight  $w$  of a recommendation is computed. That represents the *reputation* of the recommendation. For computing  $w$ , we consider  $w_x$ ,  $w_y$  and the cohesion  $c_i$  of the ranked list  $R_i$ . Figure 5.3 illustrates how a recommendation is performed for a given ranked list  $R_i$ . It considers two images  $img_x, img_y \in R_{ki}$  and takes into account their positions in the ranked list for computing the weights  $w_x$  and  $w_y$ .

In Line 9, a coefficient  $\lambda$  is computed in the interval  $[0,1]$ . This coefficient is used for determining how the distances between  $img_x$  and  $img_y$  should be decreased. For computing  $\lambda$ , we multiply the weight  $w$  of the recommendation and a constant  $L$ . The goal of constant  $L$  is to adjust the “*speed*” of the convergence of the algorithm. By increasing the value of  $L$ , the distances among images will decrease faster and the algorithm will be executed in less iterations. However, with a very high value of  $L$ <sup>1</sup>, the algorithm can not take advantage of the improvements of the ranked lists along iterations. Note also that we use a *min* function in Line 9 to avoid negative values and then to restrict the coefficient  $\lambda$  to  $[0,1]$ . Finally, the value of  $\lambda$  is multiplied by for the current distance  $A[x, y]$  for computing the new updated distance.

---

<sup>1</sup>We used  $L$  in interval  $[1,2]$  in our experiments.

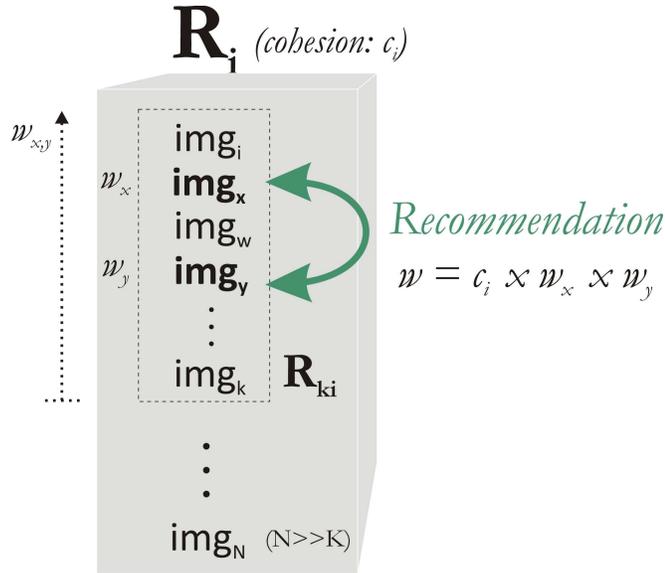


Figure 5.3: Performing recommendations.

## 5.4 Clustering Approach

High values of  $w$  (or  $L$ ) can lead to situations where  $\lambda = 0$  and, consequently,  $A[x, y] = 0$ . These situations are associated with recommendations of great confidence. The key idea of our clustering approach is to exploit these cases to group images and then making additional recommendations based on created clusters. Let  $R_i$  be a ranked list of an image  $img_i$ . A cluster  $Cl_i$  is composed by all images whose distance to  $img_i$  is equal to 0.  $Cl_i$  can be defined as follows:  $\{Cl_i \subset R_i \mid \forall img_c \in Cl_i, A[i, c] = 0\}$ .

Given a cluster  $Cl_i$  the additional recommendations consists in setting all distances among all images of  $Cl_i$  to 0. More formally: we aim at ensuring that for each cluster  $Cl_i$  and for each pair of images  $img_x, img_y \in Cl_i$ , we have  $A[x, y] = A[y, x] = 0$ .

## 5.5 Convergence Criterion

In general, an iterative method is said to converge, if the difference between results obtained along iterations decreases, tending to reach an ultimate result. In our case, it is expected that the proposed re-ranking algorithm converges, improving the quality of the ranked lists along the iterations, tending to a final ranking.

In Section 5.2, we described the *cohesion* measure, whose main goal is to estimate the quality of ranked lists. This measure is also used as a convergence criterion, according to the following conjecture: “the re-ranking procedure should be iteratively executed while the quality of ranked lists (measured by cohesion) is increasing”. Therefore, at each iteration,

the average cohesion of all ranked lists is computed and compared with the one computed in the previous iteration. The convergence criterion of the re-ranking algorithm is tested in Line 22 of Algorithm 5.1. The convergence condition checks if the variation of cohesion is greater than a given threshold. The threshold is computed proportionally to the current cohesion, using the parameter  $\epsilon_{cohesion}$ . For the convergence criterion, the computation of cohesion considers the  $2 \times K$  top positions of ranked lists (initial value of  $K$ ).

In the following, we present a brief discussion about the method's convergence. Let  $\mathcal{C}$  be an image collection. Let  $S_i$  be a set of similar images such that  $S_1 \cup S_2 \cup \dots \cup S_m = \mathcal{C}$  and  $|S_i| \geq K$ . We consider three hypothetical scenarios, given the effectiveness of CBIR descriptors:

1. *“Highly-effective” descriptor*: by using the highly-effective descriptor for collection  $\mathcal{C}$ , images found at the top  $K$  positions of a ranked list  $R_{ki}$  of an image  $img_i \in S_i$  are all similar to each other, that is  $R_{ki} \subset S_i$ . In this scenario, the average cohesion of ranked lists is very high, since all similar images refer to each other at the top positions of their ranked lists. Therefore, the recommendations produce small changes in the ranked lists. In this way, the variation of average cohesion is very low and the convergence is reached very quickly.
2. *“Real-world” descriptor*: for a real-word descriptor, the ranked list  $R_{ki}$  may include some incorrect results, that is, some non-similar images are found at the top  $K$  positions of  $R_{ki}$ . Let  $img_j \in R_{ki}$  be an image non-similar to  $img_i$ . In that case, recommendations defined for ranked lists of similar images to  $img_i$  can improve  $R_{ki}$ , by moving the non-similar image  $img_j$  out of the first positions of  $R_{ki}$ . In other words, when correct results represent the common case, the recommendation method can improve ranked lists. While these improvements occur, the average cohesion of ranked lists increase. That process is repeated until convergence is reached.
3. *“Non-effective” descriptor*: for non-effective descriptors, the created ranked lists can be seen as a result of a random permutation of images. In that case, the method convergence would be slow as the number of similar images found at the top positions of ranked lists are very small.

An experimental analysis of convergence is presented in Section 5.7.4.

## 5.6 The Rank Aggregation Algorithm

Recently, several methods have been proposed aiming at combining ranked lists produced by different descriptors. The objective is to produce more effective results [4, 28, 95]. We propose the use of our re-ranking algorithm for combining descriptors (rank aggregation).

Let  $\mathcal{C}$  be an image collection and let  $\mathcal{D} = \{D_1, D_2, \dots, D_m\}$  be a set of CBIR descriptors. We can use the set of descriptors  $\mathcal{D}$  for computing a set of distance matrices  $\mathcal{A} = \{A_1, A_2, \dots, A_m\}$ . Our approach for descriptor combination works as follows. The first step is to combine the set  $\mathcal{A}$  in a unique matrix  $A_c$ . For the matrices combination we use a multiplicative approach. Every  $(i, j)$  position of matrix is computed as follows:

$$A_c[i, j] = A_1[i, j] \times A_2[i, j] \times \dots \times A_m[i, j]. \quad (5.3)$$

By multiplying the distances between the same images (say  $img_i$  and  $img_j$ ) considering different descriptors, high distances obtained by one descriptor will be propagated to the others, leading to high aggregate values. Another reasoning behind the multiplication approach is inspired by the Naïve Bayes classifiers [118]. In a general way, the Naïve Bayes works based on the probability of an instance  $E$  be of a class  $c$ , given a set of features, assuming conditional independence among features. In a simplified manner, a Naïve Bayes classifier assumes that the presence of a particular feature of a class is unrelated to the presence (or absence) of any other feature. Under the independence assumption, the probabilities of each feature be of a given class are multiplied. In this case, as an analogy, the proposed multiplication approach can be seen as the computation of the probability of images  $img_i$  and  $img_j$  are non-similar, considering independent features (CBIR descriptors).

After the multiplication step, once we have a combined matrix  $A_c$ , we compute a set of ranked lists  $\mathcal{R}_c$  based on this matrix. Then, we perform the Pairwise Recommendation algorithm now using the matrix  $A_c$  and the set  $\mathcal{R}_c$ .

## 5.7 Experimental Evaluation

In this section, we present a set of conducted experiments for demonstrating the effectiveness of our method. We analyzed and compared our method under several aspects.

Section 5.7.1 aims at evaluating the impact of different values used for the method parameters with regard to effectiveness and efficiency criteria. Section 5.7.2 presents results concerning the use of our method to several shape descriptors, considering the MPEG-7 dataset [48]. Sections 5.7.3 aims at validating the hypothesis that our method can be used in general image retrieval tasks. In addition to shape descriptors, we conduct experiments with color and texture descriptors. The objective of the experiments presented in these sections is to assess the effectiveness of the method considering different visual properties and different datasets. Section 5.7.4 discusses convergence aspects of the re-ranking method.

Finally, Section 5.7.5 presents experimental results of our re-ranking method when

used to combine descriptors in rank aggregation tasks. We conducted experiments for shape, color, and texture descriptors.

All experiments were conducted considering all images in the collections as query images. Results presented (in terms of MAP and Recall@40 scores) represent an average scores considering all queries.

### 5.7.1 Experiment 1 - Impact of Parameters

The execution of Algorithm 5.1 considers three parameters: (i)  $K$  - number of initial neighbors considered for recommendations; (ii)  $L$  - a constant that controls the influence of weights; and (iii)  $\epsilon_{cohesion}$  - the threshold parameter in the convergence criterion (which determines the number of  $T$  iterations along which the algorithm is executed). In order to evaluate the influence of different parameter settings on the retrieval scores and for determining the best parameters values we conducted a set of experiments. We use the MPEG-7 dataset [48] with the bullseye score measure. For distance computation, we used the CFD [68] shape descriptor. Retrieval scores are computed ranging parameters  $K$  in the interval  $[1,15]$  and  $T$  in the interval  $[1,30]$  (with increments of 5) for each value of  $L$ . Figures 5.4 and 5.5 show surfaces that represent retrieval scores for  $L$  equal to 1 and 2, respectively. We can observe optimal combinations of values for regions close to  $K = 8$  and  $T = 15$ , for which the best retrieval scores are observed. In the following experiments, parameters are set to  $K = 8$  and  $\epsilon_{cohesion} = 0.0125$  (threshold that reaches convergence in about 15 iterations). Note that these parameters were defined considering a single descriptor/dataset, but they were used in all conducted experiments with good results.

Figure 5.6 shows the impact of different values of  $L$  in the method's precision. We fixed the values of  $K = 8$  and  $T = 15$  and computed the retrieval scores for  $L$  in the interval  $[0,3]$ . In this case, the best retrieval score was reached for  $L = 2$ . The value of  $L = 2$  indicates that a high weight can be assigned to the recommendations.

Finally, we analyze the impact of parameters  $K$  and  $T$  on computation time. Figure 5.7 illustrates a surface representing the variation of computation time as a function of  $K$  and  $T$ . We can observe a approximate quadratic behavior for the surface. The computation time of recommendation process increases proportionally to  $K^2 \times T$ . Note that, although the asymptotic complexity of recommendation and clustering steps is quadratic for parameters  $K$  and  $T$ , it is linear ( $O(N)$ ) for the size of collection  $N$ , since the recommendations are considered for  $K \ll N$  images.

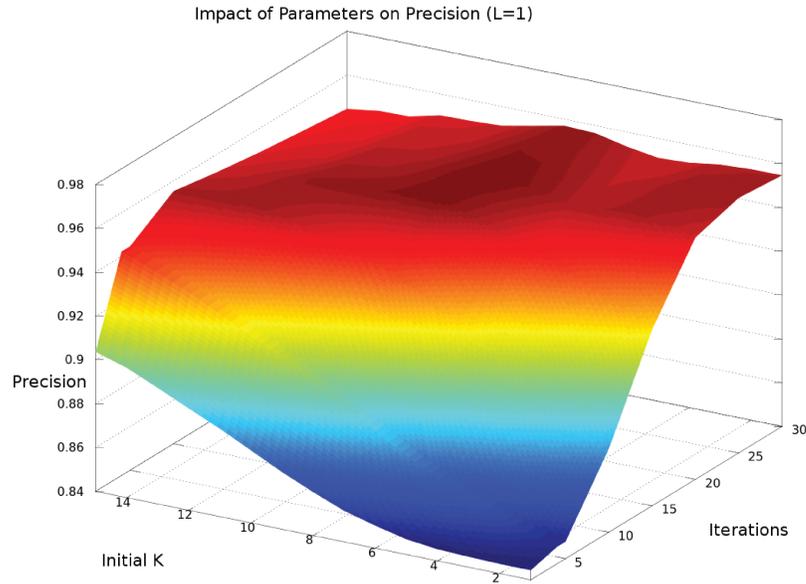
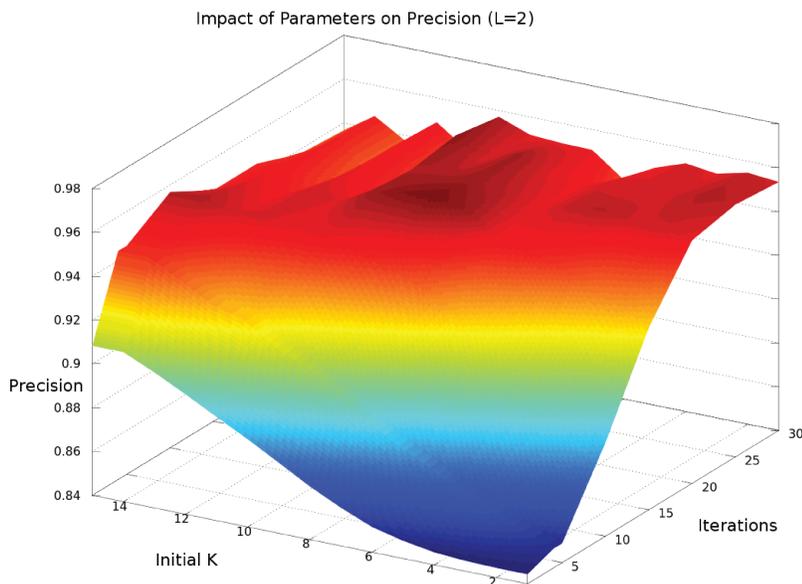


Figure 5.4: Impact of parameters  $K$  and  $T$  for  $L=1$ .

### 5.7.2 Experiment 2 - Shape Descriptors

We evaluate the use of our method with six shape descriptors: Segment Saliences (SS) [17], Beam Angle Statistics (BAS) [2], Inner Distance Shape Context (IDSC) [52], Contour Features Descriptor (CFD) [68], Aspect Shape Context (ASC) [53], and Articulation-Invariant Representation (AIR) [35]. This experiment considered the MPEG-7 dataset and the bullseye score. Parameters are set according to experimental analysis presented in the previous section:  $K = 8$ ,  $\epsilon_{cohesion} = 0.0125$ , and  $L = 2$ . Figure 5.8 illustrates an example of results comparison for a MPEG-7 shape. The comparison considers the CFD [68] shape descriptor before and after the use of the proposed re-ranking method. The first row presents the retrieval results for the CFD [68] shape descriptor (first image as a query). The second row presents retrieval results for the same shape descriptor after using the Pairwise Recommendation re-ranking algorithm.

Results of bullseye score for all descriptors are presented in Table 5.1. Note that the effectiveness gains are always positive and represent very significant improvement of effectiveness, ranging from +7.97% to +23.57%. In Figure 5.9, we report the percentage gains obtained by the Pairwise Recommendation algorithm for each of 70 shape classes in the MPEG-7 dataset. Note that bullseye score was improved by over 10% on average, and over 50% for two classes.

Figure 5.5: Impact of parameters  $K$  and  $T$  for  $L=2$ .Table 5.1: Pairwise Recommendation for Shape Descriptors on the MPEG-7 dataset ( $Recall@40$ ).

Shape Descriptor	Score	Pairwise Recommendation	Gain
SS [17]	43.99%	54.36%	+23.57%
BAS [2]	75.20%	84.03%	+11.74%
IDSC [52]	85.40%	92.21%	+7.97%
CFD [68]	84.43%	96.15%	+13.88%
ASC [53]	88.39%	94.66%	+7.09%
AIR [35]	93.67%	99.36%	+6.15%

### 5.7.3 Experiment 3 - General CBIR Tasks

Our goal here is to evaluate the use of our method for several CBIR tasks involving shape, color, and texture descriptors. The measure adopted is *Mean Average Precision (MAP)*. Results are presented in Table 5.2. As we can observe, the *Pairwise Recommendation Re-Ranking* method presents positive effectiveness gains for all descriptors (including shape, color, and texture), ranging from +1.27% to +20.47%. We conducted a paired t-test and conclude that there is a 99% of chance of difference between the means (before and after the re-ranking) being statistical significantly.

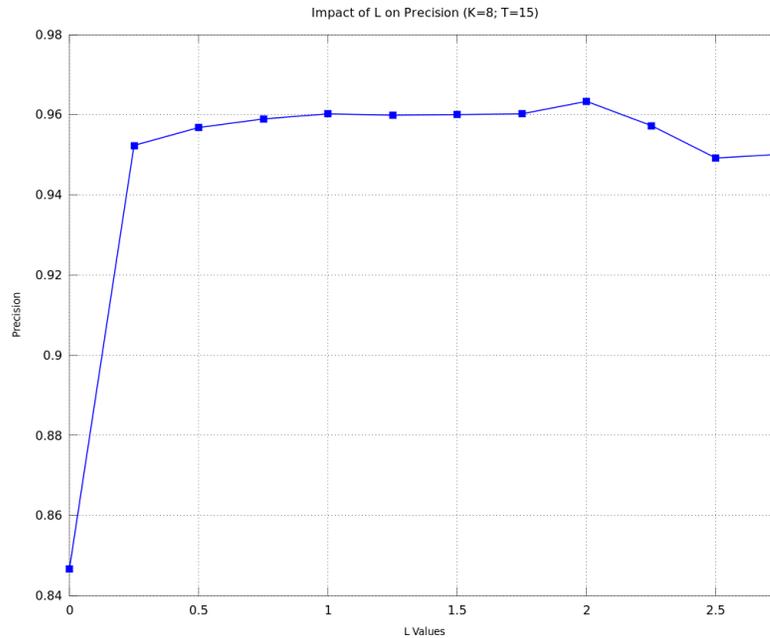


Figure 5.6: Impact of parameter L (K=8, T=15).

### Shape Descriptors

For the experiments with the shape collection, we used the same descriptors and dataset considered for previous section, using MAP as score. Results are similar to those obtained considering bulleyes score, with positive gains ranging from +5.92% to +13.22%.

### Texture Descriptors

In this section we aim at validating our method in image retrieval tasks using texture descriptors. The experiments consider three texture descriptors: LBP [60], CCOM [46], LAS [94]. We used the Brodatz [7] dataset, for texture descriptors evaluation. For parameters setting we use (both for texture and color descriptors) the same values used for shape descriptors (in this experiment, we take  $L = 1$ ). Our re-ranking method presents positive gains ranging from +7.27% to 15.44%.

### Color Descriptors

We evaluate our method for three color descriptors: BIC [90], ACC [41], and Global GCH [93]. The experiments were conducted on the Soccer dataset [100]. We can observe a positive gain for all color descriptors ranging from 0.34% to 8.61% (considering MAP as score).

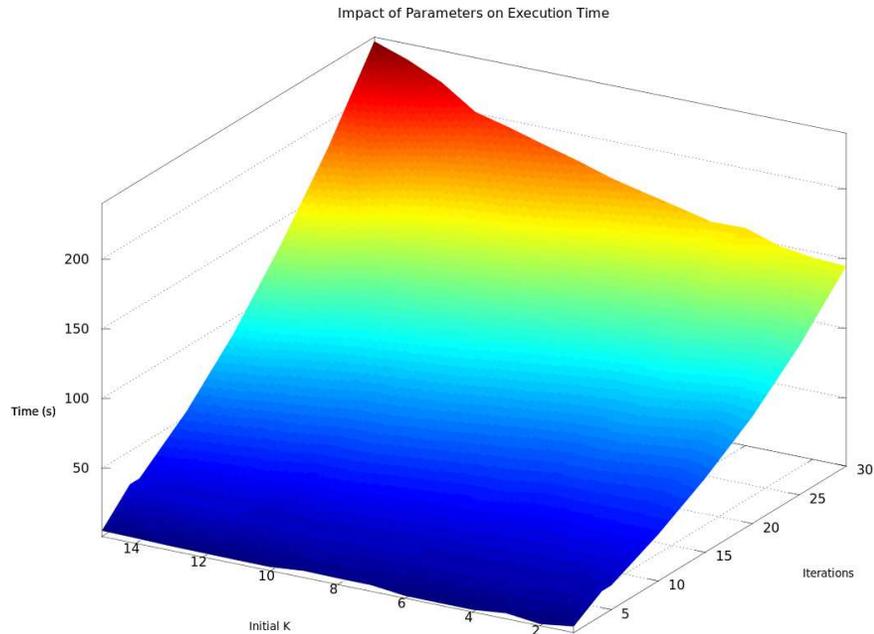


Figure 5.7: Impact of parameters on execution time.

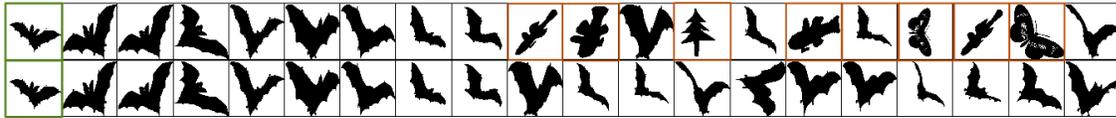


Figure 5.8: Impact of Pairwise Recommendation re-ranking on ranked lists.

#### 5.7.4 Experiment 4 - Analysis of Convergence

This section aims at discussing and experimentally evaluating the convergence of the proposed re-ranking method.

Figure 5.10 shows the evolution of cohesion measure, whose variation is used as convergence criterion. We considered three different descriptors/datasets: the CFD [68] shape descriptor on the MPEG-7 dataset, the BIC [90] color descriptor on the Soccer dataset, and LAS [94] texture descriptor on the Brodatz dataset. We can observe a similar behavior for the three curves: at the beginning, the cohesion measure increases quickly and, at the end, it converges for a constant value. As discussed in Section 5.5, in scenarios with less effective descriptors, the converge is slower. That can be observed for the BIC [90] descriptor on the Soccer dataset, which has the lowest effectiveness performance.

Besides cohesion measure, we also consider the difference between ranked lists along iterations. Intuitively, we consider that an iterative re-ranking algorithm converges if, after a certain number of iterations, it produces a small number of changes in the generated ranked lists. More formally, we consider a definition of  $\varepsilon$ -convergence for rankings

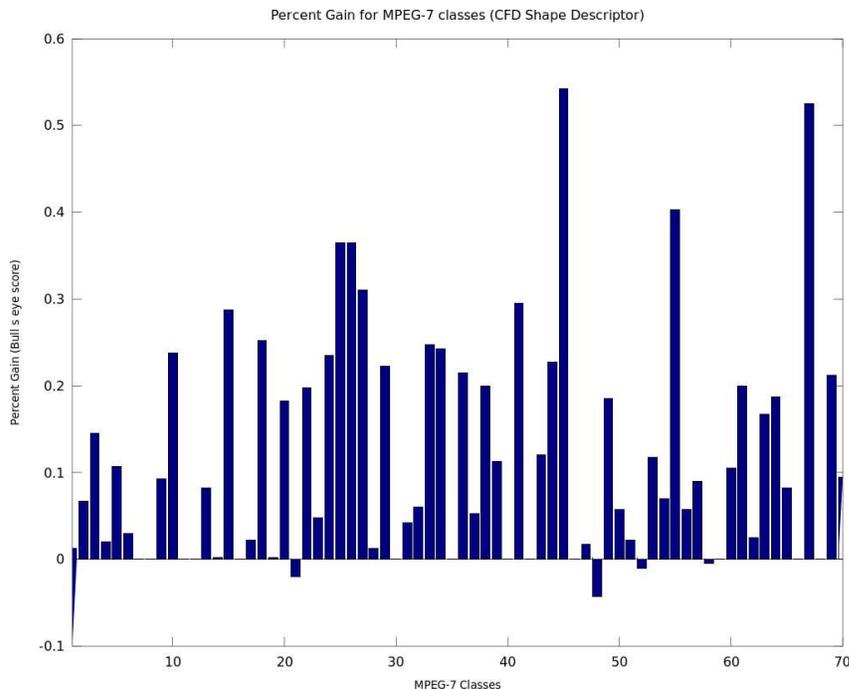


Figure 5.9: Percentage gains in bulls-eye score for each class of MPEG-7 dataset considering CFD [68] shape descriptor.

presented in [78]:

Let  $\mathcal{C}=\{img_1, img_2, \dots, img_N\}$  be an *image collection* and let  $R$  be a *ranked list*  $R=(img_1, img_2, \dots, img_N)$ , which can be defined as a permutation of the collection  $\mathcal{C}$ . An iterative ranking algorithm that generates a ranked list  $R(t)$  at each iteration  $t$ ,  $\varepsilon$ -converges in (at most)  $T$  iterations in a metric  $d(\cdot, \cdot)$ , if there exists a ranked list  $R$  such that, for every  $t \geq T$ ,  $d(R(t), R) < \varepsilon$ .

A natural distance metric to use for this definition is the Kendall's tau metric. This metric turns out to be equal to the number of exchanges needed in a bubble sort to convert one permutation to the other. The use of this metric for comparing top-k lists is detailed in [27] and a definition is presented in Equation 7.11.

In this scenario, we have measured the evolution of Kendall's tau distance between rankings at each iteration for the three descriptors. For measuring the Kendall's tau distance, we considered the  $2 \times K$  top images of ranked lists (same size considered for cohesion measure). Figure 5.11 shows the evolution of average Kendall's tau distance between rankings along iterations.

The results obtained by using the Kendall's tau distance is consistent with the cohesion measure evolution (Figure 5.10). The Kendall's tau distance *decreases* at the same pace as the cohesion measure *increases*. A high distance can be observed at first iterations,

Table 5.2: Pairwise Recommendation Evaluation on Several Content-Based Image Retrieval Tasks (*MAP*).

<b>Image Descriptor</b>	<b>Type</b>	<b>Dataset</b>	<b>Score (<i>MAP</i>)</b>	<b>Pairwise Recomm.</b>	<b>Gain</b>
SS [17]	Shape	MPEG-7	37.67%	39.90%	+5.92%
BAS [2]	Shape	MPEG-7	71.52%	77.65%	+8.57%
IDSC [52]	Shape	MPEG-7	81.70%	86.83%	+6.28%
CFD [68]	Shape	MPEG-7	80.71%	91.38%	+13.22%
ASC [53]	Shape	MPEG-7	85.28%	89.55%	+5.01%
AIR [35]	Shape	MPEG-7	89.39%	94.71%	+5.95%
GCH [93]	Color	Soccer	32.24%	32.35%	+0.34%
ACC [41]	Color	Soccer	37.23%	40.31%	+8.27%
BIC [90]	Color	Soccer	39.26%	42.64%	+8.61%
LBP [60]	Texture	Brodatz	48.40%	51.92%	+7.27%
CCOM [46]	Texture	Brodatz	57.57%	66.46%	+15.44%
LAS [94]	Texture	Brodatz	75.15%	80.73%	+7.43%

indicating a lot of changes in the ranked lists. After some iterations, the convergence criterion is reached (distances get lower values).

The same results were observed for the other descriptors used in the three image collections. On average, all descriptors converged in 17 iterations.

### 5.7.5 Experiment 5 - Rank Aggregation

This section aims at evaluating the use of our re-ranking method to combine different CBIR descriptors. We selected two descriptors for each visual property. Descriptors with best effectiveness results were selected. Table 5.3 presents the results of MAP score for these descriptors. We observe significant gains compared with the use of each descriptor in isolation.

Figure 5.12 illustrates the Precision  $\times$  Recall curves of shape descriptors CFD [68] and IDSC [52] using the MPEG-7 dataset. It considers different situations: before and after using the Pairwise Recommendation Re-Ranking and after the use of the rank aggregation approaches that uses the Pairwise Recommendation algorithm. For rank aggregation tasks, we obtain 99.52% considering the bullseye score (95% confidence interval: 99.22%, 99.82%).

Finally, we analyze the impact of our combination method on the distance matrix. Figure 5.13 illustrates a subset ( $200 \times 200$ ) of distance matrices for the MPEG-7 dataset considering descriptors CFD [68], IDSC [52], and the combination using the Pairwise

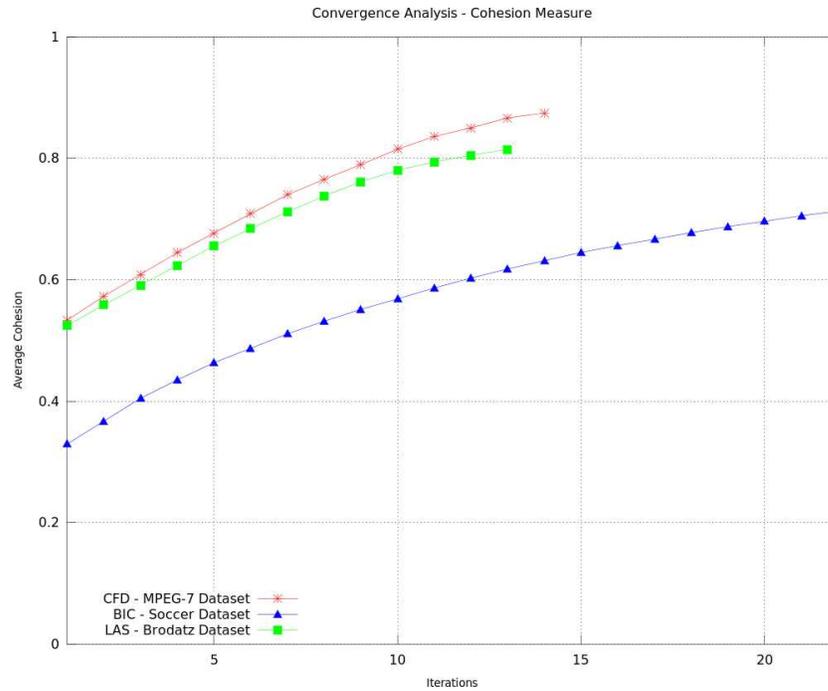


Figure 5.10: Analysis of convergence - Cohesion measure.

Recommendation algorithm. The dark pixels indicate low distances between images. In the matrix which represents the combination using the Pairwise Recommendation algorithm, we can observe very distinct squares that illustrate the low distances among shapes from the same classes.

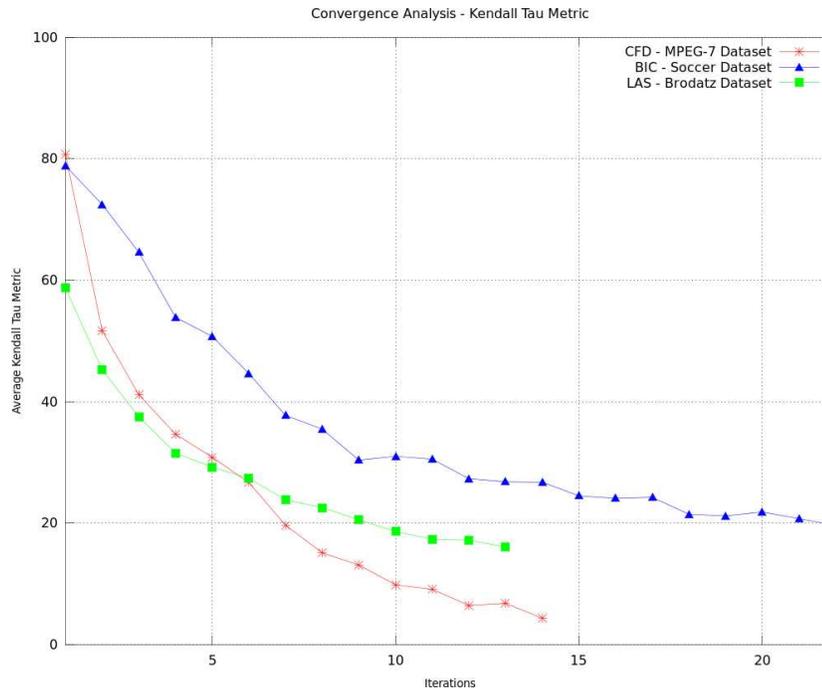


Figure 5.11: Analysis of convergence - Kendall's tau distance.

Table 5.3: Pairwise Recommendation for Descriptors Combination (*MAP*).

Descriptor	Type	Dataset	Score[%]
CFD [68]	Shape	MPEG-7	80.71%
IDSC [52]	Shape	MPEG-7	81.70%
<b>CFD [68] + IDSC [52]</b>	<b>Shape</b>	<b>MPEG-7</b>	<b>98.78%</b>
ACC [41]	Color	Soccer	37.23%
BIC [90]	Color	Soccer	39.26%
<b>ACC [41] + BIC [90]</b>	<b>Color</b>	<b>Soccer</b>	<b>42.20%</b>
CCOM [46]	Texture	Brodatz	63.67%
LAS [94]	Texture	Brodatz	75.15%
<b>LAS [94] + CCOM [46]</b>	<b>Texture</b>	<b>Brodatz</b>	<b>79.91%</b>

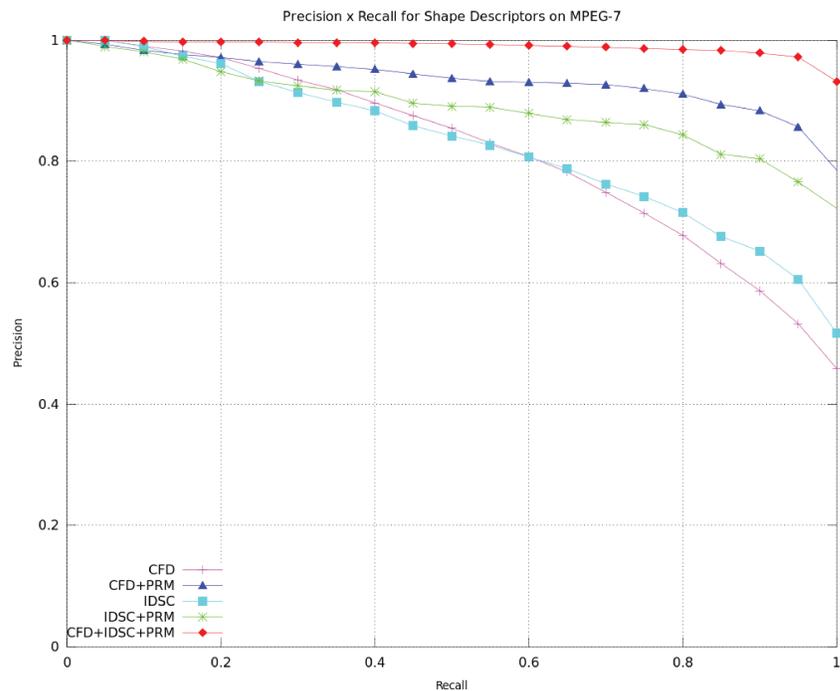


Figure 5.12: Pairwise Recommendation for Shape Descriptors on the MPEG-7 dataset.

