Unsupervised and Semi-supervised Methods for Human Action Analysis

Simon Jones

September 22, 2014

A thesis submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Department of Electronic and Electrical Engineering
The University of Sheffield

Declaration

Parts of this thesis have been taken from certain published academic conference/journal papers. All of these papers were written primarily by me, Simon Jones, during and as a result of my Ph.D. research. These papers are listed below:

Simon Jones and Ling Shao, A Multigraph Representation for Improved Unsupervised / Semi-supervised Learning of Human Actions, CVPR, 2014 (Chapter 7)

Simon Jones and Ling Shao, Unsupervised Spectral Dual Assignment Clustering of Human Actions in Context, CVPR, 2014 (Chapter 5)

Simon Jones and Ling Shao. Linear Regression Motion Analysis for Unsupervised Temporal Segmentation of Human Actions, WACV, 2014 (Chapter 4)

Ling Shao, **Simon Jones** and Xuelong Li, Efficient Search and Localisation of Human Actions in Video Databases, *Transactions on Systems and Circuits for Video Technology*, vol. 24, no. 3, pp. 504–512, Mar. 2014 (Chapter 4)

Simon Jones and Ling Shao, Rapid Localisation and Retrieval of Human Actions with Relevance Feedback, *Proceedings of CAIP*, 2013 (Chapter 4)

Simon Jones and Ling Shao. Content-based retrieval of human actions from realistic video databases. *Information Sciences*, vol. 236, pp. 56–65, Jul. 2012 (Chapter 3)

Simon Jones and Ling Shao. Action retrieval with relevance feedback on youtube videos. In *Proceedings of International Conference on Internet Multimedia Computing and Service*, 2011. (Chapter 3)

Simon Jones, Ling Shao, Jianguo Zhang, and Yan Liu. Relevance feedback for real-world human action retrieval. *Pattern Recognition Letters*, vol. 33, no. 4, Mar. 2012. (Chapter 3)

Acknowledgements

I would like to extend my sincerest thanks to the following, who have all helped to a greater or smaller extent in the completion of this thesis.

Firstly, many thanks to my family, Michael, Susan and Kirsty Jones; Sarah, Brandon, Ngaire and Elias Lim; and Phyllis Piercy. They have given me much support to complete my Ph.D. studies.

I very much appreciate the support of my supervisor, Dr. Ling Shao, who has helped to give my Ph.D. direction from the very beginning, and has provided guidance every step of the way. His advice has also been instrumental in the publication of my research.

I would like to thank my fellow students, especially Dr. Xiantong Zhen and Di Wu, who have at various stages helped me in discussing and implementing ideas.

Prof. Dave Robertson and Prof. Bob Fisher from the University of Edinburgh both gave me excellent guidance back in 2009 when I first considered starting doing a Ph.D.. Prof. Robertson also provided me a reference to help me secure my place at the University of Sheffield. Many thanks for their help.

The administrative staff at the Electronic and Engineering Department of the University of Sheffield, including James Screaton, Dianne Webster and Hilary Levesley, have provided me with support in various matters. I appreciate their effort and kindness in these.

My friends have helped me relax and give me the energy to continue with my Ph.D. Among these are my Sheffielder friends from PROGRESS, including Chris Quickfall and others, and friends from various places, including James

Mathews, Jon Chan, Fraser Mitchell and Chris Zealand.

Thanks to the Engineering and Physical Sciences Research Council (EPSRC), who provided the financial support required for me to complete my Ph.D.

Many thanks to my examiners, Prof. Shaogang Gong and Dr. Luke Seed, who gave me invaluable suggestions for the improvement of this thesis, and who made my viva an enjoyable experience.

Finally, I would like to offer my thanks to my wife Ruomei Yan, without whom I would not have been able to stay sane for the past four years! She has given me much support and has helped me smile even at times when I didn't think I could. To her, I dedicate this thesis.

Abstract

While human action recognition is a very well studied topic, semi-supervised and unsupervised tasks such as human action retrieval and human action clustering have received relatively little attention. These topics are important to study, as they require far less or no annotated training data, making it more feasible to apply these methods to real-world data, where neatly annotated data are far too rare and costly to obtain. In this thesis, several projects have been undertaken, focused on performing semi-supervised and unsupervised tasks on human actions, with potential for application to more complex systems.

The first topic for study is human action retrieval. Various methods for action representation, ranking and relevance feedback are implemented, and compared to one another. The result is a highly accurate human action retrieval system, outperforming the state-of-the-art.

This initial investigation is extended with the exploration of human action localisation. Two approaches to this problem are considered. First, a novel, efficient algorithm is introduced for performing temporally unconstrained retrieval and localisation of multimedia human action videos. This algorithm runs several orders of magnitude better than the best contemporary work on several action datasets, while maintaining practical accuracy. Then, a novel algorithm for performing unsupervised temporal localisation of discrete human motions is designed, based on the first two principal components of optical flow. A full human action recognition system is designed around this algorithm to provide an experimental validation of this concept. Experiments show state-of-the-art performance on two popular human action datasets.

An entirely new class of clustering problems is introduced for the clustering of human actions while incorporating scene context. To solve this new problem, a novel clustering algorithm – Dual-assignment K-means Clustering – is proposed. Unlike previous work, which integrates scene context in a supervised fashion, the proposed algorithm is entirely unsupervised, estimating two clustering solutions and the mutual information between them simultaneously. In experiments, the proposed algorithm out-perform state-of-the-art clustering algorithms by using scene context to improve accuracy.

The dual-assignment clustering work is extended to consider the clustering of actions with object context, rather than scene context. As there may be multiple objects per single action, an algorithm is introduced to handle this complexity, based on a combination of existing probabilistic graphical models. This algorithm is termed Multiple Object Single Action Clustering, and is compared favourably to Dual-assignment k-means and other existing clustering algorithms on a cooking dataset.

The final work presents a novel method for representing human actions that can enhance performance on unsupervised and semi-supervised tasks. It is applied to clustering, retrieval and recognition tasks, tying together with works presented in previous chapters. The proposed representation considerably outperforms existing action representations on clustering and retrieval tasks.

It is concluded that these introduced methods provide a sufficient groundwork for more advanced practical systems based on such semi-supervised/unsupervised techniques – such as human activity retrieval.

Contents

Declaration **I**

Acknowledgements **III**

Abstract **V**

List of Tables **XI**

List of Figures **XII**

1 Introduction **1**

1.1 Motivation 2

1.1.1 Human Action Analysis 3

1.1.2 Unsupervised/Semi-Supervised Learning 3

1.1.3 Practical Human Activity Systems 5

1.2 Thesis Outline 6

2 Background and Related Works **8**

2.1 Action representation 9

2.1.1 Global Feature-based Methods 9

2.1.2 Local features 11

2.1.3 Other methods 17

2.2 Multi-view and contextual recognition 18

2.3 Action Localisation 20

2.4 Action Retrieval 22

2.4.1 Relevance Feedback 23
2.5 Action Clustering 25
2.6 Datasets 25
2.7 Discussion 28

3 Human Action Retrieval 33
3.1 Introduction 33
3.2 Action Representation 35
3.2.1 Local Feature Extraction 35
3.2.2 Comparison Model 36
3.3 Video Retrieval 39
3.3.1 Relevance Feedback 39
3.4 Experiments 43
3.4.1 Setup 43
3.4.2 Results 44
3.5 Discussion 51

4 Human Action Localisation and Temporal Segmentation 53
4.1 Introduction 53
4.2 Examplar-based Human Action Localisation 55
4.2.1 Video Representation 57
4.2.2 Localisation 58
4.2.3 Ranking 63
4.2.4 Relevance Feedback 64
4.2.5 Experiments 65
4.3 Unsupervised temporal segmentation of human actions 73
4.3.1 Person Tracking 75
4.3.2 Motion Segmentation 76
4.3.3 Representation/Recognition 80
4.3.4 Experiments 81
4.4 Discussion 84

5 Clustering Human Actions with Scene Context 86

5.1 Introduction 86

5.2 Dual Assignment Clustering 89

5.2.1 Definition 89

5.2.2 Relation to Previous Works 90

5.2.3 Optimisation Problem 92

5.3 Dual Assignment k-Means Algorithm (DAkM) 93

5.3.1 Spectral Representation 96

5.4 Experiments 97

5.4.1 Synthetic Data Clustering 97

5.4.2 Human Action Clustering 98

5.5 Discussion 104

6 Dual-Assignment Clustering of Single Actions with Multiple Objects 107

6.1 Introduction 107

6.2 Graphical models for Dual-Assignment Clustering 108

6.3 Graphical models for the Dual-Assignment Clustering of Multiple Objects 112

6.4 Nested Expectation-Maximisation 116

6.4.1 Nested LDA Variational EM 117

6.4.2 Nested GMM EM 118

6.4.3 Full MOSAC Inference 119

6.5 Representing a Multiple-Object Dataset 121

6.6 Cooking Activity Experiments 122

6.6.1 Dataset 123

6.6.2 Setup 123

6.6.3 Results 124

6.7 Discussion 124

7 Multi-Spectral Representation for Human Actions 127

7.1 Introduction 127

7.2 Multigraph Representation 129

7.2.1 Feature Grouping 129

7.2.2 Multigraph Spectral Embedding 131

7.3 Experiments 133

7.3.1 Setup 133

7.3.2 Clustering 133

7.3.3 Content-based Retrieval with Relevance Feedback 135

7.3.4 Recognition 136

7.4 Discussion 138

8 Conclusions **142**

8.1 Discussion 142

8.1.1 Action Retrieval 142

8.1.2 Action Localisation and Temporal Segmentation 143

8.1.3 Contextual Clustering 143

8.1.4 Multigraph Representations 144

8.2 Future Work 144

Glossary **146**

Bibliography **152**

List of Tables

2.1 Various human action datasets used in this thesis. Num Act. shows the total number of action instances in the dataset; Num Class. shows the number of action categories; Diff. indicates the approximated subjective difficulty of the dataset according to this researcher; Loc. indicates whether the dataset has the ground truth for localisation. 26

4.1 MSR2 query time costs 70
4.2 UT query time costs 70
4.3 KTH dataset: comparison of various methods 82
4.4 MSR2 dataset: comparison of various methods 83

5.1 Performance of DAkM when view difficulty is varied. 98
5.2 Clustering performance of various methods on realistic datasets. 101

6.1 Variable definitions for GMM 108
6.2 Variable definitions for Latent Dirichlet Allocation 112
6.3 Variable definitions for MOSAC 114
6.4 Action clustering results on the MPII dataset using various techniques 124

7.1 Clustering performance of various methods on each dataset. . . . 134
7.2 Retrieval performance of various methods on each dataset, before (B) and after (A) relevance feedback (RF). 136
7.3 Recognition performance of various methods on each dataset. . . 137

List of Figures

2.1 Examples of actions from the KTH dataset. 29

2.2 Examples of actions from the MSR2 dataset. 29

2.3 Examples of actions from the UCF YouTube dataset. 30

2.4 Examples of actions from the UCF Sports dataset. 30

2.5 Examples of actions from the UT-Interaction dataset. 31

2.6 Examples of actions from the HOHA2 dataset. Clockwise from the top left: sit up, kissing, answer phone, handshake. 31

2.7 Examples of shots from the MPII dataset. 32

3.1 A simplified diagram of a working information retrieval system. . 40

3.2 How varying levels of soft-assignment affect first-query action retrieval for 3 different methods of feature extraction on each dataset. See §3.2.2 for an explanation of σ. 45

3.3 Soft and hard assignment after relevance feedback, for 3 methods of feature extraction on each dataset. 46

3.4 Various combinations of feature hybrids for each dataset. 47

3.5 Methods of representation, using [1] for the ST pyramid 49

3.6 Methods of relevance feedback, where MoS stands for Maximum of Similarities as described in 3.3.1 50

4.1 An overview of the spatio-temporal localisation and ranking aspects of the algorithm. Note that the spatial localisation part of the algorithm has been simplified here for convenience. 56

4.2 An overview of how the Branch and Bound algorithm is applied for spatial localisation. 62
4.3 Comparison of the contribution of various relevance feedback methods. 69
4.4 MSR2: The modified histogram intersection against the original. 70
4.5 Precision/Recall after different levels of relevance feedback. . . . 71
4.6 Effect on the accuracy of various spatial localisation methods, as well as temporal localisation alone. 72
4.7 A system diagram capturing the main stages of the action recognition system. 74
4.8 L' applied to two KTH dataset videos, one for boxing and one for handwaving. The peaks of the graph occur in the middle of a linear motion. The valleys occur as the actors' arms change direction. 79
4.9 KTH dataset: action class confusion matrix 82
4.10 MSR2 dataset: action class confusion matrix 83

5.1 Above: various stills from videos, showing human actions happening in different scene contexts. Below: a simple model example of an action/scene relationship table. The information in this table can be directly inferred from training data in supervised learning, but in unsupervised learning it must be estimated simultaneously with the action/scene categories. 87
5.2 A visualisation of various clustering approaches, showing the dependence relationship between latent categorisations of the dataset, G, and the observable views on that dataset, V. (a) Ordinary Clustering. (b) Multiview Clustering for two views. (c) Alternative Clustering for two solutions. (d) Dual Assignment Clustering. 91
5.3 How improvement in DAkM accuracy (averaged over $\mathbf{x}$ and $\mathbf{y}$) changes with the number of relationships between $\mathbf{x}$ and $\mathbf{y}$. . . . 99
5.4 Performance of SDAkM with various λ. 102

5.5 SDAkM on the YouTube dataset with $\lambda = 1$ – a high peak performance after few iterations, but eventually the algorithm terminates with poor performance. 103
5.6 Examples of scene categories discovered in the UCF YouTube dataset, and their most strongly associated actions below. 106

6.1 A GMM visualised with plate notation. Evidence nodes are shaded. 109
6.2 A Dual-Assignment GMM visualised with plate notation. Evidence nodes are shaded. 111
6.3 Latent Dirichlet Allocation visualised with plate notation. Evidence nodes are shaded. 113
6.4 MOSAC visualised with plate notation. Evidence nodes are shaded. 115
6.5 How the action clustering performance of MOSAC on the MPII dataset varies with the number of scene topics 125

7.1 Certain BoF histogram bins may be associated more with one component of the video than any other. In this simplified 6-bin example, bins 1 and 2 are strongly associated with the upper body, bins 3 and 4 are strongly associated with the background, and bins 5 and 6 are strongly associated with the lower body. The high separability of the three components would make this histogram ideal for FGSM. 139
7.2 Precision/recall curves for video retrieval on various datasets. . . 140
7.3 The effect of varying parameter m on retrieval performance in the top 20 results, versus manifold ranking baseline. 141

Chapter 1

Introduction

Year on year, machine learning and computer vision research grows more sophisticated. Each new technology allows us to improve our approach to many different real-world problems. As available computing resources grow more abundant and cheaper, it becomes increasingly feasible to implement complex algorithms to analyse the content of images and videos, to facilitate tasks such as surveillance, automated driving, information storage/retrieval, or assisted living, for example.