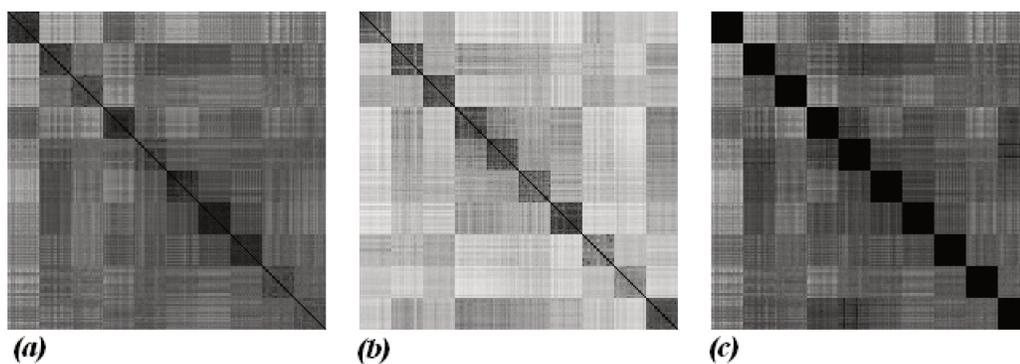


Figure 5.13: Distance matrices by shape descriptors on the MPEG-7 dataset: (a) CFD [68]; (b) IDSC [52]; (c) CFD+IDSC+Pairwise Recommendation.



# Chapter 6

## Contextual Spaces

In this chapter, we present two novel re-ranking approaches that take into account *contextual information* defined by the K-Nearest Neighbours (KNN) of a query image. We propose the creation of *contextual spaces* for analysing the contextual information, *i.e.*, for characterizing the *local* context of a query image defined by its neighbourhood. A contextual space is constructed considering the most similar images to a query image. Later, new distances are computed by taking into account the distances (encoded in those contextual spaces) among these neighbours to other collection images. The image collection is re-ranked based on the new distances and this process is repeated along iterations.

This chapter presents two variations of our re-ranking method: the *Contextual Spaces based on KNN* and *Contextual Spaces based on Mutual-KNN* algorithms. The main difference between them relies on the method for selecting the  $K$  most similar images which are used for constructing the contextual spaces. As any other general re-ranking approach, the proposed methods can be applied to any distance matrix, whether those distances are calculated from a single image descriptor or from some combination of a number of distances.

The main contributions of this chapter are:

- the definition of the concept of contextual spaces for encoding contextual information of images;
- the definition of two new re-ranking algorithms that exploit contextual information encoded in contextual spaces;
- the evaluation of the proposed algorithms in several CBIR tasks related to the combination of image descriptors; combination of visual and textual descriptors; and combination of post-processing (re-ranking) methods.

Different from other methods [66, 68, 69], this approach does not consider the costs involved with clustering strategies used to update the distances among images. The

contextual spaces proposed here are similar to the context images proposed in [67] to encode contextual information. One advantage of the proposed approach when compared to [45, 113, 114] relies on its flexibility in the sense it can be easily tailored to other CBIR tasks such as: rank aggregation, multimodal retrieval, combination of post-processing methods.

We conducted a large evaluation protocol involving visual descriptors (considering shape, color, and texture) and textual descriptors, various datasets, and comparisons with other post-processing methods. Experimental results demonstrate the effectiveness of our approaches. The proposed re-ranking algorithms yield better results in terms of effectiveness performance than various post-processing and rank aggregations methods recently proposed in the literature.

This chapter is organized as follows: Section 6.1 introduces the re-ranking algorithms based on contextual spaces. Section 6.2 presents the experimental evaluation.

## 6.1 The Contextual Spaces Algorithms

This section presents the *Contextual Spaces Algorithms*. Section 6.1.1 describes the contextual spaces representation. Section 6.1.2 presents the re-ranking algorithms. Section 6.1.3 discusses the impact of re-ranking algorithms on distances among images. Section 6.1.4 describes our rank aggregation method.

### 6.1.1 Contextual Spaces Representation

Consider the bidimensional space  $\mathbb{R}^2$  constructed by taking into account pairwise image distances defined by function  $\rho : \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$ . We can use this space for analyzing the similarity of collection images with regard to two arbitrary images  $img_i, img_j \in \mathcal{C}$  (these images are used as reference). Consider a graphic representation of the image collection  $\mathcal{C}$  on a Cartesian coordinate system. Let  $img_l \in \mathcal{C}$  be a collection image, we can represent  $img_l$  as a point on the plane, considering its distances to the reference images  $img_i$  and  $img_j$ .

Given two reference images  $img_i$  and  $img_j$ , we can consider a plane where the  $x$  axis represents the distances of collection images with regard to image  $img_i$  and the  $y$  axis represents the values of distances of collection images with regard to  $img_j$ . The position of an image  $img_l \in \mathcal{C}$  is given by the ordered pair  $(\rho(img_i, img_l), \rho(img_j, img_l))$ , where  $\rho(img_i, img_l)$  and  $\rho(img_j, img_l)$  are the distances of  $img_l$  to the reference images  $img_i$  and  $img_j$ , respectively. We can use this same approach to determine the position of all collection images.

Figure 6.2 shows the graphic representation of an image collection (MPEG-7

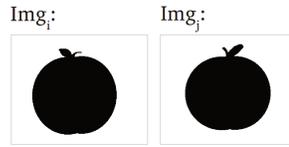


Figure 6.1: Similar reference images.

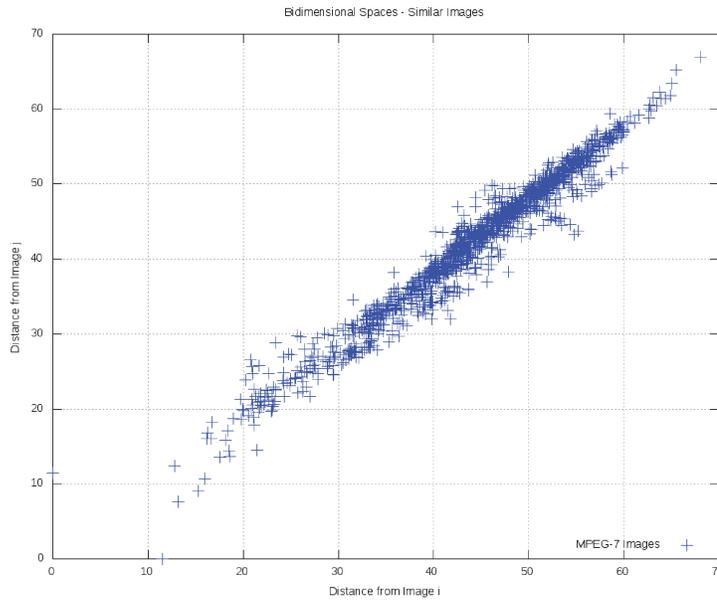


Figure 6.2: Bidimensional space representation for two similar images.

dataset [48]) by taking into account two reference images which are very similar (Figure 6.1). In this example we have used the CFD descriptor [68] to compute the distances among images. As the reference images are similar, their distance to other collection images are similar as well.

Figure 6.4 shows the same representation, now considering two reference images which are not similar (Figure 6.3), according to the same descriptor. Note that the two graphic representations present very distinct characteristics.

Our goal consists in exploiting the information of these bidimensional spaces. In this way, we propose the concept of *contextual spaces*. The *contextual spaces* are bidimensional spaces considering as reference images an arbitrary image  $img_i$  and each one of its  $K$  selected neighbours. We aim at exploiting *contextual spaces* for image re-ranking tasks.

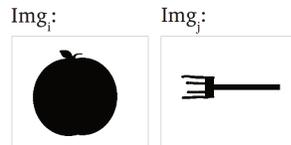


Figure 6.3: Non-similar reference images.

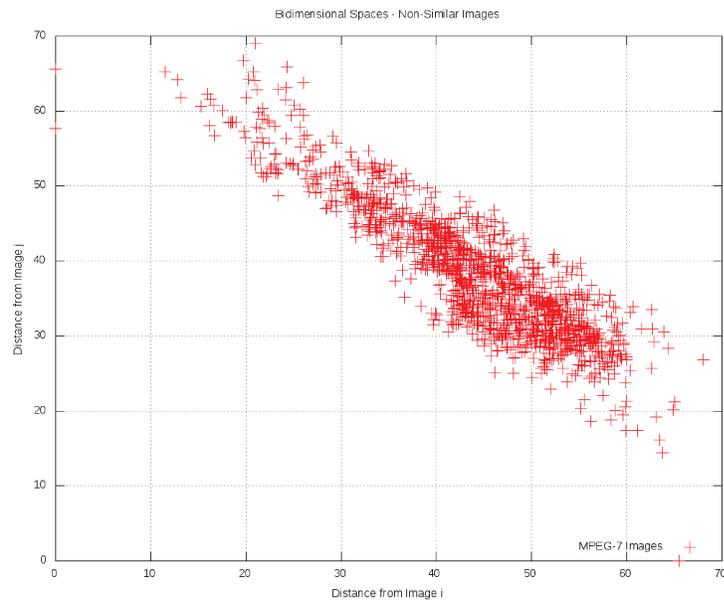


Figure 6.4: Bidimensional space representation for two non-similar images.

### 6.1.2 The Re-Ranking Algorithms

The re-ranking algorithms based on *contextual spaces* aim at redefining distances among images by taking into account information encoded in the distance matrix  $A$  and in the set of ranked lists  $\mathcal{R}$ . Based on the new distances, new ranked lists can be computed.

The main motivation of our re-ranking algorithm relies on the following question: “Given a query image, what information can similar images provide about other collection images?”

In our method, this question is answered by exploiting *contextual spaces*. The reasoning behind the use of *contextual spaces* consists in taking into account relationships of images in the context of the query (and not only pairwise distances). Given a query image, a selected set of similar images to the query is considered for constructing the contextual spaces. In the following, information encoded in these contextual spaces is used for computing a new distance from the query image to other collection images. In this way, distances from an image to other collection images are redefined considering the distances to their similar images.

Since the distances among images are redefined so as to become more effective, the process is repeated iteratively, aiming at further improving the effectiveness along iterations. Next sections detail two algorithms based on contextual spaces, which differ from each other mainly on the approach for selecting the similar images that will be used for constructing the contextual spaces.

#### Contextual Spaces based on KNN (CS-KNN)

The *Contextual Spaces based on KNN* algorithm considers the  $K$ -nearest neighbours as similar images for constructing the contextual spaces. Algorithm 6.1 presents the main steps of the proposed re-ranking algorithm. Let  $img_i$  be a collection image ( $img_i \in \mathcal{C}$ ). The response set for  $K$ -nearest neighbours  $KNN(img_i)$  is formally defined as follows:

$$KNN(q) = \{\mathcal{R} \subseteq \mathcal{C}, |\mathcal{R}| = K \wedge \forall x \in \mathcal{R}, y \in \mathcal{C} - \mathcal{R} : \rho(q, x) \leq \rho(q, y)\}. \quad (6.1)$$

Given an image  $img_i \in \mathcal{C}$ , for each of its  $K$  most similar images  $img_j \in KNN(img_i)$ , a *contextual space* is constructed aiming at computing new distances from  $img_i$  to other collection images. New distances are computed by combining information of  $K$  contextual spaces in a unique space, defined by two dimensions:  $d_i$  and  $d_j$ . Let  $img_l$  be another collection image whose distance to  $img_i$  we want to compute. The dimension  $d_i$  represents the distance between  $img_i$  and  $img_l$ . The dimension  $d_j$  represents the distance information from neighbours  $img_j \in KNN(img_i)$  to  $img_l$ .

Algorithm 6.1 shows the main steps of the method. For computing a new distance  $A_{t+i}[i, l]$  between images  $img_i$  and  $img_l$  (where  $t$  denotes the iteration being executed), the re-ranking algorithm considers the position of image  $img_l$  in this merged space, given by dimensions  $d_i$  and  $d_j$  (Line 17 of Algorithm 6.1). Images close to the origin (low distances between  $img_i$  and its neighbors) present the lowest scores.

Basically, at each iteration, given any two images  $img_i, img_l \in \mathcal{C}$ , the distance  $A_{t+1}[i, l] = \rho(img_i, img_l)$  is redefined. This is done by using information of contextual spaces defined for each  $img_j \in KNN(img_i)$ .

Note that, in Line 11 of Algorithm 6.1, the dimension  $d_j$  (distance from neighbours of  $img_i$ ) is incremented with weighted distances, computed between each  $img_j \in KNN(img_i)$  and  $img_l$ . In this way, we consider that the elements of  $KNN(img_i)$  are retrieved in increasing order of distances. The concept behind these weights (given by term  $K - c_k$ ) consists in considering as more relevant, distances from the first neighbours. In Line 16, the dimension  $d_j$  is computed dividing the accumulated weighted distances by the sum of the weights, given by the arithmetic sum. The dimension  $d_i$  is computed based on the distance from  $img_i$  to  $img_l$  divided proportionally by the number of neighbours considered in the iteration (Line 15 of Algorithm 6.1). The new distance  $A_{t+1}[i, l]$  is computed in Line 17, based on the orthogonal dimensions  $d_i$  (distance from  $img_i$ ) and  $d_j$  (distance from neighbours of  $img_i$ ).

Once the distances among images are redefined, a set of ranked lists is computed and the re-ranking algorithm is performed again, in an iterative way. At each iteration  $t$ , we increment the number of  $K$  neighbors considered for constructing the contextual spaces (Line 21 of Algorithm 6.1). The motivation behind this increment relies on the fact that the effectiveness of ranked lists increases along iterations. In this way, non-relevant images are moved out from the first positions of the ranked lists and  $K$  can be increased for considering more images.

The algorithm has two parameters: the initial number of neighbours  $K_s$  to be considered, and the final number of neighbours  $K_e$ . Note that the difference between these parameters defines the number of iterations of the algorithm ( $t = K_e - K_s$ ).

### Contextual Spaces based on Mutual-KNN (CS-MKNN)

Both *Contextual Spaces based on KNN* and *Contextual Spaces based on Mutual-KNN* re-ranking algorithms aim at redefining distance among images based on contextual spaces. The main difference between them relies on the method for selecting the  $K$  most similar images which are used for constructing the contextual spaces. The *Contextual Spaces based on Mutual-KNN* considers not only the information of nearest neighbours, but also the mutual reference between images and their neighbours in their ranked lists.

Algorithm 6.2 presents the main steps of *Contextual Spaces based on Mutual-KNN*

---

**Algorithm 6.1** Re-Ranking Algorithm: Contextual Spaces KNN
 

---

**Require:** Original distance matrix  $A$ ; set of ranked lists  $\mathcal{R}$ ; parameters  $K_s$  and  $K_e$

**Ensure:** Processed distance matrix  $\hat{A}$ ; set of ranked lists  $\hat{\mathcal{R}}$

```

1:  $t \leftarrow 0$ 
2:  $A_t \leftarrow A$ 
3:  $K \leftarrow K_s$ 
4: while  $K \leq K_e$  do
5:    $A_{t+1} \leftarrow 0$ 
6:   for all  $img_i \in C$  do
7:     for all  $img_l \in C$  do
8:       {Considering contextual spaces}
9:        $c_k \leftarrow 0; d_j \leftarrow 0$ 
10:      for all  $img_j \in KNN(img_i)$  do
11:         $d_j \leftarrow d_j + A[j, l] \times (K - c_k)$ 
12:         $c_k \leftarrow c_k + 1$ 
13:      end for
14:      {Computing distance  $\rho(img_i, img_l)$ }
15:       $d_i \leftarrow A[i, l]/K$ 
16:       $d_j \leftarrow d_j / (\frac{K \times (K-1)}{2})$ 
17:       $A_{t+1}[i, l] \leftarrow \sqrt{d_i^2 + d_j^2}$ 
18:    end for
19:  end for
20:   $\mathcal{R}_{t+1} \leftarrow performReRanking(A_{t+1})$ 
21:   $K \leftarrow K + 1$ 
22:   $t \leftarrow t + 1$ 
23: end while
24:  $\hat{A} \leftarrow A_t$ 
25:  $\hat{\mathcal{R}} \leftarrow \mathcal{R}_t$ 

```

---

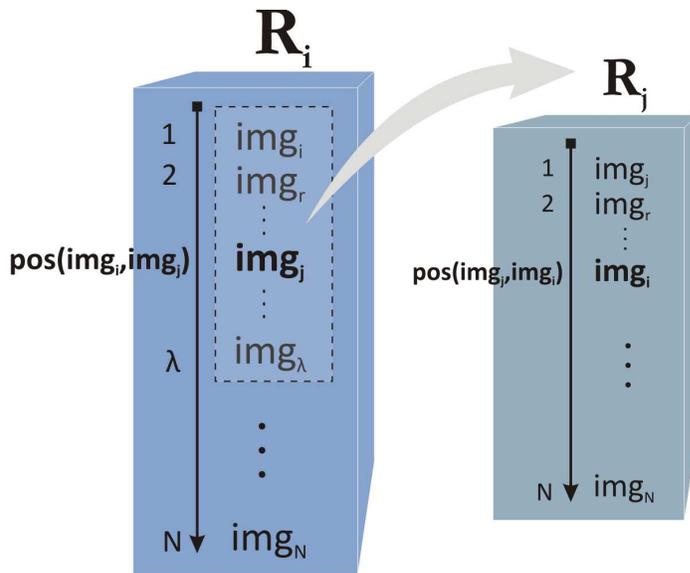


Figure 6.5: Contextual Spaces based on Mutual-KNN: selection of images for constructing the contextual spaces.

algorithm. Line 9 refers to the method for selecting the similar images, which is detailed in the following. Let  $img_i \in \mathcal{C}$  be the image whose the  $K$  most similar images we want to select. The main idea for the selection approach consists in using a  $\lambda > K$  parameter for filtering a subset of the ranked list and selecting images in this subset which refer to  $img_i$  at the first positions of their ranked list. We describe this approach in the following. Let  $pos(img_j, img_i)$  be the position of  $img_j$  in the ranked list of  $img_i$  and let  $pos(img_i, img_j)$  be the position of image  $img_i$  in the ranked list of  $img_j$ . The function  $pos(img_j, img_i)$  is defined as the size of the set of images whose distance to  $img_i$  is less than the distance from  $img_i$  to  $img_j$ . Figure 6.5 illustrates the ranked lists  $R_i$  and  $R_j$ , the parameter  $\lambda$ , and the positions  $pos(img_j, img_i)$  and  $pos(img_i, img_j)$  in these ranked lists. The image  $img_j$  is considered one of the  $K$  most similar images of  $img_i$  if:

1. the position  $pos(img_j, img_i) < \lambda$ ;
2. the value of  $pos(img_j, img_i) + pos(img_i, img_j)$  is one of the  $K$  lowest values obtained, considering the ranked list of image  $img_i$ .

We can formally define the  $pos$  function and the Mutual-KNN set as follows:

$$pos(q, x) = | \{ \mathcal{R} \subseteq \mathcal{C}, \forall x \in \mathcal{R}, y \in \mathcal{C} - \mathcal{R} : \rho(q, x) \leq \rho(q, y) \} |, \quad (6.2)$$

$$MKNN(q) = \{\mathcal{R} \subseteq \mathcal{C}, |\mathcal{R}| = K \wedge \forall x \in \mathcal{R} : (pos(q, x) \leq \lambda \wedge \forall x \in \mathcal{R}, y \in \mathcal{C} - \mathcal{R} : (pos(q, x) + pos(x, q)) \leq (pos(q, y) + pos(y, q)))\}. \quad (6.3)$$

We consider that the elements of  $MKNN(img_i)$  are retrieved in increasing order of distances. Once the  $K$  most similar images have been selected, a contextual space is constructed for each selected image, in the same way of *Contextual Spaces based on KNN* algorithm. New distances are computed merging information of  $K$  contextual spaces in an unique space, defined by two dimensions:  $d_i$  and  $d_j$ . The dimension  $d_i$  represents the distance information from  $img_i$  to  $img_l$ . The dimension  $d_j$  represents the distance information from selected similar images  $img_j \in MKNN(img_i)$  to  $img_l$ .

The new distance is computed based on values of  $d_i$  and  $d_j$  (Line 17 of Algorithm 6.2). The dimension  $d_j$  (distance from neighbours of  $img_i$ ) is computed as a weighted average from  $img_j \in MKNN(img_i)$  to  $img_l$ . High weights (given by the term  $K - c_K$ ) will be assigned to images from set  $MKNN(img_i)$  which refer and are referred by image  $img_i$  at the first positions of their ranked lists. The dimension  $d_i$  is computed based on the distance between  $img_i$  and  $img_l$  divided proportionally by the number of neighbours considered in the iteration.

As the *Contextual Spaces based on KNN* algorithm, at each iteration, given any two images  $img_i, img_l \in \mathcal{C}$ , the distance  $A_{t+1}[i, l] = \rho(img_i, img_l)$  is redefined. Once the distances among images are redefined, a set of ranked lists can be computed and a re-ranking can be performed again, in an iterative way.

Note that the parameter  $K$  (number of images used for constructing the contextual spaces) is not incremented at each iteration (as in *Contextual Spaces based on KNN*). That happens because the method used for selecting similar images (considering mutual reference in ranked lists) consider more information than using only the nearest neighbours and, therefore, can start considering a larger number of neighbours. In this way, the number of iterations of the algorithm is executed is not related to the size of  $K$ .

For *Contextual Spaces based on Mutual-KNN*, three parameters are considered by the algorithm: (i)  $K$ : the number of similar images used for constructing the contextual spaces; (ii)  $\lambda$ : the maximum position considered for selecting the similar images; and (iii)  $T$ : number of iterations along which the algorithm is executed.

### 6.1.3 Impact of Re-Ranking Algorithms on Distances

This section aims at analysing the impact of the proposed algorithms on the distribution of distances among images in a given dataset. For this analysis, we construct the same bidimensional representation presented in Section 6.1.1. We considered the MPEG-7

---

**Algorithm 6.2** Re-Ranking Algorithm: Contextual Spaces Mutual-KNN
 

---

**Require:** Original distance matrix  $A$ ; set of ranked lists  $\mathcal{R}$ ; parameters  $K$ ,  $\lambda$ , and  $T$

**Ensure:** Processed distance matrix  $\hat{A}$ ; set of ranked lists  $\hat{\mathcal{R}}$

```

1:  $t \leftarrow 0$ 
2:  $A_t \leftarrow A$ 
3: while  $t \leq T$  do
4:    $A_{t+1} \leftarrow 0$ 
5:   for all  $img_i \in C$  do
6:     for all  $img_l \in C$  do
7:       {Considering contextual spaces}
8:        $c_k \leftarrow 0; d_j \leftarrow 0$ 
9:       for all  $img_j \in MKNN(img_i)$  do
10:         $d_j \leftarrow d_j + A[j, l] \times (K - c_k)$ 
11:         $c_k \leftarrow c_k + 1$ 
12:       end for
13:       {Computing distance  $\rho(img_i, img_l)$ }
14:        $d_i \leftarrow A[i, l] / K$ 
15:        $d_j \leftarrow d_j / (\frac{K \times (K-1)}{2})$ 
16:        $A_{t+1}[i, l] \leftarrow \sqrt{d_i^2 + d_j^2}$ 
17:     end for
18:   end for
19:    $\mathcal{R}_{t+1} \leftarrow performReRanking(A_{t+1})$ 
20:    $t \leftarrow t + 1$ 
21: end while
22:  $\hat{A} \leftarrow A_T$ 
23:  $\hat{\mathcal{R}} \leftarrow \mathcal{R}_T$ 

```

---

dataset and ASC [53] shape descriptor, after the execution of the *Contextual Spaces based on Mutual-KNN* algorithm. Both re-ranking algorithms, *Contextual Spaces based on KNN* and *Mutual-KNN* have similar impact on distances.

Figure 6.7 illustrates the bidimensional representation for two similar images (illustrated in Figure 6.6). We can observe a set of points representing similar images to reference images very close to the origin. The remaining images of the dataset are in very distinct positions. In fact, considering similar reference images, the algorithm divides the dataset in two clusters: one for images that are similar to the reference images and another for non-similar images.

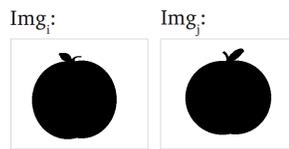


Figure 6.6: Similar reference images.

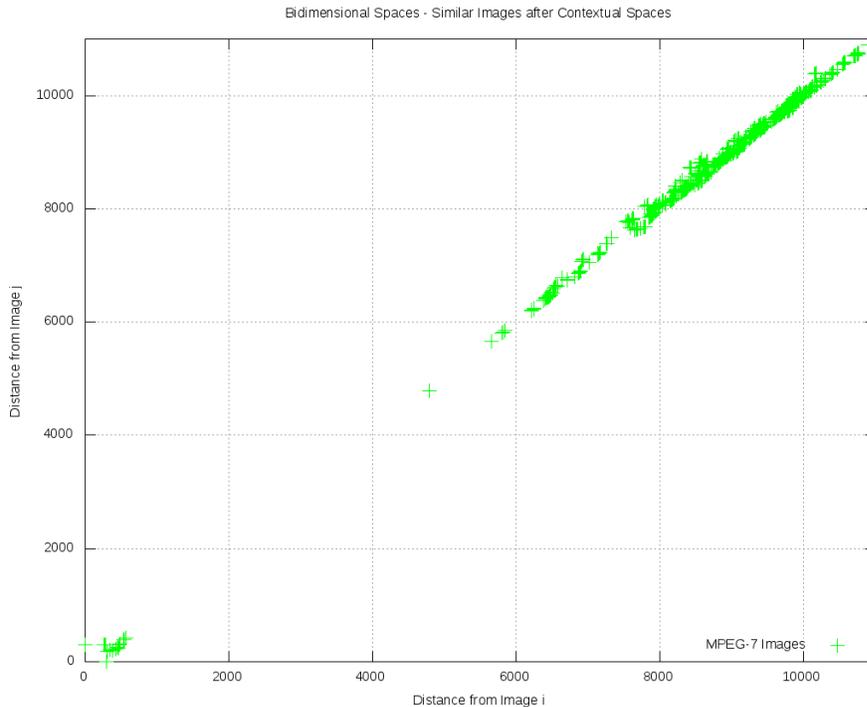


Figure 6.7: Bidimensional space representation for two similar images after *Contextual Spaces based on Mutual-KNN* algorithm execution.

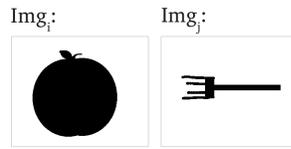


Figure 6.8: Non-similar reference images.

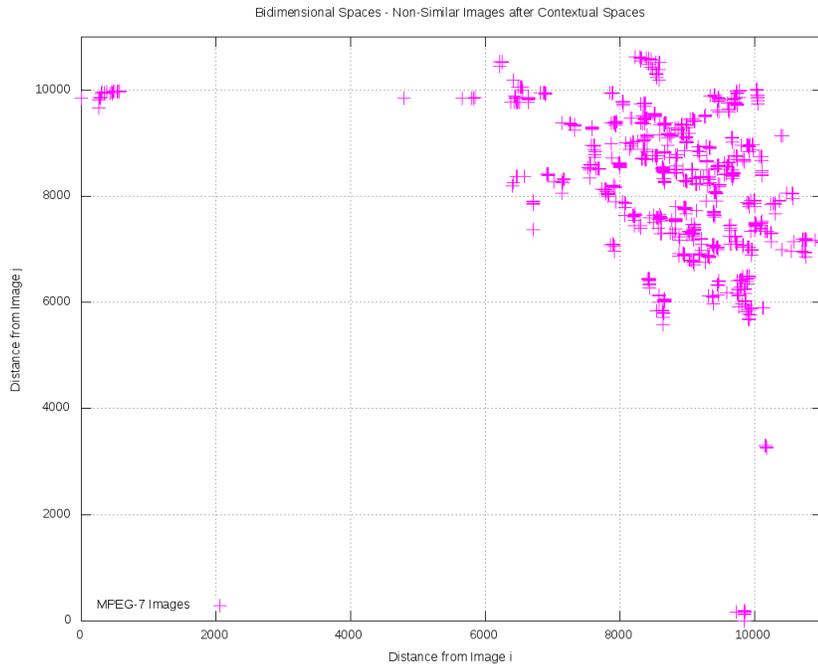


Figure 6.9: Bidimensional space representation for two non-similar images after *Contextual Spaces* based on *Mutual-KNN* algorithm execution.

We can observe the same behaviour for non-similar images. Figure 6.9 illustrates the bidimensional representation for non-similar images after the execution of the algorithm. Three distinct sets of points can be distinguished: (i) points next to x axis (images similar to image  $i$ ); (ii) points next to y axis (images similar to image  $j$ ); (iii) central points (remaining images). Note also that the remaining images are grouped in small sets, when compared to the distribution of images before the execution of algorithm (Figure 6.4).

### 6.1.4 The Rank Aggregation Algorithm

Let  $\mathcal{C}$  be an image collection and let  $\mathcal{D} = \{D_1, D_2, \dots, D_m\}$  be a set of CBIR descriptors. We can use the set of descriptors  $\mathcal{D}$  for computing a set of distances matrices  $\mathcal{A} = \{A_1, A_2, \dots, A_m\}$ . Our approach for combining descriptors works as follows: first, we combine the set  $\mathcal{A}$  in a unique matrix  $A_c$ . For the matrices combination we use a multiplicative approach. Each position  $(i, j)$  of the matrix is computed as follows:

$$A_c[i, j] = \sqrt[m]{(A_1[i, j] + 1) \times (A_2[i, j] + 1) \times \dots \times (A_m[i, j] + 1)}. \quad (6.4)$$

Once we have a matrix  $A_c$ , we can compute a set of ranked lists  $\mathcal{R}_c$  based on this matrix. Then, we can submit the matrix  $A_c$  and the set  $\mathcal{R}_c$  for our original re-ranking algorithm. This approach is very similar to that presented in Chapter 5, except for the constant 1 added to all terms, (aiming at avoiding interferences for very small distances) and for the square (aiming at mitigating noise).

### 6.1.5 Aspects of Efficiency

This work has as its focus the presentation of re-ranking algorithm based on Contextual Spaces and its effectiveness evaluation. The focus in effectiveness is justified by the fact that the execution of the algorithm is expected to be off-line, as in other post-processing methods [102]. This subsection aims at briefly discussing some aspects of efficiency and complexity of the algorithm.

Let  $\mathcal{C}$  be an image collection with  $N$  images, the number of elements in distance matrix  $A$  that should be redefined is equal to  $N^2$ . The complexity of the distances computing step is given by  $(N^2 \times K \times T)$  and therefore, the asymptotic computational complexity is  $O(N^2)$ . The re-ranking step computes a sort operation ( $O(N \log N)$ ) for all images ( $O(N^2 \log N)$ ). Other post-processing methods use matrices multiplication approaches [113, 114] and graph algorithms [102], both with complexity of  $O(N^3)$ .

Note that the re-ranking algorithm admits several optimizations (left as future work). A natural optimization consists in redefining only the distances between images at the top positions of ranked lists. In this way, the distance matrix does not require to be totally recomputed and the ranked lists does not require to be totally resorted. The re-ranking algorithm can also be massively parallelized, since there is no dependence between processing of different distances at a same iteration.

## 6.2 Experimental Evaluation

This section presents the set of conducted experiments for demonstrating the effectiveness of our methods.

Section 6.2.1 presents an analysis of the impact of the parameters of the re-ranking algorithms. Section 6.2.2 describes conducted experiments on CBIR tasks, considering color (Section 6.2.2), texture (Section 6.2.2), and shape (Section 6.2.2) descriptors. Section 6.2.3 discusses convergence aspects of the re-ranking method. Section 6.2.4 presents experimental results concerning the use of our methods in rank aggregation tasks. In Section 6.2.5, our approach is evaluated in multimodal (textual and visual) retrieval tasks.

### 6.2.1 Experiment 1: Impact of Parameters

This section describes experiments conducted to evaluate the influence of different parameter settings on the retrieval scores. The objective is to determine the best parameters values to be used with our methods.

We use the MPEG-7 database [48] and the CFD [68] and ASC [53] shape descriptors for distance computation. We present the experiments for *Contextual Spaces based on KNN* and *Contextual Spaces based on Mutual-KNN*.

#### Contextual Spaces based on KNN

The execution of *Contextual Spaces based KNN* (Algorithm 6.1) considers only two parameters: (i)  $K_s$  - initial K; (ii)  $K_e$  - end value of K. These two parameters also define the number of iterations along which the re-ranking algorithm should be executed.

Retrieval scores are computed ranging parameters  $K_s$  in the interval [1,5] and  $K_e$  in the interval [1,10]. Results are showed in Figure 6.10. We can observe that the best retrieval scores increased when parameters converged to regions next to values  $K_s = 1$  and  $K_e = 10$ . The best retrieval score was reached for  $K_s=2$  and  $K_e=9$ . These parameters are used in all experiments.

#### Contextual Spaces based on Mutual-KNN

The *Contextual Spaces based on Mutual-KNN* (Algorithm 6.2) considers three parameters: (i)  $K$ : number of similar images for constructing the contextual spaces; (ii)  $\lambda$ : maximum position in ranked lists considered for selecting similar images; and (iii)  $T$ : number of iterations along which the algorithm is executed.

In the first experiment, we used  $\lambda = 40$ . Retrieval scores are computed ranging parameters  $K$  in the interval [1,15] and  $T$  in the interval [1,15]. Results are showed in Figure 6.11. The best retrieval score (94.79%) was reached for  $K=8$  and  $T=13$ .