One of the most popular topics in computer vision in the past decade has been that of human action analysis – more specifically, human action *recognition*. It is easy to understand this focus, as the human body is often involved in computer vision tasks, and is also present in a great deal of existing visual media. Recognising and understanding a human's action is fundamental to many higher level tasks, such as behaviour prediction, plan recognition, or activity recognition, each of which has many practical applications. Human actions also receive much attention because they are considered particularly difficult to model, even compared to other computer vision tasks (for various reasons that are stated in §1.1). Some of the most highly cited Computer Vision papers of the past decade, therefore, are focused on human action recognition.[2, 3]

However, because human action *recognition* already receives considerable attention, this thesis aims to cover several less common topics on human action analysis. Within the individual chapters, various tasks are performed, including retrieval, localisation, temporal segmentation, clustering, and representation. The unifying theme of these topics is the use of *unsupervised/semi-supervised* machine learning, which also serves to distinguish this work from many previous efforts in action analysis, the majority of which only consider fully-supervised learning. (However, there is still a significant body of existing work that does consider unsupervised/semi-supervised learning, and these works are cited throughout this thesis.) unsupervised/semi-supervised learning can be applied in many situations that supervised learning cannot, and can be used as a basis for higher-level tasks, such as activity recognition, making it highly practical. A more complete justification of this thesis' focus is given in §1.1.

This thesis contains roughly two categories of work, with one category focusing on semi-supervised retrieval and localisation of human actions with relevance feedback – the other category is focused on fully unsupervised learning, performing action clustering and segmentation. In the semi-supervised learning tasks there is typically limited labeled data available, but a much larger quantity of unlabeled data that can assist in the learning task. In unsupervised learning, the data is entirely unlabeled. While the topics included in this thesis are disparate, as listed above, the novel techniques introduced in this thesis are unified, in that they find an underlying structure inherent to human action, according to some representation method, relying minimally on training data to learn a task model. This is a necessary and under-explored direction for human action research, as is argued below.

1.1 Motivation

In this section, the motivation for this thesis is provided. There are three aspects to justify: first, the focus on human actions is addressed; then, the decision to concentrate on unsupervised/semi-supervised learning is explained; finally, some potential applications of the work are described.

1.1.1 Human Action Analysis

It is proposed that when performing analysis of existing real world video media, the most important aspect of the video is often a human's actions. There are two reasons that this is the case. First of all, the majority of existing video media tends to focus on humans as subjects. These video media include, for instance, TV, film, surveillance footage, and online videos. Indeed, video media that does not include any humans, such as nature documentaries, have a comparatively low frequency. The second reason is that practical video analysis tasks are often related to humans in some way – CCTV surveillance is mostly concerned with preventing/recording human crime; assisted living is used to help a human perform daily tasks; YouTube searches will often relate to human activities, such as in sports, dramas or comedies.

There is another reason that human actions are often studied: because of the challenge that they pose for computer vision/machine learning. For instance, there is enormous variety in the visual appearance of the human body, depending on the clothes worn, the size/gender of the actor, and the ambient lighting conditions. The human body has many degrees of freedom, and the same action semantically could be performed in a variety of ways visually, or appear very differently from multiple viewpoints. Indeed, even the question of how to define the semantics of an action can be hard and application-dependent. All of these problems make human actions an interesting and practical problem to approach.

1.1.2 Unsupervised/Semi-Supervised Learning

Machine learning tasks can be categorised according to whether labeled data is provided for training a model, and how much labeled data is provided. When fully-labeled data are provided for training, the task is *supervised* learning. If some of the training data are unlabeled, the task is *semi-supervised* learning. If no labels for the data are provided at all, and only the underlying structure of the data can be used for learning, this is *unsupervised* learning.

This thesis is only concerned with semi-supervised and unsupervised learning of human actions. Supervised learning (typically used to perform tasks such as video classification, localisation, or detection) is a well-explored topic, perhaps because it is easier than semi-supervised/unsupervised learning, and can give practical results that are easy to assess. Additionally, many researchers work on low-level feature extraction, and supervised learning provides a convenient way to show their features' representational power. However, all supervised learning methods share a weakness – they make the assumption that there will be sufficient training data available to fully model the underlying data structure.

In the real world, the assumption that we have enough data for fully supervised learning is problematic. Even the largest existing datasets for video analysis, such as HMDB51, contain only a few thousand annotated clips, and several tens of action classes, representing a infinitesimal proportion of existing video media – this is nowhere near the sufficient size to perform effective training on real-world video databases, such as YouTube, or on surveillance archives. Instead, research datasets typically limit the number of action classes, and the intraclass variability, so that reasonably performing algorithms can be trained on relatively few samples. This is because annotating each video as training data requires manual human effort, which is prohibitively expensive for large amounts of data. Some supervised learning research focuses on reducing the amount of training data required for good performance – for instance, using dimensionality reduction, mid-level features or hierarchical modeling patterns. Even these methods, however, perform better as more training data is introduced.

Another issue with supervised human action learning is that the training labels are often too simplistic to be useful. That is to say, action datasets are often annotated with simple action classes as they are understood by humans, such as "kicking a ball", or "cycling", which are highly semantic representations of the action. For certain high-level tasks, such as activity/behaviour recognition, these simplistic semantic abstractions are not particularly useful. This is because they don't capture the attributes of the action, such as its speed, the intention of the actor, or the different stages of the action, which can be very informative in activity recognition. It is possible to annotate a dataset with such attributes, but then the cost of annotation becomes even more expensive.

In this thesis, *semi-supervised* and *unsupervised* methods for human action analysis are considered. Rather than relying on labeled training data to find the underlying decision boundaries between classes, *unsupervised* learning attempts to model and utilise the underlying structure of unlabeled data to perform a task, such as clustering or dimensionality reduction.

Semi-supervised learning combines relatively little labeled training data with significantly larger quantities of unlabeled data to perform tasks like classification or ranking – the intention here is that the combined labeled/unlabeled data can provide information about the domain's underlying structure, while the labeled data can be used to perform more accurate classification/regression on this underlying structure.

Several works are considered in this thesis, which are detailed further in §1.2 below. These works share a common theme, in that they are all semi-supervised or unsupervised tasks, applied to human actions, and therefore utilise the underlying data structure of the actions in order to achieve high performance. For semi-supervised methods, action retrieval with relevance feedback is studied. Then, spatio-temporal localisation is applied in the context of action retrieval. For unsupervised methods, a method for unsupervised temporal segmentation of human actions is proposed, and then a method for using scene context to enhance action clustering is given. A multigraph representation for human actions is also devised, which can enhance the accuracy of both action clustering and action retrieval.

1.1.3 Practical Human Activity Systems

Now we consider the practical applications of unsupervised/semi-supervised methods for human action analysis.

Unsupervised/semi-supervised techniques could be used for the automatic annotation of the vast quantities of video data found in multimedia databases, such as YouTube, and in surveillance systems. In current multimedia databases, annotations of a video's content are manually provided by the uploading user, which may therefore be incomplete or inaccurate. The recorded videos from surveillance systems are almost never annotated. By using effective ranking and clustering methods, along with minimal user interaction (through an active learning system, for instance), it will be possible to automatically annotate the data in such video databases. These annotations can then be used for such applications as search.

One particular form of search that could be performed on top of a well-annotated database of human actions is that of human *activity* retrieval. In this type of search, a user would provide a query video containing a complex, perhaps non-linear sequence of human actions that they want to search for, perhaps in a surveillance system. For instance, a user looking for instances of shoplifting in a supermarket could provide a video that has the following actions: lots of glancing around; taking an item from the shelf; walking out without paying. The system would then provide a set of localised video results that most closely match the semantics of the provided query. The action retrieval, clustering and temporal segmentation work presented in this thesis could potentially be used as groundwork for the development of such a system.

In addition to these applications, each of the chapters below discuss the potential applications of the individual works as they are introduced.

1.2 Thesis Outline

The rest of this thesis is structured as follows.

Chapter 2 contains a full literature review of all of the previous works relevant to human actions analysis, and unsupervised/semi-supervised machine learning. Low-level action representation is first briefly considered, as it is necessary for all higher-level work. Then, some popular mid-level representations are also described. Various other topics that are relevant to this thesis are considered: information retrieval (with a focus on action retrieval); action localisation; temporal segmentation of human activities; clustering of human actions. Finally, all of the human action datasets used in this thesis are described.

Chapter 3 researches existing techniques for their applicability to human action retrieval. Several composite action retrieval systems are considered for ranking. Also evaluated are various methods for relevance feedback.

Chapter 4 extends the work of Chapter 3 to develop new algorithms for human action localisation, in two different approaches. The first approach performs spatio-temporal localisation within the context of human action retrieval. A novel algorithm is implemented, and experiments on several human action datasets are shown. In the second approach a new fully *unsupervised* algorithm is developed for performing temporal localisation of human actions based on analysis of humans' motions. This method's efficacy is demonstrated by applying it to a recognition framework – however, as it is unsupervised, it could also be applied for clustering, or for higher level tasks such as activity recognition.

In Chapter 5, a novel class of unsupervised algorithms is proposed to find the link between actions and scene context when performing human action clustering. Spectral Dual-Assignment K-means Clustering (SDAKM) is proposed as a specific implementation of this algorithm class, and it is proven effective on four action datasets.

Chapter 6 extends the work of Chapter 5 to consider how more complex contexts can be used to assist in the clustering of human actions. In particular, SDAKM is unsuitable when there may be multiple context instances, and so a novel algorithm called Multiple Object Single Action Clustering (MOSAC) is proposed to improve on SDAKM in this particular instance, and is tested on a cooking dataset.

Chapter 7 details a novel local-feature-based representation method for human actions using multiple graphs, that can be applied to enhance the accuracy of action clustering, retrieval and recognition. This technique, termed FGSM, is proven through extensive experiments for a variety of tasks on a variety of datasets.

Finally, conclusions are drawn and future work is considered in Chapter 8.

Chapter 2

Background and Related Works

This chapter provides a background to the general topic of human action analysis and certain related machine learning topics – concepts and specific related works are both covered. The topics discussed are as follows:

- Action representation – how to perform feature extraction and action representation, which are fundamental problems to all human action analysis tasks.
- Action localisation, detection and segmentation – how to find a human action spatio-temporally within a larger video.
- Action retrieval – finding actions within a database based on an exemplar.
- Action clustering – finding unsupervised groupings of the actions.

These topics are relatively broad, and apply to all of the work detailed within this thesis. However, some subsequent chapters also require specific knowledge from other fields of study, such as data mining – in those chapters, a further background section is provided, to give the proper context to the research.

2.1 Action representation

Feature extraction is typically the first and lowest level task of action representation, performed on a pixel level representation of an action video. The goal of this step is to obtain salient features in a video; these features should contain all the information required for the identification of the action, and minimal irrelevant information. After extraction, the features are either aggregated into a representation model (e.g., Bag of Features) or used directly for the learning task (such as in Naive Bayes Nearest Neighbour).

Action representation methods can be categorised according to the feature extraction method used: local feature-based methods, or global feature-based methods. The work in this thesis uses a variety of different action representations, both global and local, so a full review is given under these categories

2.1.1 Global Feature-based Methods

Global feature-based methods are concerned with the overall appearance of a human action in the video, and usually rely on some pre-processing or segmentation to remove irrelevant background information and isolate the human silhouette, such as background subtraction. Historically, techniques in this category have proven to be the most accurate on very clean datasets, such as the Weizmann [4]. However, their reliance on primitive pre-processing steps to isolate the action makes them poor at dealing with noisy conditions that appear in more realistic datasets; for instance, occlusion of the human subject, viewpoint variance, clothing changes, and ego motion all significantly degrade classification accuracy.