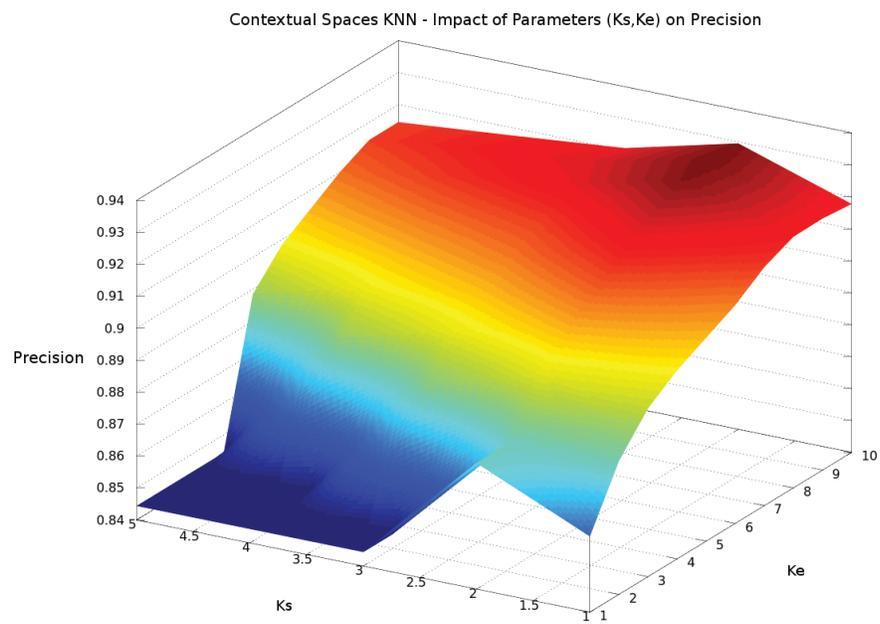


Figure 6.10: Impact of parameters  $K_s$  and  $K_e$  on precision score for *Contextual Spaces based on KNN*.

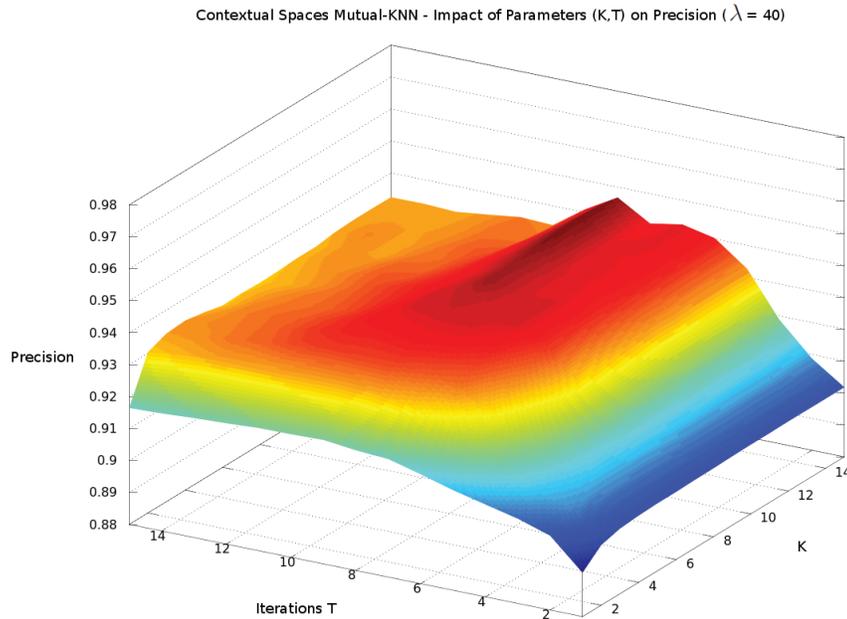


Figure 6.11: Impact of parameters  $K$  and  $T$  on *Contextual Spaces based on Mutual-KNN*.

After the parameter setting defined for  $K$  and  $T$ , a second experiment with parameter  $\lambda$  ranging in the interval  $[10,40]$  was conducted. Figure 6.12 shows the results of the experiment. The best retrieval score is obtained with  $\lambda = 23$ : **95.03%**. Note that retrieval scores are almost the same for  $\lambda > 30$ . That occurs because images at last positions of ranked lists are very unlikely to be selected as similar images. Then, even with the increase of lambda the results are almost the same.

## 6.2.2 CBIR Tasks Involving Color, Texture, and Shape Descriptors

Our goal here is to evaluate the use of our method in CBIR tasks involving shape, color, and texture descriptors. Our methods are compared with several descriptors and different datasets.

Results are presented in Table 6.1. The measure adopted is *Mean Average Precision (MAP)*. We conducted a paired t-test and conclude that there is a 99.9% of chance of difference between the means (before and after executing our re-ranking methods) being statistically significant. Next subsections present the descriptors and datasets used in the experiments.

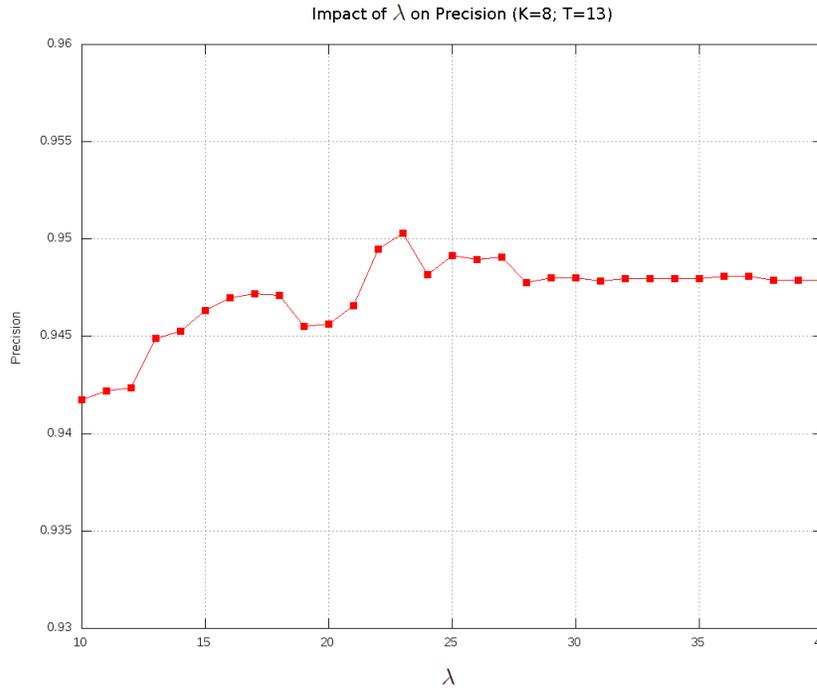


Figure 6.12: Impact of parameter  $\lambda$  on *Contextual Spaces based on Mutual-KNN*.

### Experiment 2: Color Descriptors

Three color descriptors were used in our experiments: BIC [90], ACC [41], and GCH [93]. The experiments were conducted on the Soccer dataset [100]. In Table 6.1, it can be observed a positive gain for all color descriptors for both re-ranking algorithms ranging from +2.26% to +21.06% (considering MAP as score).

### Experiment 3: Texture Descriptors

Regarding texture, three texture descriptors were considered: LBP [60], CCOM [46], and LAS [94]. We use the Brodatz [7] dataset.

Table 6.1 presents the results. We can observe that our re-ranking methods present positive gains ranging from +1.94% to +6.80%, considering texture descriptors. One exception is concerned with the use of CS-MKNN with the LBP descriptor. In that case, the ranked lists produced by LBP have few relevant images at first positions, what impact the quality of generated contextual spaces and therefore the effectiveness performance of the re-ranking method.

Table 6.1: Contextual Spaces evaluation on several content-based image retrieval tasks involving color, texture, and shape descriptors.

Image Descriptor	Type	Dataset	Score [%] (MAP)	CS- KNN	Gain	CS- MKNN	Gain
GCH [93]	Color	Soccer	32.24%	32.97%	+2.26%	33.96%	+5.33%
ACC [41]	Color	Soccer	37.23%	39.35%	+5.69%	45.07%	+21.06%
BIC [90]	Color	Soccer	39.26%	43.07%	+9.70%	45.00%	+14.62%
LBP [60]	Texture	Brodatz	48.40%	49.34%	+1.94%	48.20%	-0.41%
CCOM [46]	Texture	Brodatz	57.57%	61.49%	+6.80%	61.44%	+6.72%
LAS [94]	Texture	Brodatz	75.15%	79.67%	+6.01%	77.18%	+2.70%
SS [17]	Shape	MPEG-7	37.67%	40.74%	+8.15%	42.52%	+12.87%
BAS [2]	Shape	MPEG-7	71.52%	74.71%	+4.46%	76.07%	+6.36%
IDSC [52]	Shape	MPEG-7	81.70%	85.87%	+5.10%	88.05%	+7.78%
CFD [68]	Shape	MPEG-7	80.71%	90.00%	+11.51%	90.41%	+12.02%
ASC [53]	Shape	MPEG-7	85.28%	90.51%	+6.13%	91.87%	+7.72%
AIR [35]	Shape	MPEG-7	89.39%	93.16%	+4.22%	96.07%	+7.47%

#### Experiment 4: Shape Descriptors

We evaluate the use of our method with five shape descriptors: Segment Saliences (SS) [17], Beam Angle Statistics (BAS) [2], Inner Distance Shape Context (IDSC) [52], Contour Features Descriptor (CFD) [68], Aspect Shape Context (ASC) [53] and AIR [35]. We consider the MPEG-7 dataset, described in Chapter 3. Figure 6.13 illustrates an example of results for a MPEG-7 shape. The figure considers the CFD [68] shape descriptor before and after executing the re-ranking methods. The first row presents the retrieval results for the CFD [68] shape descriptor (first image as a query). The second row presents retrieval results for the same shape descriptor after using the *Contextual Spaces* re-ranking algorithm.

Results for MAP score are presented in Table 6.1. It can be observed significant gains for all shape descriptors, ranging from +4.46% to +12.87%. For shape descriptors, we also consider the bullseye score for the MPEG-7 dataset. Results of bullseye score for all descriptors are presented in Table 6.2. Note that the effectiveness gains are always positive and represent very significant improvement of effectiveness, ranging from +5.90% to +21.10%. Figure 6.14 shows the percentage gain obtained by Contextual Spaces algorithm for the CFD [68] descriptor considering each of 70 shape classes in MPEG-7 dataset. Note that bullseye score was improved by over 10% on average, and over 30% for some classes.



Figure 6.13: Impact of *Contextual Spaces* re-ranking algorithm on ranked lists.

Table 6.2: Contextual Spaces for shape descriptors on the MPEG-7 dataset (*Recall@40*).

Shape Descriptor	Score	CS-KNN	Gain	CS-MKNN	Gain
SS [17]	43.99%	48.50%	+10.25%	53.27%	+21.10%
BAS [2]	75.20%	80.53%	+7.09%	81.54%	+8.43%
IDSC [52]	85.40%	90.44%	+5.90%	90.91%	+6.46%
CFD [68]	84.43%	93.02%	+10.17%	93.00%	+10.15%
ASC [53]	88.39%	93.28%	+5.53%	95.03%	+7.51%
AIR [35]	93.67%	99.75%	+6.49%	99.92%	+6.67%

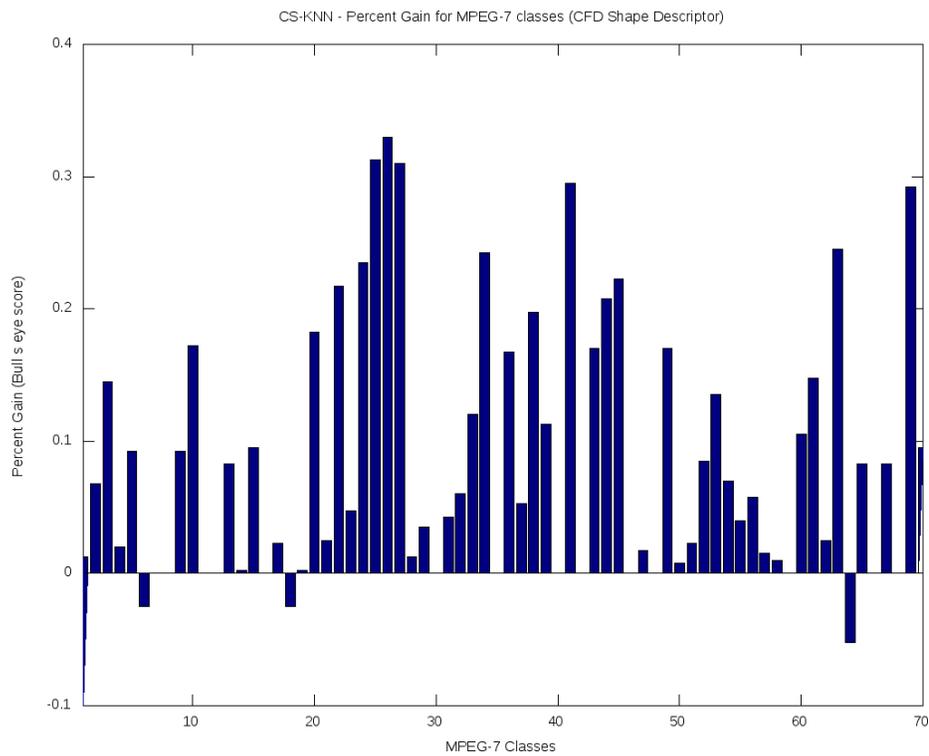


Figure 6.14: Percentage gain in bulls-eye score for each class of MPEG-7 dataset considering CFD [68] shape descriptor and *Contextual Spaces based on KNN* algorithm.

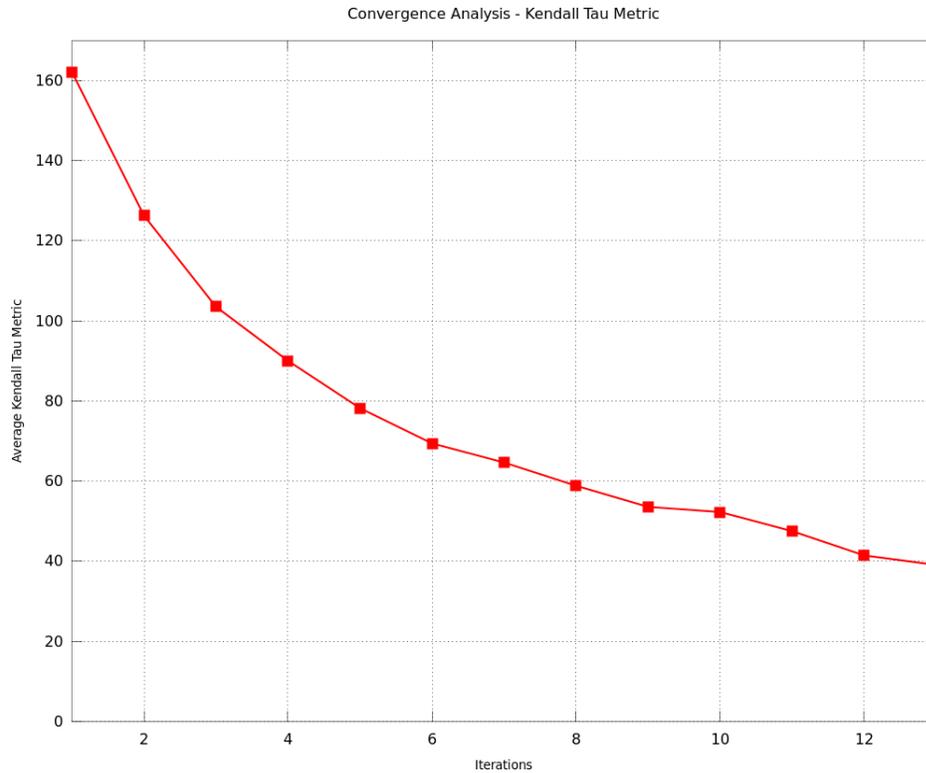


Figure 6.15: Analysis of convergence - Kendall’s tau distance along iterations.

### 6.2.3 Experiment 5: Analysis of Convergence

This section aims at discussing and experimentally evaluating the convergence of the proposed re-ranking methods. We consider the difference between ranked lists along iterations. As discussed in Chapter 5, an iterative re-ranking algorithm converges if, after a certain number of iterations, it produces a small number of changes in the generated ranked lists (a formal definition of  $\varepsilon$ -convergence for rankings is presented in [78]).

In this scenario, we have measured the evolution of Kendall’s tau distance between rankings at each iteration. The use of Kendall’s tau distance for comparing top-k lists is detailed in [27] and its definition is presented in Chapter 7. For the experiment, we considered the ASC [53] shape descriptor on the MPEG-7 [48] dataset and the Contextual Spaces based on Mutual-KNN. For measuring the Kendall’s tau distance, we considered the 20 top images of ranked lists. Figure 6.15 shows the evolution of the average Kendall’s tau distance between rankings along iterations.

Note that the Kendall’s tau distance *decreases* along the iterations. A high distance can be observed at first iterations, indicating a lot of changes in the ranked lists. After some iterations (next to the parameter  $T = 13$ ) the distances between ranked lists get lower values, indicating small changes in the ranked lists.

Table 6.3: Contextual Spaces for rank aggregation (*MAP*)

Descriptor	Type	Dataset	Method	Score (MAP)
CFD [68]	Shape	MPEG-7	-	80.71%
IDSC [52]	Shape	MPEG-7	-	81.70%
ASC [53]	Shape	MPEG-7	-	85.28%
CFD [68] + IDSC [52]	Shape	MPEG-7	CS-KNN	98.54%
CFD [68] + IDSC [52]	Shape	MPEG-7	CS-MKNN	98.47%
CFD [68] + ASC [53]	Shape	MPEG-7	CS-KNN	98.67%
CFD [68] + ASC [53]	Shape	MPEG-7	CS-MKNN	99.36%
ACC [41]	Color	Soccer	-	37.23%
BIC [90]	Color	Soccer	-	39.26%
BIC [90] + ACC [41]	Color	Soccer	CS-KNN	42.44%
BIC [90] + ACC [41]	Color	Soccer	CS-MKNN	46.10%
CCOM [46]	Texture	Brodatz	-	57.57%
LAS [94]	Texture	Brodatz	-	75.15%
LAS [94] + CCOM [46]	Texture	Brodatz	CS-KNN	81.94%
LAS [94] + CCOM [46]	Texture	Brodatz	CS-MKNN	84.50%

### 6.2.4 Experiment 6: Rank Aggregation

This section aims at evaluating the use of our re-ranking methods to combine different CBIR descriptors. We selected the two best descriptors for each visual property, considering the effectiveness results of our previous experiments. Table 6.3 presents the MAP scores obtained considering the use of contextual spaces for rank aggregation. It can be observed significant gains compared with the results of each descriptor in isolation. For example, by using CS-MKNN for combining the CFD and ASC descriptors, the effectiveness MAP score reaches 99.36% (against 80.71% and 85.28%, in the case of using CFD and ASC, respectively, in isolation).

Figure 6.16 shows the Precision  $\times$  Recall curves of shape descriptors CFD [68] and ASC [53] in different situations: before and after using the Contextual Spaces Re-Ranking methods, and after using it for rank aggregation. Both re-ranking methods improve a lot the effectiveness performance of the descriptors, while the use of the proposed rank aggregation methods yield almost perfect results for all queries.

### 6.2.5 Experiment 7: Multimodal Retrieval

This section presents the evaluation of our method with regard to a multimodal retrieval task, which considers both visual and textual descriptors. We consider each descriptor

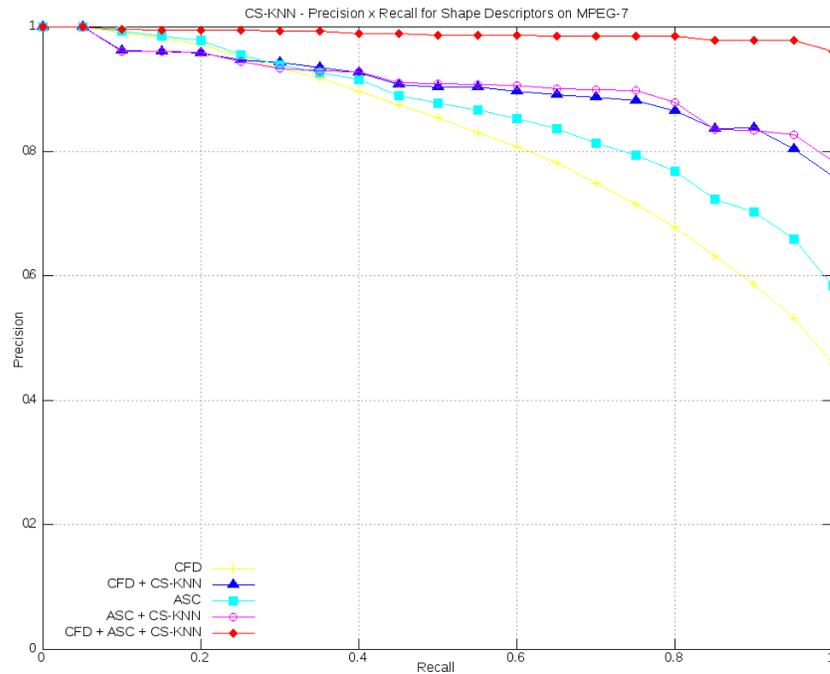


Figure 6.16: *Contextual Spaces based on KNN* for shape descriptors on the MPEG-7 dataset.

individually and on rank aggregation tasks. Used descriptors, dataset, are presented in the following.

### Dataset and Descriptors

We considered the *UW Dataset* and descriptors discussed in Chapter 3 for the experiments. Table 6.4 presents the scores (MAP) for used descriptors (visual and textual) on the UW dataset. It can be observed that the best descriptor effectiveness score in terms of MAP reaches 52.26%.

### Results

Experiments were conducted for both algorithms *Contextual Spaces based on KNN* and *based on Mutual-KNN*, considering visual, textual, and multimodal retrieval tasks. Two CBIR scenarios were evaluated: when all descriptors are used; and when only the best descriptors for each modality are used. Two baselines are considered in the experiments: the traditional Borda [116] method and the recently proposed Reciprocal Rank Fusion [14].

Table 6.5 presents the MAP results for all conducted experiments. It can be observed that, except for the combination of all visual descriptors, all the remaining results overcome the best individual descriptor (52.26%). Observe also that the use of Contextual

Table 6.4: Descriptors scores (*MAP*) on UW Database

Descriptor	Type	Score (MAP)
GCH [93]	Visual - Color	31.75%
BIC [90]	Visual - Color	43.46%
JAC [104]	Visual - Color	<b>52.26%</b>
QCCH [40]	Visual - Texture	17.81%
LAS [94]	Visual - Texture	20.44%
HTD [105]	Visual - Texture	22.61%
DICE [50]	Textual	50.73%
OKAPI [81]	Textual	51.68%
BOW [9]	Textual	48.84%
COS [3]	Textual	41.80%
JACCARD [50]	Textual	50.29%
TF-IDF [3]	Textual	49.25%

Table 6.5: Contextual Spaces on multimodal retrieval tasks (*MAP* as score)

Retrieval Task	Descriptors	CS-KNN	CS-MKNN	Borda [116]	Reciprocal [14]
Visual	All visual descriptors	46.11%	47.72%	40.29%	43.29%
Textual	All textual descriptors	61.75%	60.87%	53.07%	53.14%
Multimodal	All descriptors	67.25%	67.64%	54.89%	59.34%
Visual	BIC+JAC	54.94%	60.59%	52.54%	53.00%
Textual	DICE+OKAPI	59.13%	61.16%	54.57%	54.31%
Multimodal	BIC+JAC + DICE+OKAPI	70.73%	<b>75.19%</b>	61.91%	63.67%

Spaces yields better results than the baselines methods. The best multimodal retrieval result (75.19%) presents a gain of **+43.88%** over the best individual descriptor.

# Chapter 7

## RL-Sim Re-Ranking

In this chapter, we present the *RL-Sim Re-Ranking* algorithm, a new post-processing method that considers the similarity among ranked lists (*RankedLists-Similarities*) for characterizing contextual information in CBIR systems. The main motivation of our re-ranking algorithm relies on the conjecture that *contextual information encoded in the similarity between ranked lists can provide useful information for improving the effectiveness of CBIR descriptors*. In general, if two images are similar, their ranked lists should be similar as well [68]. It is somehow close to the the cluster hypothesis [101], which states that “*closely associated documents tend to be relevant to the same requests*”.

We believe that the modeling of contextual information considering only the similarity between ranked lists represents an advantage of our strategy. Since the re-ranking method does not depend on distances or similarity scores, it can be used for different CBIR tasks and can be easily adapted for other information retrieval tasks (*e.g.*, text or multimodal retrieval). Beyond that, the re-ranking method can use different similarity/distance measures among ranked lists, a well-established research area [27, 103, 106]. Therefore, the re-ranking algorithm can be easily extended using different similarity measures.

Our experimental evaluation demonstrates that the proposed method can improve the retrieval results of different CBIR tasks, considering different datasets. We evaluated the proposed method with shape, color, and texture descriptors considering re-ranking and rank aggregation tasks. Experimental results demonstrate that the proposed method yields better results in terms of effectiveness performance than various post-processing algorithms recently proposed in the literature. We also evaluated other aspects of the proposed algorithm (as efficiency and impact of parameters) in our experimental evaluation.

This chapter is organized as follows. In Section 7.1, we present our approach for unsupervised distance learning based on the similarity of ranked lists. Section 7.2 discusses approaches for comparing ranked lists. Section 7.3 presents the RL-Sim Re-Ranking

Algorithm. In Section 7.4, we describe how the proposed algorithm can be used in rank aggregation tasks. Section 7.5 presents the experimental evaluation.

## 7.1 Contextual Distance Measure based on Ranked Lists

In this section, we define a contextual distance measure based on similarity/dissimilarity of ranked lists. The contextual distance measure represents the basis of our proposed re-ranking algorithm. According to the formalization presented in Chapter 2, a given image descriptor  $\mathcal{D}$  can compute a distance  $\rho(img_i, img_j)$  between two images  $img_i, img_j \in \mathcal{C}$ . Therefore, this distance value considers only the two images  $img_i, img_j$ .

In order to compute the ranked lists  $R_i, R_j$  for images  $img_i, img_j$ , distances from these images to all other collection images need to be computed. In this way, the ranked lists represent, by itself, a contextual description of images with regard to the whole dataset. The images at the top positions of ranked lists often represent the most relevant images, in the sense that they usually represent the results in which users are interested. In this scenario, we conjecture that *given any two images and their respective ranked lists, a new and more effective distance measure between the two images can be computed by considering the images at the top positions of their ranked lists.*

The proposed contextual distance measure is based on this conjecture. In the common case, the top positions of ranked lists contain many images that are similar to the query image and some “*wrong*” (non-similar) images. Those images placed at the top positions usually are similar to each other and, therefore, there are many images in common in their ranked lists. We can observe that this set of images (similar to the query image and similar to each other) appears in the ranked lists of all images that compose the set. The same behavior can not be observed when analysing the top positions of the ranked lists of non-similar images (the same set of images does not appear at the top positions). In this scenario, a low contextual distance score is computed, since there are few images in common at the top positions of ranked lists of non-similar images. The objective of the proposed re-ranking algorithm is to move the non-similar images down in the ranked lists, and as a result of this process, the quality of ranked lists is improved. Note that, in extreme situations, in which the CBIR descriptors completely confuse similar and non-similar images, there is no contextual information available for improving the ranked lists.

A new contextual distance measure, defined in the following, is iteratively learned in a unsupervised setting. That distance measure is able to incorporate the contextual information, improving retrieval results. Let us consider the *neighborhood set*  $\mathcal{N}(i)$  of an image

$img_i$ , which contains images similar to  $img_i$  according to a given distance, say  $\rho$  defined by the image descriptor. The set  $\mathcal{N}(i)$  can be obtained, for example, by the well-known *k-Nearest Neighbor* approach, where the cardinality of the set is denoted by  $|\mathcal{N}(i)| = k$ . In Section 7.2.1, we formally define approaches for obtaining the neighborhood set  $\mathcal{N}$ .

In the following, we formally define the top positions of a ranked list as a *top k list*, according to [27]. We define a ranked list  $R_i$  as a permutation of collection  $\mathcal{C}$ , given by a bijection  $\sigma_i$  from the collection  $\mathcal{C}$  onto the set  $[N] = \{1, 2, \dots, N\}$ . Similarly, a *top k list*  $\tau_i$  is a bijection from a domain  $\mathcal{N}(i)$  (the members of the top  $k$  list) to  $[k] = \{1, 2, \dots, k\}$ . We say that  $img_j$  appears in the top  $k$  list  $\tau_i$  if  $img_j \in \mathcal{N}(i)$ . We interpret  $\tau_i(j)$  as the position (or rank) of image  $img_j$  in  $\tau_i$ . For the well-known *k-Nearest Neighbor* approach, we can say that if  $img_1$  is ranked before  $img_2$  ( $\tau_i(1) < \tau_i(2)$ ), then  $\rho(img_i, img_1) \leq \rho(img_i, img_2)$ . Approaches for computing top  $k$  lists are formally defined in Section 7.2.1.

Assume that  $\tau_i$  and  $\tau_j$  are top  $k$  lists computed for images  $img_i, img_j$  respectively. Several similarity (or dissimilarity) measures for comparing  $\tau_i$  and  $\tau_j$  can be defined [27, 103, 106] (different distance measures are discussed in Section 7.2.2). Let  $d(\tau_i, \tau_j, k)$  denote a given distance measure for comparing top  $k$  lists, we define a non-iterative contextual distance measure  $\rho_c(img_i, img_j)$  based on the comparison of the top  $k$  lists, as follows:

$$\rho_c(img_i, img_j) = d(\tau_i, \tau_j, k). \quad (7.1)$$

Based on the conjecture that the contextual distance measure  $\rho_c$  represents a more effective distance between images, we can recompute the distance among all images in a collection based on this measure. In this way, a new set of ranked lists (and their respective top  $k$  lists) can be obtained, such that the contextual distance can also be recomputed. Therefore, this process can be repeated in an iterative manner. Let  $(t)$  be a superscript that denotes the iteration. Let  $\tau_i^{(t)}$  be the top  $k$  list for image  $img_i$  at iteration  $t$ , which is computed based on contextual distance  $\rho_c^{(t)}$ . Let  $\rho_c^{(0)}$  be the contextual distance at first iteration, which is equal to the distance defined by the image descriptor, such that  $\rho_c^{(0)}(img_i, img_j) = \rho(img_i, img_j)$  for all images  $img_i, img_j \in \mathcal{C}$ . We can define an iterative contextual measure as follows:

$$\rho_c^{(t+1)}(img_i, img_j) = d(\tau_i^{(t)}, \tau_j^{(t)}, k). \quad (7.2)$$

Once the effectiveness of the contextual distance measure improves along iterations, the effectiveness of ranked lists also improve. Non-relevant images are moved out from the first positions of the ranked lists and, therefore,  $k$  is increased for considering more images. In this way, a larger  $k$  is considered for computation of top  $k$  lists along iterations, as follows:

$$\rho_c^{(t+1)}(img_i, img_j) = d(\tau_i^{(t)}, \tau_j^{(t)}, k + t). \quad (7.3)$$

After a given number of  $T$  iterations, a new distance  $\hat{\rho}$  is computed based on the contextual distance measure  $\rho_c$ :

$$\hat{\rho}(img_i, img_j) = \rho_c^{(T)}(img_i, img_j). \quad (7.4)$$

Finally, a new distance matrix  $\hat{A}$  is computed based on  $\hat{\rho}$ , such that for all images  $img_i, img_j \in \mathcal{C}$  we have  $\hat{A}_{ij} = \hat{\rho}(img_i, img_j)$ . Based on  $\hat{A}$ , a new set of ranked lists  $\hat{\mathcal{R}}$  is computed completing the re-ranking process.

## 7.2 Comparing Ranked Lists

The comparison of ranked lists is the basis of our proposed contextual measure. This section discusses and formalizes the process of comparison, which can be divided in two main steps: (i) how to retrieve a *neighborhood set* for a given image  $img_i$ , which is used to compose a top  $k$  list  $\tau_i$ ; (ii) and how to compute a distance  $d(\tau_i, \tau_j, k)$  between two top  $k$  lists  $\tau_i$  and  $\tau_j$ .

Sections 7.2.1 and 7.2.2 discuss respectively steps (i) and (ii).

### 7.2.1 Neighborhood Set

This section presents and formally defines two different approaches for computing the top  $k$  lists for a given image: the well-known k-Nearest Neighbors ( $kNN$ ) method and the Mutual k-Nearest Neighbors ( $MkNN$ ).

#### k-Nearest Neighbors

Let  $img_i$  be a collection image ( $img_i \in \mathcal{C}$ ) whose the  $k$  most similar images (neighborhood set) we want to select. Let  $\mathcal{N}_{kNN}(i)$  be the neighborhood set obtained using the  $k$ -nearest neighbors method, which is defined as follows:

$$\mathcal{N}_{kNN}(i, k) = \{\mathcal{R} \subseteq \mathcal{C}, |\mathcal{R}| = k \wedge \forall x \in \mathcal{R}, y \in \mathcal{C} - \mathcal{R} : \rho(i, x) \leq \rho(i, y)\}. \quad (7.5)$$

Based on the neighborhood set  $\mathcal{N}_{kNN}(i)$  we also want define the top  $k$  list  $\tau_{i_{kNN}}$  using the k-Nearest Neighbors. Let  $\tau_{i_{kNN}}(j)$  be the position (or rank) of image  $img_j$  in  $\tau_{i_{kNN}}$ , we can say that if  $img_x$  is ranked before  $img_y$ , that is  $\tau_{i_{kNN}}(x) < \tau_{i_{kNN}}(y)$ , then  $\rho(img_i, img_x) \leq \rho(img_i, img_y)$ .

More formally, let's consider that no distance score is repeated if we compute the distance between  $img_i$  and the images in its neighborhood set  $\mathcal{N}_{kNN}(i, k)$  (or there is a pre-processing step for tie breaking distances), such that  $\{\forall x, y \in \mathcal{N}_{kNN}(i, k) : \rho(i, x) =$

$\rho(i, y)\} = \emptyset$ . The top  $k$  list  $\tau_{i_{kNN}}(j)$  is a permutation of  $\mathcal{C}$ , that can also be considered as a bijection  $\tau_{i_{kNN}} : \mathcal{C} \rightarrow [1, \dots, k]$  defined as follows:

$$\tau_{i_{kNN}}(j) = |\{j \in \mathcal{N}_{kNN}(i, k), \forall x \in \mathcal{N}_{kNN}(i, k) : \rho(i, x) < \rho(i, j)\}| + 1. \quad (7.6)$$

### Mutual k-Nearest Neighbors

Let  $\tau_{i_{kNN}}(j)$  be the position (or rank) of image  $img_j$  in the top  $k$  list  $\tau_{i_{kNN}}$ . Let  $\tau_{j_{kNN}}(i)$  be the position of  $img_i$  in the top  $k$  list  $\tau_{j_{kNN}}$ . It is very common in CBIR systems that  $\tau_{i_{kNN}}(j) \neq \tau_{j_{kNN}}(i)$ . However, when the difference between these positions is large, it may indicate an incorrect position of the image in one of the top  $k$  lists.