One of the earlier papers on human action recognition, Davis and Bobick [5] represented the motion of a person as two types of binary image – the motion-energy image and the motion-history image – and then used Hu moments to represent this image as a vector, with the Mahalanobis distance to distinguish the vectors, and the nearest neighbour algorithm for classification, as in ordinary image recognition. Such a technique, however, loses most temporal information, which is of considerable importance when distinguishing between certain actions

– for instance, running and walking. There have also been approaches to extract 3D, spatio-temporal human volumes from a video. For instance, Niebles et al. [6] implement such a system using a person detector, pose estimation, and temporal belief propagation of the changing shape.

In order to capture temporal sequence information, but retain partial invariance to temporal distortion of the action, Yamato et al. applied Hidden Markov Models (HMMs) [7] to a time-sequence of images. Silhouettes are extracted from each frame by background subtraction, are reduced dimensionally using moment invariants, and matched to a particular HMM state. A HMM is trained for each action, and Bayes rule is used in combination with these HMMs to classify unlabelled actions. This initial paper resulted in a considerable amount of human action recognition research into HMMs as an action classifier [8, 9, 10].

The original HMM models the duration of each state with an geometrically decaying probability, due to its looping structure, but for most action states this is not a realistic representation of the true state duration distribution, as it tends to favour only short actions. The Switching Hidden semi-Markov Model (S-HSMM), developed by Duong et al. [11], models duration probability by removing loops from the model and then explicitly including duration as an output from each state. Also due to its switching nature, an S-HSMM models more complex relationships between states than a simple first-order markov chain, which is of particular use in recognising more complex actions.

Other global feature methods rely on tracking of the body. Ikeda et al. [12] localise both the hands and face, and use the distance between them at each frame to classify actions. There is also considerable research in body pose estimation, from both multiple and single cameras, such as in Andriluka et al. [13]. Gu et al. [14] have shown promising results for action recognition using such an approach, classifying the actions based on extracted body joint angles from multiple cameras.

More recently, Zhen et al. [15] have achieved state-of-the-art results on several realistic action datasets, including the HMDB51 [16]. applied a global method called Laplacian pyramid coding to several realistic datasets. This holistic method implicitly extracts features by first creating a spatio-temporal Laplacian pyramid of the video, which permits analysis of the video at multiple scales, and then applying a bank of 3D Gabor filters to find salient spatio-temporal edges.

2.1.2 Local features

While global features have given impressive results for clean data, it has been noted that they are not at all robust to noise introduced by issues such as occlusion. In addition, research has shown, such as in Johansson [17], that it is unnecessary to extract a complete body pose at every frame before performing recognition; humans in particular are capable of distinguishing actions given surprisingly little information, such as the movement of point light sources attached to key points on the human body. This has motivated research into local features for action recognition, which are small video patches extracted from a local neighbourhood within the overall video. Initial work, such as that by Laptev [18], showed that, by extracting many video patches from areas of interest in an action video, and using a statistical method to model the distribution of these patches within a dimensionally-reduced feature space, it was possible to distinguish between actions in a robust manner. As local features do not rely on techniques such as body segmentation or background subtraction, they are a lot more robust to noise and variables such as rotation and viewpoint, making them more suitable for tasks in realistic settings.

Local Feature Detection

When using local features for recognition, the first task is to detect areas of interest within the video that are relevant to the action. If each frame of the video is conceptually stacked together to view the video as a 3-dimensional volume (a *spatio-temporal* volume), feature detection finds a series of (x, y, t) coordinates, the local neigbourhoods of which are considered particularly informative in recognising the action. Laptev [18] refers to these points as Spatio-temporal Interest Points (STIPs) which this report will adopt for convenience.

There are many methods for performing local feature detection for human action recognition. The earliest of these, such as the Harris detector [19], MSER [20] and SIFT [21] originated in image recognition, and work only on individual frames, discarding temporal information. These detectors were later extended to the spatio-temporal domain, such as the 3D Harris-Laplace detector [18] and the 3D SIFT detector [22]. Each of these functions looks for a specific structural feature in the image or video; for instance, the Harris detector finds corners, and the SIFT detector finds minima/maxima after applying a Difference of Gaussian (DoG) function.

Identifying features in a spatio-temporal volume using the same techniques as for an image is unlikely to give optimal results, as the temporal information is of a different nature to spatial information. Dollar's feature detector [23] takes this into account. It applies a 2D Gaussian smoothing kernel on the spatial dimensions, and combines this with two 1D Gabor filters in quadrature on the temporal dimension. Resultingly, this detector finds highly discriminative STIPs at points of complex motion. Because of its general strength, it has been very popular in further human action research since its introduction in 2005.

There have been other attempts at creating improved action motion detectors in the subsequent years, such as Ning's method [24]; Recently MoSIFT [25] has been shown to considerably outperform Dollar's method, its primary drawback being that it is computationally expensive – though this is mitigated to a degree as it is highly parallelisable. Shabani et al. [26] introduce their own detector after determining that Dollar's method has two fundamental weaknesses – that it loses important high gradient information during spatial Gaussian filtering, and that it does not account for ego-motion (such as camera-shaking). They fail, however, to make a quantitative evaluation between their proposed detector and Dollar's method.

The feature detection step may be skipped, in which case every possible local neighbourhood of the video is treated as a feature; this technique is known as densely sampled local features.

Local Feature Description

After feature detection, it is necessary to extract a video patch around each STIP to provide a description of each feature. It is possible to use the raw pixel values of each video patch for description; however, we usually want to group together local patches which have similar motion patterns and raw pixel values are not effective for this task. Instead, a feature descriptor is usually applied to each video patch which discriminates more accurately between similar features. Dollar's gradient descriptor was introduced in the same paper as his popular detector [23]. It takes the brightness gradients from the x, y and t dimensions and concatenates them to form a descriptor.

While the gradient descriptor performs better than pixel values, there are yet superior methods. Laptev et al. [3] used the combined Histogram of Oriented Gradients and Histogram of Oriented Optical Flow (HOG-HOF) descriptors, which were, once again, originally developed for image recognition. The first of these, HOG, divides the spatio-temporal patch into regions, and calculates the orientation of the gradient at every pixel. The orientation space is divided into bins, and according to these bins, a histogram of the orientations of the gradients is made for each region, and then concatenated for the final descriptor. HOF is similar, except that optical flow is used instead of the gradient. This combination of a static descriptor (HOG) with a dynamic descriptor (HOF) has been shown to be more effective than either descriptor alone.

The SIFT descriptor [21], as with its detector, has been extended to 3D by Scovanner et al. [22]. Histogram of Oriented Gradients, which is similar to the SIFT descriptor, has also recently been extended to 3D by Kläser et al. [27], who used a series of implementation innovations that made its calculations efficient enough to be practical. HOG3D has proven itself to be a particularly effective descriptor, achieving state-of-the-art results compared to all other existing local descriptors.

Once a feature descriptor has been calculated, it is often of an extremely high dimensionality. If this is the case, we can improve the efficiency and accuracy of the subsequent representation and classification by performing dimensionality reduction. The most popular dimensionality reduction technique is Principal Components Analysis (PCA) [23]. PCA, however, has limited performance in certain circumstances because of its assumption of a linear relationship between the input dimensions. Therefore there have been a considerable number of recent papers showing the use of alternative techniques in human action recognition, such as the linear method LPP [28], and non-linear LLE [29]. In Zhang et al. [30] the experimental results of many different dimensionality reduction techniques show that PCA is a comparatively weak algorithm, and also highlight the performance of supervised dimensionality reduction techniques such as FLDA, RLDA and particularly their own technique, SSCP.

Local Feature Representation

It is necessary to transform the features of each video into a representation that can be used in conjunction with a classification algorithm to recognise the action. Unlike with many global methods, the feature extraction does not result in a uniformly sized vector that can be directly used with classification techniques like SVMs or kNN.

One of the simplest types of action representations is the Bag-of-Words model (BoW) [23], which discards all structural information between the local features and makes a purely frequentative analysis. The first step of this representation is to quantize the feature descriptors into bins by their similarity to each other. To achieve this, the descriptors are clustered, ordinarily with k-means clustering, into k pre-specified groups, or *codewords*. For each action sequence, the number of videos falling into each bin is calculated and the bins are concatenated into a histogram of codeword frequencies. These histograms then map each action sequence into a k-dimensional space, and in combination with a distance metric this information can be used to train a classifier such as an SVM or kNN. Optionally, if the number of features in each action varies greatly, the histograms can be normalised. Popular distance metrics for action histograms include the χ^2 goodness-of-fit test, the Euclidean distance, and the histogram intersection.

Despite its ubiquity in both image recognition and human action recognition research, BoW has been criticised as a weak model for several reasons. Boiman et al. [31] claim that the quantization step loses a great deal of discriminative information. They instead advocate the use of the Naive-Bayes Nearest Neighbour (NBNN) algorithm. This algorithm skips the representation step and classifies an image or action based purely on local features by minimizing the distance:

$$\sum_{i=1}^{n} ||d_i - NN_C(d_i)||^2 \tag{2.1}$$

with respect to C, where d_i is the ith feature of the action and $NN_C(d)$ is the distance (by some metric) from the nearest feature in class C to feature d. The set of features in class C, D_C, is simply the set of all features in the training data for that class. As NBNN effectually requires no training step, nor the calculation of a representation such as a histogram, it has a low initial cost. Additionally, Boiman et al. showed it to be far more accurate than the BoW method. However, these gains are made at the expense of a very computationally expensive classification process, which requires $O(FT)$ time, where F is the number of features in the image/action and T is the number of features in the training set. By comparison, to perform classification with kNN in combination with BoW requires only $O(H)$ time, where H is the number of images/actions in the training set.

One of the more interesting improvements to the original BoW model is the Fisher Vector, proposed in multiple works for image representation, such as Perronnin et al. [32]. Instead of representing the visual vocabulary of image descriptors as a set of hard clusters generated by k-means, a Gaussian Mixture Model [33] is instead trained, more fully capturing the distribution of the descriptors. Let K be the number of components in the GMM, and D be the number of dimension in the descriptor. A $2KD$ Fisher Vector is created over an image by aggregating a mean-derived term and a variance-derived term for each component/dimension combination over all of the image descriptors. This approach results in a significant improvement over the BoW approach in most cases, and also significantly reduces dimensionality. Its most significant drawback is perhaps that it takes considerably longer to compute than the original BoW representation, though it is still more efficient than NBNN.

Another approach to reduce the impact of BoW quantization is by Grauman and Darrell [34], who introduced the original pyramid match algorithm. They iteratively divide the feature space into smaller and smaller bins, and at each level, count the features in each bin. This generates hierarchical histograms to represent each image, and then use these histograms in combination with a distance metric such as the Euclidean or χ^2 distances. They updated this method in [35] to improve its time and storage efficiencies. Lazebnik et al. [36] and Choi et al. [1] extend this technique by applying the pyramid match to the spatial, and spatio-temporal dimensions, rather than the feature space.

Another technique for capturing structural information is the shape-context, which has several different forms. The original shape context by Belongie et al. [37] was created for images. While ordinarily local features are described from the appearance of the local area around each feature, a shape context describes a local feature by its spatial relationship to other detected local features. A set of bins is centred around a local feature in a log-polar pattern. The value of each bin is determined by the number of other local features in the image that fall within that bin; these bins are then concatenated into a histogram that describes the central local feature. Using the χ^2 goodness-of-fit test it is possible to then find corresponding STIPs on two different images, and using these matched points as a reference, an affine transformation is performed on one of the images that maps it on to the other image. After this transformation, the mean squared error (MSE) of the distance between the matched points in the two images is used as a distance metric.

The shape context has been applied several times in human action recognition. Kortgen et al.'s [38] used the original algorithm but extended the log-polar pattern to 3D. Grundmann and Meier [39] improved this by evenly distributing the bins, as in Kortgen et al's implementation the bins nearer the poles of the shape context were smaller than those at the equator. Shao and Du [40] similarly used shape contexts, but discarded appearance information and used the shape-context histograms as semi-local feature descriptors in the ordinary

BoW paradigm, clustering the histograms into k groups and creating a single histogram representing each action sequence.

2.1.3 Other methods

There are certain techniques which do not fall cleanly into the local/global feature paradigm. For instance, densely-sampled features – local features extracted from every location on an image or video – are effectively describing the whole image/video, so while they are individually local features, together they could be considered a global feature method. One example of this comes from the work of Bregonzio et al. [41] who suggest that global-like features can be extracted from a dense cloud of local features, providing a descriptor that successfully combines the discriminative power of global features with the robustness of local features. This global-from-local approach also appears in in Shechtmann and Irani [42], where they extract the overall motion from many individual local patches, and combine the results to build up a global motion field representation for each action. Recently, one of the most successful representations for human action recognition is that of the dense trajectory descriptor by Wang et al. [43]. This method finds the trajectory of densely sampled points in a video, and describes them using HOG, HOF and MBH descriptors. Dense trajectories are particularly effective as they describe the relative motion of the objects within a scene. However, they are much more computationally expensive than sparse features.

Also related to densely-sampled features are mid-level representations, which are named mid-level because they fall in between the lowest level local features, and whole-video representations such as Bag-of-Words. The most popular examples include the deformable parts model [44] and the related poselets [45]. These representations describe the appearance of local parts of an object or person, and then model the approximate relationship between these parts. These methods are relatively slow (although faster than dense local features) and highly accurate, in part because they balance the low local variance and high global variance of the appearance of deformable objects. A good example of mid-level features applied to actions appears in Raptis et al. [46].

Work has also been done to more closely approximate how the biological brain processes vision, such as in Jhuang et al. [47] and Escobar et al. [48]. Ostensibly, these methods could be classified both as a global and a local method. The representation model retains structural information, much like a spatio-temporal shape context [40], in order to enhance accuracy to levels close to the best global methods. Also inspired by humans' and animals' visual systems, Convolutional Neural Networks(CNNS) [49] are similar in nature to the biological methods described above. They model an alternating sequence of convolutions and subsampling steps performed on an image as layers of a neural network, and use back propagation to train the weights of the convolution and subsampling kernels. Later works by Ji et al. [50] and Baccouche et al. [51] have independently extended CNNs to the video domain, with reasonable success on the KTH and Weizmann datasets. Recently, Karpathy et al. [52] have seen remarkable success in applying a CNN to a new very large scale video dataset called Sports-1M. In CNN based approaches, as all the model parameters are trained automatically (except for the network topology) the algorithm is not "handcrafted" for a dataset to get the best results; this makes the algorithm more flexible, and should achieve reasonable performance on any new dataset with minimal changes.

2.2 Multi-view and contextual recognition

While the majority of research is focused on visual, single viewpoint recognition of actions, there have been several worthwhile explorations of using additional information to augment recognition accuracies.

The first broad category here is that of multi-camera recognition, where a single action is viewed from multiple perspectives. Many of these techniques take a global approach and construct a visual hull from the background-subtracted silhouettes of each camera. Weinland et al. [53] suggest the use of Motion History Volumes on these 3D volumes; MHVs are the 3D analogue of Motion History Images introduced in [5]. Yan et al. [54] construct a full 4D spatio-temporal visual hull and extract geometric features for comparison. Peng et al. [55] decompose the visual hull as voxels and on these they perform multilinear analysis, and Turaga et al. [56] learn the manifolds on which the hulls lie for better representation. Despite achieving considerable accuracy with these techniques, they are especially prone to noise. Cilla et al. [57] avoid construction of the visual hull, and instead use a correlation model to integrate the cameras' observations for a probabilistic classifier. Naiel et al. [58] calculate the MEI/MHI and extract 2DPCA features for each camera, and combine the classification results by majority voting. Srivastava et al. [59] adopt a local feature-based approach to get a BoW histogram for each camera, and combine these histograms for the final action representation.

Contextual information relating to an action can also be used to improve accuracy, and this contextual information can be gathered from a variety of sources. Marszałek et al. [60] exploit the correlation between the type of movie scene and the probability of a certain action being performed to create a joint-SVM classifier, improving both action and scene recognition accuracy. Jiang et al. [61] take this one step further and perform both scene and object recognition, combining this information in a probabilistic framework for enhanced recognition. Han et al. [62] use object recognition, object-part recognition and body-part recognition to provide contextual data for action recognition. Everingham et al. [63] use the subtitles and script associated with TV programs to automatically learn the appearance of TV characters' faces and track them. Schroff et al. [64] use image-associated metadata to automatically "harvest" an image database from the web; this idea could be extended to human action videos by using metadata on websites such as YouTube.

Ikizler-Cinbis and Sclaroff [65] combine object, scene and action information in a multiple instances learning framework, to improve the classification performance of YouTube videos. Prest et al. [66] use a weakly supervised framework to learn the interaction between human actions and the objects in the scene, in particular learning the spatial relationship between actions and objects. Liu et al. [67] create an action recognition system where contextual information (as well as action execution details) are integrated as *attributes* of the action – these attributes are modeled as latent variables. Li et al. [68] attempt to find correlations between events in various traffic datasets, which can be used for enhanced abnormal behaviour detection, using a cascaded topic model (Latent Dirichlet Allocation) to find co-occurrence relations between events in data. It is important to note, however, that all of these techniques rely upon training data to learn the relationship between actions and context – later in this thesis, Chapters 5 and 6 present novel methods to consider unsupervised context for actions.

We can also improve action recognition in certain circumstances by using intersensory data, such as audio, in addition to the visual data. Intersensory data has proven particularly useful in speech recognition, as demonstrated by the coupled audio-visual HMM by Nefian et al. [69] In a human action setting, Töreyin et al. used audio in addition to visual data to help distinguish between falling down and controlled sitting. Wu et al. [70] use audio for context in the Hollywood dataset and show that audio is more informative than scene context in their dataset. Abdullah and Noah [71] fuse high-level audio features with a HMM-based action recognition approach to recognise violent actions in movies.

2.3 Action Localisation

While action recognition is generally aimed at temporally pre-segmented actions, there have more recently been attempts to perform action *localisation*. Localisation in the context of actions refers to determining the bounding region of where an action occurs – spatially, temporally or both. This could have many direct applications, such as for searching through long video sequences such as films. It is also particularly useful in the context of activity recognition, where often the component atomic actions must first be recognised and localised before the activity as a whole can be recognised.

There have been several different approaches to action localisation to date – initially these approaches have mostly been based on global techniques. If it is possible to perform background subtraction, or extract a spatio-temporal volume, then the task of localisation can be trivial. However, there have been some more sophisticated global approaches that can temporally segment actions of interest from irrelevant noise. Sullivan and Carlsson [72] perform 3D body joint tracking using keyframe matching on global features. In Kläser et al. [73], spatial body tracking and a temporal sliding window are used to perform action localisation in the noisy Hollywood Localization dataset.

There have recently been more attempts to use local features in localisation. In Thi et al. [74] local features are used to perform localisation in several datasets, by attaching a relevancy weight to each local feature (which measures how relevant that local feature is to the classified action) and then creating a spatio-temporal bounding box around all the local features which pass a relevancy threshold. Ryoo and Aggarwal [75] demonstrate the effectiveness of local feature voting in their activity recognition system – where each feature casts a "vote" for the spatio-temporal boundaries of the action. Oikonomopoulos et al. [76] also use feature voting in combination with mean shift mode.

One of the better tools for performing human action localisation is person detection and tracking. With the recent improvement of person detectors, such as poselets, first published by Bourdev and Malik [45], and person trackers, such as the recent GMCP-tracker by Zamir et al. [77], it is now possible to get relatively stable human tracks from many action datasets. Even for more realistic datasets for which person tracking might not work, such as UCF YouTube [78], bounding box annotations have been published. Because of this, it is now also possible to apply global representation methods even to videos that have significant ego motion, scale and translation variations, and where background subtraction/silhouette extraction is not possible. Recent action recognition works that rely on bounding boxes of persons include Zhen et al. [79] who use embedded motion and structure features within the bounding box and Ji et al. [80], which describes 3D convolutional neural networks for feature extraction from the person tracks.

Another approach to action localisation is the temporal segmentation of human actions. Many existing works, such as Kläser et al. [73], take a fully supervised approach to this problem, training a model to localise specific actions within a longer video. However, these techniques require training and prior knowledge of the actions to be localised. Additionally, the complexity of this type of localisation scales linearly with the number of action classes to be found. The goal of *unsupervised temporal segmentation*, on the other hand, is to split a video or track into temporally discrete blocks, based on some inherent property of the video (e.g. start/end of linear motion), rather than relying on training data. Unsupervised temporal segmentation could potentially be applied in many scenarios, including video representation for multimedia retrieval, keyframe analysis, unsupervised action categorisation, and efficient action detection. Research on unsupervised temporal segmentation of human actions has been sparse. One work that performs unsupervised temporal segmentation, Xiang and Gong [81], does so using blob trajectories in surveillance, which is a method that will not work for highly complex human actions. However, there are some notable recent works [82, 83, 84] that have started researching unsupervised temporal segmentation for action clustering.

2.4 Action Retrieval

Human action *retrieval* is a subtly different task to recognition, with no training set available (or rather, only a single training sample available) and results judged on retrieval ranking rather than classification; it is a subfield of Content-based Video Retrieval (CBVR), which is in turn an extension of Content-based Image Retrieval (CBIR) to the video domain. While CBVR – which has its roots in CBIR – has been studied for almost the same length of time [85, 86], video retrieval has to date seen relatively little attention. Clearly defined, the task of human action retrieval is to search through a database of human actions and return a set of human actions in ranked order by their relevance to the query. The query can be textual in nature; however, most ordinarily the query is itself a video example of the action that is to be retrieved. The main problem here then, is to establish an efficient and effective measure of the distance between the query and each database video. Finding such a measure can be of particular difficulty for human actions, as two semantically identical actions can have very different appearances, and a single query action is often insufficient to capture this intraclass variance, causing incomplete/poor results.

Jin and Shao [87] have performed human action retrieval, representing actions using local features and performing relevance feedback to improve results. However, the most effective retrieval method to date was applied to image retrieval: manifold ranking, presented in He et al. [88], which is a graph-based algorithm. Manifold ranking incorporates the underlying structure of the dataset to rank the database items according to their similarity to the query. It can also elegantly incorporate positive and negative feedback to improve its rankings further. There are also two tasks closely related to content-based retrieval, which is explored later in this thesis. Firstly Localised Content-based Retrieval, as presented in Rahmani et al. [89] and Zhang et al. [90], attempts not only to rank database images in response to a query, but localise the object of interest within the image. The second task related to retrieval is that of action recognition using only a single training sample, or one-shot learning – this is similar to retrieval, as the retrieval query can be considered equivalent to a single training sample. The primary difference is that while action retrieval seeks to rank the database items in terms of their relevance to a query, action recognition from a single sample attempts to classify the database items into categories. Seo and Milanfar [91] present surprisingly promising results on the Weizmann and KTH dataset using a classifier trained only from a single sample.

2.4.1 Relevance Feedback

Relevance feedback is a technique used within retrieval systems to improve results by altering the query. Once a user has retrieved results from the system, he/she can then choose to mark some of these results as relevant or irrelevant to the query. The system then integrates this additional information with the original query, in order to create a better result ranking. This two steps – feedback and result refinement – can be performed iteratively until there is no improvement in results or the user is satisfied. Relevance feedback was first applied to textual information in Salton [92] and has been of considerable interest in content-based image retrieval for some time – two notable examples of this are in Tong and Chang [93] and Hong et al. [94]. These systems rely on training binary SVMs to incorporate relevance feedback, which, while effective for relatively simple tasks, are highly dependent on the quality of the feedback, and assume that relevant/irrelevant samples can be accurately described by linearly separable vectors. Tao et al. [95] introduced a partial solution to these issues in the form of ABRS-SVMs, which are more robust against disproportionately sized positive and negative feedback sets, as a response to most existing retrieval systems that will, on the first few iterations, return more irrelevant than relevant results. Based on this, Zhang et al. [96] combine asymmetric bagging with soft query expansion. Bian and Tao [97] introduced biased discriminant euclidean embedding, which attempts to model a non-linear structure in low-level image features. Tian et al. [98] In order to more accurately model the widely distributed negative samples, Tao et al. [99] first cluster the negative samples into groups and then train a series of marginal convex machine subclassifiers between each of these groups and the positive samples, combining the sub-classifiers into a single relevance classifier. However, all of these methods have been applied only to low-level global features such as texture and colour in images, so it is unclear how they would perform in a task such as human action retrieval, that typically relies on local features. All of the above examples are based on image retrieval, but there have been a few explorations of relevance feedback on video and human actions. Yan et al. [100] applied pseudo-relevance feedback to the TREC2002 video track database, but only used simple, static features like global colour and texture descriptors. Jin and Shao [87] examined applying a single round of relevance feedback to human action retrieval with a simple aggregate measure for combining the relevance feedback. Active learning is an alternative to relevance feedback, which once again relies on user interaction to improve results. Instead of choosing feedback samples from an improving result set, however, the system provides the user with a set of highly informative samples for which the user should provide (ir)relevant labels. It is expected that this method could more swiftly improve results than relevance feedback. Some examples of these techniques are found in Zhang and Chen [101] and Tong and Chang [102].

2.5 Action Clustering

Other works have focused on fully unsupervised clustering of human actions. To solve this problem, research is typically focused on improvements in three areas: either the action representation (for better cluster separation), the metric for comparing actions, or the clustering technique (for finding non-linear correspondences in the data). Yang et al. [103] demonstrate that a global action descriptor and a temporal matching algorithm provide superior results to local feature based methods for clustering. Niebles et al. [104] use probabilistic Latent Semantic Analysis (pLSA) and Latent Dirichlet Allocation (LDA) – techniques originating from natural language processing – to cluster the actions based on the intermediate topics associated with them. Wang et al. [105] show the effectiveness of spectral clustering, using a linear programming technique to find the distance between pairs of action images. Topic models have also been of use in performing mining of behaviour in videos, such as in Hospedales et al. [106], which introduced a novel clustering method, the Markov Clustering Topic Model (MCTM), to perform unsupervised scene interpretation. Chaudhuri et al. [128]

2.6 Datasets

Due to the great breadth of the human action recognition field, there is no one canonical dataset for researchers to use to compare their methods. Indeed, as research has progressed, the growth in number of datasets has accelerated, as old datasets are found lacking in addressing new methods and research goals. In this section an overview of the many datasets in the field are given. Table 2.1 summarises the datasets.