Based on this observation, we define a *Mutual k-Nearest Neighbors* method that considers reciprocal positions of images in their ranked lists. In fact, we select the k-Nearest Neighbors considering  $c \times k$  neighbors (where  $c$  is a constant<sup>1</sup>). Given a neighborhood set  $\mathcal{N}_{kNN}(i, c \times k)$ , we select the  $k$  most similar of this set by taking into account both: (i) the position of images in ranked list of  $img_i$  and (ii) the position of  $img_i$  in the ranked list of these images. We formally define the neighborhood set based on the Mutual k-Nearest Neighbor as follows:

$$\begin{aligned} \mathcal{N}_{MkNN}(i, k) = \{ \mathcal{R} \subseteq \mathcal{N}_{kNN}(i, c \times k), |\mathcal{R}| = k \wedge \forall x \in \mathcal{R}, y \in \mathcal{C} - \mathcal{R} : \\ \tau_{i_{kNN}}(x) + \tau_{x_{kNN}}(i) \leq \tau_{i_{kNN}}(y) + \tau_{y_{kNN}}(i) \}. \end{aligned} \quad (7.7)$$

We also define the top  $k$  list  $\tau_{i_{MkNN}}$  using the Mutual k-Nearest Neighbors:

$$\begin{aligned} \tau_{i_{MkNN}}(j) = |\{j \in \mathcal{N}_{MkNN}(i, k), \forall x \in \mathcal{N}_{MkNN}(i, k) : \\ \tau_{i_{kNN}}(x) + \tau_{x_{kNN}}(i) \leq \tau_{i_{kNN}}(y) + \tau_{y_{kNN}}(i)\}| + 1. \end{aligned} \quad (7.8)$$

### 7.2.2 Distance Measures between Top k Lists

Given the methods for obtaining neighborhood sets and top  $k$  lists, we now discuss how to compute a distance  $d(\tau_i, \tau_j, k)$  between two retrieved top  $k$  lists  $\tau_i$  and  $\tau_j$ . Note that the distance measure adopted  $d(\cdot, \cdot, k)$  does not depend on the method used for computing the neighborhood set  $\mathcal{N}$  and the top  $k$  lists  $\tau$ . Therefore, different combinations can be used in our re-ranking algorithm.

---

<sup>1</sup>We used  $c = 2$  in our experiments.

### Intersection Measure

An approach to define the distance between two top  $k$  lists  $\tau_i$  and  $\tau_j$  proposed in [27] is to capture the extent of overlap between  $\tau_i$  and  $\tau_j$ . This idea of overlap can be extended by considering not only the overlap at depth  $k$ , but also the cumulative overlap at increasing depths [72, 103]. For each  $k_c \in \{1 \dots k\}$ , it is computed the overlap at  $k_c$ , and then those overlaps are averaged to derive a similarity measure. The measure gives higher weights to the first positions of top  $k$  lists, which are considered many times. Equation 7.9 formally defines the intersection similarity measure  $\psi$ .

$$\psi(\tau_i, \tau_j, k) = \frac{\sum_{k_c=1}^k |\mathcal{N}(i, k_c) \cap \mathcal{N}(j, k_c)|}{k}. \quad (7.9)$$

Note that if two ranked lists present the same images at the first positions, the size of the intersection set is greater, and the value of  $\psi$  is greater as well. Figure 7.1 illustrates the computation of  $\psi$  considering multiscale values of  $k$ .

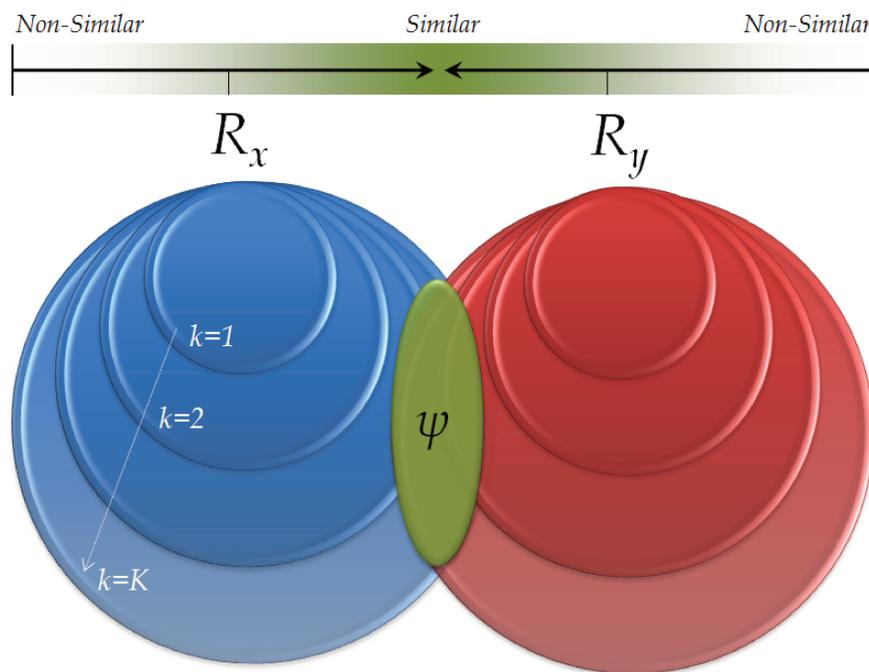


Figure 7.1: Computation of measure  $\psi$ : intersection of ranked lists with different sizes.

Since we are interested in a distance measure between top  $k$  lists, we define  $d_\psi$  as follows:

$$d_\psi(\tau_i, \tau_j, k) = \frac{1}{1 + \psi(\tau_i, \tau_j, k)}. \quad (7.10)$$

### Kendal's Tau Measure

The Kendall's tau is a distance measure between permutations, used to measure rank correlation. Its value is equal to the number of exchanges needed in a bubble sort to convert one permutation to the other [27]. The normalized Kendall's tau measure is defined as follows:

$$d_{\tau}(\tau_i, \tau_j, k) = \frac{\sum_{x,y \in \mathcal{N}(i,k) \cup \mathcal{N}(j,k)} \bar{K}_{x,y}(\tau_i, \tau_j)}{k \times (k-1)}, \quad (7.11)$$

where  $\bar{K}_{x,y}(\tau_i, \tau_j)$  is a function that determines if images  $img_x$  and  $img_y$  are in the same order in compared ranked lists  $R_i$  and  $R_j$ . This function can be defined as follows:

$$\bar{K}_{x,y}(\tau_i, \tau_j) = \begin{cases} 0 & \text{if } (\sigma_i(x) \leq \sigma_i(y) \wedge \sigma_j(x) \leq \sigma_j(y)) \vee (\sigma_i(x) \geq \sigma_i(y) \wedge \sigma_j(x) \geq \sigma_j(y)). \\ 1 & \text{otherwise} \end{cases}$$

The maximum value of defined Kendall's tau measure is given by  $k \times (k-1)$ , which occurs when  $\mathcal{N}(i, k) \cap \mathcal{N}(j, k) = \emptyset$  and  $\sigma_i$  is the reverse of  $\sigma_j$ .

Note that, although our goal is to compute the distance between the top  $k$  lists  $\tau_i$  and  $\tau_j$ , we considered the ranked lists positions  $\sigma_i$  and  $\sigma_j$  because we may have an image that is in only one of the top  $k$  lists (for example,  $img_x \in \tau_i$  and  $img_x \notin \tau_j$ ).

## 7.3 The RL-Sim Re-Ranking Algorithm

The goal of our re-ranking algorithm is to exploit the initial set of ranked lists  $\mathcal{R} = \{R_1, R_2, \dots, R_N\}$  for computing a more effective distance matrix  $\hat{A}$  and, therefore, a more effective set of ranked lists  $\hat{\mathcal{R}}$ . The *RL-Sim Re-Ranking Algorithm* is based on the presented contextual measure  $\rho_c$ , which takes into account the similarity between ranked lists on an iterative way.

An iterative approach is proposed. Let the superscript  $(t)$  denote the current iteration, a new (and more effective) set of ranked lists  $\mathcal{R}^{(t+1)}$  is computed by taking into account distances among top  $k$  lists. Next,  $\mathcal{R}^{(t+1)}$  is used for the next execution of our re-ranking algorithm and so on. These steps are repeated along several iterations aiming at improving the effectiveness incrementally. After a number  $T$  of iterations a re-ranking is performed based on the final distance matrix  $\hat{A}$ . Based on matrix  $\hat{A}$ , a final set of ranked lists  $\hat{\mathcal{R}}$  can be computed. Algorithm 7.1 outlines the proposed *RL-Sim Re-Ranking Algorithm*.

Observe that the distances are redefined considering the function  $d(\tau_i, \tau_j, k)$  for the first  $\lambda$  positions of the each ranked list, such that  $\lambda \in \mathbb{N}$  and  $0 \leq \lambda \leq N$ . For images in the remaining positions of the ranked lists, the new distance is redefined (Line 12)

---

**Algorithm 7.1** RL-Sim Re-Ranking Algorithm
 

---

**Require:** Original set of ranked lists  $\mathcal{R}$  and parameters  $k_s, T, \lambda$

**Ensure:** Processed set of ranked lists  $\hat{\mathcal{R}}$

```

1:  $t \leftarrow 0$ 
2:  $\mathcal{R}^{(t)} \leftarrow \mathcal{R}$ 
3:  $A^{(t)} \leftarrow A$ 
4:  $k \leftarrow k_s$ 
5: while  $t < T$  do
6:   for all  $R_i \in \mathcal{R}^{(t)}$  do
7:      $c \leftarrow 0$ 
8:     for all  $img_j \in R_i$  do
9:       if  $c \leq \lambda$  then
10:         $A^{(t+1)}[i, j] \leftarrow d(\tau_i, \tau_j, k)$ 
11:       else
12:         $A^{(t+1)}[i, j] \leftarrow 1 + A^{(t)}[i, j]$ 
13:       end if
14:        $c \leftarrow c + 1$ 
15:     end for
16:   end for
17:    $\mathcal{R}^{(t+1)} \leftarrow \text{performReRanking}(A^{(t+1)})$ 
18:    $k \leftarrow k + 1$ 
19:    $t \leftarrow t + 1$ 
20: end while
21:  $\hat{\mathcal{R}} \leftarrow \mathcal{R}^{(T)}$ 

```

---

based on the current distances. In these cases, the function  $d(\tau_i, \tau_j, k)$  does not need to be computed, considering that relevant images should be at the beginning of the ranked lists. In this way, the computational efforts decrease, making this step of the algorithm not dependent on the collection size  $N$ .

In Line 18, at each iteration  $t$ , we increment the number of  $k$  neighbors considered. The motivation behind this increment relies on the fact that the effectiveness of ranked lists increase along iterations. In this way, non-relevant images are moved out from the first positions of the ranked lists and  $k$  can be increased for considering more images. This strategy is also used by the re-ranking algorithms based Contextual Spaces, presented in Chapter 6.

Note that the re-ranking algorithm does not depend on specific measures between top  $k$  lists. In this way, an important advantage of our re-ranking algorithm is the possibility of using different approaches for retrieving the neighborhood set (we discussed the  $kNN$  and *Mutual  $kNN$*  methods) and different measures for comparing top  $k$  lists (we discussed the *intersection* and *Kendall's tau* measures). Therefore, the proposed RL-Sim Re-Ranking algorithm can be easily extended in order to consider different and even more complex approaches to compute the similarity between top  $k$  lists.

## 7.4 Rank Aggregation

Let  $\mathcal{C}$  be an image collection and let  $\mathcal{D}_s = \{D_1, D_2, \dots, D_m\}$  be a set of CBIR descriptors. We can use the set of descriptors  $\mathcal{D}$  for computing a set of distances matrices  $\mathcal{A}_s = \{A_1, A_2, \dots, A_m\}$ . Our approach for combining descriptors works as follows: first, we combine the set  $\mathcal{A}$  in a unique matrix  $A_c$ . For the matrices combination we use a multiplicative approach. Each position  $(i, j)$  of the matrix is computed as follows:

$$A_c[i, j] = (1 + A_1[i, j]) \times (1 + A_2[i, j]) \times \dots \times (1 + A_m[i, j]). \quad (7.12)$$

Once we have a matrix  $A_c$ , we can compute a set of ranked lists  $\mathcal{R}_c$  based on this matrix. Then, we can submit the matrix  $A_c$  and the set  $\mathcal{R}_c$  for our original re-ranking algorithm. This approach is very similar to that presented in Chapter 5, except for the constant 1 added to all terms, aiming at avoiding interferences for very small distances.

## 7.5 Experimental Evaluation

This section presents the set of conducted experiments for demonstrating the effectiveness of our method. We analyzed and compared our method under several aspects. Section 7.5.1 presents an analysis of the re-ranking algorithm considering the impact of

parameters. Section 7.5.2 presents a brief discussion about complexity and efficiency aspects.

Section 7.5.3 discusses the experimental results for our re-ranking method. It presents results of the use of our method for several shape descriptors, considering the MPEG-7 dataset [48]. In addition to shape descriptors, we conduct experiments with color and texture descriptors. Finally, Section 7.5.4 presents experimental results of our method on rank aggregation tasks.

### 7.5.1 Experiment 1: Impact of Parameters

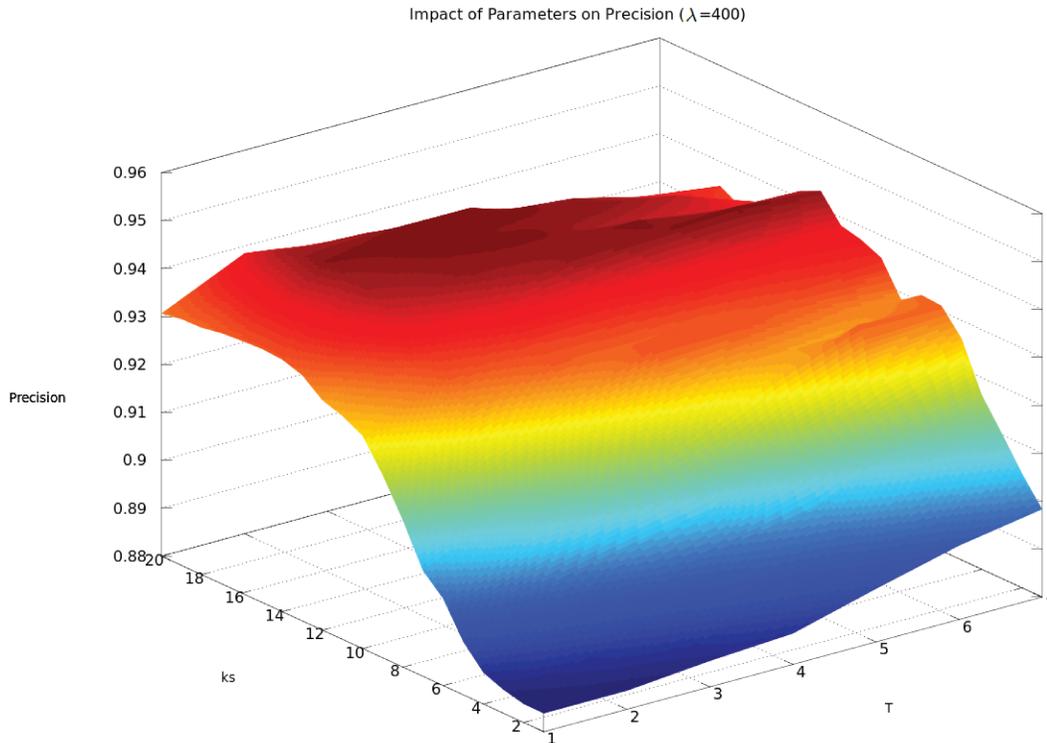
The execution of Algorithm 7.1 considers three parameters: *(i)*  $k_s$  - number of neighbors considered when algorithm starts; *(ii)*  $\lambda$  - number of images of each ranked list that are considered for redefining distances; and *(iii)*  $T$  - number of iterations along which the algorithm is executed.

To evaluate the influence of different parameter settings on the retrieval scores and for determining the best parameters values, we conducted a set of experiments considering the MPEG-7 [48] dataset. For distance computation, we used the ASC [53] shape descriptor.

Retrieval scores are computed considering the  $kNN$  method for the intersection measure. Parameter  $k_s$  varies in the interval [1,20] while parameter  $T$  varies in the interval [1,7]. Figure 7.2 illustrates the results of precision scores for different values of parameters  $k_s$  and  $T$ . We observed that best retrieval scores increased along iterations yielding the best precision score (94.69%) for  $k_s = 15$  and  $T = 3$ . We used these values in all experiments involving the intersection measure (for  $kNN$  and *Mutual kNN*). Analogous experiments were conducted for the Kendall's tau measure and very similar values were obtained:  $k_s = 15$  and  $T = 2$ . Those values were also used in all experiments involving the Kendall's tau measure.

Note that all variations of the algorithm (considering the intersection measure and the Kendall's tau measure) presented a very fast convergence reached with few iterations ( $T = 2$  and  $T = 3$ ).

We also analyzed the impact of parameter  $\lambda$  on precision. As discussed before, the objective of  $\lambda$  consists in decreasing computation efforts needed for the algorithm. It can be seen as a tradeoff between effectiveness and efficiency. In this way, we ranged  $\lambda$  in the interval [0, $N$ ] (considering the MPEG-7 collection). Results are illustrated in Figure 7.3. Note that the precision scores achieve the stability for  $\lambda = 700$  (value used in our experiments).

Figure 7.2: Impact of parameters:  $k_s$  and  $T$ .

### 7.5.2 Experiment 2: Aspects of Efficiency

This work has as its focus on the presentation of *RL-Sim Re-Ranking Algorithm* and on its effectiveness evaluation. This subsection aims at briefly discussing some aspects of efficiency and computational complexity.

The complexity of computation of comparison between top  $k$  lists, considering both intersection and Kendall's tau measures, is  $O(k^2)$ . The number of comparisons that should be processed is equal to  $(N \times \lambda)$ . Since the parameters  $\lambda$  and  $k$  have fixed values independent on  $N$ , the asymptotic computational complexity of the main step of the algorithm (computing distance between top  $k$  lists) is  $O(N)$ . Note, however, that the parameter  $\lambda$  is given in the interval  $0 \leq \lambda \leq N$  and, if defined with the maximum value  $\lambda = N$ , that makes this step of the algorithm  $O(N^2)$ .

Other steps of the algorithm have different complexities. The matrix  $A$  are recomputed ( $O(N^2)$ ) at each iteration. The re-ranking step computes a sort operation ( $O(N \log N)$ ) for all images ( $O(N^2 \log N)$ ). However, these steps admit optimizations: the matrix does not require to be totally recomputed and the ranked lists do not require to be totally sorted again at each iteration. The *RL-Sim Re-Ranking Algorithm* can also be massively parallelized, since there is no dependence between comparisons between ranked

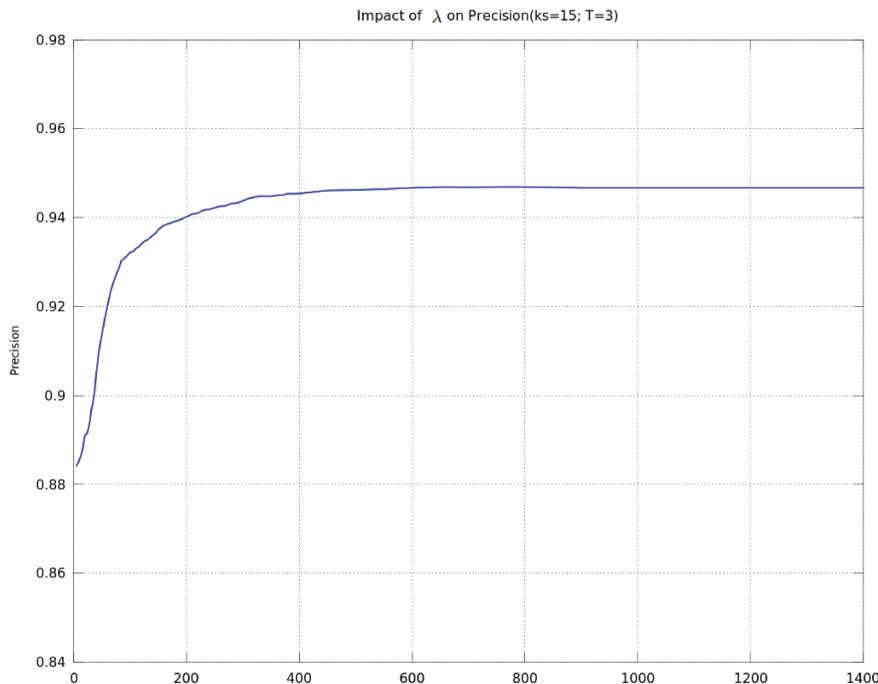


Figure 7.3: Impact of parameter  $\lambda$  on precision.

lists in a same iteration. Optimizations and parallelization issues will be investigated in future work. Note also that other post-processing methods use matrices multiplication approaches [113, 114] and graph algorithms [102], both with complexity of  $O(N^3)$ .

We evaluated the computation time of RL-Sim Re-Ranking algorithm for the MPEG-7 dataset ( $N = 1400$ ), using the parameters defined in Section 7.5.1 ( $k_s = 15$ ,  $T = 3$ , and  $\lambda = 700$ ), executing in a Linux PC Core 2 Duo and using a Java implementation. This execution took approximately 2 minutes (125.2s).

### 7.5.3 Experiment 3: Re-Ranking Evaluation

In this section, we present a set of conducted experiments for demonstrating the effectiveness of our method. We analyzed our method in the task of re-ranking images considering shape, color, and texture descriptors.

#### Shape Descriptors

We evaluate the use of our method with six shape descriptors: SS [17], BAS [2], IDSC [52], CFD [68], ASC [53], and AIR [35]. We consider the the MPEG-7 [48] dataset, described in Chapter 3.

Table 7.1: RL-Sim Re-Ranking using Intersection Distance Measure for shape descriptors on the MPEG-7 dataset (*Recall@40*).

Shape Descriptor	Score	kNN + Intersection	Gain	Mutual kNN + Intersection	Gain
SS [17]	43.99%	53.15%	+20.82%	57.58%	+30.90%
BAS [2]	75.20%	82.94%	+10.29%	85.87%	+14.19%
IDSC [52]	85.40%	92.18%	+7.94%	92.62%	+8.45%
CFD [68]	84.43%	94.13%	+11.49%	95.33%	+12.91%
ASC [53]	88.39%	94.69%	+7.13%	95.75%	+8.33%
AIR [35]	93.67%	99.90%	+6.65%	99.87%	+6.62%

Table 7.1 presents results (bullseye score - Recall@40) for shape descriptors using the *intersection measure* on the MPEG-7 [48] dataset. Both *kNN* and *Mutual kNN* methods are considered in the experiments. We can observe significant gains from +6.62% to +30.90% in relation to the use of the original descriptors.

The iterative behavior of the *RL-Sim Re-Ranking* algorithm can be observed in results illustrated in Figure 7.4. The figure shows the evolution of rankings (and their precision) along iterations. Each row presents 20 results for a query image (first column with green border). The first row presents the results of CFD [68] shape descriptor (wrong results with red borders). The remaining rows present the results of *RL-Sim Re-Ranking* algorithm for each iteration. We can observe that wrong results contain images from different classes, situation in which the re-ranking algorithm can *correct* the rankings based on the contextual information. The remaining rows present the results for each iteration of *RL-Sim Re-Ranking* algorithm, considering the *Mutual kNN* and *intersection measure* approaches. We can observe the significant improvement in terms of precision, ranging from 45% (on the ranking computed by the CFD [68] descriptor) to 100% at the third iteration of the re-ranking algorithm. The wrong results are progressively removed from top positions of ranked lists.

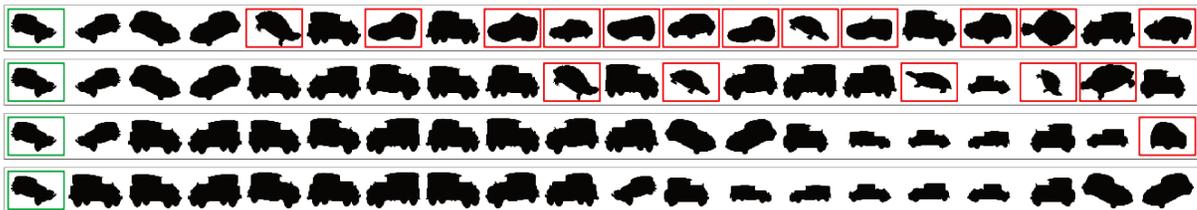


Figure 7.4: Evolution of rankings along iterations on the MPEG-7 [48] dataset.

Table 7.2 presents results for shape descriptors using the *Kendall's tau measure* on

Table 7.2: RL-Sim Re-Ranking using Kendall’s Tau Distance Measure for shape descriptors on the MPEG-7 dataset (*Recall@40*).

Shape Descriptor	Score	kNN + Kendall’s Tau	Gain	Mutual kNN + Kendall’s Tau	Gain
SS [17]	43.99%	52.67%	+19.73%	56.06%	+27.44%
BAS [2]	75.20%	81.16%	+7.93%	83.44%	+10.96%
IDSC [52]	85.40%	91.12%	+6.70%	92.06%	+7.80%
CFD [68]	84.43%	93.12%	+10.29%	94.27%	+11.65%
ASC [53]	88.39%	93.68%	+5.98%	94.56%	+6.98%
AIR [35]	93.67%	99.94%	+6.69%	99.93%	+6.68%

the MPEG-7 dataset. Results of both *kNN* and *Mutual kNN* methods are presented. We can also observe significant gains ranging from +5.98% to +27.44%.

Results for MAP (*Mean Average Precision*) score are presented in Table 7.3, considering the intersection measure; and Table 7.4, considering the Kendall’s tau measure. Both tables present results considering the *kNN* and *Mutual kNN* approaches. Positive gains can be observed for all shape descriptors in all combinations of approaches, ranging from +2.42% to +26.63%.

In addition to shape descriptors, we conducted experiments with color and texture descriptors, considering 12 image descriptors in 3 different datasets. Experiments with color and texture descriptors are described in next sections.

We also conducted a paired t-test aiming at evaluating the chance of difference between the means (before and after executing the proposed re-ranking method considering all descriptors) being statistical significantly. We conclude that there is a 99.9% of chance of difference being statistical significantly considering the intersection measure and 99% considering the Kendall’s Tau measure.

## Color Descriptors

We evaluate our method with three color descriptors: BIC [90], ACC [41], and GCH [93]. The experiments were conducted on the Soccer dataset [100].

Table 7.3 presents the experimental results considering the intersection measure while Table 7.4 considers the Kendall’s tau measure. Both tables present results considering the *kNN* and *Mutual kNN* approaches. We can observe a positive gain for all color descriptors for approaches ranging from +2.18% to +23.18% (considering the MAP score).

Table 7.3: MAP scores for RL-Sim Re-Ranking using Intersection Distance Measure in different CBIR tasks.

Descriptor	Type	Dataset	Score (MAP)	kNN + Intersection	Gain	M-kNN + Intersection	Gain
SS [17]	Shape	MPEG-7	37.67%	43.06%	+14.31%	47.70%	+26.63%
BAS [2]	Shape	MPEG-7	71.52%	74.57%	+4.25%	78.16%	+9.28%
IDSC [52]	Shape	MPEG-7	81.70%	86.75%	+6.18%	87.67%	+7.31%
CFD [68]	Shape	MPEG-7	80.71%	88.97%	+10.23%	90.78%	+12.48%
ASC [53]	Shape	MPEG-7	85.28%	88.81%	+4.14%	90.88%	+6.57%
AIR [35]	Shape	MPEG-7	89.39%	93.54%	+4.64%	93.52%	+4.62%
GCH [93]	Color	Soccer	32.24%	33.66%	+4.40%	33.84%	+4.96%
ACC [41]	Color	Soccer	37.23%	43.54%	+16.95%	44.78%	+20.28%
BIC [90]	Color	Soccer	39.26%	43.45%	+10.67%	44.08%	+12.28%
LBP [60]	Texture	Brodatz	48.40%	47.77%	-1.30%	48.51%	+0.23%
CCOM [46]	Texture	Brodatz	57.57%	62.01%	+7.72%	63.48%	+10.27%
LAS [94]	Texture	Brodatz	75.15%	77.81%	+3.54%	78.11%	+3.94%

### Texture Descriptors

The experiments consider three texture descriptors: LBP [60], CCOM [46], and LAS [94]. We used the Brodatz [7] dataset.

Table 7.3 presents the experimental results considering the intersection measure while Table 7.4 considers the Kendall’s tau measure. Both tables present results considering the *kNN* and *Mutual kNN* approaches. We can observe that our re-ranking methods yield positive gains ranging from +0.63% to +10.27%, except for LBP [60], which presents loss of effectiveness in some cases. The LBP [60] descriptor on the Brodatz dataset represents the extreme situations, discussed in Section 7.1, in which the CBIR descriptor confuses classes of images. In these situations, there is no enough contextual information available in the ranked lists to distinguish the classes, causing the loss of effectiveness. This situation is contrary to that illustrated in Figure 7.4, in which wrong results contain few images from different classes.

#### 7.5.4 Experiment 4: Rank Aggregation Evaluation

We evaluate the use of our re-ranking method to combine different CBIR descriptors. We select two descriptors for each visual property. Table 7.5 presents the MAP scores observed for the proposed rank aggregation approaches. We can observe that significant gains are obtained when compared with the results of descriptors in isolation.

Table 7.4: MAP scores for RL-Sim Re-Ranking using Kendall's Tau Distance Measure in different CBIR tasks.

Descriptor	Type	Dataset	Score (MAP)	kNN + Kendall's Tau	Gain	M-kNN + Kendall's Tau	Gain
SS [17]	Shape	MPEG-7	37.67%	44.24%	+17.92%	46.74%	+24.08%
BAS [2]	Shape	MPEG-7	71.52%	73.25%	+2.42%	75.38%	+5.40%
IDSC [52]	Shape	MPEG-7	81.70%	85.93%	+8.18%	86.53%	+5.91%
CFD [68]	Shape	MPEG-7	80.71%	88.40%	+9.53%	89.50%	+9.55%
ASC [53]	Shape	MPEG-7	85.28%	88.10%	+3.31%	89.92%	+5.44%
AIR [35]	Shape	MPEG-7	89.39%	96.27%	+7.70%	95.72%	+7.08%
GCH [93]	Color	Soccer	32.24%	32.96%	+2.18%	33.76%	+4.71%
ACC [41]	Color	Soccer	37.23%	44.29%	+18.96%	46.02%	+23.61%
BIC [90]	Color	Soccer	39.26%	43.76%	+11.46%	45.58%	+16.35%
LBP [60]	Texture	Brodatz	48.40%	45.20%	-6.61%	45.78%	-5.41%
CCOM [46]	Texture	Brodatz	57.57%	60.30%	+4.74%	61.41%	+6.67%
LAS [94]	Texture	Brodatz	75.15%	75.62%	+0.63%	76.13%	+1.30%

Table 7.5: MAP scores regarding the use of RL-Sim Algorithm for Rank Aggregation

Descriptor	Type	Dataset	Neighbor Set	Measure	Score (MAP)
CFD [68]	Shape	MPEG-7	-	-	80.71%
ASC [53]	Shape	MPEG-7	-	-	85.28%
CFD [68] + ASC [53]	Shape	MPEG-7	kNN	Intersection	98.75%
CFD [68] + ASC [53]	Shape	MPEG-7	M-kNN	Intersection	98.96%
CFD [68] + ASC [53]	Shape	MPEG-7	kNN	Kendall's Tau	98.57%
CFD [68] + ASC [53]	Shape	MPEG-7	M-kNN	Kendall's Tau	98.57%
ACC [41]	Color	Soccer	-	-	37.23%
BIC [90]	Color	Soccer	-	-	39.26%
BIC [90] + ACC [41]	Color	Soccer	kNN	Intersection	44.49%
BIC [90] + ACC [41]	Color	Soccer	M-kNN	Intersection	44.16%
BIC [90] + ACC [41]	Color	Soccer	kNN	Kendall's Tau	44.45%
BIC [90] + ACC [41]	Color	Soccer	M-kNN	Kendall's Tau	45.16%
CCOM [46]	Texture	Brodatz	-	-	57.57%
LAS [94]	Texture	Brodatz	-	-	75.15%
LAS [94] + CCOM [46]	Texture	Brodatz	kNN	Intersection	80.26%
LAS [94] + CCOM [46]	Texture	Brodatz	M-kNN	Intersection	83.39%
LAS [94] + CCOM [46]	Texture	Brodatz	kNN	Kendall's Tau	80.51%
LAS [94] + CCOM [46]	Texture	Brodatz	M-kNN	Kendall's Tau	81.68%

# Chapter 8

## Contextual Re-Ranking

In this chapter, we present a new post-processing method that re-ranks images by taking into account *contextual information* encoded in ranked lists and distance among images. We propose a novel approach for retrieving contextual information, by creating a *gray scale image* representation of distance matrices computed by CBIR descriptors (referenced in this thesis as *context image*). The context image is constructed for the  $k$ -nearest neighbors of a query image and analysed using image processing techniques.

The use of image processing techniques for *contextual information* representation and processing is an important novelty of our work. Our method uses distance matrices computed by CBIR descriptors that are later processed considering their image representation. The median filter, for instance, which is a well-known non-linear filter often used for removing noise, is exploited in our approach to improve the quality of distance scores. Other filters could have been used, but we chose the median filter just because it is the most suitable for noise removal. Basically, we consider that “*wrong*” distances can be considered and represented as “*noise*” in the context image, and the median filter is used for filtering this noise out.

In fact, a very large number of image processing techniques can be used for extracting useful information from context images. We believe that our strategy opens a new area of investigation related to the used of image processing approaches for analyzing distances computed by CBIR descriptor, in tasks such as image re-ranking, rank aggregation, and clustering.

The chapter is organized as follows: Section 8.1 describes the contextual information representation, while Sections 8.2 and 8.3 describe the re-ranking and rank aggregation methods, respectively. Experimental design and results are reported in Section 8.4.

## 8.1 Contextual Information Representation

Let  $\mathcal{C}$  be an image collection and let  $\mathcal{D}$  be an image descriptor. The distance function  $\rho$  defined by  $\mathcal{D}$  can be used for computing the distance  $\rho(img_i, img_j)$  among all images  $img_i, img_j \in \mathcal{C}$  in order to obtain an  $N \times N$  distance matrix  $A$ .

Our goal is to represent the distance matrix  $A$  as a gray scale image and to analyze this image for extracting *contextual information* using image processing techniques. For the gray scale image representation, referenced in this thesis as *context image*  $\hat{I}$ , we consider two reference images  $img_i, img_j \in \mathcal{C}$ .

Let the *context image*  $\hat{I}$  be a gray scale image defined by the pair  $(D_I, f)$ , where  $D_I$  is a finite set of pixels (points in  $\mathbb{N}^2$ , defined by a pair  $(x, y)$ ) and  $f : D_I \rightarrow \mathbb{R}$  is a function that assigns to each pixel  $p \in D_I$  a real number. We define the values of  $f$  function in terms of the distance function  $\rho$  (encoded into matrix  $A$ ) and reference images  $img_i, img_j \in \mathcal{C}$ .

Let  $R_i = (img_{i_1}, img_{i_2}, \dots, img_{i_N})$  be the ranked list defined by the matrix  $A$  considering the reference image  $img_i$  as query image; and  $R_j = (img_{j_1}, img_{j_2}, \dots, img_{j_N})$  the ranked list of reference image  $img_j$ . In this way, the axis of *context image*  $\hat{I}$  are ordered according to the order defined by ranked lists  $R_i$  and  $R_j$ . Let  $img_{i_x} \in R_i$  be an image at  $x$  position of ranked list  $R_i$  and  $img_{j_y} \in R_j$  an image at  $y$  position of the ranked list  $R_j$ , the value of  $f(x, y)$  (function that defines the gray scale of pixel  $p(x, y)$ ) is defined as follows:  $f(x, y) = \bar{\rho}(img_{i_x}, img_{j_y})$ , where  $\bar{\rho}$  is defined by the distance function  $\rho$  normalized in the interval  $[0, 255]$ .

An example, considering two similar reference images (from the MPEG-7 dataset [48]), is illustrated in Figure 8.1. The respective gray scale image representing matrix  $A$  is illustrated in Figure 8.3. An analogous example for non-similar images is showed in Figures 8.2 and 8.4.

The *context images* can represent a great source of information about the image collection. A single context image contains information about all distances among images and their spatial relationship defined by the ranked lists of the reference images. In other words, a single pixel is related to four collection images: the two reference images (that define the position of the pixel, according to their ranked lists) and the two images whose distance defines the grayscale value of the pixel. Another important advantage of this image representation relies on the possibility of using a large number of image processing techniques.

In this work, our goal is to exploit useful *contextual information* provided by context images. Low distance values (similar images) are associated with dark pixels in the image, while high values (non-similar images) refers to non-black pixels. Considering two similar images as reference images, the beginning of two ranked lists should have similar images as well. This behavior creates a *dark region* at the top left corner of a context image (as



Figure 8.1: Similar reference images.

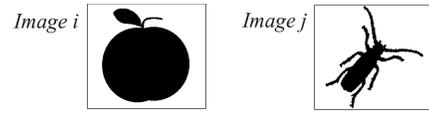


Figure 8.2: Non-similar reference images.

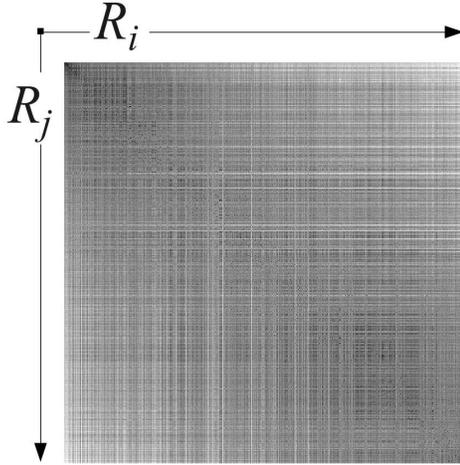


Figure 8.3: Context image for similar reference images.

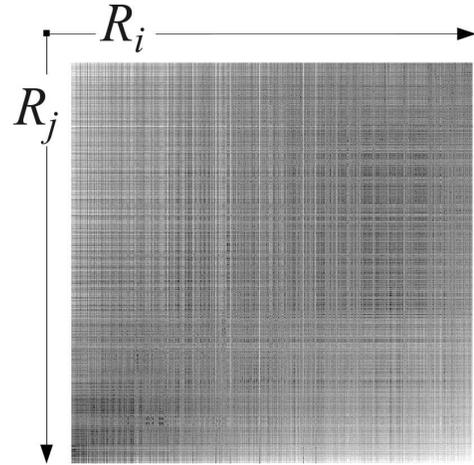


Figure 8.4: Context image for non-similar reference images.

we can observe in Figure 8.3). This region represents a neighborhood of similar images with low distances.

The top left corner represents images at the first position of the ranked lists of the two reference images, whose accuracy is higher than any other region in context image. We aim at characterizing *contextual information* by analyzing this region using image processing techniques. These information will be used by the re-ranking method presented in next section.

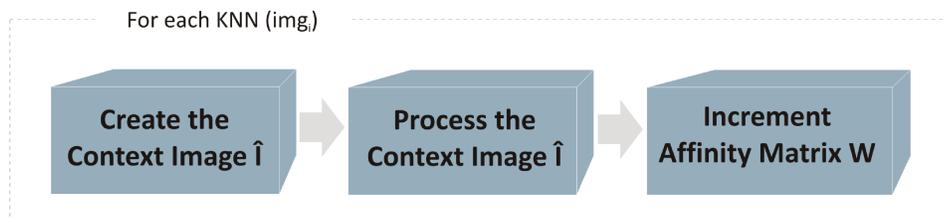
Other regions of context images could also be of interest. Considering similar reference images, the region close to the main diagonal, for example, contains more dark pixels (low distances) than the remaining of the image. Once the ranked lists of reference images are similar, pixels close to the main diagonal represent distances between similar images. The use of other regions of context images in image re-ranking tasks is left as future work.

## 8.2 The Contextual Re-Ranking Algorithm

Given an image  $img_i \in \mathcal{C}$ , we aim at processing *contextual information* of  $img_i$  by constructing *context images* for each one of its  $k$ -nearest neighbors (based on the distance matrix  $A$ ). We use an affinity matrix  $W$  to store the results of processing *contextual*

**Perform along T iterations:**

**For each collection image:  $img_i$**  ↴



**Compute Distance Matrix  $A_{t+1}$  (from Affinity Matrix  $W$ )**

**Perform Re-Ranking (based on  $A_{t+1}$ )**

Figure 8.5: The Contextual Re-Ranking Algorithm.

*information.* Let  $N$  be the size of collection  $\mathcal{C}$ , the affinity matrix  $W$  is an  $N \times N$  matrix where  $W[k, l]$  represents the similarity between images  $img_k$  and  $img_l$ .

We use image processing techniques to process the *context images* that consider  $img_i$  and each one of its *k-nearest neighbor* and then update the affinity matrix  $W$ . The same process is performed for all  $img_i \in \mathcal{C}$ . Since all images of  $\mathcal{C}$  are processed, the affinity matrix  $W$  is used as input for computing a new distance matrix  $A_{t+1}$  (where  $t$  indicates the current iteration).

Based on the new distance matrix  $A_{t+1}$ , a new set of ranked lists are computed. These steps are repeated along several iterations. Finally, after a number  $T$  of iterations a re-ranking is performed based on the final distance matrix  $A_T$  in order to obtain the final set of ranked lists. The main steps of the Contextual Re-Ranking Algorithm are illustrated in Figure 8.5. Algorithm 8.1 outlines the complete re-ranking method, that is detailed in the following.

The affinity matrix  $W$  is initialized with value 1 for all positions in Line 4. *Context images* are created in Line 7, as explained in Section 8.1, considering  $img_i$  (image being processed) and  $img_j$  (current neighbor of  $img_i$ ) as reference images. The parameter  $L$  refers to the size of the square at the top left corner of *context image* that will be analyzed.

Image processing techniques are applied to *context images* in Line 8. Our goal is to identify dense regions of dark pixels. Dark pixels indicate low distance values and, therefore, similar images. These regions represent the set of similar images at first positions of both ranked lists whose distances to each other are low. We use a threshold for obtaining a binary image and then identifying dark pixels. The threshold  $l$  used is computed based on normalization defined by the average and maximum distance values contained in the

---

**Algorithm 8.1** Contextual Re-Ranking Algorithm

---

**Require:** Original distance matrix  $A$   
**Ensure:** Processed distance matrix  $A_T$

- 1:  $t \leftarrow 0$
- 2:  $A_t \leftarrow A$
- 3: **while**  $t < T$  **do**
- 4:    $initializeAffinityMatrix(W, 1)$
- 5:   **for all**  $img_i \in C$  **do**
- 6:      $k \leftarrow 1$
- 7:     **for all**  $img_j \in KNN(img_i)$  **do**
- 8:        $ctxImg \leftarrow createContextImage(img_i, img_j, A_t, L)$
- 9:        $ctxImg' \leftarrow processContextImage(ctxImg, L)$
- 10:        $W \leftarrow updateAffinityMatrix(ctxImg', W, k)$
- 11:        $k \leftarrow k + 1$
- 12:     **end for**
- 13:   **end for**
- 14:    $A_{t+1} \leftarrow computeDistanceMatrix(W)$
- 15:    $performReRanking(A_{t+1})$
- 16:    $t = t + 1$
- 17: **end while**

---

$L \times L$  square at the top left corner of *context image*:

$$l = \frac{avg(\rho(img_p, img_q))}{max(\rho(img_p, img_q))}, \quad (8.1)$$

with  $p, q < L$ .

Next, we use a median filter for determining regions of dense black pixels. The non-linear median filter, often used for removing noise, is used in our approach aiming at correcting distances among images. Basically, we consider that “*wrong*” distances can be considered and represented as “*noise*” and the median filter is used to filter this noise out. More specifically, consider a dense region of black pixels at the top left corner of a context image. It represents a set of similar images (low distances) at the top positions of ranked lists of reference images. Consider a white pixel in this region, indicating a high distance between two images. By taking into account the contextual information given by the region of the pixel (defined by its position and pixels) and its neighborhood, it is very likely that the distance represented by this pixel is incorrect. In this scenario, the median filter replaces the white pixel by a black pixel. Similar reasoning can be applied to isolated black pixels in white regions. We should note that, in extreme situations, in which the CBIR descriptors completely confuse similar and non-similar images, there is less contextual information available in the context images.

Figure 8.6 illustrates an example of a binary image and Figure 8.7 shows the same image after applying the median filter (with a  $3 \times 3$  mask).

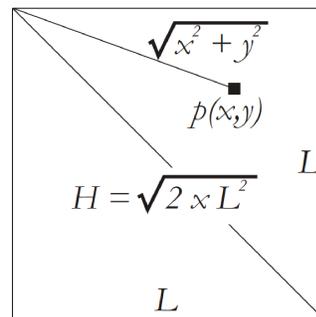
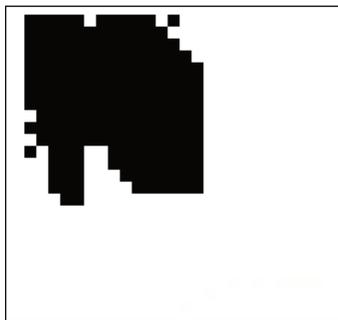
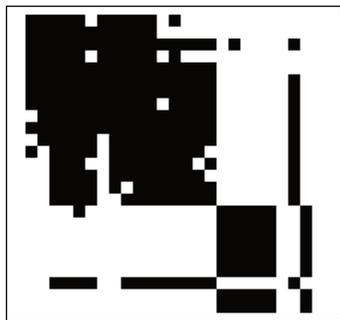


Figure 8.6: Example of binary image.

Figure 8.7: Example of filtered image.

Figure 8.8: Updates of matrix  $W$ .

Line 9 updates the affinity matrix  $W$  by taking into account the *context images*. For updating, only black pixels (and their positions) are considered. The objective is to give more relevance to pixels next to the origin  $(0,0)$ , i.e., pixels that represent the beginning of ranked lists. The importance of neighbors should also be considered: neighbors at first positions should be considered more relevant when updating  $W$ .

Let  $img_i \in \mathcal{C}$  be the current image being processed. Let  $img_j$  be the  $k$  (such that  $k < K$ ) neighbor of  $img_i$ . Let  $img_i$  and  $img_j$  be reference images and let  $\hat{I}(D_I, f)$  be the *context image* after thresholding and applying the median filter. Let  $L$  be the size of the top left corner square that should be processed and let  $p(x, y) \in D_I$  be a black pixel ( $f(x, y) = 0$ ), such that  $x, y < L$ . The pixel  $p(x, y)$  represents the distance between images  $img_x$  and  $img_y$  such that the image  $img_x$  is the image at the position  $x$  of the ranked list  $R_i$  and the image  $img_y$  is the image at position  $y$  of ranked list  $R_j$ .

Let  $H = \sqrt{2 \times L^2}$  be the maximum distance of a pixel  $p(x, y)$  to origin  $(0,0)$ , as illustrated in Figure 8.8. Let  $W[x, y]$  represent the similarity between images  $img_{i_x}$  and  $img_{i_y}$ . Then, for each black pixel  $p(x, y)$ , the matrix  $W$  receives five updates: the most relevant one refers to the similarity between images  $img_x$  and  $img_y$ ; two updates refers to the relationship between the reference image  $img_i$  with images  $img_x$  and  $img_y$ ; and two updates refers to the relationships between reference image  $img_j$  and images  $img_x$  and  $img_y$ . Figure 8.9 illustrates the relationship among these images provided by each black pixel in the context image. The update of the similarity score between  $img_x$  and  $img_y$  (the most relevant one) is computed as follows:

$$W[x, y] \leftarrow W[x, y] + [(K - k) \times (H / \sqrt{x^2 + y^2})]. \quad (8.2)$$

Note that low values of  $k, x, y$  (the beginning of ranked lists) leads to high increments of  $W$ . Smaller increments occur when  $k$  has high values and  $x, y = L$ . In this case,

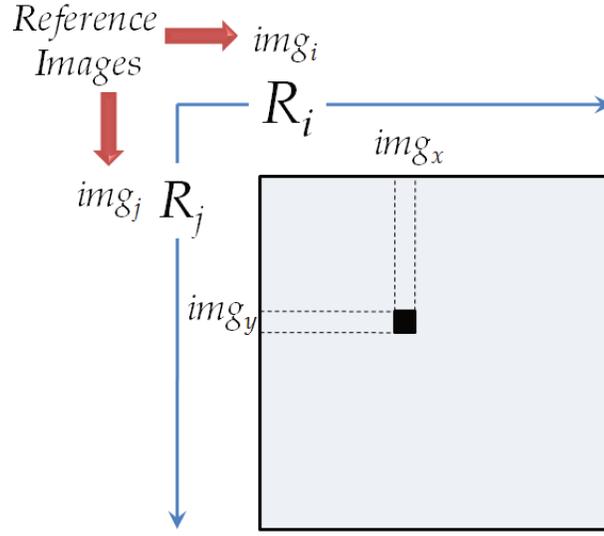


Figure 8.9: Relationship among images provided by a single pixel of a context image.

the term  $H/\sqrt{x^2 + y^2}$  is equal to 1. The remaining four updates (relationship among reference images and images  $img_x, img_y$ ) are computed as follows:

$$W[i, x] \leftarrow W[i, x] + \frac{[(K - k) \times (H/\sqrt{x^2 + y^2})]}{4}, \quad (8.3)$$

$$W[i, y] \leftarrow W[i, y] + \frac{[(K - k) \times (H/\sqrt{x^2 + y^2})]}{4}, \quad (8.4)$$

$$W[j, x] \leftarrow W[j, x] + \frac{[(K - k) \times (H/\sqrt{x^2 + y^2})]}{4}, \quad (8.5)$$

$$W[j, y] \leftarrow W[j, y] + \frac{[(K - k) \times (H/\sqrt{x^2 + y^2})]}{4}. \quad (8.6)$$

Observe that these four updates have together the same weight of the first update. They are computed based on the position of  $img_x$  and  $img_y$  ( $x, y$ ) in the ranked the lists of other images (reference images  $img_i, img_j$ ), while the first update is given by a pixel that represents the distance between images  $img_x$  and  $img_y$ .

When all images have been processed, and therefore an iteration has finished, the affinity matrix  $W$  presents high values for similar images. But there may be positions of  $W$  that was not updated (e.g., in the case of non-similar reference images), and have the initial value 1. The new distance matrix  $A_{t+1}$  (Line 12 of Algorithm 8.1) is computed as follows:

$$A_{t+1}[x, y] = \begin{cases} 1 + \bar{A}_t[x, y], & \text{if } W[x, y] = 1 \\ 2 \times (1/W[x, y]), & \text{if } W[x, y] > 1. \end{cases} \quad (8.7)$$

where  $\bar{A}_t$  is the distance matrix  $A_t$  normalized in the interval  $[0,1]$ . When  $W[x, y] = 1$ , i.e.,  $W[x, y]$  was not updated by Equation 8.2, we use the old distance matrix  $A_t$  for determining values of  $A_{t+1}$ . Otherwise (when  $W[x, y] > 1$ ), values of new distance matrix  $A_{t+1}$  is equal to the inverse of the values found in the affinity matrix  $W$ . Since the smallest increment for  $W$  is 1 (and therefore  $W[x, y] = 2$ ), the largest value of a new distance in  $A_{t+1}$  is 0.5. Therefore, we normalize the new distance values in the interval  $[0,1]$  by multiplying distances by 2.  $A_{t+1}$  will have values in the interval  $[0,2]$ : (i) in the interval  $[0,1]$ , if  $W[x, y] > 1$ , and (ii) in the interval  $[1,2]$ , if  $W[x, y] = 1$ . A last operation is performed on the new distance matrix  $A_{t+1}$  for ensuring the symmetry of distances between images ( $\rho(x, y) = \rho(y, x)$ ):

$$A_{t+1}[x, y] \leftarrow A_{t+1}[y, x] \leftarrow \min(A_{t+1}[x, y], A_{t+1}[y, x]). \quad (8.8)$$

Finally, a re-ranking is performed based on values of  $A_{t+1}$  (Line 15 of Algorithm 8.1). At the end of  $T$  iterations, a new computed distance matrix  $A_T$  and a set of new ranked list are obtained.

### 8.3 The Contextual Rank Aggregation Algorithm

The presented re-ranking algorithm can be easily tailored to rank aggregation tasks. In this section, we present the *Contextual Rank Aggregation Algorithm*, aiming at combining the results of different descriptors. The main idea consists in using the same iterative approach based on context images, but using the affinity matrix  $W$  for accumulating updates of *different descriptors* at the first iteration.

Algorithm 8.2 outlines the rank aggregation algorithm. We can observe that the algorithm is very similar to the re-ranking algorithm (Algorithm 8.1). It also considers an iterative approach and the context images for the contextual information processing. Note that the main difference relies on lines 8-13 of Algorithm 8.2, that are executed only at the first iteration, when different matrices  $A_d \in \mathcal{A}$  of different descriptors are being combined.

### 8.4 Experimental Evaluation

In this section, we present the set of conducted experiments for demonstrating the effectiveness of our method. We analyzed and evaluated our method under several aspects. In

---

**Algorithm 8.2** Contextual Rank Aggregation Algorithm
 

---

**Require:** Set of distance matrices  $\mathcal{A}$

**Ensure:** Processed distance matrix  $A_T$

```

1:  $t \leftarrow 1$ 
2: while  $t < T$  do
3:   initializeAffinityMatrix( $W, 1$ )
4:   for all  $img_i \in C$  do
5:     for all  $img_j \in KNN(img_i)$  do
6:        $k \leftarrow 1$ 
7:       if  $t = 1$  then
8:         for all  $A_d \in \mathcal{A}$  do
9:            $ctxImg \leftarrow createContextImage(img_i, img_j, A_d, L)$ 
10:           $ctxImg' \leftarrow processContextImage(ctxImg, L)$ 
11:           $W \leftarrow updateAffinityMatrix(ctxImg', W, k)$ 
12:           $k \leftarrow k + 1$ 
13:        end for
14:       else
15:          $ctxImg \leftarrow createContextImage(img_i, img_j, A_t, L)$ 
16:          $ctxImg' \leftarrow processContextImage(ctxImg, L)$ 
17:          $W \leftarrow updateAffinityMatrix(ctxImg', W, k)$ 
18:          $k \leftarrow k + 1$ 
19:       end if
20:     end for
21:   end for
22:    $A_{t+1} \leftarrow computeDistanceMatrix(W)$ 
23:   performReRanking( $A_{t+1}$ )
24:    $t = t + 1$ 
25: end while

```

---

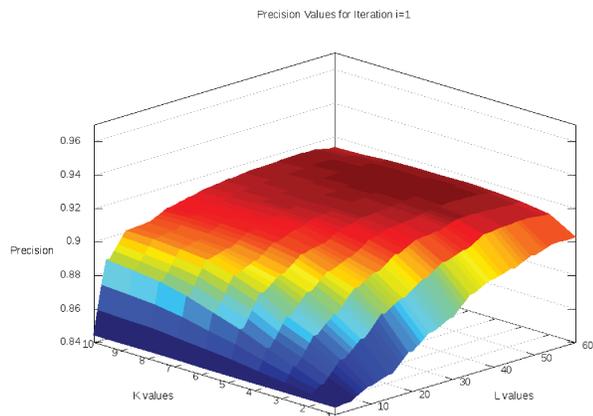


Figure 8.10: Iteration 1: 91.84%, K=5, L=50.

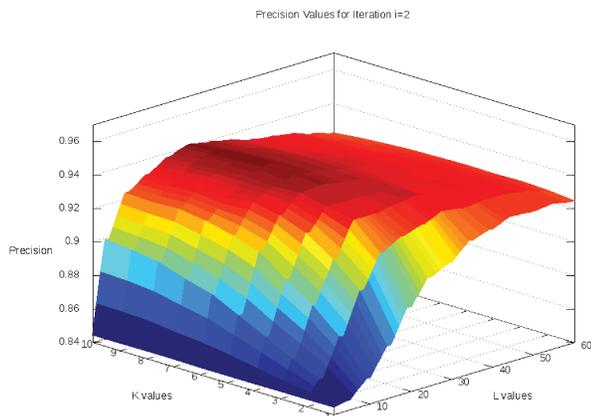


Figure 8.11: Iteration 2: 94.41%, K=7, L=25.

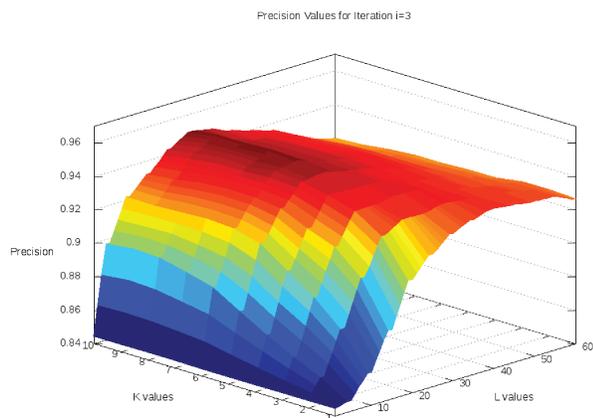


Figure 8.12: Iteration 3: 95.29%, K=7, L=25.

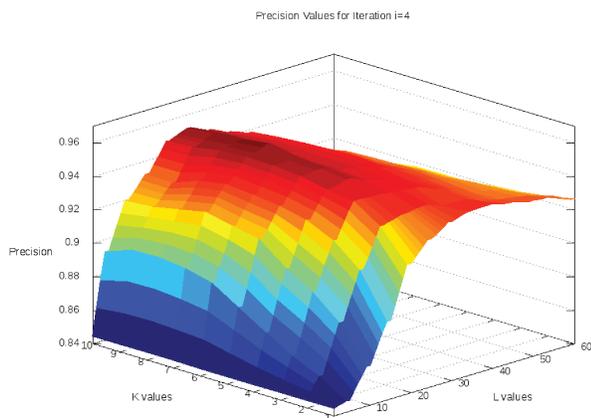


Figure 8.13: Iteration 4: 95.66%, K=7, L=25.

Section 8.4.1, we present an analysis of the Contextual Algorithm considering: the impact of parameters and image processing techniques on the re-ranking algorithm; and a brief discussion about complexity and efficiency.

In Section 8.4.2, we discuss the experimental results for our re-ranking method. We present results of the use of our method for several shape descriptors, considering the well-known MPEG-7 dataset [48]. Another goal of Section 8.4.2 is to validate the hypothesis that our method can be used in general image retrieval tasks. In addition to shape descriptors, we conducted experiments with color and texture descriptors. Section 8.4.3 presents experimental results of our method on rank aggregation tasks.

### 8.4.1 Experiment 1: Analysis of Contextual Re-Ranking Algorithm

In this section, we evaluated the Contextual Re-Ranking Algorithm with regard to different aspects. This section analyzes the impact of parameters in effectiveness results, evaluates the relevance of image processing techniques for the algorithm, and discusses aspects of efficiency and computational complexity.

#### Impact of Parameters

The execution of Algorithms 8.1 and 8.2 considers three parameters: (i)  $K$  - number of neighbors used as reference images; (ii)  $L$  - size of top left square of context image to be analyzed; and (iii)  $T$  - number of iterations that the algorithm is executed.

To evaluate the influence of different parameter settings on the retrieval scores and for determining the best parameters values we conducted a set of experiments. We use the MPEG-7 dataset [48] with the bullseye score. For distance computation, we used the CFD [68] shape descriptor.

Retrieval scores are computed ranging parameters  $K$  in the interval [1,10] and  $L$  in the interval [1,60] (with increments of 5) for each iteration. Figures 8.10, 8.11, 8.12, and 8.13 show surfaces that represent retrieval scores for iterations 1, 2, 3, and 4, respectively. For each iteration, the best retrieval score was determined.

We observed that the best retrieval scores increased along the first iterations and parameters converged for  $K = 7$  and  $L = 25$ . Figure 8.14 illustrates the evolution of precision according to the iterations of re-ranking algorithm. The best retrieval score was reached at iteration  $T = 5$ : **95.71%**.

Note that these parameters may change for datasets with very different sizes. The parameter values  $K = 7$ ,  $L = 25$ , and  $T = 5$  were used for all experiments, except for the Soccer color dataset (described in Section 10.5.1). Since this dataset is smaller than others, we used  $K = 3$ .

#### Impact of Image Processing Techniques

In this section we aim at evaluating the impact of the image processing techniques in the effectiveness results. For the experiments, we consider the MPEG-7 [48] dataset (with Recall@40 score), the CFD [68] shape descriptor and the parameters values defined in Section 8.4.1. We evaluated the method with regard to the follows aspects:

- **Median filter:** we have disabled the median filter (considering only the thresholding). The retrieval score obtained was 93.94%.

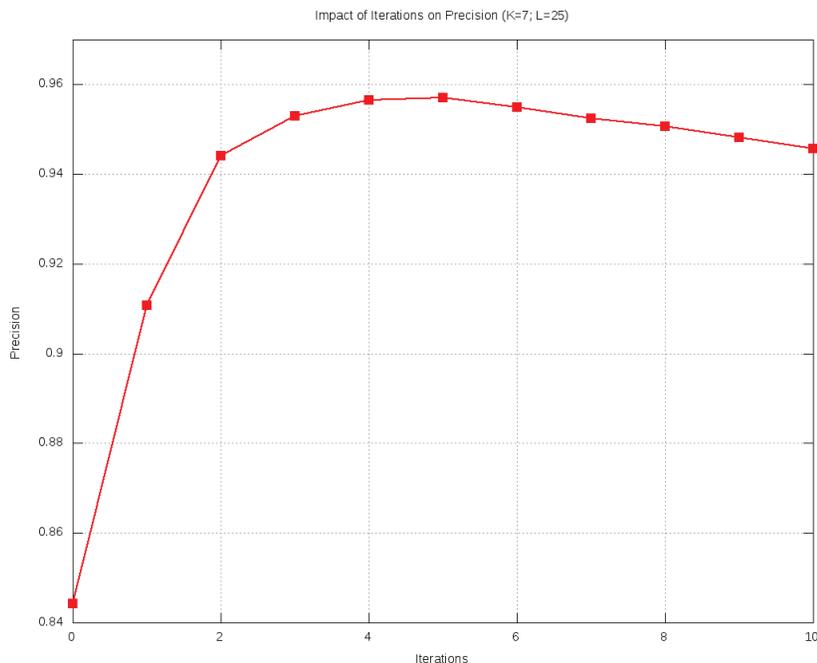


Figure 8.14: Impact of iterations on precision.

- **Thresholding:** we have disabled the thresholding and filter steps (considering updating for all pixels in context images). The effectiveness result obtained was 92.89%.
- **Masks of median filter:** we evaluated the effectiveness of the method for different sizes of masks (3, 5, and 7). We have obtained for masks 3, 5, and 7, respectively **95.71%**, 95.36%, and 95.18%.

The experimental results demonstrate the positive impact of image processing techniques in effectiveness results of Contextual Re-Ranking Algorithm. The best retrieval score (95.71%) was obtained when a thresholding and a median filter (a  $3 \times 3$  mask is used).

### Aspects of Efficiency

This subsection aims at briefly discussing some aspects of efficiency and computational complexity. Let  $\mathcal{C}$  be an image collection with  $N$  images. The number of context images that should be processed is equal to  $(N \times K \times T)$ . The size of context images that impacts the number of updates in matrix  $W$  is given by  $L^2$  pixels. Since the parameters  $K$ ,  $T$ , and  $L$  have fixed values independent of  $N$ , the asymptotic computational complexity of the main steps of the algorithm (image processing and  $W$  matrix updating steps) is  $O(N)$ .

Other steps of the algorithm have different complexities. The matrices  $A$  and  $W$  are recomputed ( $O(N^2)$ ) at each iteration. The re-ranking step computes a sort operation ( $O(N \log N)$ ) for all images ( $O(N^2 \log N)$ ). However, these steps admit optimizations: once the updatings for matrix  $W$  impact a small subset of positions (depending on the size  $L^2$  of context image), the matrices do not require to be totally recomputed and the ranked lists do not require to be totally sorted again. The Contextual Re-Ranking algorithm can also be massively parallelized, since there is no dependence between processing of different context images at a same iteration.

We evaluated the computation time of Contextual Re-Ranking algorithm for the MPEG-7 dataset ( $N = 1400$ ), using the parameters defined in Section 8.4.1 ( $K = 7$ ,  $L = 25$  and  $T = 5$ ), executing in a AMD Opteron 6168 (1.9GHz - 12 cores) and using a Java implementation. This execution took approximately 155 s. Next chapter discusses an efficient and parallel implementation of the Contextual Re-Ranking algorithm.

## 8.4.2 Experiment 2: Re-Ranking

In this section, we present a set of experiments conducted for demonstrating the effectiveness of our method. We compared results for several descriptors (shape, color, and texture) in different datasets. The measure adopted is *Mean Average Precision (MAP)*. Table 8.1 presents results for 12 image descriptors in 3 different datasets. As we can observe in Table 8.1, the Contextual Re-Ranking method presents positive effectiveness gains for all descriptors (including shape, color, and texture). The gains ranged from +1.37% to +18.90%, with 8.57% on the average. We conducted a paired t-test and conclude that there is a 99.9% of chance of difference between the means (before and after the re-ranking) being statistically significant. Next subsections present the descriptors and datasets used for shape, color and texture experiments.

### Shape Descriptors

We evaluate the use of our method with six shape descriptors considering the MPEG-7 dataset [48]: BAS [2], SS [17], IDSC [52], CFD [68], ASC [53], and AIR [35]. Results of bullseye score for all descriptors are presented in Table 8.2. Note that the effectiveness gains are always positive and represent very significant improvements of effectiveness, ranging from +5.29% to +16.80%, with 10.56% on average. Figure 8.15 presents the percentage gain obtained by Contextual Re-Ranking algorithm for CFD [68] descriptor considering each of 70 shape classes in the MPEG-7 dataset. Note that the bullseye score was improved over 30% for several classes.

The iterative behavior of the Contextual Re-Ranking algorithm can be observed in results illustrated in Figure 8.16. The figure shows the evolution of rankings along the

Table 8.1: Contextual Re-Ranking Evaluation in Content-Based Image Retrieval Tasks.

Descriptor	Type	Dataset	Score (MAP)	Contextual Re-Ranking	Gain
SS [17]	Shape	MPEG-7	37.67%	<b>44.79%</b>	<b>+18.90%</b>
BAS [2]	Shape	MPEG-7	71.52%	<b>76.60%</b>	<b>+7.10%</b>
IDSC [52]	Shape	MPEG-7	81.70%	<b>87.39%</b>	<b>+6.96%</b>
ASC [53]	Shape	MPEG-7	85.28%	<b>89.82%</b>	<b>+5.32%</b>
CFD [68]	Shape	MPEG-7	80.71%	<b>92.76%</b>	<b>+14.93%</b>
AIR [35]	Shape	MPEG-7	89.39%	<b>94.49%</b>	<b>+5.71%</b>
GCH [93]	Color	Soccer	32.24%	<b>33.02%</b>	<b>+2.42%</b>
ACC [41]	Color	Soccer	37.23%	<b>39.86%</b>	<b>+7.06%</b>
BIC [90]	Color	Soccer	39.26%	<b>43.04%</b>	<b>+9.63%</b>
LBP [60]	Texture	Brodatz	48.40%	<b>49.06%</b>	<b>+1.37%</b>
CCOM [46]	Texture	Brodatz	57.57%	<b>63.67%</b>	<b>+10.60%</b>
LAS [94]	Texture	Brodatz	75.15%	<b>78.48%</b>	<b>+4.43%</b>

Table 8.2: Contextual Re-Ranking for Shape Descriptors on the MPEG-7 dataset (*Recall@40*).