The Weizmann dataset, created by Gorelick et al. [4], is a simple dataset

Dataset	Num. Act.	Num Class	Diff.	Loc.
Weizmann	93	9	Easy	No
KTH	2391	11	Easy	No
UCF Sports	150	8	Easy	No
MSR2 Localisation	203	2	Easy	Yes
UT-Interaction	164	6	Medium	Yes
YouTube	1168	11	Medium	No
Hollywood-2	3669	12	Hard	No
MPII Cooking Activities	5609	65	Hard	Yes

Table 2.1: Various human action datasets used in this thesis. Num Act. shows the total number of action instances in the dataset; Num Class. shows the number of action categories; Diff. indicates the approximated subjective difficulty of the dataset according to this researcher; Loc. indicates whether the dataset has the ground truth for localisation.

consisting of 10 actions performed by 9 actors, with a total of 93 sequences. All of the actions are shot from an orthogonal perspective, with a static background, and there is little variation in clothing or body types. The Weizmann dataset has seen 100% classification accuracy for whole sequences even in its original paper, using background subtraction and global feature-based methods for classification. However, due to its extremely clean nature, the best performing algorithms on the Weizmann tend to perform poorly on more realistic datasets.

The KTH dataset [2], shown in Figure 2.1, has 6 actions performed by 25 subjects in different conditions, for a total of 2391 sequences. Alternatively, the KTH dataset can be viewed as a series of 599 longer sequences, concatenating several of the smaller sequences for each example. The actions are shot from an orthogonal perspective, and in 4 different conditions: inside; outside; outside with scale variations; and outside with different clothes;. The KTH dataset has proved much more challenging to global feature-based methods, especially because of the non-static background due to ego motion, precluding clean background subtraction. The video quality is poor, and the lighting conditions are quite variable. 3 of the actions – walking, running and jogging – also have very similar motion patterns and are difficult to distinguish between. For this reason, no technique so far has been able to achieve 100% accuracy on the dataset. The best techniques, based on local features, achieve accuracy between 91 and 96%, but due to the diminishing returns of the incremental improvements to techniques thus far, progression to 100% may require a fundamentally different approach.

While the KTH dataset is a step closer to realistic datasets, it is still shot from a single perspective and includes only very simple, cyclical actions. Many of the more recent datasets have a more practical focus. The UCF datasets are all composed of human action videos drawn from existing YouTube videos, and are aimed at researchers who wish to design algorithms for online videos. The actions in the UCF Sports dataset [107] are short, atomic actions, such as *Shoot Basketball* or *Ride Bicycle*, and they are generally shot from the same perspective, but do not necessarily have static backgrounds. Examples are shown in Figure 2.4. The UCF YouTube [78] and UCF YouTube 50 [108] datasets contain similar actions that are not restricted to sports, such as walking, and can be shot from a variety of perspectives. In the UCF YouTube 50 dataset, some of the actions given are very general (such as *Tai Chi*) and might be considered as higher level activities. Due to the low quality of Internet video, and the great intraclass variety in these realistic datasets, the UCF datasets are still considered to be open problems. Examples of the YouTube dataset are given in Figure 2.3.

The MSR2 Localisation Dataset [109] consists of a 3 action subset of the KTH dataset – specifically the waving, punching and clapping actions. What distinguishes this dataset from the KTH is primarily that the actions are spatio-temporally unsegmented from the rest of the scene, so that the MSR2 dataset can be used for action localisation/detection experiments. Additionally, the MSR2 dataset has multiple actors per scene, a variety of locations, and considerable background noise, so that it poses a more significant feature extraction challenge than the KTH. Examples of this dataset are in Figure 2.2.

Attempting to capture more realistic data for human action recognition, Laptev et al. [3] and Marszałek et al. [60] have created the Hollywood dataset, and subsequently extended it with the Hollywood 2 dataset. The Hollywood 2 dataset, examples shown in Figure 2.6 consists of 12 action classes, in 10 types of "scene" (used for context-assisted classification), extracted from 69 different movies. It totals 3669 sequences and 20.1 hours of video. These action classes are mostly simple and atomic, such as *Kiss* and *Sit Down*, but also contains a more activity-based *Fight Person* class. The actions are performed in a variety of ways, from many perspectives, in black and white and colour, with varying lighting and occlusion (in some cases, extreme occlusion), and with many different actors. These highly realistic datasets can be expected to present a challenge in the field for many years to come due to this huge variety, and especially as some of the actions can only be recognised through contextual inference. The best performing approach on the Hollywood 2 dataset in the literature is Wang and Schmid [110].

The UT Interaction dataset [111] features videos captured from a surveillance perspective of 6 classes of 2-person interactions. These interactions are captured against an unmoving background at high resolution, but are performed from a variety of viewpoints and by a variety of actors. It is chiefly of interest because it looks at interactions rather than simple isolated 1-person actions. Examples of the UT Interaction dataset are found in Figure 2.5.

The MPII Cooking Activities dataset, introduced in Rohrbach et al. [112], is designed for the analysis of fine-grained human actions that might occur in a kitchen. The actions, such as *stirring* and *chopping* have very subtle visual differences, so techniques used for whole-body action classification aren't typically as effective on this dataset as they are on others. There are 65 distinct action classes, which contribute to the difficulty of analysis, and 5609 actions. These are taken from 44 videos that each contain the completion of a single recipe. However, all actions are captured from the same viewpoint in the same kitchen, with the same lighting, meaning that simple techniques such as background subtraction can be used in the first stage of processing. Example scenes are shown in Figure 2.7.

2.7 Discussion

This thesis covers several disparate topics within the unsupervised/semi-supervised learning of human actions – clustering, retrieval and localisation. However, these

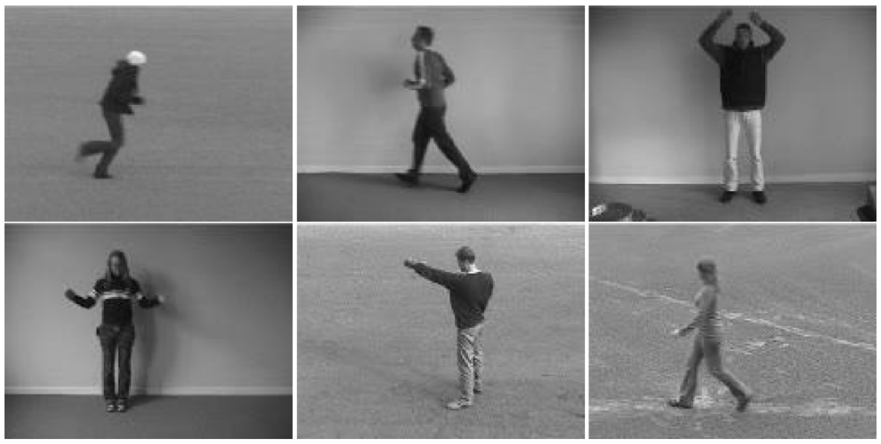


Figure 2.1: Examples of actions from the KTH dataset.

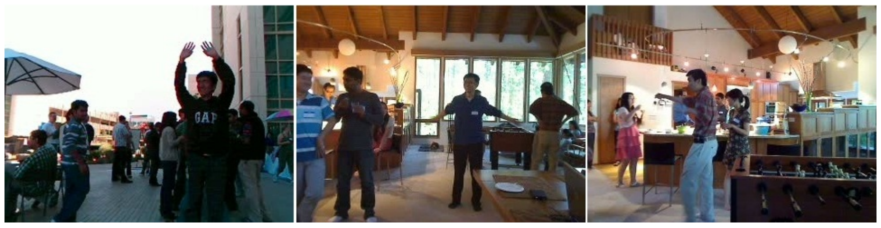


Figure 2.2: Examples of actions from the MSR2 dataset.

tasks share many commonalities, justifying their inclusion in a single body of work. Three examples of these commonalities are provided here. Firstly, these tasks are all unsupervised/semi-supervised – as seen later in Chapter 7, an action representation that performs well on one unsupervised/semi-supervised task is likely to perform well on other unsupervised/semi-supervised tasks. Secondly, findings from experiments in one of the topics are used to inform design decisions in the other topics. For instance, in the experiments of Chapter 3, this author notes the effect of scene context on *retrieval* results – this finding is later used to motivate the topic of *clustering* with context in Chapter 5. Finally, many of the tasks presented here can be used in conjunction with one another to build a higher-level system – for instance, action clustering could be used in conjunction with action retrieval to perform activity retrieval. This idea is discussed further in Chapter 8.

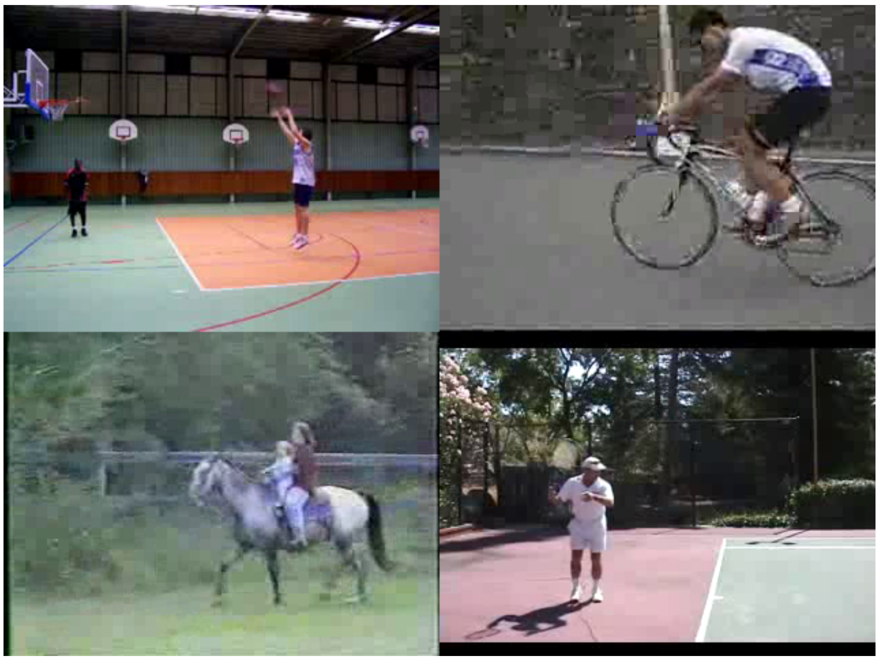


Figure 2.3: Examples of actions from the UCF YouTube dataset.

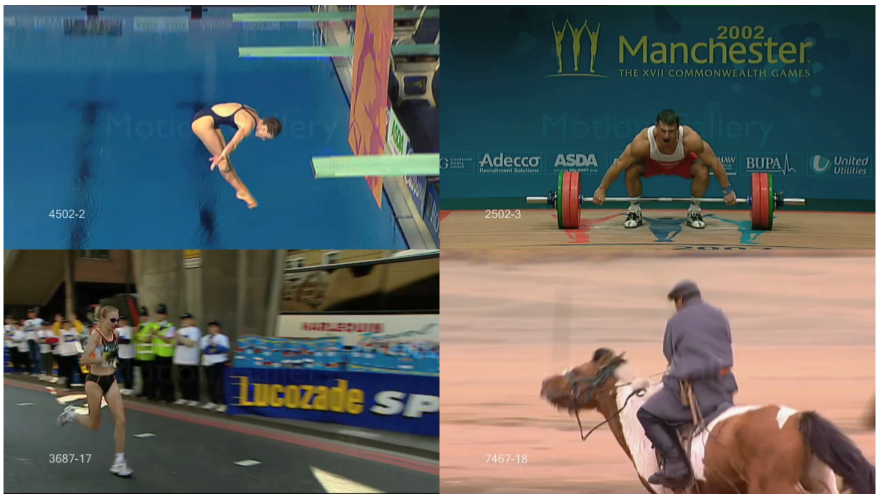


Figure 2.4: Examples of actions from the UCF Sports dataset.

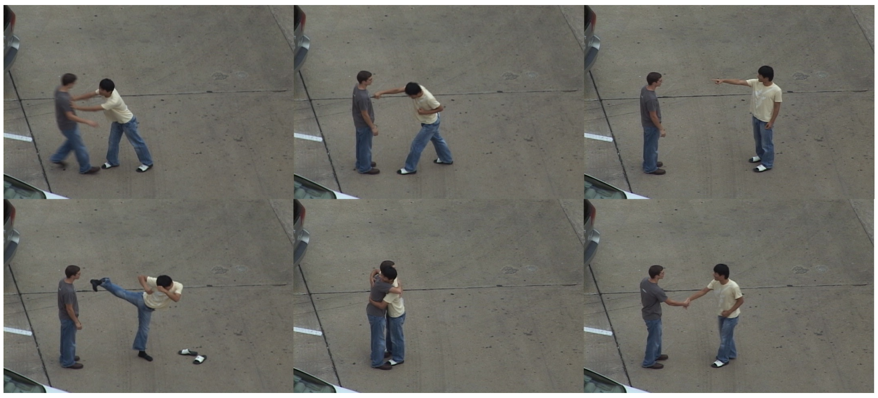


Figure 2.5: Examples of actions from the UT-Interaction dataset.

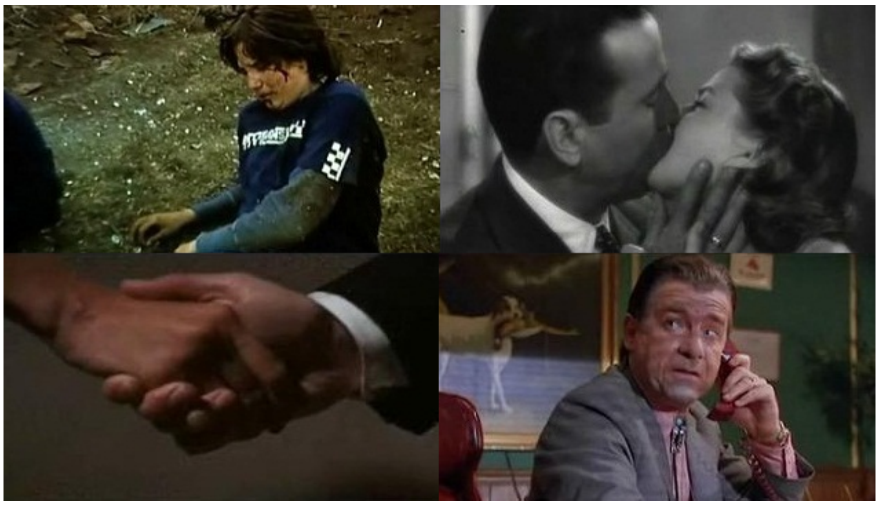


Figure 2.6: Examples of actions from the HOHA2 dataset. Clockwise from the top left: sit up, kissing, answer phone, handshake.

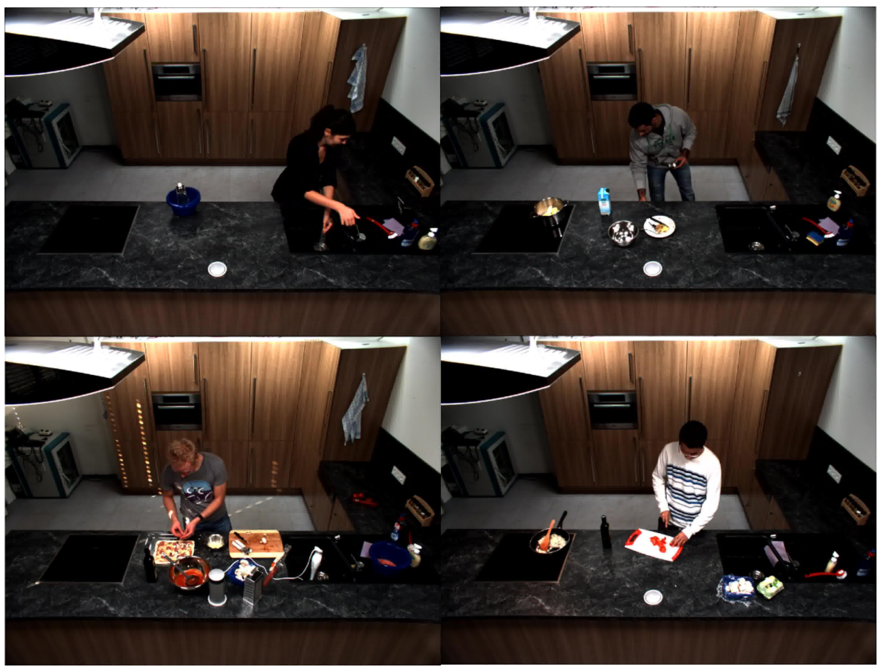


Figure 2.7: Examples of shots from the MPII dataset.

Chapter 3

Human Action Retrieval

3.1 Introduction

The recent explosion of multimedia information on the World Wide Web has simultaneously resulted in many potential applications and many challenges. Foremost among these challenges is managing the enormous amount of data. When search engines such as Google or Yahoo were introduced, the vast majority of data available on the Web was in textual form; however, with consumer-grade Internet speeds growing at an unprecedented rate, and with the advent of such sites as *YouTube* and *Google Video*, users are making increasing use of the Internet to look up and consume images, music and videos.

Because of the rapidity of this shift in content, methods of accessing this data have been slow to keep up. The majority of Web users still rely on textual keyword or phrase searches to perform all of their searches, whether they are looking for modes of information as disparate as, for instance, a newspaper article or a music video. There are several disadvantages to using a keyword search to find multimedia information. Firstly, this method requires that the correct textual data is associated with the multimedia object in question. To give an example, most current image search engines find images embedded on webpages in close proximity to the given search keywords, and YouTube video searches look primarily at the title and description of each video, as well as the keyword “tags” that can be attached to them. Unfortunately, this textual data is often inaccurate, and always incomplete; words cannot be used to fully describe a multimedia object.

The second problem with textual searches is that often a user will not know the name of what he/she is looking for, but only some knowledge of its contents, such as the visual appearance of an actor, or what an object looks like. The solution then to both of these issues is to perform a search directly on the contents of the objects, a practice referred to as Content-based Information Retrieval (CBIR). This technique has been well-explored in the audio and image domains, resulting in commercially available systems. For instance, *Shazam* is a popular service that allows users to find a music track by playing or humming a short clip of the track into a microphone; *Google* has recently extended their keyword image search so that users can provide an image to begin a content-based search.

The most challenging area, however, remains largely unsolved – video search engines still rely primarily on textual keywords to retrieve results. There are several reasons that video content-based searches prove more challenging than either audio or image content-based searches: searches must be performed not only on appearance but also on motion; videos are multi-modal, meaning that they include both visual and audio information; because the amount of data involved in a video is an order of magnitude higher than in images or audio, it is correspondingly more difficult to extract the pertinent information; additionally, because of this huge amount of data, algorithms must be extremely efficient to be practical on large video databases.

This chapter focuses on one sub-domain of CVPR – that of Human Action Retrieval. Human actions are of common interest to researchers both because of the difficulty in recognising them, and because of their ubiquity in most available video media. There are a great variety of possible human body appearances and poses, as well as a multitude of ways an action can be performed. Even the semantics of what constitutes an particular action might not be well defined. Any practical CBVR system would, however, be expected by its users to overcome these problems.

One previously popular class of techniques for improving search results on difficult subjects is known as *relevance feedback*. Here, a search engine will return an initial small set of results for a query, and the user will provide the search engine with information about which results are relevant to the query, and which are not. The search engine can then incorporate this additional information into an enhanced query, perform an improved search and retrieve better results. This process can be iterated as often as required, until the user is satisfied with the results.

This chapter explores the application of a form of relevance feedback to the retrieval of human actions. This technique has previously been applied in the image domain, and this chapter shows that it can be extended to the video domain, even for very noisy datasets, such as those found on Youtube, or in Hollywood movies. In particular, the algorithms are tested on the *Hollywood* dataset [3] of complex and realistic human actions. It is shown that the use of Relevance Feedback (RF) can be used to greatly augment the accuracy of such a system after only a few iterations.

3.2 Action Representation

In this section several approaches are examined to represent and compare human actions, divided into two categories: local feature extraction methods, and techniques for aggregating these features into a model for comparison.

3.2.1 Local Feature Extraction

Local Feature extraction refers to the dual-process of first detecting salient points in an image or video, and then describing the area around them. For detection of features Dollar's method is used [23], as it is the best detector as shown in [113]. For description, two methods are compared: Dollar's Gradient [23] and the state-of-the-art HOG3D [27]. In addition to motion features, static SIFT [21] features are extracted from keyframes in the video. Finally the use of combinations of the above extraction methods is explored.

3.2.2 Comparison Model

When performing action retrieval, a metric is necessary to compare how close the query action is to each database sample. Several methods of measuring this similarity are investigated.

Bag-of-Words

Under the Bag-of-Words model, dimensionality reduction is first performed on the extracted features. Then, the features are clustered into codewords using k-means clustering. According to the codewords of an action's component features, a frequency histogram is generated for each action.

To measure the similarity between histograms several metrics have been considered. Because the histograms generated in the BoW model tend to sparsity, the χ^2 distance and Bhattacharyya distance are unsuitable, as they tend to become unrepresentative on sparsely sampled distributions. The Earth Mover's Distance, popular for comparing colour histograms of images, is also not suitable, as in its standard form it assumes that the histogram bins are in order – but in BoW histograms, bin adjacency has no meaning. It is also possible to apply the EMD without considering bin adjacency, but then it simply reduces to the L_1 norm. The L_1 norm, L_2 norm and Kullback-Leibler divergence empirically gives poor results for comparing BoW histograms for action recognition. Ultimately, the histogram intersection is considered the least flawed metric in comparing BoW histograms, and as such is commonly found in the literature [23, 43]:

$$s(H_q, H_t) = \sum_{i=1}^{k} min(H_q^i, H_t^i) \tag{3.1}$$

Codeword assignment to the features can either be *hard* or *soft*. For hard codeword assignment, each feature is assigned to a single codeword corresponding to the nearest cluster centroid. In soft assignment, each feature is assigned proportionately to each codeword in the codebook, according to the following formula:

$$B_{f,C} = \frac{e^{\frac{d_{f,C}}{\sigma^2}}}{\sum e^{\frac{d_{f,C}}{\sigma^2}}} \tag{3.2}$$

where $B_{f,C}$ is a normalised vector (summing to 1) of the proportional belonging of feature f to each of the clusters C, $d_{f,C}$ is a vector of the Euclidean distance between the feature and every cluster centroid, and σ is a constant which determines the "hardness" of the soft assign. Generating a codeword frequency histogram for a video is then performed by aggregating the soft assignment vectors for each feature in the video.

Vocabulary-Guided Pyramid Match

The original pyramid match algorithm, introduced by Grauman and Darrell [34], attempts to improve on the bag-of-words model by using multi-resolution histograms to describe an image. Initially, the feature space is divided into 2^d equally sized bins, where d is the dimensionality of the feature space. Then, the feature space is iteratively divided into smaller bins to give increasing levels of resolution, until each bin contains at most 1 feature. There are 2^{i^d} bins on the ith level. The resulting set of histograms can be used to represent an action; to measure the similarity of two images, the histogram intersection at each level is calculated, weighted by the inverse of the size of the bins at that level, and the resulting values are summed. This similarity measure takes the following form:

$$\begin{aligned} sim(X,Y) = \sum_{i=0}^{L} \frac{1}{2^i}(\sum min(X_i, Y_i) - \\ \sum min(X_{i-1}, Y_{i-1})) \end{aligned} \tag{3.3}$$

where X_i is the ith level of the multi-resolution histogram X, L is the total number of levels in the histogram, and $\sum min(a,b)$ is the histogram intersection.

Unfortunately, this pyramid match algorithm is intractable for high dimensionality feature spaces, as it has $O(2^d)$ time and storage complexity. Grauman and Darrell addressed this issue with their vocabulary-guided pyramid match

(VGPM) [35], which serves as an efficient approximation of the original pyramid match. Rather than divide the feature space into uniformly sized bins along every dimension, the features are clustered using a k-means clustering algorithm, resulting in unevenly sized and shaped bins. The features within each of these bins is then clustered again using k-means, and this process is repeated hierarchically, with smaller and smaller bins, until each bin contains at most one feature. The benefit here is that the bins follow the structure of the feature space, resulting in several orders of magnitude less redundancy in the repesentation than in the original pyramid match. The similarity measure here is similar in concept to that shown in Equation 3.3; however, the weight term is adjusted for the size of each bin. Each bin has the following similarity measure:

$$sim_b(X,Y) = \frac{1}{d(b)} min(X_b, Y_b) - \sum_{b'}^{ch(b)} min(X_{b'}, Y_{b'}) \tag{3.4}$$

The size term of bin b, $d(b)$, is calculated during the hierarchical k-means process as the maximum distance between any two features within b. $ch(b)$ is the set of all children of bin b. The total similarity is calculated by summing over the similarity of every bin.

Spatio-temporal Pyramid Match

The original pyramid match was also modified separately by Lazebnik et al. [36] to be applied spatially to an image. Initially, the features are clustered into codewords as within the Bag-of-Words model. Then, the original pyramid match is applied iteratively to divide the image spatially into multi-resolution bins along both dimensions, and a codeword frequency histogram is generated for each bin at every level. The advantage of this approach is that the spatial correlation between the features is partially retained, which can lead to improved results; the disadvantage, however, is that this representation is not invariant to most geometric transformations on the video, such as translation, scaling and rotation. It also has greater computational complexity than its sister VGPM algorithm. Its similarity measure is identical in form to the original pyramid match.