Shape Descriptor	Score [%]	Contextual Re-Ranking	Gain
SS [17]	43.99%	<b>51.38%</b>	<b>+16.80%</b>
BAS [2]	75.20%	<b>82.43%</b>	<b>+9.61%</b>
IDSC [52]	85.40%	<b>91.84%</b>	<b>+7.54%</b>
ASC [53]	88.39%	<b>93.07%</b>	<b>+5.29%</b>
CFD [68]	84.43%	<b>95.71%</b>	<b>+13.36%</b>
AIR [35]	93.67%	<b>99.80%</b>	<b>+6.54%</b>

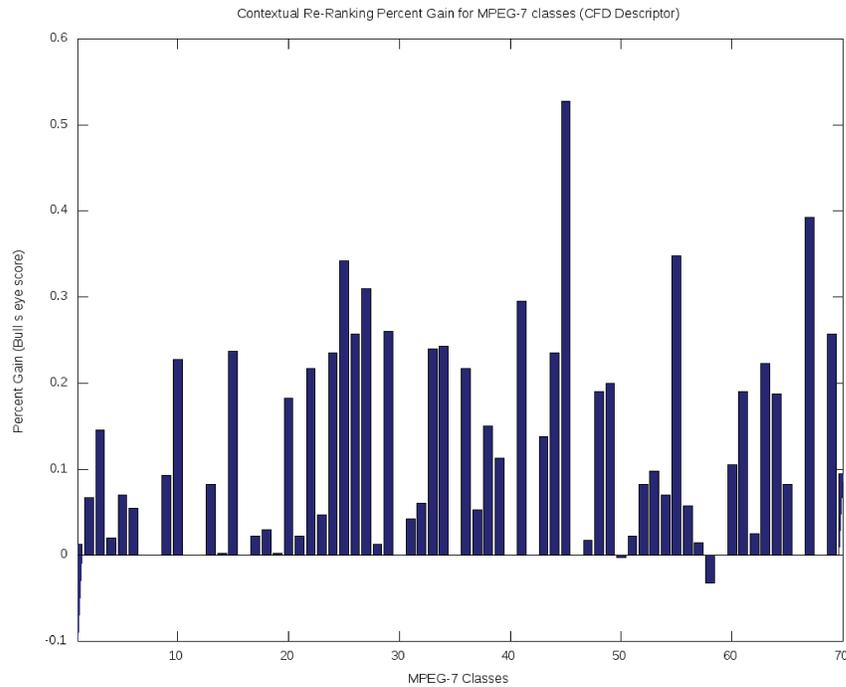


Figure 8.15: Contextual re-ranking percent gain for CFD [68] shape descriptor on the MPEG-7 classes.

iterations. The first row presents 20 results for a query image (first column) according to the CFD [68] shape descriptor. The remaining rows present the results for each iteration of Contextual Re-Ranking algorithm. We can observe the significant improvement in terms of precision, ranging from 40% (on the ranking computed by the CFD [68] descriptor) to 100% at the fifth iteration of the re-ranking algorithm. Note that the wrong results are progressively removed from top positions of ranked lists.

### Texture Descriptors

The experiments considered three texture descriptors: LBP [60], LAS [94], and CCOM [46]. We used the Brodatz [7] dataset, a popular dataset for texture descriptors evaluation. Our re-ranking method presents positive gains, presented in Table 8.1, ranging from +1.37% to 10.60%.

### Color Descriptors

Three color descriptors were considered for our evaluation: ACC [41], BIC [90], and GCH [93]. The experiments were conducted on the Soccer dataset [100] We can observe positive gains for all color descriptors, presented in Table 8.1, ranging from +2.42% to



Figure 8.16: Evolution of rankings along iterations on the MPEG-7 [48] dataset (first column contains the query image): the first row presents the results of CFD [68] shape descriptor; the remaining rows present the results of the *Contextual Re-Ranking Algorithm* for each iteration.

9.63%.

### 8.4.3 Experiment 3: Rank Aggregation

This section aims at evaluating the use of our re-ranking method to combine different CBIR descriptors. We selected two descriptors for each visual property (shape, color, and texture): descriptors with the best effectiveness results are selected (except for MPEG-7 dataset, for which the AIR [35] descriptor yields results very close to the maximum scores). Table 8.3 presents the MAP scores obtained for the rank aggregation considering these descriptors. We can observe significant gains compared with each isolated descriptors results. Figure 8.17 illustrates the Precision x Recall curves of shape descriptors CFD [68] and ASC [53] in different situations: before and after using the Contextual Re-Ranking algorithm, and after using it for rank aggregation. As it can be observed, for both re-ranking and rank aggregation, very significant gains in terms of precision have been achieved.

Table 8.3: Contextual rank aggregation on several content-based image retrieval tasks (*Mean Average Precision*)

Image Descriptors	Type	Dataset	Score ( <i>MAP</i> )
CFD [68]	Shape	MPEG-7	80.71%
ASC [53]	Shape	MPEG-7	85.28%
<b>CFD [68] + ASC [53]</b>	<b>Shape</b>	<b>MPEG-7</b>	<b>98.77%</b>
ACC [41]	Color	Soccer	37.23%
BIC [90]	Color	Soccer	39.26%
<b>ACC [41] + BIC [90]</b>	<b>Color</b>	<b>Soccer</b>	<b>42.14%</b>
CCOM [46]	Texture	Brodatz	63.67%
LAS [94]	Texture	Brodatz	75.15%
<b>CCOM [46] + LAS [94]</b>	<b>Texture</b>	<b>Brodatz</b>	<b>81.63%</b>

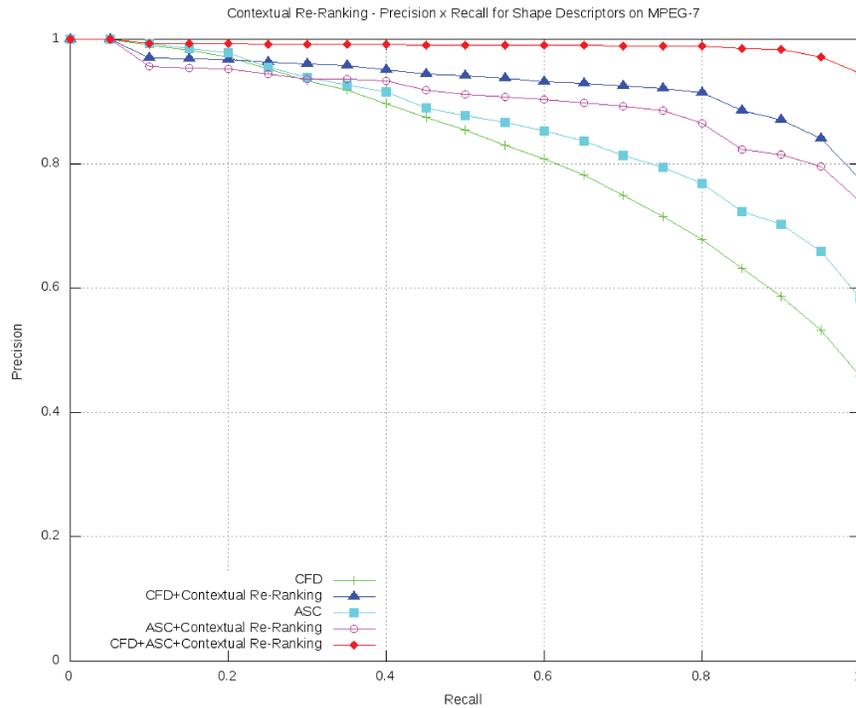


Figure 8.17: Contextual re-ranking and rank aggregation for shape descriptors.



## Chapter 9

# Efficient Image Re-Ranking Computation on GPUs

In the past few years, there have been considerable research to improve the distance measures in CBIR systems, using contextual information. Re-ranking approaches have been proposed to improve the effectiveness of search tasks, replacing pairwise similarities by more global affinity measures that consider the relationships among the collection images. The goal of these methods is to mimic the human behavior on judging the similarity among objects by considering specific *contexts*.

The usefulness of CBIR systems depends on both the *effectiveness* and the *efficiency* of the retrieval process. While the effectiveness is related to the quality of retrieved images, the efficiency is related to the time spent to obtain the results. Both are indispensable for useful and real world systems. Aiming at computing the relationship among images, re-ranking algorithms often consider all distances among images in a given dataset, which represent a large computational effort. Typically, image re-ranking algorithms assume  $O(N^2)$  to  $O(N^3)$  complexity, which poses a great challenge on efficiently executing the re-ranking methods.

The computation of re-ranking algorithms can be parallelized using Graphic Processing Units (GPUs) devices. Despite the fact that these distances can be computed concurrently in multi-core machines and the execution is likely to achieve linear speedups, current multi-core machines are still limited to a couple of cores, which limit the maximum available parallelism. GPUs, on the contrary, are capable of executing up to 1600 threads at the same time, two orders of magnitude higher than state-of-the-art multi-core CPUs. Graphic Processing Units (GPUs) have evolved into massive parallel architectures capable of executing hundreds of operations per cycle. These devices are no longer used exclusively for graphics processing. In fact, GPU hardware and programming models are constantly being improved to enable general purpose applications to execute on it.

General Purpose Graphic Processing Units (GPGPUs), as it is known, is already present on several of the Top 500 high performance computing systems list. Despite the computational power, its execution and programming models are different from traditional General Purpose Processors (GPPs), which renders simple recompilation techniques useless when porting applications from GPPs to GPGPUs.

In this chapter, we address the image re-ranking performance challenges by designing and implementing a re-ranking algorithm that takes advantage of the massive amount of parallelism at GPUs. We propose a parallel GPU-based solution which can speed up the *Contextual Re-Ranking* algorithm computation (described in Chapter 8).

The chapter is organized as follows: Section 9.1 discusses General Purpose computing on GPUs. Section 9.2 describes the proposed parallel solution for re-ranking computing. Finally, experimental design and results are reported in Section 9.3.

## 9.1 General Purpose computing on GPUs (GPGPUs)

Graphics Processing Units (GPUs) are power efficient massively parallel computing devices. GPUs are fast emerging parallel processors due to their high computation power and low price. The massive data processing capability of the GPU has been attracting researchers to exploit it for general purpose computing [85]. In this way, once specially designed for computer graphics, today's GPUs are general-purpose parallel processors with support for accessible programming interfaces, as OpenCL (detailed in Section 9.2.1).

Iterative algorithms are at the core of several scientific applications, which have traditionally been parallelized and optimized for large multi-processors, either based on shared memory or clusters of interconnected nodes. Gunarahne et al. [98] proposed a GPU-based solution for iterative statistical applications. Three iterative statistical algorithms (K-Means, Multi-Dimensional Scaling (MDS), and PageRank) were designed and implemented using OpenCL. A GPGPU approach with a large number of processing units for on-line machine learning was proposed by Webers [108]. The work considers the Stochastic Gradient Descent algorithm, discussing a parallel solution, its performance gain, and variations in accuracy. A study on efficient execution of PageRank algorithm on AMD GPUs is presented by Wu et al. [107]. They analyze the characteristic of the sparse matrices used in PageRank, and introduce a fast sparse matrix-vector multiplication (SpMV) implementation using a modified Compressed Sparse Row (CSR) format.

Strong and Gong [92] discuss how to efficiently organize a collection of images based on their similarities. The objective is to facilitate photo browsing and searching. The proposed approach first generates a feature vector for each image in the collection. The

feature vectors are then used to train a Self Organizing Map (SOM) on the GPUs. An image retrieval approach using GPUs is proposed by Pham et al. [79], applying the Factorial Correspondence Analysis (FCA). FCA is a method for analyzing textual data, which is adapted to images using SIFT local descriptors. FCA is used to reduce dimensions and to limit the number of images to be considered during the search. Two algorithms on GPU for image retrieval using FCA are proposed.

Given that both image re-ranking and GPGPUs are recent approaches, to the best of our knowledge, there is no study about the use of GPUs for efficient image re-ranking computation.

## 9.2 GPU Acceleration of the Contextual Re-Ranking Algorithm

This section discusses the design of a parallel implementation of the Contextual Re-Ranking algorithm. A briefly presentation of the OpenCL standard, used for this implementation, is presented in the following.

### 9.2.1 OpenCL

OpenCL is a new industry standard for task-parallel and data-parallel heterogeneous computing on a variety of modern CPUs, GPUs, DSPs, and other microprocessor designs [91]. This trend toward heterogeneous computing and highly parallel architectures has created a strong need for software development infrastructure in the form of parallel programming languages and subroutine libraries that can support heterogeneous computing on multiple vendors hardware platforms. In OpenCL, a program is executed on a computational *device*, which can be a CPU, GPU, or another accelerator. GPU devices typically contain one or more *compute units* (processor cores). These units are themselves composed by one or more single-instruction multiple-data (SIMD) processing elements (PE) that execute instructions in lock-step.

A *kernel* is a function declared in an OpenCL program and is executed on an OpenCL *device*. The *kernels* are computing functions dynamically compiled and scheduled for execution by calling a C runtime library. Parallel executions of a *kernel* are invoked on a device by a *command*. Each instance of a kernel running on a compute unit is called a work-item. A *work-item* is executed by one or more processing elements as part of a *work-group* executing on a compute unit. OpenCL maps the total number of work-items to be launched onto an *n-dimensional* grid, called as *ND-Range*.

### 9.2.2 Parallel Contextual Re-Ranking

The *Contextual Re-Ranking* algorithm presents great potential for parallelization, as a large number of context images ( $N \times K$ ) are created at each iteration and they do not depend on each other for processing. The re-ranking step (the re-sort of  $N$  ranked lists) also can be computed in a parallel way for each ranked list. However, the parallelization of the algorithm presents several challenges (e.g., concurrent accesses to the affinity matrix  $W$ ) that may require the design of synchronization approaches.

The re-ranking algorithm could be designed in a single OpenCL kernel, repeatedly executed at each iteration. However, some steps must be completed in a predefined order. This occurs for example when the affinity matrix  $W$  and the distance matrix  $A$  are computed. The matrix  $W$  should be completely computed before starting the computation of  $A$ .

In a parallel implementation of the re-ranking algorithm, some barriers should be respected to ensure the correct data dependence when executing the algorithm. Although barriers are available in OpenCL for synchronization, they only provide synchronization among kernels in a same work-group. Global synchronization can be obtained using atomic operations in global memory or using different kernels through the *dispatch queue*. In this way, the need for global synchronization between steps of the algorithm is the main motivation for dividing the algorithm in different kernels. As an initial point, the re-ranking algorithm was divided into three OpenCL kernels and three segments of serial code. Figure 9.1 illustrates the overall design of the parallel algorithm. In the following, we described the proposed kernels and the segments of serial code:

- **Context Images Processing:** given the distance matrix  $A_t$  and the set of ranked lists  $\mathcal{R}_t$ , this kernel processes a *context image*. It constructs a *context image*, applies the image processing procedures (thresholding and filtering) and obtains the resulting black pixels. Based on these pixels, it computes the increment values that are used later to update the matrix  $W$ . Two dimensions of work items are created to compute this kernel, in a total of  $N \times K$  work items.
- **Distances Computation:** based on the current distance matrix  $A_t$  and the affinity matrix  $W$ , computed by the previous kernel, this kernel computes a new distance matrix  $A_{t+1}$  for the next iteration. Each cell of the matrix can be computed independently from others. In this way,  $N \times N$  work items (one for each cell) are created, divided into two dimensions.
- **Sort of Ranked Lists:** based on the new distance matrix  $A_{t+1}$  (computed for the next iteration), a new set of ranked lists  $\mathcal{R}_{t+1}$  can also be computed. A single dimension is considered and  $N$  work items are created - one for each ranked list.

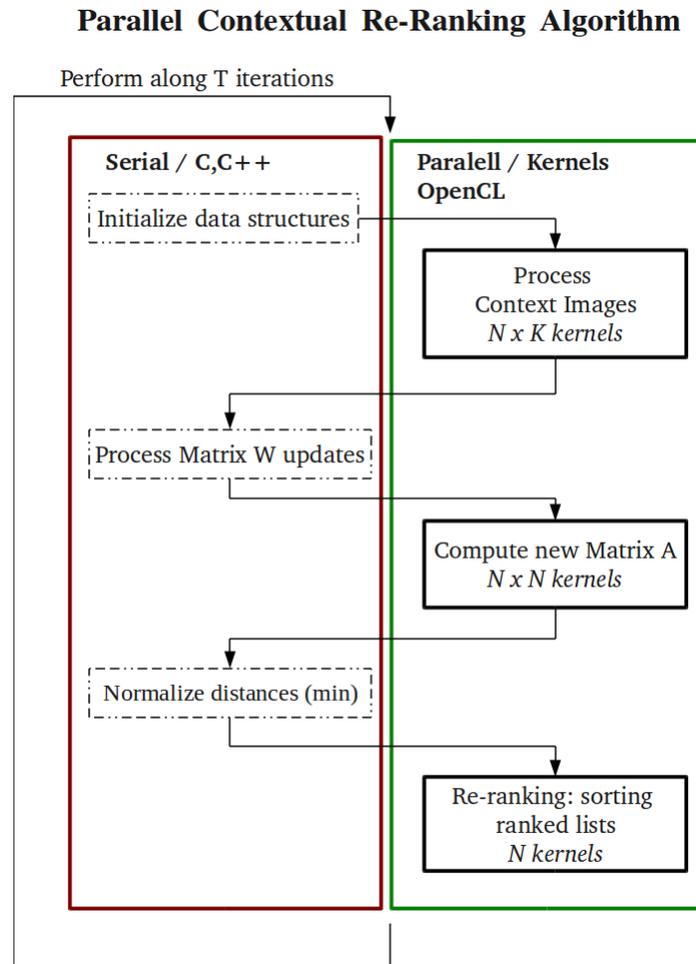


Figure 9.1: Parallel project of the Contextual Re-Ranking Algorithm.

Although the kernels represent the majority of the computational effort needed by the re-ranking algorithm, there are some *gaps* between them. These gaps, which were also considered when designing the algorithm, represent segments of code that should be synchronized and executed in a serial way.

The first kernel, for example, processes context images computing the increments of the matrix  $W$ . However, these increments are produced by different threads and the concurrent writing in matrix  $W$  may produce loss of increments. In this way, the kernel uses a list of increments for storing the increments without concurrent access among different threads and returns this list to the host device. In the host device, the list is sequentially addressed to increment the matrix  $W$ . This process is illustrated in Figure 9.2.

Other steps, as the initialization of matrix  $W$  and normalization of matrix  $A$ , are also computed in the host device. The initialization of matrix  $W$  consists in assigning the

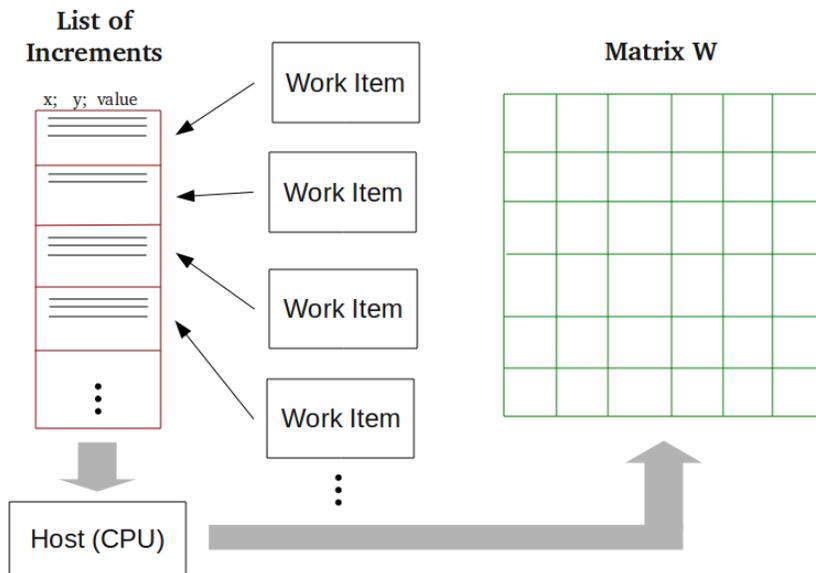


Figure 9.2: Update model using list of increments.

value 1 to all positions of the matrix. Although this step could be computed in parallel by the first kernel, it has a non-compatible number of work items. While the first kernel has  $N \times K$  work items, the initialization requires  $N \times N$  (one for each cell) or  $N$  (on for each row). Therefore, this step is serially computed. The normalization of matrix  $A$  consists in the computation of the *min* value between correspondent positions of matrix  $A$ . This step is serially computed because it involves different positions of the matrix  $A$  and it should be globally synchronized if computed in parallel.

### 9.2.3 Optimizations

The existence of serial segments of code always represents bottlenecks, where no gains can be obtained by parallelization. In this section, we present strategies for exploiting parallelization even in segments initially designed for being executed in a sequential way.

The first segment considered for optimization is concerned with the processing the increments of matrix  $W$ . The list of increments is sequentially checked and processed in the matrix  $W$ . In addition to the sequential processing of the list of increments, other negative implications related to memory storage and data transferring are expected. To ensure the inexistence of concurrent access over the list of increments, a predefined number of list elements (maximum of possible increments) are reserved for each work item. In this way, since the number of effective increments is often very lower than the maximum possible, a large portion of memory needed to store the list is wasted. Another negative impact is the fact of the whole list (including wasted elements) is transferred back to

the host device. Considering this characteristics, the ideal implementation computes the increments direct in the matrix  $W$ . As discussed in previous section, the inexistence of global synchronization (and lock engines) in OpenCL motivates the use of the list of the increments, once concurrent writing in matrix  $W$  can cause loss of increments.

However, in the common case, the matrix  $W$  is very sparse and the number of increments for each context image is lower than the maximum possible. This behavior indicates a low probability for concurrent increments in the same positions of the matrix  $W$ . In this scenario, we propose to change the initial implementation in order to reduce concurrent writing in matrix  $W$  and to enable direct increments on the matrix  $W$ . The direct increments on the matrix  $W$  are illustrated in Figure 9.3.

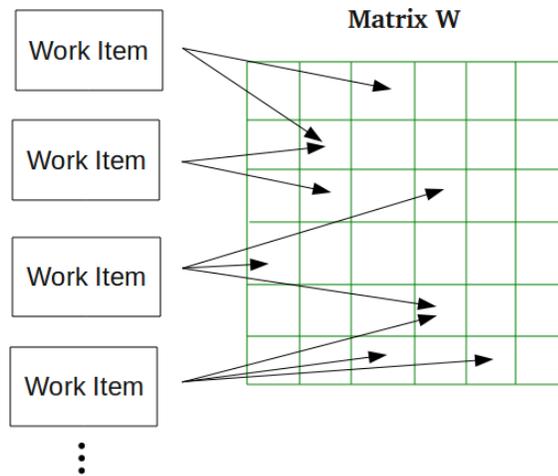


Figure 9.3: Update model with increments direct on Matrix  $W$ .

The number of work items and dimensions configurations of the first kernel also were changed. We consider a single dimension with  $N$  work items, such that the  $K$  context images produced for a given image are processed in a unique work item. In this situation, in which different context images likely have increments in common cells, they are processed in a single work item avoiding loss of increments.

Removing the synchronization mechanism and allowing direct updates on matrix the  $W$  increases the performance of the algorithm. It is important to notice that concurrent accesses to the same elements of  $W$  may lead to the loss of some increments. Theoretically, this loss of information may affect the accuracy of the re-ranking algorithm. However, as we show in Section 9.3.3, a very low impact on the accuracy of the algorithm was observed in practice.

The second situation where the parallelization is not exploited is the normalization of the matrix  $A$ . This step is sequentially processed because it computes the *min* value between transposed positions of matrix, involving different work items (one for each po-

sition of the matrix) and causing synchronization problems. We aim at parallelizing even the normalization of the matrix, decreasing the segments of serial code. In this case, we propose a dynamic programming based strategy. Instead of two dimensions and  $N \times N$  work items, with one for each position of the matrix, we use a single dimension with  $N$  work items. Let  $x$  be the global identifier of a work item. This work item is responsible for computing the distances of  $img_x$  to  $img_y$ , where  $y > x$ . In this way, the kernel can compute the matrix  $A$  in positions  $A_{xy}$ ,  $A_{yx}$  and the *min* value between them without synchronization.

## 9.3 Experimental Evaluation

This section presents the experiments conducted aiming at assessing the impact of the proposed parallel strategies in both effectiveness and efficiency. We also compare segments of the algorithm considering executions in different languages (C/C++ and OpenCL) and different devices (CPU/GPU).

### 9.3.1 Experimental Setup

The hardware environment is composed by the devices:

- *CPU*: AMD Opteron 6168 1.9GHz - 12 cores
- *GPU*: ATI FirePro V7800

The software environment is given by:

- *Operating System*: Linux 2.6.32-33 - Ubuntu 10.04
- *OpenCL SDK*: OpenCL 1.1 AMD-APP-SDK-v2.4

We use the MPEG-7 [48] dataset and the CFD [68] shape descriptor for main experimental evaluations.

### 9.3.2 Performance Results

We conducted a set of experiments aiming at evaluating the performance of the *Contextual Re-Ranking* algorithm with regard to different aspects:

- *Parallel and serial implementations*: we considered equivalent implementations of the re-ranking algorithm in OpenCL and C/C++ for experimental evaluation.

Table 9.1: Performance comparison for different kernel implementations - No optimizations, using list of increments

Language/Device	Measure Time (s)	Context Image Processing	Distances Computing	Sorting Ranked Lists	Serial Code
Serial C/C++	<b>Total Time</b>	7.3661 ± 0.0125	0.2359 ± 0.0006	0.9766 ± 0.0067	~ 1.9808 ± 0.0047
OpenCL CPU	<i>Computing Time</i>	0.7329 ± 0.0057	0.1388 ± 0.0059	0.0508 ± 0.0018	
	<i>Memory Transf.</i>	0.8030 ± 0.0030	0.1355 ± 0.0033	0.1403 ± 0.0032	
	<b>Total Time</b>	1.7058 ± 0.0100	0.2877 ± 0.0065	<b>0.2111 ± 0.0030</b>	
OpenCL GPU	<i>Computing Time</i>	0.3466 ± 0.0052	0.0234 ± 0.0001	2.9105 ± 0.0014	
	<i>Memory Transf.</i>	0.7491 ± 0.0007	0.0600 ± 0.0004	0.0588 ± 0.0002	
	<b>Total Time</b>	<b>1.0992 ± 0.0057</b>	<b>0.0861 ± 0.0006</b>	2.9709 ± 0.0014	

Table 9.2: Performance comparison for different kernel implementations - Optimizations and increments direct on Affinity Matrix W

Language/Device	Measure Time (s)	Context Image Processing	Distances Computing	Sorting Ranked Lists	Serial Code
Serial C/C++	<b>Total Time</b>	6.1602 ± 0.0120	0.4049 ± 0.0027	0.9683 ± 0.0001	~ 0.1290 ± 0.0045
OpenCL CPU	<i>Computing Time</i>	0.6451 ± 0.0029	0.0532 ± 0.0015	0.0499 ± 0.0014	
	<i>Memory Transf.</i>	0.1909 ± 0.0030	0.1418 ± 0.0025	0.1380 ± 0.0048	
	<b>Total Time</b>	0.8534 ± 0.0054	0.2145 ± 0.0042	<b>0.2068 ± 0.0040</b>	
OpenCL - GPU	<i>Computing Time</i>	0.5059 ± 0.0014	0.0574 ± 0.0001	2.7927 ± 0.0249	
	<i>Memory Transf.</i>	0.0845 ± 0.0023	0.0541 ± 0.0001	0.0539 ± 0.0001	
	<b>Total Time</b>	<b>0.5926 ± 0.0011</b>	<b>0.1130 ± 0.0001</b>	2.8481 ± 0.0248	

- *Different segments of code:* we considered the division of re-ranking algorithm in three kernels, measuring independently the run time of each one.
- *Different devices:* the parallel executions were performed in CPU and GPU devices.
- *Optimizations:* we evaluated our parallel implementation considering the proposed optimizations.

We measured the run time repeating 5 executions and computing the average time with correspondent 95% confidence intervals. Table 9.1 presents the results for the initial parallel implementation, which does not include the optimizations discussed in Section 9.2.3. The best performance, for each kernel/segment code is presented in bold. We can observe that the context image processing kernel, which represents the major computational effort of the algorithm, presents a significantly performance increase. The speedup from serial implementation (7.3661 s) to OpenCL GPU (1.0992 s) in this kernel is  $6.7 \times$ . Table 9.2 presents the results for the optimized version of the algorithm. We can observe that both serial and memory transfers times decreased significantly.

We also considered the execution of the optimized implementation of the algorithm combining the best combination of kernels and devices in which they are executed. The first two kernels (context images processing and distance computation) are computed in OpenCL GPU and the third kernel (sort of ranked lists) is computed using OpenCL CPU. Table 9.3 presents the results for the best combination of kernels, summing up parallel and

Table 9.3: Best Combination of Kernels/Devices - Total Run Time

Kernel/Segment Code	Language/Device	Run Time (s)
Context Image Processing	OpenCL - GPU	$0.5926 \pm 0.0011$
Distances Computing	OpenCL - GPU	$0.1130 \pm 0.0001$
Sorting Ranked Lists	OpenCL - CPU	$0.2068 \pm 0.0040$
Serial Code	Serial C/C++	$0.1290 \pm 0.0045$
<b>Sub-Total</b>		<b>1.0414</b>
OpenCL Environment	-	$0.8995 \pm 0.0416$
<b>Total</b>		<b>1.9409</b>

serial run time. We also report the OpenCL environment creation time. Although very significant for the considered execution, the OpenCL environment represents a fixed time that does not increase when increasing the datasets. Considering the total elapsed time of the optimized version (1.9409 s), including the OpenCL environment time, the speedup to the C/C++ serial version (7.6624 s) is  $3.9 \times$ . Considering only the memory transfers and the computing time (1.0414 s), the speedup is  $7.4 \times$ . For comparison, the original Java implementation [67] takes 155 s when performing the re-ranking algorithm, which represents a speedup of  $79.9 \times$  considering the OpenCL environment time and  $148.9 \times$  considering only the computing time and memory transfers.

### 9.3.3 Effectiveness Analysis

In this section, we evaluate the impact of the optimization strategy on the effectiveness of the re-ranking algorithm, since there can be loss of increments on the matrix  $W$  and consequently effectiveness variations. We aim at assessing the impact of this approach on the final effectiveness of the CBIR system. We consider the same CBIR descriptors and datasets used in the evaluation of Contextual Re-Ranking algorithm.

Table 9.4 presents a comparison of the serial C/C++ with the optimized implementation in OpenCL. The experiments were performed on the GPU, since it is more susceptible to synchronization issues due to the large number of threads. An average of 5 executions were considered. As we can observe, the variations of effectiveness results are very small. The use of the optimized version is then justified by the significant gains in performance.

Table 9.4: Contextual Re-Ranking Effectiveness Evaluation - MAP.

<b>Descriptor</b>	<b>Type</b>	<b>Dataset</b>	<b>Score (MAP)</b>	<b>CR-R Serial C/C++</b>	<b>CR-R OpenCL GPU</b>
SS [17]	Shape	MPEG-7	37.67%	<b>44.79%</b>	<b>44.87%</b>
BAS [2]	Shape	MPEG-7	71.52%	<b>76.60%</b>	<b>76.59%</b>
IDSC [52]	Shape	MPEG-7	81.70%	<b>87.39%</b>	<b>87.60%</b>
ASC [53]	Shape	MPEG-7	85.28%	<b>89.82%</b>	<b>90.12%</b>
CFD [68]	Shape	MPEG-7	80.71%	<b>92.76%</b>	<b>92.98%</b>
GCH [93]	Color	Soccer	32.24%	<b>33.02%</b>	<b>32.70%</b>
ACC [41]	Color	Soccer	37.23%	<b>39.86%</b>	<b>39.99%</b>
BIC [90]	Color	Soccer	39.26%	<b>43.04%</b>	<b>42.80%</b>
LBP [60]	Texture	Brodatz	48.40%	<b>49.06%</b>	<b>49.32%</b>
CCOM [46]	Texture	Brodatz	57.57%	<b>63.67%</b>	<b>63.69%</b>
LAS [94]	Texture	Brodatz	75.15%	<b>78.48%</b>	<b>79.16%</b>



# Chapter 10

## Combining Re-Ranking and Rank Aggregation Methods

As we have been discussed, in the past few years, there has been considerable research on exploiting *contextual information* for improving the distance measures and *re-ranking images* in CBIR systems [42, 45, 62, 77, 84, 114, 115]. Another approach for improving CBIR systems is based on *rank aggregation* techniques [4, 12, 28]. Promising results have been obtained considering several approaches and techniques, including those proposed in this thesis.

However, although a lot of efforts have been employed to develop new re-ranking and rank aggregation methods, few initiatives aim at combining the existing methods. Besides that, in the same way that different CBIR descriptors produce different and complementary rankings, results of *re-ranking* and *rank aggregation* methods can also be combined to obtain more effective results. In this chapter, we propose three novel approaches for combining re-ranking and rank aggregation methods aiming at improving the effectiveness of CBIR systems. We discuss how to combine (i) re-ranking algorithms; (ii) rank aggregation algorithms, and both (iii) re-ranking and rank aggregation algorithms. We conducted a large evaluation protocol involving shape, color, and texture descriptors datasets and comparisons with baseline methods. Experimental results demonstrate that our combination approaches can further improve the effectiveness of CBIR systems.

This chapter is organized as follows. Section 10.1 presents our approach for combining re-ranking methods. Section 10.2 presents the combination of re-ranking using rank aggregation methods. The combination of rank aggregation methods is discussed in Section 10.3. Section 10.4 presents the re-ranking and the rank aggregation methods considered in experiments. Section 10.5 presents the experimental evaluation.

## 10.1 Cascading Re-Ranking

In Chapter 2, we defined a re-ranking algorithm as an implementation of a function  $f_r(A, \mathcal{R})$ . Note that both input and output of the function  $f_r$  are distance matrices (the set of ranked lists  $\mathcal{R}$  can be computed based on this distance matrix). In this way, an output matrix obtained from a given function  $f_{r_1}$ , implemented by a re-ranking algorithm, can be sent to another re-ranking algorithm  $f_{r_2}$ , aiming at further improving its effectiveness. We call this combination approach as “*cascading re-ranking*”, as it can be applied to a chain of re-ranking algorithms. The main motivation of this approach is based on two facts: (i) different re-ranking algorithms exploit contextual information in different ways and can be complementary (one algorithm can improve the quality of ranked lists that others did not); (ii) the second re-ranking algorithm can take advantage of improvements obtained by the first one. Figure 10.1 illustrates this combination approach, considering two re-ranking algorithms.



Figure 10.1: Cascading Re-Ranking.

## 10.2 Re-Ranking with Rank Aggregation Combination

A single image descriptor can be submitted to different re-ranking algorithms, defined by a set of functions  $\{f_{r_1}, f_{r_2}, \dots, f_{r_m}\}$ . In this scenario, a different distance matrix is produced for each re-ranking algorithm. However, the results can be complementary (one re-ranking can exploit contextual information that others did not). In this way, a rank aggregation method can combine the results of different re-ranking algorithms in order to obtain a single, and more effective distance matrix. Figure 10.2 illustrates this process, considering the use of two re-ranking methods followed by a rank aggregation step.