Choi et al. [1] extended the Spatial Pyramid Match (SPM) to the video domain with the Spatio-Temporal Pyramid Match (STPM). The STPM algorithm is near identical in form to SPM, but is additionally applied to the temporal dimension. Consequently, the same benefits and drawbacks apply to STPM as SPM; this may adversely affect STPM's performance on realistic human action datasets.

3.3 Video Retrieval

In a content-based video retrieval system the goal is, supplied with a query video as an input, to find set of videos within a database that are most relevant to the query. The system presented in this chapter achieves this as follows. Firstly, all of the database videos are pre-processed in an offline step to extract all of the local features, generate a visual codebook, and represent the local features using a model such as Bag-of-Words or a Spatio-Temporal pyramid. This step is done once, offline, which improves the speed of queries by several orders of magnitude. When a query video is supplied by a user, this video is first processed as for the database videos, and then compared to the video database, to get a ranked list of database videos by similarity, of which the top X are returned to the user. Then, in the relevance feedback step, the user marks some of these results as positive (relevant) and others as negative (irrelevant). The system incorporates the feedback to generate an improved set of results. The result generation and feedback steps are performed iteratively until the user is satisfied or no further improvement of the results is possible. Figure 3.1 illustrates the system.

3.3.1 Relevance Feedback

As described in §2.4.1 in the literature review, to improve accuracy, the user can provide feedback on the results returned after retrieval, marking them either as relevant or irrelevant to their query. The retrieval system will perform an initial query and return the top X most relevant results to the user. The user can then mark as many or as few of these as "relevant" and "irrelevant" as desired, and

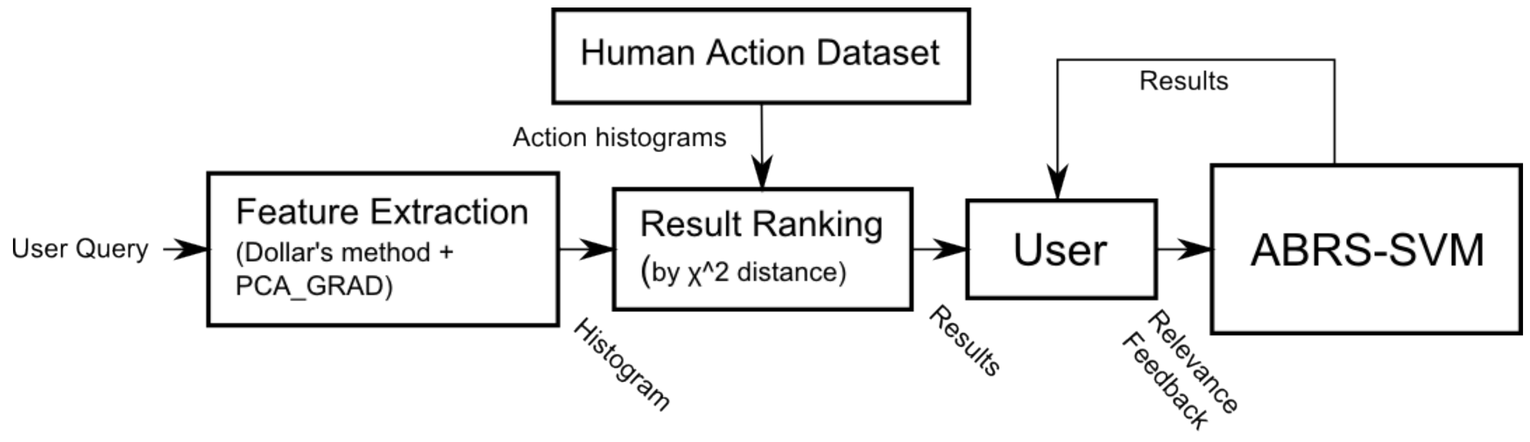


Figure 3.1: A simplified diagram of a working information retrieval system.

then run the query again to generate improved results. There are several methods for incorporating this relevance feedback into the retrieval system. Relevance feedback can be considered to be a basic type of active learning.[1]

Support Vector Machine

Perhaps the simplest relevance feedback technique is the Support Vector Machine (SVM). Let us define a set of positive feedback $P = \{pos_1, ..., pos_n\}$ and a set of negative feedback $N = \{neg_1, ..., neg_n\}$, where neg_i and pos_i are data points of d dimensionality. An SVM is trained by finding the optimally separating hyperplane in d-dimensional space between the two sets P and N. Typically the SVM can then be used to classify further data points into P or N by identifying on which side of the hyperplane that point lies.

In a retrieval system, the two classes are "relevant to the query" and "irrelevant", but it is not sufficient to find the class of the data points. It is also necessary to rank the points on the relative confidence of their relevance. Rather than performing classification, therefore, the SVM can be used for regression to get a real-valued relevance score. This is done by measuring the perpendicular distance of each datapoint from the hyperplane.

ABRS-SVM

SVMs are not ideal for relevance feedback, however, as they tend to be unstable when few training examples are available, or when there is a significant disparity between the number of examples in each class. In relevance feedback it is often expected that the majority of feedback will belong to the "irrelevant" class, rather than the "relevant" class. To overcome these limitations in image retrieval, Tao et al. [95] outline the Asymmetric Bagging and Random Subspace for SVM (ABRS-SVM), which learns a series of weak classifiers to distinguish between P and N

As the name suggests, the ABRS-SVM is made of two components. The

[1]NOTE: A further work has been published to extend the findings of this chapter to more sophisticated forms of active learning, but has not been included as many of the experimental contributions come from other the other co-authors. Please see Jones et al. [115].

first component, called Asymmetric Bagging SVM (AB-SVM), is a model that compensates for unequal numbers of examples in each class. AB-SVMs work as follows. First, random sampling with replacement is performed s times on N, to generate s new negative sets: $N'_1..N'_s$, where each N'_i has the same number of samples as P. Then, s linear SVM estimators $\{c_1, ..., c_s\}$ are trained, using $\{P, N'_i\}$ as training data for the ith estimator. Effectively, each weak estimator s_i is trained using the same number of positive and negative data points, so the hyperplanes of each are not distorted by the greater size of N compared to P.

The second component of the ABRS-SVM is the Random Subspace SVM (RS-SVM) method, which aims to reduce overfitting to a small number of training samples. The first step of the RS-SVM model is random sampling with replacement, applied to both P and N, but to the feature space rather than the example space. A small subset of the features in P and N is extracted. This random sampling is performed f times, generating f new positive sets $P'_1..P'_f$ and negative sets $N'_1..N'_f$, each with small subsets of the full feature set. f linear SVM estimators $\{c_1, ..., c_s\}$ are then trained, using $\{P'_i, N'_i\}$ as training data for the ith estimator c_i. As the features in N'_i and P'_i are a small subset of those in N and P, the overfitting problem is mitigated.

The full ABRS-SVM combines these two components. First, the Asymmetric Bagging technique is applied to N to generate s negative sets $N'_1, ..., N'_s$. Then, the Random Subspace method is applied to P and N', to give feedback sets $P'_1, ..., P'_f$ and $N''_{1,1}, N''_{1,f}, ..., N''_{s,f}$. A number $(s.f)$ of weak estimators, defined as $\{c_{1,1}, ..., c_{s,f}\}$, are trained using $\{P'_i, N''_j, i\}$ as training data for the jth iteration of Asymmetric Bagging and ith iteration of Random Subspace sampling. The results from all estimators s are then aggregated using the Bayes Sum Rule (BSR) [95] to get a final relevance score, by which the results are ordered. BSR biases classifiers according to their relative informational value, and is defined as follows:

$$C^*(x) = argmax_k \left[(1-R)P(y_k) + \sum_{i=1}^{R} P(y_k|c_i) \right] \tag{3.5}$$

where $c_i (1 \leq i \leq s.f)$ is the ith classifier, $P(y_k)$ is the prior probability of the ith class, R is the number of classifiers, and $P(y_k|z_i)$ is defined as:

$$P(y_k|z_i) = 1/\{1 + exp(-|f_i(x)|)\} \tag{3.6}$$

f_i is the output from the ith classifier.

Naive Algorithm (Maximum of Similarities)

A naive algorithm is presented to serve as a comparison to the more sophisticated relevance feedback techniques. Here, the negative feedback is discarded. Then, a database video, the similarity between that video and each of the feedback videos is measured using one of the representation metrics, such as the χ^2 test between histograms. The final similarity that a video is given is the maximum of these similarities. As for the other techniques, the similarities for all database videos are sorted and the top X of these are given to the user as results. In mathematical form:

$$sim(h, pos) = min\{sim(h, p)|p \in pos\} \tag{3.7}$$

where $sim(h, p)$ is the similarity between database video h and positive feedback example p, and pos is the set of all positive feedback.

3.4 Experiments

In this section experiments are detailed using the above techniques, applied to two datasets.

3.4.1 Setup

The UCF YouTube Action dataset [78], the UCF Sports dataset [107] and the Hollywood Human Actions 2 dataset (HOHA2) [60] are used for the experiments, as they contain unconstrained, real footage, therefore providing a good test-bed for real-world use.

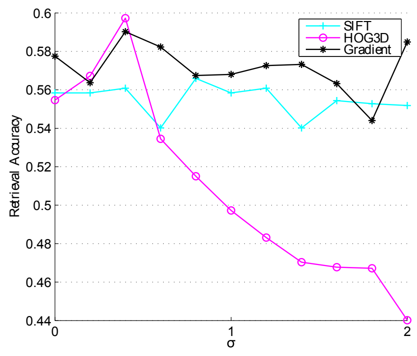
Tests are performed on these datasets with leave-one-out cross validation. 9 iterations of relevance feedback are applied for each search. At maximum, 5 positive and 5 negative feedback samples are taken from the top 20 results on each iteration. To calculate the accuracy of an individual query, let N be the count of all database videos that have the same action category as the query. Then, the first $ceil(\frac{N}{5})$ results are taken, and the percentage of them that belong to the query's action category is calculated.

When performing BoW clustering, first PCA is performed, retaining 95% total variance, and then k-means clustering with $k = 1500$. For the VG pyramids, $k = 10$ is used at each level, and for the ST pyramids, $k = 200$ for the codebook clustering. Both pyramid methods have a total depth of 4, due to time and memory constraints; for each VG pyramid a total of $(10 + 100 + 1000 + 10000) = 11110$ bins were used, and for the ST pyramid, $(1 + 8 + 64 + 512) * 200 = 117000$ bins. For soft-assign clustering (see Equation 3.2) $\sigma = 1.2$. All of the above were empirically determined through experimentation.

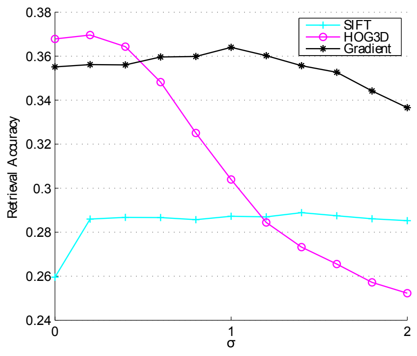
3.4.2 Results

Figure 3.2 shows the effect of soft assignment. For both the Dollar+Gradient and SIFT methods, there is almost no effect on introducing soft-assignment; this is in line with earlier findings. However, with HOG3D, soft assignment significantly negatively affects the results, worsening dramatically with increasing values for σ. This is a surprising result. It is perhaps because the relationship between HOG3D features is too complex to describe as the Euclidean distance between PCA-reduced vectors; while this is purely speculative, further investigations may consider the use of non-linear dimensionality reduction techniques such as LLE [114] in combination with HOG3D.

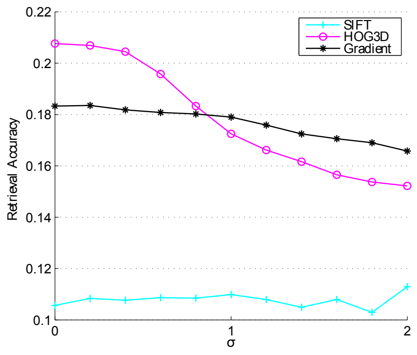
To further the examination of soft-assignment, several iterations of Maximums of Similarity relevance feedback are applied in combination with varying values of σ. As can be seen from Figure 3.3, the results are worse for soft-assignment across all types of descriptor – the descriptor most negatively affected by soft assignment is, as expected, HOG3D.



(a) UCF Sports

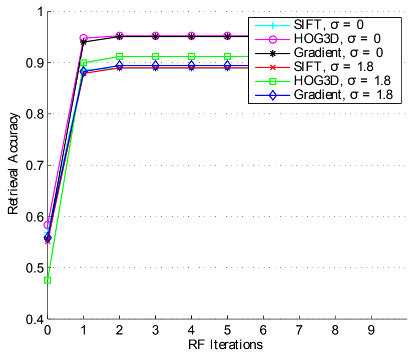


(b) UCF YouTube

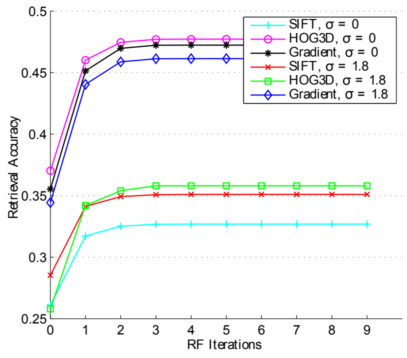


(c) HOHA2

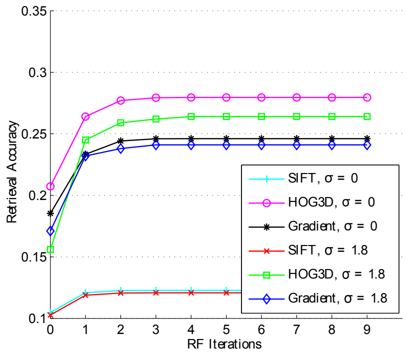
Figure 3.2: How varying levels of soft-assignment affect first-query action retrieval for 3 different methods of feature extraction on each dataset. See §3.2.2 for an explanation of σ.