## 10.3 Agglomerative Rank Aggregation

The two previous combination approaches consider that only one image descriptor is available. This section presents the “*Agglomerative Rank Aggregation*” approach that

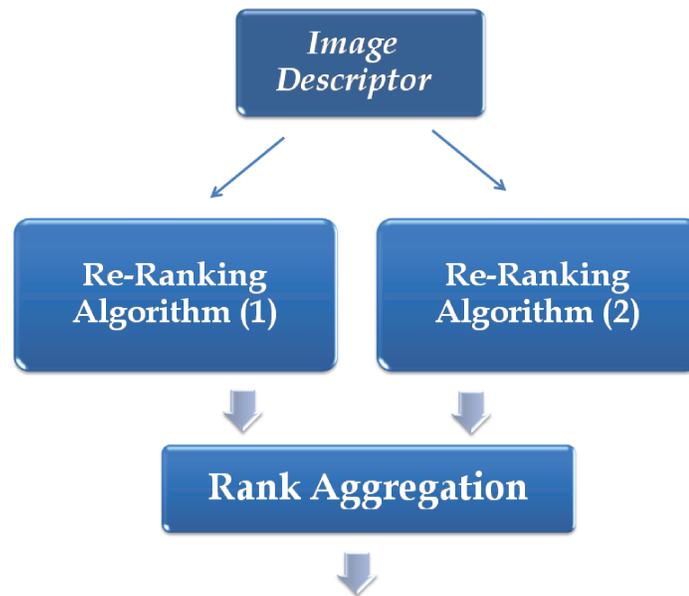


Figure 10.2: Re-Ranking with Rank Aggregation Combination.

uses several image descriptors as input.

Given a set of image descriptors, different rank aggregation methods can be employed for combining them. However, each rank aggregation method produces a different output. In this way, another rank aggregation method can be employed for combining the results of the first rank aggregation methods. This combination approach uses a hierarchical agglomerative method, in which the rank aggregation methods can be divided in layers. Figure 10.3 illustrates our proposed approach for a two-layer rank aggregation scenario.

## 10.4 Used Re-Ranking and Rank Aggregation Methods

### 10.4.1 Re-Ranking Methods

This section briefly describes the two re-ranking methods considered for combination in our experimental evaluation. We consider the Contextual Re-Ranking and the RL-Sim Re-Ranking because these methods use very different approaches for processing contextual information, and therefore, are expected to be complementary.

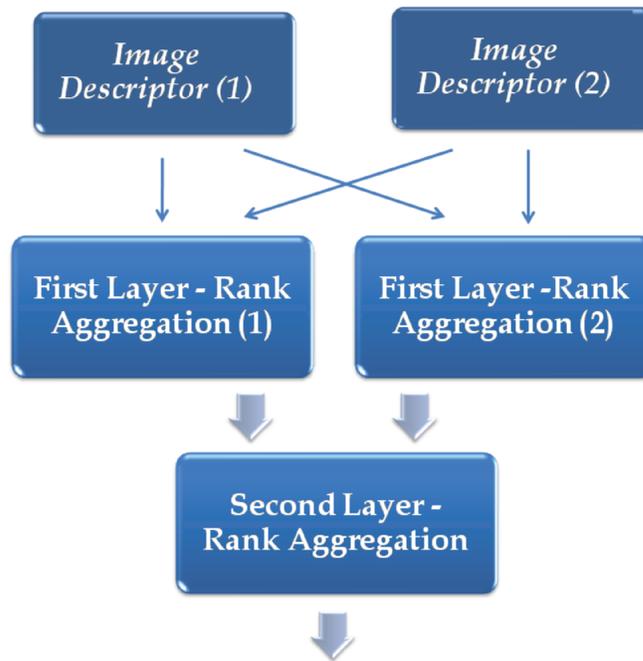


Figure 10.3: Agglomerative Rank Aggregation.

### Contextual Re-Ranking

The Contextual Re-Ranking (CRR) algorithm, presented in Chapter 8, re-ranks images by taking into account *contextual information* encoded in ranked lists and distance among images. The algorithm uses a *gray scale image* representation of distance matrices computed by CBIR descriptors, referenced as *context image*. The context image is constructed for the  $k$ -nearest neighbors of a query image and analyzed using image processing techniques.

### RL-Sim Re-Ranking Algorithm

The main motivation of RL-Sim Re-Ranking algorithm, presented in Chapter 7, relies on the conjecture that *contextual information encoded in the similarity between ranked lists can provide useful information for improving effectiveness of CBIR descriptors*. In general, if two images are similar, their ranked lists should be similar as well. A function that measures the similarity between ranked lists are considered for processing the contextual information. We considered the  $kNN$  approach for neighborhood set retrieval and the *intersection metric* for measuring the similarity between ranked lists.

## 10.4.2 Rank Aggregation Methods

This section briefly describes the rank aggregation approaches considered in our experimental evaluation.

### Contextual Rank Aggregation

The *Contextual Rank Aggregation (CRA)* algorithm aims at combining the results of different descriptors. The main idea consists in using the Contextual Re-Ranking [67] algorithm, discussed in Chapter 8, but using the affinity matrix  $W$  for accumulating updates of different descriptors at the first iteration.

### RL-Sim Rank Aggregation

We used the multiplication approach with the RL-Sim Algorithm, proposed in Chapter 7. The distance matrices are first combined and later submitted to the RL-Sim Re-Ranking algorithm.

### Set Rank Aggregation (SetRA)

In this section, we propose a simple method to be used as a *second layer* rank aggregation in the agglomerative approach, presented in Section 10.3. We consider the strategy of modeling the ranked lists as *sets* of different sizes. This strategy is also used by the RL-Sim [72] algorithm, discussed in Chapter 7. We use the same function  $\psi$  described in Chapter 7 for computing the similarity between ranked lists. We refer to this method as *Set Rank Aggregation (SetRA)* along the chapter.

The main idea is to compute the similarity between ranked lists, defined by each distance matrix being combined, and sum up these similarity scores in order to obtain a new combined score. Let  $m$  be the number of matrices being combined and  $R_{i_x}$  be the ranked list produced by matrix  $A_i$  for image  $img_x$  and  $R_{i_y}$  for image  $img_y$ , then the new combined similarity score  $\psi_c$  is computed as follows:

$$\psi_c(img_x, img_y, K) = \sum_{i=1}^m \psi(R_{i_x}, R_{i_y}, K). \quad (10.1)$$

The new combined distance is computed using the Equation 10.1.

## 10.5 Experimental Evaluation

This section presents a set of conducted experiments for demonstrating the effectiveness of our combination approaches. We analysed our approaches under several aspects and

compared our results with other methods from the literature. Section 10.5.1 describes the datasets and descriptors used. Section 10.5.2 presents the experimental results concerning the cascading re-ranking methods. Section 10.5.3 presents results for combination of re-ranking methods, and Section 10.5.4 presents the evaluation of agglomerative rank aggregation approach.

### 10.5.1 Descriptors and Datasets

#### Shape

We use the MPEG-7 dataset [48], considering the MAP and the bullseye score as effectiveness measures. We consider six shape descriptors: SS [17], BAS [2], IDSC [52], CFD [68], ASC [53], and AIR [35].

#### Color

We evaluate our method for three color descriptors: BIC [90], ACC [41], and GCH [93]. The experiments were conducted on the Soccer dataset [100].

#### Texture

The experiments consider three texture descriptors: LBP [60], CCOM [46], and LAS [94]. We used the Brodatz [7] dataset.

### 10.5.2 Cascading Re-Ranking

This section discusses the use of our cascading approach for combining re-ranking methods. The main goal of this experiment is to validate that our proposed approach can be used with different re-ranking methods. In this way, we considered the MPEG-7 dataset, which has available various baselines.

We considered four different re-ranking approaches: the Mutual kNN Graph [45], the Distance Optimization Algorithm, presented in Chapter 4, the Contextual Re-Ranking, presented in Chapter 8, and RL-Sim algorithm, presented in Chapter 7. We also considered the Contextual Re-Ranking and RL-Sim applied to all algorithms.

Table 10.1 presents the results for *Recall@40* measure. We can observe that the gains are positives for all combinations, ranging from +0.11% to +1.99% . The positive gains shows that, even with contextual information already exploited by the first re-ranking employed, the second re-ranking can further improve the effectiveness when combined by our cascading approach.

Table 10.1: Cascading Re-Ranking Methods on the MPEG-7 dataset (*Recall@40*).

Descriptor	Score	Re-Ranking Algorithm 1	Score	Re-Ranking Algorithm 2	Cascade Score	Gain
CFD [68]	84.43%	Distance Optimization	92.56%	Contextual Re-Ranking	93.39%	+10.61%
CFD [68]	84.43%	Distance Optimization	92.56%	RL-Sim Re-Ranking	94.40%	+11.81%
IDSC [52]	85.40%	Mutual kNN Graph [45]	93.40%	Contextual Re-Ranking	93.68%	+9.70%
IDSC [52]	85.40%	Mutual kNN Graph [45]	93.40%	RL-Sim Re-Ranking	94.09%	+10.18%
CFD [68]	84.43%	RL-Sim Re-Ranking	94.13%	Contextual Re-Ranking	94.23%	+11.61%
CFD [68]	84.43%	Contextual Re-Ranking	95.71%	RL-Sim Re-Ranking	95.94%	+13.63%

### 10.5.3 Combining Re-Ranking methods with Rank Aggregation

This section presents the evaluation of our approach for combining re-ranking with rank aggregation algorithms. We consider the Contextual Re-Ranking (CRR) [67] and the RL-Sim [72] re-ranking algorithms, discussed in Section 10.4.1 and the Set Rank Aggregation presented in Section 10.4.2. We aim also at evaluating the use of the proposed approach for different descriptors and datasets. We considered three datasets and twelve descriptors, including shape, color, and texture descriptors.

Table 10.2 presents the MAP scores for the RL-Sim [72] and Contextual Re-Ranking [67] algorithms in isolation (as baselines), and considering their combination. As we can observe, for almost all descriptors our combination approach presents a higher MAP score than both baselines, with significant gains. Exceptions are the LBP [60] and LAS [94] descriptors, in which the RL-Sim [72] presents low gains. However, we should note that, even for those cases, our combination approach presents a MAP score higher than the worst re-ranking methods (in fact, close to the average result between the two methods).

Our approach also presents a higher average score when compared with both re-ranking algorithms. The combination score (67.72%) represents a gain of +2.48% for the Contextual Re-Ranking [67] algorithm (66.08%) and +3.67% for the RL-Sim [72] algorithm (65.32%). The last column presents the effectiveness gains considering the original descriptor score, with a significant average gain of +11.52%.

We also considered the bullseye score (*Recall@40*) for shape descriptors on the MPEG-7 dataset. Table 10.3 present the effectiveness results considering the *Recall@40* measure. Similar results to MAP measure can be observed. Our combination approach also presents better average scores (86.69%) than both re-ranking algorithms. Note that the RL-Sim [72] algorithm yields better average *Recall@40* scores than the Contextual Re-Ranking [67] algorithm. The contrary occurs considering the MAP scores (Contextual Re-Ranking [67] yields better MAP scores in Table 10.2). However, our combination approach presents better scores considering both measures.

[!ht]

Table 10.2: Re-Ranking with Rank Aggregation Combination on CBIR Tasks (*MAP*).

Image Descriptor	Type	Dataset	Score	Re-Ranking 1: <i>RL-Sim</i>	Re-Ranking 2: <i>Contextual Re-Ranking</i>	Rank Aggregation: <i>SetRA</i>	Gain
SS [17]	Shape	MPEG-7	37.67%	43.06%	44.79%	<b>47.33%</b>	+25.64%
BAS [2]	Shape	MPEG-7	71.52%	74.57%	76.60%	<b>78.31%</b>	+9.49%
IDSC [52]	Shape	MPEG-7	81.70%	86.75%	87.39%	<b>88.66%</b>	+8.52%
CFD [68]	Shape	MPEG-7	80.71%	88.97%	92.76%	<b>92.94%</b>	+15.15%
ASC [53]	Shape	MPEG-7	85.28%	88.81%	89.82%	<b>90.62%</b>	+6.26%
AIR [35]	Shape	MPEG-7	89.39%	93.54%	94.49%	<b>97.15%</b>	+8.68%
GCH [93]	Color	Soccer	32.24%	33.66%	33.02%	<b>33.78%</b>	+4.78%
ACC [41]	Color	Soccer	37.23%	43.54%	39.86%	<b>46.60%</b>	+25.17%
BIC [90]	Color	Soccer	39.26%	43.45%	43.04%	<b>47.27%</b>	+20.40%
LBP [60]	Texture	Brodatz	48.40%	47.77%	49.06%	<b>47.93%</b>	-0.97%
CCOM [46]	Texture	Brodatz	57.57%	62.01%	63.67%	<b>64.20%</b>	+11.52%
LAS [94]	Texture	Brodatz	75.15%	77.81%	78.48%	<b>77.89%</b>	+3.65%
<b>Average</b>			61.34%	65.32%	66.08%	<b>67.72%</b>	+11.52%

Table 10.3: Re-Ranking and Rank Aggregation Combination for Shape Descriptors on the MPEG-7 dataset (*Recall@40*).

Shape Descriptor	Score	Re-Ranking 1: <i>RL-Sim</i>	Re-Ranking 2: <i>Contextual Re-Ranking</i>	Rank Aggregation: <i>SetRA</i>	Gain
SS [17]	43.99%	53.15%	51.38%	<b>54.69%</b>	+24.32%
BAS [2]	75.20%	82.94%	82.43%	<b>83.51%</b>	+11.06%
IDSC [52]	85.40%	92.18%	91.84%	<b>92.16%</b>	+7.92%
CFD [68]	84.43%	94.13%	95.71%	<b>95.98%</b>	+13.67%
ASC [53]	88.39%	94.69%	93.07%	<b>93.80%</b>	+6.12%
AIR [35]	93.67%	99.90%	99.80%	<b>99.99%</b>	+6.75%
<b>Average</b>	78.51%	86.17%	85.71%	<b>86.69%</b>	+10.42%

### 10.5.4 Agglomerative Rank Aggregation

This section presents the evaluation of our proposed Agglomerative Rank Aggregation Approach, introduced in Section 10.3. We selected two descriptors with the best effectiveness scores for each visual property (*e.g.*, shape, color, and texture). For shape descriptors, we do not consider the AIR [35] descriptor because this descriptor presents a very high effectiveness score by itself. Table 10.4 presents the MAP score of our combination approach. For comparison, we also present the MAP score for descriptors in isolation and combined with first-layer rank aggregation method, considering the RL-Sim [72] and the Contextual Rank Aggregation (CRA) [70] algorithms. We can observe that significant gains are obtained by our combination approach when compared with the results of the descriptors and with the first-layer rank aggregation method.

Table 10.4: Agglomerative Rank Aggregation Combination for CBIR Tasks (*MAP*).

Descriptor	Type	Dataset	First Layer - Rank Aggregation	Second Layer - Rank Aggregation	Score (MAP)
CFD [68]	Shape	MPEG-7	-	-	80.71%
ASC [53]	Shape	MPEG-7	-	-	85.28%
CFD [68] + ASC [53]	Shape	MPEG-7	RL-Sim	-	98.75%
CFD [68] + ASC [53]	Shape	MPEG-7	CRA	-	98.77%
<b>CFD [68] + ASC [53]</b>	<b>Shape</b>	<b>MPEG-7</b>	<b>RL-Sim + CRA</b>	<b>Set Rank Aggregation</b>	<b>99.41%</b>
ACC [41]	Color	Soccer	-	-	37.23%
BIC [90]	Color	Soccer	-	-	39.26%
BIC [90] + ACC [41]	Color	Soccer	RL-Sim	-	44.49%
BIC [90] + ACC [41]	Color	Soccer	CRA	-	42.14%
<b>BIC [90] + ACC [41]</b>	<b>Color</b>	<b>Soccer</b>	<b>RL-Sim + CRA</b>	<b>Set Rank Aggregation</b>	<b>49.00%</b>
CCOM [46]	Texture	Brodatz	-	-	57.57%
LAS [94]	Texture	Brodatz	-	-	75.15%
LAS [94] + CCOM [46]	Texture	Brodatz	RL-Sim	-	80.26%
LAS [94] + CCOM [46]	Texture	Brodatz	CRA	-	81.63%
<b>LAS [94] + CCOM [46]</b>	<b>Texture</b>	<b>Brodatz</b>	<b>RL-Sim + CRA</b>	<b>Set Rank Aggregation</b>	<b>83.70%</b>



# Chapter 11

## Comparing Re-Ranking and Rank Aggregation Methods

This chapter summarizes the experimental results presented along this thesis, presenting various comparisons considering the proposed re-ranking and rank aggregation methods. First we present an effectiveness comparison among the proposed methods, using different datasets, descriptors, and measures. In the following, we compared our approaches with various state-of-the-art methods using previous results reported for well-known datasets. The proposed methods present better effectiveness performance when compared with various methods recently proposed in the literature.

### 11.1 Comparison of the proposed Re-Ranking Methods

This section summarizes the experimental results of the proposed re-ranking methods presented along this thesis, considering the MPEG-7, Soccer, and Brodatz datasets. Table 11.1 presents the MAP scores for the proposed methods (and their respective variations), considering shape, color, and texture descriptors. The best effectiveness result for each descriptor is presented in bold. We can observe that, for each image descriptor, the best effectiveness result is obtained by a different re-ranking method. It indicates that there is no agreement about the *best method* for all descriptors, but that each re-ranking method can be the most appropriate for a given specific descriptor. It occurs because each re-ranking method processes the contextual information considering different strategies and, therefore, providing different and complementary views of the same problem. That confirms our claims related to the usefulness of using combination strategies to further improve the effectiveness of re-ranking approaches.

Table 11.2 presents the bullseye scores ( $Recall@40$ ) for the proposed re-ranking methods considering shape descriptors on the MPEG-7 dataset. We can observe that the RL-Sim re-ranking method presents the best scores for five out of six descriptors, considering this measure. Note, however, that the RL-Sim method has not achieved such good results considering the MAP score.

Table 11.1: MAP scores for proposed re-ranking methods in different CBIR tasks.

Descriptor	Type	Dataset	Score (MAP)	DOA RL-Cor	DOA Cor-Cor	Pairwise Recommend.	Context Spaces kNN	Context Spaces MkNN	RL-Sim kNN Intersection	RL-Sim M-kNN Intersection	RL-Sim kNN Kendall's Tau	RL-Sim M-kNN Kendall's Tau	Context Re-Ranking
SS [17]	Shape	MPEG-7	37.67%	46.53%	<b>48.76%</b>	39.90%	40.74%	42.52%	43.06%	47.70%	44.24%	46.74%	44.79%
BAS [2]	Shape	MPEG-7	71.52%	<b>81.05%</b>	80.84%	77.65%	74.71%	76.07%	74.57%	78.16%	73.25%	75.38%	76.60%
IDSC [52]	Shape	MPEG-7	81.70%	86.94%	87.77%	86.83%	85.87%	<b>88.05%</b>	86.75%	87.67%	85.93%	86.53%	87.39%
CFD [68]	Shape	MPEG-7	80.71%	91.79%	91.40%	91.38%	90.00%	90.41%	88.97%	90.78%	88.40%	89.50%	<b>92.76%</b>
ASC [53]	Shape	MPEG-7	85.28%	88.41%	91.60%	89.55%	90.51%	<b>91.87%</b>	88.81%	90.88%	88.10%	89.92%	89.82%
AIR [35]	Shape	MPEG-7	89.39%	93.54%	95.77%	94.71%	93.16%	96.07%	93.54%	93.52%	<b>96.27%</b>	95.72%	94.49%
GCH [93]	Color	Soccer	32.24%	30.78%	33.13%	32.35%	32.97%	<b>33.96%</b>	33.66%	33.84%	32.96%	33.76%	33.02%
ACC [41]	Color	Soccer	37.23%	42.46%	45.24%	40.31%	39.35%	45.07%	43.54%	44.78%	44.29%	<b>46.02%</b>	39.86%
BIC [90]	Color	Soccer	39.26%	38.16%	44.23%	42.64%	43.07%	45.00%	43.45%	44.08%	43.76%	<b>45.58%</b>	43.04%
LBP [60]	Texture	Brodatz	48.40%	<b>52.31%</b>	49.34%	51.92%	49.34%	48.20%	47.77%	48.51%	45.20%	45.78%	49.06%
CCOM [46]	Texture	Brodatz	57.57%	59.27%	64.60%	<b>66.46%</b>	61.49%	61.44%	62.01%	63.48%	60.30%	61.41%	63.67%
LAS [94]	Texture	Brodatz	75.15%	80.36%	<b>81.17%</b>	80.73%	79.67%	77.18%	77.81%	78.11%	75.62%	76.13%	78.48%

Table 11.2: Recall@40 scores for proposed re-ranking methods in shape retrieval tasks.

Descriptor	Score (Recall@40)	DOA RL-Cor	DOA Cor-Cor	Pairwise Recommend.	Context Spaces kNN	Context Spaces MkNN	RL-Sim kNN Intersection	RL-Sim M-kNN Intersection	RL-Sim kNN Kendall's Tau	RL-Sim M-kNN Kendall's Tau	Context Re-Ranking
SS [17]	43.99%	50.93%	53.23%	54.36%	48.50%	53.27%	53.15%	<b>57.58%</b>	52.67%	56.06%	51.38%
BAS [2]	75.20%	85.11%	84.15%	84.03%	80.53%	81.54%	82.94%	<b>85.87%</b>	81.16%	83.44%	82.43%
IDSC [52]	85.40%	90.02%	90.39%	92.21%	90.44%	90.91%	92.18%	<b>92.62%</b>	91.12%	92.06%	91.84%
CFD [68]	84.43%	93.62%	92.90%	<b>96.15%</b>	93.02%	93.00%	94.13%	95.33%	93.12%	94.27%	95.71%
ASC [53]	88.39%	90.66%	93.61%	94.66%	93.28%	95.03%	94.69%	<b>95.75%</b>	93.68%	94.56%	93.07%
AIR [35]	93.67%	97.68%	98.81%	99.36%	99.75%	99.92%	99.90%	99.87%	<b>99.94%</b>	99.93%	99.80%

## 11.2 Comparison of the proposed Rank Aggregation Methods

This section summarizes the results of the rank aggregation methods, presented along this thesis. Table 11.3 presents the MAP scores for the combination of two image descriptors for each visual property (shape, color, and texture), considering the various rank aggregation methods. The results of each image descriptor, in isolation is also presented. We can observe very significant gains for all rank aggregation methods in comparison with image descriptors in isolation. In general, the rank aggregation methods yield very close results. The best effectiveness result for each visual property was achieved by the *Contextual Spaces* rank aggregation method.

Table 11.3: MAP scores for proposed Rank Aggregation methods in general CBIR tasks.

Descriptor	Type	Dataset	Rank Aggregation Method	Variation	Score (MAP)
CFD [68]	Shape	MPEG-7	-	-	80.71%
ASC [53]	Shape	MPEG-7	-	-	85.28%
CFD [68] + ASC [53]	Shape	MPEG-7	Pairwise Recommend.	-	99.34%
CFD [68] + ASC [53]	Shape	MPEG-7	Contextual Spaces	kNN	98.67%
CFD [68] + ASC [53]	Shape	MPEG-7	Contextual Spaces	M-kNN	<b>99.36%</b>
CFD [68] + ASC [53]	Shape	MPEG-7	RL-Sim	kNN - Intersection	98.75%
CFD [68] + ASC [53]	Shape	MPEG-7	RL-Sim	M-kNN - Intersection	98.96%
CFD [68] + ASC [53]	Shape	MPEG-7	RL-Sim	kNN - Kendall's Tau	98.57%
CFD [68] + ASC [53]	Shape	MPEG-7	RL-Sim	M-kNN - Kendall's Tau	98.57%
CFD [68] + ASC [53]	Shape	MPEG-7	Contextual Re-Ranking	-	98.77%
ACC [41]	Color	Soccer	-	-	37.23%
BIC [90]	Color	Soccer	-	-	39.26%
BIC [90] + ACC [41]	Color	Soccer	Pairwise Recommend.	-	42.20%
BIC [90] + ACC [41]	Color	Soccer	Contextual Spaces	kNN	42.44%
BIC [90] + ACC [41]	Color	Soccer	Contextual Spaces	M-kNN	<b>46.10%</b>
BIC [90] + ACC [41]	Color	Soccer	RL-Sim	kNN - Intersection	44.49%
BIC [90] + ACC [41]	Color	Soccer	RL-Sim	M-kNN - Intersection	44.16%
BIC [90] + ACC [41]	Color	Soccer	RL-Sim	kNN - Kendall's Tau	44.45%
BIC [90] + ACC [41]	Color	Soccer	RL-Sim	M-kNN - Kendall's Tau	45.16%
BIC [90] + ACC [41]	Color	Soccer	Contextual Re-Ranking	-	42.14%
CCOM [46]	Texture	Brodatz	-	-	57.57%
LAS [94]	Texture	Brodatz	-	-	75.15%
LAS [94] + CCOM [46]	Texture	Brodatz	Pairwise Recommend.	-	79.91%
LAS [94] + CCOM [46]	Texture	Brodatz	Contextual Spaces	kNN	81.94%
LAS [94] + CCOM [46]	Texture	Brodatz	Context Spaces	M-kNN	<b>84.50%</b>
LAS [94] + CCOM [46]	Texture	Brodatz	RL-Sim	kNN - Intersection	80.26%
LAS [94] + CCOM [46]	Texture	Brodatz	RL-Sim	M-kNN - Intersection	83.39%
LAS [94] + CCOM [46]	Texture	Brodatz	RL-Sim	kNN - Kendall's Tau	80.51%
LAS [94] + CCOM [46]	Texture	Brodatz	RL-Sim	M-kNN - Kendall's Tau	81.68%
LAS [94] + CCOM [46]	Texture	Brodatz	Contextual Re-Ranking	-	81.63%

## 11.3 Comparison with other Re-Ranking Approaches

We also evaluated our proposed methods in comparison with other state-of-the-art re-ranking and post-processing methods. We consider several results reported in the literature, considering two well-known datasets, MPEG-7 and Kimia99, and various shape descriptors. Regarding to our re-ranking methods, we selected some combinations of methods and shape descriptors presented in this thesis. Table 11.4 presents these results for the MPEG-7 dataset. The third column presents the scores for the combination of shape descriptor and re-ranking method. The fourth column presents the gains in relation to the use of each descriptor in isolation. Note that results of our methods (in bold) present comparable effectiveness performance when compared with several other post-processing methods recently proposed in the literature.

Table 11.4: Re-Ranking methods comparison on the MPEG-7 dataset (*Recall@40*).

Algorithm	Shape Descriptor	Score	Gain
<b>Shape Descriptors</b>			
Data Driven Generative Models (DDGM) [99]	-	80.03%	-
Contour Features Descripor (CFD) [68]	-	84.43%	-
Inner Distance Shape Context (IDSC) [52]	-	85.40%	-
Shape Context (SC) [5]	-	86.80%	-
Aspect Shape Context (ASC) [53]	-	88.39%	-
Articulation-Invariant Representation (AIR) [35]	-	93.67%	-
<b>Re-Ranking and Post-Processing Methods</b>			
Graph Transduction (LP) [113]	IDSC [52]	91.00%	+6.56%
<b>DOA [RL+DU]</b>	<b>CFD [68]</b>	<b>92.56%</b>	<b>+9.63%</b>
<b>DOA [Cor+Cor]</b>	<b>CFD [68]</b>	<b>92.90%</b>	<b>+10.03%</b>
<b>Contextual Spaces [MKNN]</b>	<b>CFD [68]</b>	<b>93.00%</b>	<b>+10.15%</b>
<b>Contextual Spaces [KNN]</b>	<b>CFD [68]</b>	<b>93.02%</b>	<b>+10.17%</b>
<b>Contextual Re-Ranking</b>	<b>ASC [53]</b>	<b>93.07%</b>	<b>+5.29%</b>
<b>Contextual Spaces [KNN]</b>	<b>ASC [53]</b>	<b>93.28%</b>	<b>+5.53%</b>
Locally Constrained Diffusion Process [114]	IDSC [52]	93.32%	+9.27%
Shortest Path Propagation [102]	IDSC [52]	93.35%	+9.31%
Mutual kNN Graph [45]	IDSC [52]	93.40%	+9.37%
<b>DOA [RL+Cor]</b>	<b>CFD [68]</b>	<b>93.62%</b>	<b>+10.88%</b>
Locally Constrained Diffusion Process [114]	IDSC [52]+St. I [95]	93.80%	+9.84%
<b>RL-Sim Re-Ranking [kNN+Intersection]</b>	<b>ASC [53]</b>	<b>94.69%</b>	<b>+7.13%</b>
Locally Constrained Diffusion Process [114]	IDSC [52]+St. II [95]	94.85%	+11.07%
<b>Contextual Spaces [M-kNN]</b>	<b>ASC [53]</b>	<b>95.03%</b>	<b>+7.51%</b>
<b>RL-Sim Re-Ranking [M-kNN+Intersection]</b>	<b>CFD [68]</b>	<b>95.33%</b>	<b>+12.91%</b>
Locally Constrained Diffusion Process [114]	IDSC [52]+St. I&II [95]	95.60%	+11.94%
<b>Contextual Re-Ranking</b>	<b>CFD [68]</b>	<b>95.71%</b>	<b>+13.36%</b>
<b>RL-Sim Re-Ranking [M-kNN+Intersection]</b>	<b>ASC [53]</b>	<b>95.75%</b>	<b>+8.33%</b>
Locally Constrained Diffusion Process [114]	ASC [53]	95.96%	+8.56%
<b>SetRA (CRR + RL-Sim[kNN+Intersection])</b>	<b>CFD [68]</b>	<b>95.98%</b>	<b>+13.68%</b>
<b>Pairwise Recommendation</b>	<b>CFD [68]</b>	<b>96.15%</b>	<b>+13.88%</b>
<b>Contextual Re-Ranking (CRR)</b>	<b>AIR [35]</b>	<b>99.80%</b>	<b>+6.54%</b>
<b>RL-Sim Re-Ranking [M-kNN+Kendall's tau]</b>	<b>AIR [35]</b>	<b>99.93%</b>	<b>+6.68%</b>
<b>RL-Sim Re-Ranking [kNN+Kendall's tau]</b>	<b>AIR [35]</b>	<b>99.94%</b>	<b>+6.69%</b>
Tensor Product Graph [115]	AIR [35]	99.99%	+6.75%
<b>SetRA (CRR+RL-Sim[kNN+Intersection])</b>	<b>AIR [35]</b>	<b>99.99%</b>	<b>+6.75%</b>

Table 11.5: Re-Ranking and post-processing methods comparison on the Kimia99 dataset.

Algorithm	Descriptor	1°	2°	3°	4°	5°	6°	7°	8°	9°	10°
CFD [68]	-	99	98	98	99	97	90	86	86	68	56
IDSC [52]	-	99	99	99	98	98	97	97	98	94	79
<b>DOA [RL+DU]</b>	<b>CFD [68]</b>	<b>98</b>	<b>99</b>	<b>99</b>	<b>99</b>	<b>98</b>	<b>99</b>	<b>99</b>	<b>97</b>	<b>98</b>	<b>99</b>
Graph Transduction [113]	IDSC [52]	99	99	99	99	99	99	99	99	97	99
Mutual kNN Graph [45]	IDSC [52]	99	99	99	99	99	99	99	99	99	99
<b>Pairwise Recommendation</b>	<b>CFD [68]</b>	<b>99</b>									

Table 11.5 presents the comparison on the Kimia99 dataset, with the results of our proposed methods in bold. Recall that scores are calculated as the sum of correctly retrieved shapes from all classes within the first 10 objects. Therefore the best resulting score for each of them is 99. Note that the maximum retrieval score was reached by the *Pairwise Recommendation* method.

## 11.4 Comparison with other Rank Aggregation Approaches

Our rank aggregation methods are also compared with other approaches on the MPEG-7 dataset. Results are presented in Table 11.6. Three baselines are considered: the traditional Borda [116] method, usually used as baseline in several works; the recently proposed Reciprocal Rank Fusion [14] method; and the Co-Transduction [4] method, recently proposed for CBIR applications. In the case of Borda and Reciprocal Rank methods, the same descriptors, which are used by some of our methods, were combined. We can observe that several proposed rank aggregation methods outperform the baselines.

## 11.5 Other aspects

In addition to effectiveness, we also compared the proposed re-ranking methods with regard to other aspects as asymptotic complexity, efficiency, and number of parameters. For asymptotic complexity, we only consider the step of updating distances, since the sorting of ranked lists is the same for all methods. For efficiency, we consider the execution time of an execution for the MPEG-7 dataset using a Java implementation of for each method. The experiments were performed on a machine with Intel Xeon 2.40GHz - 16 cores and Linux Ubuntu 10.04.

Table 11.7 presents the results. Except for DOA, all proposed re-ranking algorithms consider only three parameters, which were experimentally defined. The Pairwise Rec-

Table 11.6: Rank Aggregation methods comparison on the MPEG-7 dataset ( $Recall@40$ ).

Algorithm	Shape Descriptor	Score
<b>Shape Descriptors</b>		
Data Driven Generative Models (DDGM) [99]	-	80.03%
Contour Features Descripor (CFD) [68]	-	84.43%
Inner Distance Shape Context (IDSC) [52]	-	85.40%
Shape Context (SC) [5]	-	86.80%
Aspect Shape Context (ASC) [53]	-	88.39%
Articulation-Invariant Representation (AIR) [35]	-	93.67%
<b>Rank Aggregation Methods</b>		
Borda [116]	CFD [68]+IDSC [52]	91.06%
Borda [116]	CFD [68]+ASC [53]	93.51%
Reciprocal Rank Fusion [14]	CFD [68]+IDSC [52]	94.98%
Reciprocal Rank Fusion [14]	CFD [68]+ASC [53]	96.25%
<b>DOA [RL+Cor]</b>	<b>CFD+IDSC</b>	<b>96.46%</b>
<b>DOA [Cor+Cor]</b>	<b>CFD+IDSC</b>	<b>97.00%</b>
Co-Transduction [4]	IDSC [52]+DDGM [99]	97.31%
Co-Transduction [4]	SC [5]+DDGM [99]	97.45%
Co-Transduction [4]	SC [5]+IDSC [52]	97.72%
<b>Contextual Spaces [kNN]</b>	<b>CFD [68]+IDSC [52]</b>	<b>98.95%</b>
<b>Contextual Re-Ranking</b>	<b>CFD [68]+IDSC [52]</b>	<b>98.95%</b>
<b>Contextual Spaces [kNN]</b>	<b>CFD [68]+ASC [53]</b>	<b>99.03%</b>
<b>Contextual Spaces [M-kNN]</b>	<b>CFD [68]+IDSC [52]</b>	<b>99.15%</b>
<b>RL-Sim Re-Ranking [kNN+Intersection]</b>	<b>CFD [68]+IDSC [52]</b>	<b>99.31%</b>
<b>Contextual Re-Ranking</b>	<b>CFD [68]+ASC [53]</b>	<b>99.38%</b>
<b>RL-Sim Re-Ranking [kNN+Intersection]</b>	<b>CFD [68]+ASC [53]</b>	<b>99.44%</b>
<b>RL-Sim Re-Ranking [M-kNN+Intersection]</b>	<b>CFD [68]+IDSC [52]</b>	<b>99.49%</b>
<b>SetRA (CRA + RL-Sim[kNN+Intersection])</b>	<b>CFD [68]+ASC [53]</b>	<b>99.50%</b>
<b>Pairwise Recommendation</b>	<b>CFD [68]+IDSC [52]</b>	<b>99.52%</b>
<b>Contextual Spaces [M-kNN]</b>	<b>CFD [68]+ASC [53]</b>	<b>99.56%</b>
<b>RL-Sim Re-Ranking [M-kNN+Intersection]</b>	<b>CFD [68]+ASC [53]</b>	<b>99.65%</b>

ommendation and RL-Sim algorithms present a low asymptotic complexity ( $O(N)$ ). Regarding to execution time, the most efficient methods are the Contextual Spaces and the Pairwise Recommendation. However, we should note that the execution time results can only provide a rough comparison, since the methods were not optimally implemented. We can observe, for example that execution times presented in Chapter 9 (considering optimized implementation of Contextual Re-Ranking method) is much smaller than those in presented in Table 11.7.

## 11.6 Discussion

The objective of this section is to discuss, in a general way, the advantages and disadvantages of proposed methods considering the presented experimental results. As we mentioned before, given properties and obtained results, there is no clear definition of the *best re-ranking* method for all situations. However, we can identify scenarios in which a given method can be considered the most appropriate one.

For example, the RL-Sim re-ranking algorithm yields very high effectiveness perfor-

Table 11.7: Comparison of re-ranking methods regarding various criteria.

Method	Complexity	Execution Time	Number of Parameters
DOA [Cor-Cor]	$O(N^2)$	778 s	7
Pairwise Recommendation	$O(N)$	75 s	3
Contextual Spaces [kNN]	$O(N^2)$	40 s	3
RL-Sim [kNN+Intersection]	$O(N)$	167 s	3
Contextual Re-Ranking	$O(N^2)$	138 s	3

mance for several descriptors, in special considering the Recall@40 measure on the MPEG-7 dataset. Thus, it may be a good option for situations in which effectiveness is the most important aspect. However, it presents a large execution time.

On the other hand, the Pairwise Recommendation algorithm is very simple, presents a low execution time, and a low asymptotic complexity. Its good efficiency performance does not affect its effectiveness. The method also yields significant gains for various descriptors. Therefore, the method may be indicated for large datasets and situations in which efficiency is indispensable.

Considering rank aggregation tasks, the Contextual Spaces algorithm presents the best MAP scores considering all datasets (shape, color, and texture) and very high Recall@40 scores on the MPEG-7 dataset.

# Chapter 12

## Conclusions

### 12.1 Contributions

This thesis developed research aiming at exploiting *contextual information* for improving the effectiveness of CBIR systems. Several research challenges were addressed and various contributions were proposed.

The main contributions of this thesis is the creation, modeling, and implementation of five re-ranking and rank aggregation methods that exploit contextual information:

- **Distance Optimization Algorithm:** a cluster-based approach for image re-ranking. Two cluster-similar functions and two distances updating approaches are proposed, based on the similarity of ranked lists and the correlation of distances. A rank aggregation approach based on the cluster information were also proposed.
- **Pairwise Recommendation:** a novel re-ranking approach based on the concept of recommendation for modeling and handling relationships among images. Our strategy opens a new area of investigation, related to the use of recommendation techniques in re-ranking tasks.
- **Contextual Spaces:** we have presented the concept of *contextual spaces* and two variations of re-ranking approaches based on this concept. The main idea of the methods consists in exploiting information encoded in ranked lists and distances among images for constructing contextual spaces and, based on them, re-computing the distances among images.
- **RL-Sim Re-Ranking:** a new re-ranking method that considers the similarity between ranked lists for encoding contextual information in CBIR systems. We believe that the modeling of contextual information considering only the similarity

between ranked lists represents an advantage of our strategy, since it can use different similarity/distance measures among ranked lists, a well-established research area [27, 103, 106].

- **Contextual Re-Ranking:** a new re-ranking method based on the use of image processing techniques for *contextual information* representation and processing. The use of image processing techniques is an important novelty of our work. We believe that our strategy opens a new area of investigation related to the use of image processing approaches for analyzing distances computed by CBIR descriptor, in tasks such as image re-ranking, rank aggregation, and clustering.

We conducted a large set of experiments and experimental results demonstrated the effectiveness of our methods in several image retrieval tasks based on shape, color and texture descriptors. The proposed methods achieves very high effectiveness performance when compared with state-of-the-art post-processing and rank aggregation methods on the well-known MPEG-7 dataset. Similar results were observed for color and texture-oriented datasets.

In addition to the development of re-ranking and rank aggregation methods other contributions were also presented:

- **Combination:** we have presented three novel combination approaches for re-ranking and rank aggregation methods. The main idea of our work consists in exploiting complementary rankings obtained by different methods in order to obtain more effective results.
- **Parallel Computation of Image Re-Ranking:** we accelerate the computation of the *Contextual Re-Ranking* algorithm, designing a parallel implementation using the OpenCL standard and GPUs. Various important issues were addressed during the design of the parallel algorithm, as the need for global synchronization in OpenCL and the concurrent access on data structures. Experimental results demonstrate that very significant speedups can be achieved by the OpenCL parallel implementation.
- **Multimodal Retrieval:** we evaluated our methods considering multimodal retrieval tasks, considering textual and visual descriptors. Relevant gains were obtained in comparison with traditional rank aggregation approaches.

## 12.2 Future Work

Considering the diversity of the research work presented in this thesis, several extensions are possible:

- **Relevance Feedback:** the use of re-ranking and rank aggregation approaches jointly with relevance feedback techniques, aiming at further improving the effectiveness of retrieval tasks.
- **Collaborative Image Retrieval:** the use of re-ranking approaches (in special the *Pairwise Recommendation* method) for performing collaborative image retrieval, in which the training data is obtained by exploring historical relevance feedback log data, collected in multiple image retrieval sessions.
- **Distance Optimization Algorithm:** investigation of new cluster-similar functions and distance updating approaches.
- **Contextual Spaces:** the use of geometrical properties for extracting more information from contextual spaces; the use of contextual spaces for other tasks, such as clustering.
- **RL-Sim Re-Ranking:** investigation of other different measures between top  $k$  lists and the combination of results obtained from different measures.
- **Contextual Re-Ranking:** the use of other image processing techniques, as dynamic thresholding and other filtering approaches; the analysis of other regions of context images; the use of context images for other applications (for clustering and computing the similarity between ranked lists, for example).
- **Combination:** the use of all proposed re-ranking and rank aggregation methods in combination approaches. Investigation of new techniques using iterative approaches for combining re-ranking and rank aggregation methods.
- **Parallel Computation of Image Re-Ranking:** parallelization of all re-ranking methods, investigation of tuning approaches, simultaneous execution of kernels in CPU and GPU devices, and the use of proposed parallel algorithm in web scale image datasets.
- **Multimodal Retrieval:** evaluation of all proposed re-ranking methods on multimedia retrieval tasks, considering visual and textual descriptors.
- **Large Datasets:** evaluation of the proposed re-ranking and rank aggregation methods considering web scale datasets.

Figure 12.1 illustrates the main concepts covered in this thesis, the contributions and associated publications. Finally, it highlights in gray possible extensions mentioned along this chapter.

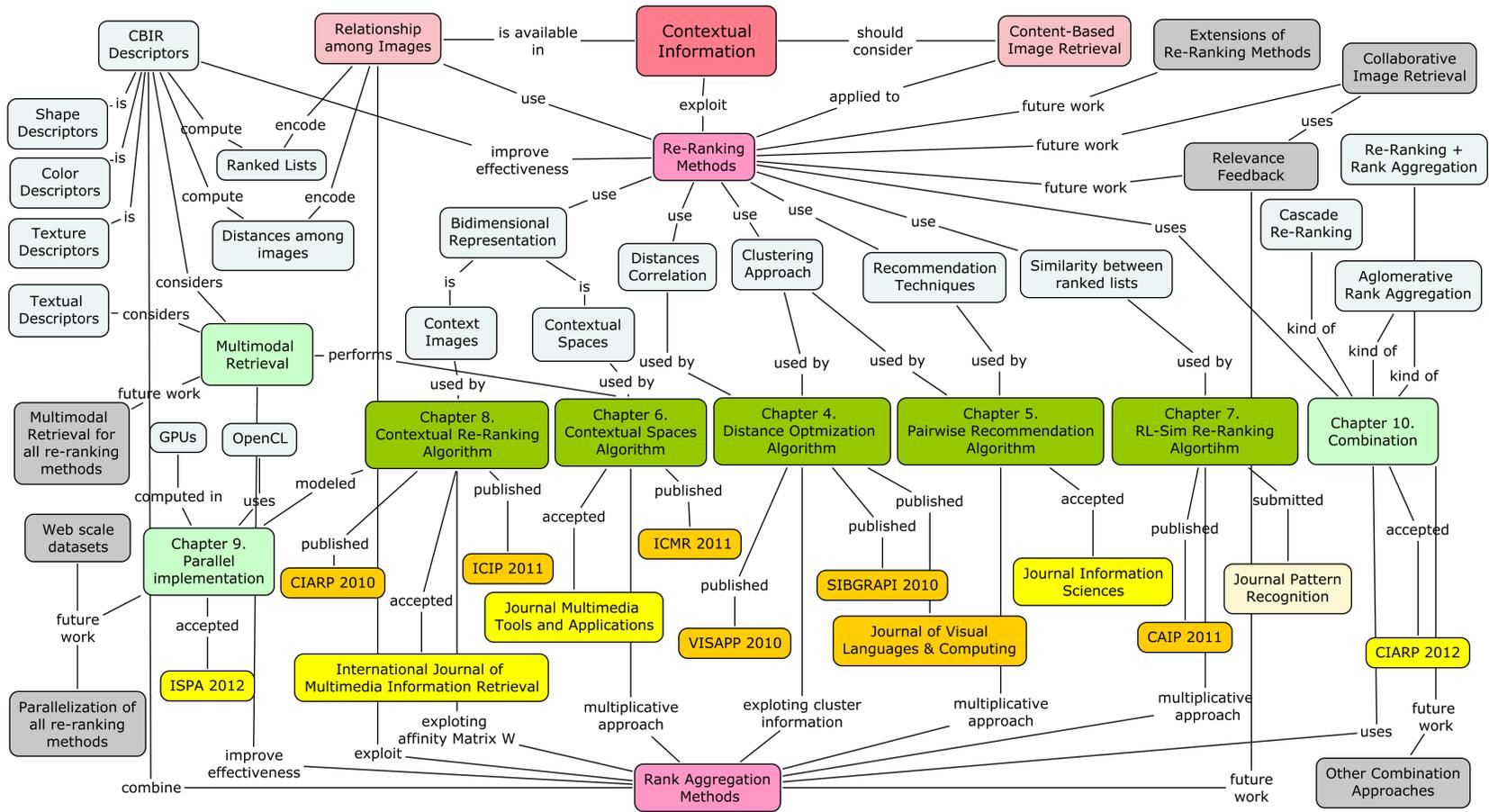


Figure 12.1: Contributions, main concepts, and possible extension of this thesis.

# Bibliography

- [1] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307, 1999.
- [2] Nafiz Arica and Fatos T. Yarman Vural. BAS: a perceptual shape descriptor based on the beam angle statistics. *Pattern Recognition Letters*, 24(9-10):1627–1639, 2003.
- [3] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.
- [4] Xiang Bai, Bo Wang, Xinggang Wang, Wenyu Liu, and Zhuowen Tu. Co-transduction for shape retrieval. In *European Conference on Computer Vision (ECCV'2010)*, volume 3, pages 328–341, 2010.
- [5] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.
- [6] Yolanda Blanco-Fernández, Martín López-Nores, Alberto Gil-Solla, Manuel Ramos-Cabrer, and José J. Pazos-Arias. Exploring synergies between content-based filtering and Spreading Activation techniques in knowledge-based recommender systems. *Information Sciences*, 181(21):4823–4846, 2011.
- [7] Phil Brodatz. *Textures: A Photographic Album for Artists and Designers*. Dover, 1966.
- [8] Rodrigo Tripodi Calumby. Recuperação multimodal de imagens com realimentação de relevância baseada em programação genética. Master’s thesis, Unicamp, Campinas - Brazil, February 2010.
- [9] Maya Carrillo, Esaú Villatoro-Tello, A. López-López, Chris Eliasmith, Manuel Montes-Y-Gómez, and Luis Villaseñor Pineda. Representing context information

- for document retrieval. In *8th International Conference on Flexible Query Answering Systems (FQAS'09)*, pages 239–250, 2009.
- [10] Guang-Ho Cha. Context-sensitive ranking for effective image retrieval. In *Advances in Multimedia Modeling*, volume 4351 of *Lecture Notes in Computer Science*, pages 344–353, 2006.
- [11] Jun Chen, Yu Zhou, Zhijun Yao, Linbo Luo, Bo Wang, and Wenyu Liu. Unsupervised clustering using graph transduction. In *International Conference on Biomedical Engineering and Computer Science (ICBECS'2010)*, pages 1–4, april 2010.
- [12] Stephane Clinchant, Julien Ah-Pine, and Gabriela Csurka. Semantic combination of textual and visual information in multimedia retrieval. In *ACM International Conference on Multimedia Retrieval (ICMR'11)*, pages 44:1–44:8, 2011.
- [13] Don Coppersmith, Lisa K. Fleischer, and Atri Rurda. Ordering by weighted number of wins gives a good ranking for weighted tournaments. *ACM Transactions Algorithms*, 6(3):55:1–55:13, July 2010.
- [14] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 758–759, 2009.
- [15] W. Bruce Croft. Combining approaches to information retrieval. In W. Bruce Croft and W. Bruce Croft, editors, *Advances in Information Retrieval*, volume 7 of *The Information Retrieval*, pages 1–36. Springer US, 2002.
- [16] Ricardo da S. Torres and Alexandre X. Falcão. Content-Based Image Retrieval: Theory and Applications. *Revista de Informática Teórica e Aplicada*, 13(2):161–185, 2006.
- [17] Ricardo da S. Torres and Alexandre X. Falcão. Contour Saliency Descriptors for Effective Image Retrieval and Analysis. *Image and Vision Computing*, 25(1):3–13, 2007.
- [18] Ricardo da S. Torres, Alexandre X. Falcão, Marcos A. Gonçalves, João P. Papa, Baoping Zhang, Weiguo Fan, and Edward A. Fox. A genetic programming framework for content-based image retrieval. *Pattern Recognition*, 42(2):283–292, 2009.
- [19] Ricardo da S. Torres, Claudia B. Medeiros, Marcos A. Gonçalves, and Edward Fox. A digital library framework for biodiversity information systems. *International Journal on Digital Libraries*, 6(1):3–17, 2006.

- [20] André Tavares da Silva, Jefersson Alex dos Santos, Alexandre X. Falcão, Ricardo da S. Torres, and Léo Pini Magalhães. Incorporating multiple distance spaces in optimum-path forest classification to improve feedback-based learning. *Computer Vision and Image Understanding*, 116(4):510–523, 2012.
- [21] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):5:1–5:60, 2008.
- [22] Thomas Deselaers, Daniel Keysers, and Hermann Ney. Features for image retrieval: an experimental comparison. *Information Retrieval*, 11(2):77–107, 2008.
- [23] Fernando Diaz. Regularizing ad hoc retrieval scores. In *ACM Conference on Information and Knowledge Management (CIKM'2005)*, pages 672–679, 2005.
- [24] Jefersson Alex dos Santos, Cristiano D. Ferreira, Ricardo da S. Torres, Marcos André Gonçalves, and Rubens A.C. Lamparelli. A relevance feedback method based on genetic programming for classification of remote sensing images. *Information Sciences*, 181(13):2671 – 2684, 2011.
- [25] Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the web. In *10th International Conference on World Wide Web (WWW'01)*, pages 613–622, 2001.
- [26] Ronald Fagin, Ravi Kumar, Mohammad Mahdian, D. Sivakumar, and Erik Vee. Comparing and aggregating rankings with ties. In *23th ACM SIGMOD Symposium on Principles of Database Systems (PODS'04)*, pages 47–58, 2004.
- [27] Ronald Fagin, Ravi Kumar, and D. Sivakumar. Comparing top k lists. In *ACM-SIAM Symposium on Discrete algorithms (SODA'03)*, pages 28–36, 2003.
- [28] Fabio F. Faria, Adriano Veloso, Humberto M. Almeida, Eduardo Valle, Ricardo da S. Torres, Marcos A. Goncalves, and Wagner Meira Jr. Learning to rank for content-based image retrieval. In *Multimedia Information Retrieval (MIR'2010)*, pages 285–294, 2010.
- [29] Pedro F. Felzenszwalb and Joshua D. Schwartz. Hierarchical matching of deformable shapes. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, pages 1–8, 2007.
- [30] Cristiano D. Ferreira, Jefersson Alex dos Santos, Ricardo da S. Torres, Marcos André Gonçalves, R. C. Rezende, and Weiguo Fan. Relevance feedback based on genetic programming for image retrieval. *Pattern Recognition Letters*, 32(1):27–37, 2011.

- [31] Edward A. Fox and Joseph A. Shaw. Combination of multiple searches. In *The Second Text REtrieval Conference (TREC-2)*, volume 500-215 of *NIST Special Publication*, pages 243–252. NIST, 1994.
- [32] Alexander Gammerman, Katy S. Azoury, and Vladimir Vapnik. Learning by transduction. In *Uncertainty in Artificial Intelligence (UAI'98)*, pages 148–155, 1998.
- [33] Zoubin Ghahramani. Unsupervised Learning. In *Advanced Lectures on Machine Learning*, pages 72–112. 2004.
- [34] Marcos André Gonçalves. *Streams, Structures, Spaces, Scenarios, and Societies (5S): A Formal Digital Library Framework and Its Applications*. PhD thesis, Virginia Polytechnic Institute and State University, November 2004.
- [35] Raghuraman Gopalan, Pavan Turaga, and Rama Chellappa. Articulation-invariant representation of non-planar shapes. In *11th European Conference on Computer Vision (ECCV'2010)*, volume 3, pages 286–299, 2010.
- [36] Abhishek A. Gupta, Dean P. Fostery, and Lyle H. Ungarz. Unsupervised distance metric learning using predictability. Technical report, University of Pennsylvania, 2008.
- [37] Tomer Hertz, Aharon Bar-Hillel, and Daphna Weinshall. Learning distance functions for image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2004)*, pages 570–577, 2004.
- [38] Steven C. Hoi, Wei Liu, and Shih-Fu Chang. Semi-supervised distance metric learning for collaborative image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2008)*, pages 1–7, 2008.
- [39] Steven C. Hoi, Wei Liu, and Shih-Fu Chang. Semi-supervised distance metric learning for collaborative image retrieval and clustering. *ACM Transactions on Multimedia Computing and Communication Applications*, 6(3):18:1–18:26, August 2010.
- [40] Chao-Bing Huang and Quan Liu. An orientation independent texture descriptor for image retrieval. In *International Conference on Communications, Circuits and Systems (ICCCAS 2007)*, pages 772–776, 2007.
- [41] Jing Huang, S. Ravi Kumar, Mandar Mitra, Wei-Jing Zhu, and Ramin Zabih. Image indexing using color correlograms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'97)*, pages 762–768, 1997.

- [42] Hervé Jegou, Cordelia Schmid, Hedi Harzallah, and Jakob Verbeek. Accurate image search using the contextual dissimilarity measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):2–11, 2010.
- [43] Shihao Ji, Ke Zhou, Ciya Liao, Zhaohui Zheng, Gui-Rong Xue, Olivier Chapelle, Gordon Sun, and Hongyuan Zha. Global ranking by exploiting user clicks. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 35–42, 2009.
- [44] Thorsten Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML'2003)*, pages 290–297, 2003.
- [45] Peter Kontschieder, Michael Donoser, and Horst Bischof. Beyond pairwise shape similarity analysis. In *Asian Conference on Computer Vision*, pages 655–666, 2009.
- [46] Vassili Kovalev and Stephan Volmer. Color co-occurrence descriptors for querying-by-example. In *International Conference on Multimedia Modeling*, page 32, 1998.
- [47] Stephane Lafon and Ann B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1393–1403, 2006.
- [48] Longin Jan Latecki, Rolf Lakmper, and Ulrich Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2000)*, pages 424–429, 2000.
- [49] Jung-Eun Lee, Rong Jin, and Anil K. Jain. Rank-based distance metric learning: An application to image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2008)*, pages 1–8, 2008.
- [50] James Lewis, Stephan Ossowski, Justin Hicks, Mounir Errami, and Harold R. Garner. Text similarity: an alternative way to search medline. *Bioinformatics*, 22(18):2298–2304, 2006.
- [51] Liang Lin, Xiaobai Liu, and Song-Chun Zhu. Layered graph matching with composite cluster sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(8):1426–1442, 2009.
- [52] Haibin Ling and David W. Jacobs. Shape classification using the inner-distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):286–299, 2007.

- [53] Haibin Ling, Xingwei Yang, and Longin Jan Latecki. Balancing deformability and discriminability for shape matching. In *European Conference on Computer Vision (ECCV'2010)*, volume 3, pages 411–424, 2010.
- [54] Dong Liu, Shuicheng Yan, Xian-Sheng Hua, and Hong-Jiang Zhang. Image retagging using collaborative tag propagation. *IEEE Transactions on Multimedia*, 13(4):702–712, 2011.
- [55] Yu-Ting Liu, Tie-Yan Liu, Tao Qin, Zhi-Ming Ma, and Hang Li. Supervised rank aggregation. In *International Conference on World Wide Web (WWW'2007)*, pages 481–490, 2007.
- [56] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.
- [57] Alistair Moffat and Justin Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems*, 27(1):2:1–2:27, December 2008.
- [58] Henning Mller, Nicolas Michoux, David Bandon, and Antoine Geissbuhler. A review of content-based image retrieval systems in medical applications - clinical benefits and future directions. *International Journal of Medical Informatics*, 73(1):1–23, 2004.
- [59] Klimis Ntalianis, Anastasios Doulamis, Nicolas Tsapatsoulis, and Nikolaos Doulamis. Human action annotation, modeling and analysis based on implicit user interaction. *Multimedia Tools and Applications*, 50(1):199–225, 2010.
- [60] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.
- [61] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.
- [62] Gunhan Park, Yunju Baek, and Heung-Kyu Lee. Re-ranking algorithm using post-retrieval clustering for content-based image retrieval. *Information Processing and Management*, 41(2):177–194, 2005.
- [63] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Exploiting contextual information for image re-ranking and rank aggregation. *International Journal of Multimedia Information Retrieval*. Accepted.

- [64] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Exploiting pairwise recommendation and clustering strategies for image re-ranking. *Information Sciences*. Accepted.
- [65] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Image re-ranking and rank aggregation based on similarity of ranked lists. *Pattern Recognition*. Submitted.
- [66] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Distances correlation for re-ranking in content-based image retrieval. In *Conference on Graphics, Patterns and Images (SIBGRAPI'2010)*, volume 1, pages 1 – 8, 2010.
- [67] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Exploiting contextual information for image re-ranking. In *Iberoamerican Congress on Pattern Recognition (CIARP'2010)*, pages 541–548, 2010.
- [68] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Shape retrieval using contour features and distance optimization. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP'2010)*, volume 1, pages 197 – 202, 2010.
- [69] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Exploiting clustering approaches for image re-ranking. *Journal of Visual Languages and Computing*, 22(6):453–466, 2011.
- [70] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Exploiting contextual information for rank aggregation. In *International Conference on Image Processing (ICIP'2011)*, pages 97–100, 2011.
- [71] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Exploiting contextual spaces for image re-ranking and rank aggregation. In *ACM International Conference on Multimedia Retrieval (ICMR'11)*, pages 13:1–13:8, 2011.
- [72] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Image re-ranking and rank aggregation based on similarity of ranked lists. In *Computer Analysis of Images and Patterns (CAIP'2011)*, volume 6854, pages 369–376, 2011.
- [73] Daniel Carlos Guimarães Pedronette and Ricardo da S. Torres. Combining re-ranking and rank aggregation methods. In *Iberoamerican Congress on Pattern Recognition (CIARP'2012)*, 2012. Accepted.
- [74] Daniel Carlos Guimarães Pedronette, Ricardo da S. Torres, Edson Borin, and Mauricio Breternitz. Efficient image re-ranking computation on GPUs. In *International Symposium on Parallel and Distributed Processing (ISPA'2012)*, 2012. Accepted.

- [75] Daniel Carlos Guimarães Pedronette, Ricardo da S. Torres, and Rodrigo Calumby Tripodi. Using contextual spaces for image re-ranking and rank aggregation. *Multimedia Tools and Applications*. Accepted.
- [76] Otávio A.B. Penatti, Eduardo Valle, and Ricardo da S. Torres. Comparative study of global color and texture descriptors for web image retrieval. *Journal of Visual Communication and Image Representation*, 23(2):359–380, 2012.
- [77] Florent Perronnin, Yan Liu, and Jean-Michel Renders. A family of contextual measures of similarity between distributions with application to image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2009)*, pages 2358–2365, 2009.
- [78] Enoch Peserico and Luca Pretto. What does it mean to converge in rank? In *1st International Conference on the Theory of Information, Studies in Theory of Information Retrieval (ICTIR 2007)*, pages 239–245, 2007.
- [79] Nguyen-Khang Pham, Annie Morin, and Patrick Gros. Accelerating image retrieval using factorial correspondence analysis on GPU. In *Computer Analysis of Images and Patterns (CAIP'2009)*, pages 565–572. 2009.
- [80] Tao Qin, Tie-Yan Liu, Xu-Dong Zhang, De-Sheng Wang, and Hang Li. Global ranking using continuous conditional random fields. In *Neural Information Processing Systems*, pages 1281–1288, 2008.
- [81] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *Text REtrieval Conference*, pages 109–126, 1994.
- [82] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
- [83] Frans Schalekamp and Anke Zuylen. Rank aggregation: Together were strong. In *11th Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 38–51, 1998.
- [84] Olivier Schwander and Frank Nielsen. Reranking with contextual dissimilarity measures from representational bregman k-means. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP'2010)*, volume 1, pages 118–122, 2010.

- [85] Shweta Srivastava Scott Rostrup and Kishore Singhal. Fast and memory-efficient minimum spanning tree on the gpu. In *In Proceedings of the Second International Workshop on GPUs and Scientific Applications (GPUScA), PACT 2011*, pages 3 – 13, 2011.
- [86] D. Sculley. Rank aggregation for similar items. In *SIAM International Conference on Data Mining (SDM 2007)*, pages 587–592, 2007.
- [87] Thomas B. Sebastian, Philip N. Klein, and Benjamin B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):550–571, May 2004.
- [88] Jesus Serrano-Guerrero, Enrique Herrera-Viedma, Jose A. Olivas, Andres Cerezo, and Francisco P. Romero. A google wave-based fuzzy recommender system to disseminate information in university digital libraries 2.0. *Information Sciences*, 181:1503–1516, 2011.
- [89] Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *22nd international conference on Machine learning (ICML'05)*, pages 824–831, 2005.
- [90] Renato O. Stehling, Mario A. Nascimento, and Alexandre X. Falcão. A compact and efficient image retrieval approach based on border/interior pixel classification. In *ACM Conference on Information and Knowledge Management (CIKM'2002)*, pages 102–109, 2002.
- [91] John .E. Stone, David Gohara, and Guochun Shi. OpenCL: A parallel programming standard for heterogeneous computing systems. *Computing in Science Engineering*, 12(3):66 –73, 2010.
- [92] Grant Strong and Minglun Gong. Browsing a large collection of community photos based on similarity on gpu. In *4th International Symposium on Advances in Visual Computing (ISVC'08)*, pages 390–399, 2008.
- [93] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal on Computer Vision*, 7(1):11–32, 1991.
- [94] Bo Tao and Bradley W. Dickinson. Texture recognition and image retrieval using gradient indexing. *Journal of Visual Communication and Image Representation*, 11(3):327–342, 2000.

- [95] Andrew Temlyakov, Brent C. Munsell, Jarrell W. Waggoner, and Song Wang. Two perceptually motivated strategies for shape classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2010)*, volume 1, pages 2289–2296, 2010.
- [96] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- [97] Nguyen Duc Thang, Tahir Rasheed, Young-Koo Lee, Sungyoung Lee, and Tae-Seong Kim. Content-based facial image retrieval using constrained independent component analysis. *Information Sciences*, 181(15):3162–3174, 2011.
- [98] Arun Chauhan Thilina Gunarathne, Bimalee Salpitikorala and Geoffrey Fox. Optimizing OpenCL kernels for iterative statistical algorithms on GPUs. In *Second International Workshop on GPUs and Scientific Applications (GPUScA), PACT 2011*, pages 33–44, 2011.
- [99] Zhuowen Tu and Alan L. Yuille. Shape matching and recognition - using generative models and informative features. In *European Conference on Computer Vision (ECCV'2004)*, pages 195–209, 2004.
- [100] Joost van de Weijer and Cordelia Schmid. Coloring local feature extraction. In *European Conference on Computer Vision (ECCV'2006)*, volume Part II, pages 334–348, 2006.
- [101] Cornelis Joost van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, London, 1979.
- [102] Jingyan Wang, Yongping Li, Xiang Bai, Ying Zhang, Chao Wang, and Ning Tang. Learning context-sensitive similarity by shortest path propagation. *Pattern Recognition*, 44(10-11):2367–2374, 2011.
- [103] William Webber, Alistair Moffat, and Justin Zobel. A similarity measure for indefinite rankings. *ACM Transactions on Information Systems*, 28(4):20:1–20:38, 2010.
- [104] Adam Williams and Peter Yoon. Content-based image retrieval using joint correlograms. *Multimedia Tools and Applications*, 34(2):239–248, August 2007.
- [105] P. Wu, B. S. Manjunanth, S. D. Newsam, and H. D. Shin. A texture descriptor for image retrieval and browsing. In *IEEE Workshop on Content-Based Access of Image and Video Libraries (CBAIVL'99)*, pages 3–7, 1999.

- [106] Shengli Wu and Fabio Crestani. Methods for ranking information retrieval systems without relevance judgments. In *ACM Symposium on Applied Computing (SAC'03)*, pages 811–816, 2003.
- [107] Tianji Wu, Bo Wang, Yi Shan, Feng Yan, Yu Wang, and Ningyi Xu. Efficient pagerank and spmv computation on AMD GPUs. In *39th International Conference on Parallel Processing (ICPP'2010)*, pages 81–89, 2010.
- [108] Fang Zhou Xiao, Eric McCreath, and Christfried Webers. Fast on-line statistical learning on a gpgpu. In *In Proceedings of the 9th Australasian Symposium on Parallel and Distributed Computing (AusPDC 2011)*, pages 3–13, 2011.
- [109] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart J. Russell. Distance metric learning with application to clustering with side-information. In *Neural Information Processing Systems (NIPS'2002)*, pages 505–512, 2002.
- [110] Hui Xiong, Shashi Shekhar, Pang-Ning Tan, and Vipin Kumar. Exploiting a support-based upper bound of pearson's correlation coefficient for efficiently identifying strongly correlated pairs. In *ACM SIGKDD international Conference on Knowledge Discovery and Data Mining (KDD'04)*, pages 334–343, 2004.
- [111] Lingpeng Yang, Donghong Ji, Guodong Zhou, Yu Nie, and Guozheng Xiao. Document re-ranking using cluster validation and label propagation. In *ACM Conference on Information and Knowledge Management (CIKM'2006)*, pages 690–697, 2006.
- [112] Liu Yang and Rong Jin. Distance Metric Learning: A Comprehensive Survey. Technical report, Department of Computer Science and Engineering, Michigan State University, 2006.
- [113] Xingwei Yang, Xiang Bai, Longin Jan Latecki, and Zhuowen Tu. Improving shape retrieval by learning graph transduction. In *European Conference on Computer Vision (ECCV'2008)*, volume 4, pages 788–801, 2008.
- [114] Xingwei Yang, Suzan Koknar-Tezel, and Longin Jan Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2009)*, pages 357–364, 2009.
- [115] Xingwei Yang and Longin Jan Latecki. Affinity learning on a tensor product graph with applications to shape and image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2011)*, pages 2369–2376, 2011.

- [116] H. Peyton Young. An axiomatization of borda's rule. *Journal of Economic Theory*, 9(1):43–52, 1974.
- [117] Jie Yu, Jaume Amores, Nicu Sebe, Petia Radeva, and Qi Tian. Distance learning for similarity estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:451–462, 2008.
- [118] Harry Zhang, Liangxiao Jiang, and Jiang Su. Augmenting naive bayes for ranking. In *22nd International Conference on Machine learning (ICML'05)*, pages 1020–1027, 2005.
- [119] Junping Zhang, Stan Z. Li, and Jue Wang. Manifold learning and applications in recognition. In *Intelligent Multimedia Processing with Soft Computing*, pages 281–300, 2004.
- [120] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Scholkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.
- [121] Wengang Zhou, Qi Tian, Yijuan Lu, Linjun Yang, and Houqiang Li. Latent visual context learning for web image applications. *Pattern Recognition*, 44(10-11):2263–2273, October 2011.
- [122] Xiaojin Zhu. Semi-supervised learning literature survey. Technical report, Computer Sciences, University of Wisconsin-Madison, 2005.
- [123] Xiaojin Zhu. *Semi-supervised learning with graphs*. PhD thesis, Pittsburgh, PA, USA, 2005. John Chair-Lafferty and Ronald Chair-Rosenfeld.
- [124] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *In International Conference on Machine Learning (ICML'2003)*, pages 912–919, 2003.
- [125] Xiaojin Zhu and Andrew B. Goldberg. *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, Washington, 2009.