(a) UCF Sports

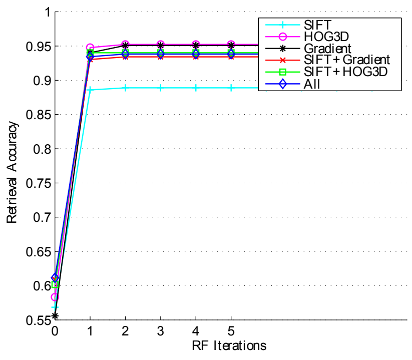


(b) UCF YouTube

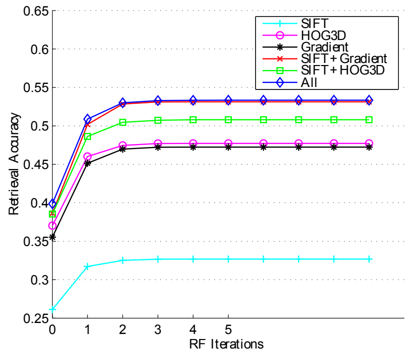


(c) HOHA2

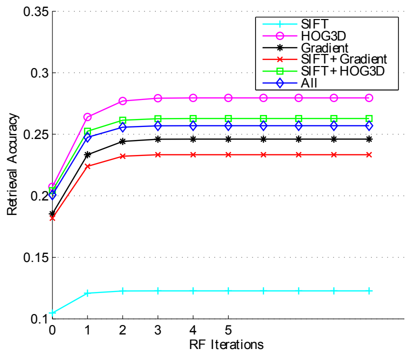
Figure 3.3: Soft and hard assignment after relevance feedback, for 3 methods of feature extraction on each dataset.



(a) UCF Sports



(b) UCF YouTube



(c) HOHA2

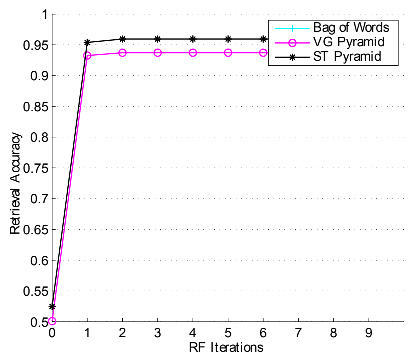
Figure 3.4: Various combinations of feature hybrids for each dataset.

Figure 3.4 demonstrates the effect of hybridising extracted features – that is extracting features using each method, and concatenating the codeword frequency histogram for each feature type together for use in the Bag-of-Words model. As can be seen, the hybrid features have a better response on the first round of relevance feedback, and converge to a higher accuracy. By combining all 3 descriptors, the best accuracy is achieved after 2 rounds of relevance feedback – however, the very small difference gained from the extra computational cost may not, in the vast majority of applications, be a worthwhile tradeoff.

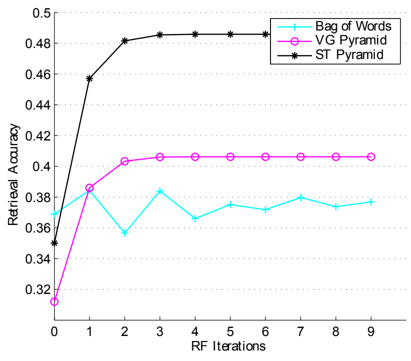
Next, the effect of different representation methods is examined in combination with naive relevance feedback, shown in Figure 3.5. HOG3D is used for all further experiments. As can be seen, the ST pyramid match performs impressively on both datasets; this is attributed in part to the nature of the datasets. Because in many of the action categories, the actions were shot with uniform perspectives and scales, this would particularly benefit the ST pyramid method. It performed with a less significant margin of improvement over the BoW for the UCF YouTube dataset, as this dataset has more variability in the aforementioned aspects.

In none of the datasets did either of the pyramid representation methods significantly outperform the Bag of Words model – indeed, in the UCF YouTube dataset, the VG Pyramid performed significantly worse than either of the alternatives. Because pyramid methods typically outperform the Bag of Words model in fully trained classification, this implies that pyramid methods require much more than one labeled data point in order to be effective. It might be that additional training data is necessary due to the higher level of quantisation present in the VG/ST pyramid approaches in comparison to the BoW model. Also, the great differences between the representation methods in the three datasets show that the VG pyramid match's performance is highly context-specific. Due to this, the use of the VG pyramid match would not generally be recommended.

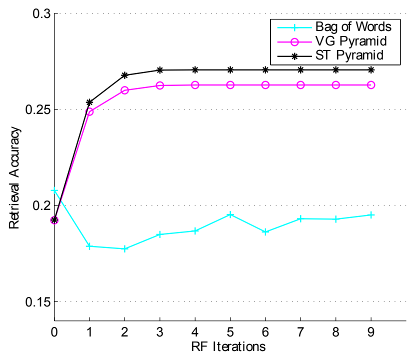
Finally, investigate the different methods of relevance feedback are investigated in Figure 3.6. Due to time constraints, it was not possible to fully investigate using ABRS-SVMs in conjunction with the pyramid methods – the combined complexity of ABRS-SVMs and pyramid matches would result in



(a) UCF Sports

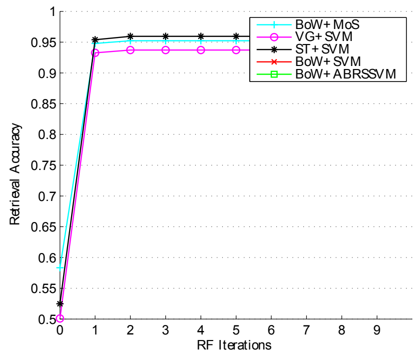


(b) UCF YouTube

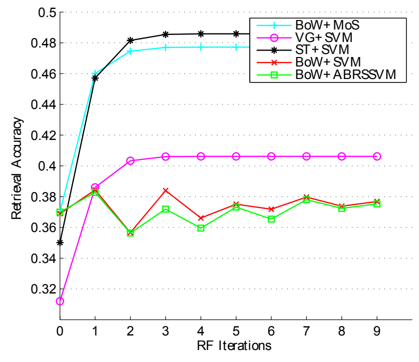


(c) HOHA2

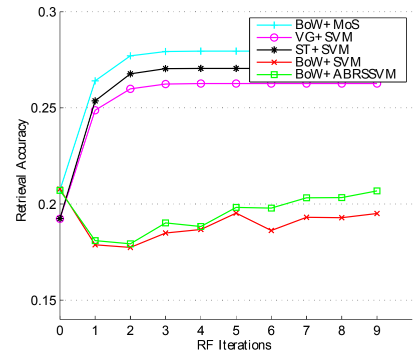
Figure 3.5: Methods of representation, using [1] for the ST pyramid



(a) UCF Sports



(b) UCF YouTube



(c) HOHA2

Figure 3.6: Methods of relevance feedback, where MoS stands for Maximum of Similarities as described in 3.3.1

impractically long calculations. It was however possible to apply these methods with an SVM, as shown.

Surprisingly, the SVMs resulted in a poor, oscillating performance after relevance feedback; clearly the level of feedback (a maximum of 5 positive and negative samples) were not sufficient to model a stable hyperplane, even using the histogram intersection kernel. The ABRS-SVM results in worse performance than the SVMs – indeed, after several rounds of relevance feedback, the results tended to favour simpler systems. Through further experimentation, the oscillating performance in successive rounds of relevance feedback is attributed to limited discriminative power of the BoW representation.

3.5 Discussion

A variety of different approaches to content-based human action retrieval has been presented, including relevance feedback techniques such as SVMs, ABRS-SVMs and the simple Maximum of Similarities; Representation methods were also looked at, including the vocabulary guided pyramid match, the spatio-temporal pyramid match, and the original bag of words. Soft assignment was explored; finally, some of the more popular feature extraction methods were combined for greater accuracy.

In general, experiments showed that simpler methods tend to perform better. The naive RF algorithm – the Maximum of Similarities – proved surprisingly effective. This is at least partly attributable to the low number of feedback samples considered; however, in practical relevance feedback, it is unlikely that the user would be inclined to provide the vast amount of feedback that would benefit an SVM-based approach, especially for video content. Additionally, while the pyramid matches had a marginal advantage over bag of words, it is unclear whether this advantage would be retained in more complex datasets, as they trade invariance for discriminative ability. It is clear that yet further research needs to be performed on human action retrieval in order to significantly surpass the results obtainable by a naive bag of words approach.

Further work on this topic would includes exploring other options for the enhancement of retrieval, such as active learning (such as in Jones et al. [115]) or the incorporation of textual metadata into the algorithms to improve results.

Chapter 4

Human Action Localisation and Temporal Segmentation

4.1 Introduction

With the increased availability of digital video recording technology, more videos are being created than ever before, with these videos coming from diverse domains such as surveillance, amateur film-making, and home recording. These videos contribute to the growth of video media databases, such as those available online to consumers (e.g., YouTube), or CCTV footage collections. From this exponential growth rises a new problem: how can these vast collections of media be accessed in the most effective way, so that users can find what they are looking for?

Currently, video databases such as YouTube employ a text-based search, where videos are returned based on a set of keywords provided by a user. Such a system, however, is flawed; text-searches can search the textual metadata associated with a video (e.g. title, description, keyword tags), but not search the videos directly. The textual metadata is rarely an accurate representation of the video's content, for several reasons: firstly, the textual information is provided by the video's uploader, whose assessment of the video may be flawed/incomplete;

secondly, the amount of information in a video cannot be represented in a few keywords without necessarily losing much potentially salient information. Most importantly, text-based searches typically do not *localise* the action within a longer video sequence. Such localisation is particularly useful, as in a real-world database a video might be quite long, but only a very short section of it relevant to the user's query.

In order to tackle this problem, this thesis chapter looks at and qualitatively compares two alternative approaches to the problem of action localisation within longer sequences to improve data search and access.

The first approach, an exemplar-based human action localisation system, is an extension of Content Based Image Retrieval (CBIR) to the video domain. Given an example video – a query – of what the user is searching for, CBVR directly searches the database's contents, meaning it can potentially return far more accurate results than existing text query systems, as it avoids the above problems associated with poor quality metadata. While CBVR has been well researched through a focus on keyframes and 2D features, only a few previous works have looked at this problem using temporal information, such as Yu et al. [116]. Human actions are the focus of this work, meaning that the vast quantity of prior human action recognition research can be used to inform this work. Human action recognition, a distinct task from retrieval, focuses on using trained models of human actions for classification. Because supervised recognition algorithms require prior knowledge of all the classes that are to classified, they are unsuitable for direct use in retrieval tasks, but many of the unsupervised action representation techniques developed for recognition are also applicable in retrieval. The second approach is an efficient method for the *unsupervised* temporal localisation of all human actions from a single human track – this is also known as temporal *segmentation*, but it is referred to here as localisation to highlight its relevance to the chapter. The algorithm presented is designed in such a way to maximise its efficiency, in order to make it applicable to extensive databases of video footage, such as those found in surveillance databases. The system works roughly as follows: first, an effective human tracker is applied to the human subject, removing any background noise, as well as compensating for translational and scale variations. Secondly, this method for unsupervised segmentation is applied to the human tracks, resulting in a series of sections of self-consistent linear motion.

The rest of this chapter describes these two approaches to human action localisation in more detail.

4.2 Examplar-based Human Action Localisation

The first system presented in this chapter is a Content-based Video Retrieval system. It distinguishes itself from the majority of previous works by not only retrieving relevant human action videos, but also localising the exact relevant part of these videos spatially and temporally. Among other improvements to the state-of-the-art, this work distinguishes itself in two facets: 1. An extremely efficient algorithm is designed for spatio-temporally localising human actions within a dataset using only a single query of the sought-after action – this algorithm is considerably computationally simpler than comparable works for action retrieval with localisation. It could also be extended into a hierarchical model for better-than-linear performance. 2. Relevance feedback is used in the context of localisation, and its efficacy is demonstrated in this application – also considered is how imperfectly localised relevance feedback can be used.

The goal of the first system is defined as follows: given a query video containing a pre-localised human action, the system searches a video database for all instances of this human action. It spatio-temporally localises and ranks these actions according to relevance, before returning them to the user. At this point, the user may mark results for relevance feedback and run the query again iteratively, until he/she is satisfied with the results.

Human actions are focused on for two reasons. Firstly, videos of humans constitute the majority of existing video media, and are therefore highly likely to be the target of a user's query. Secondly, the majority of existing video datasets for recognition and retrieval research are also focused on human actions.

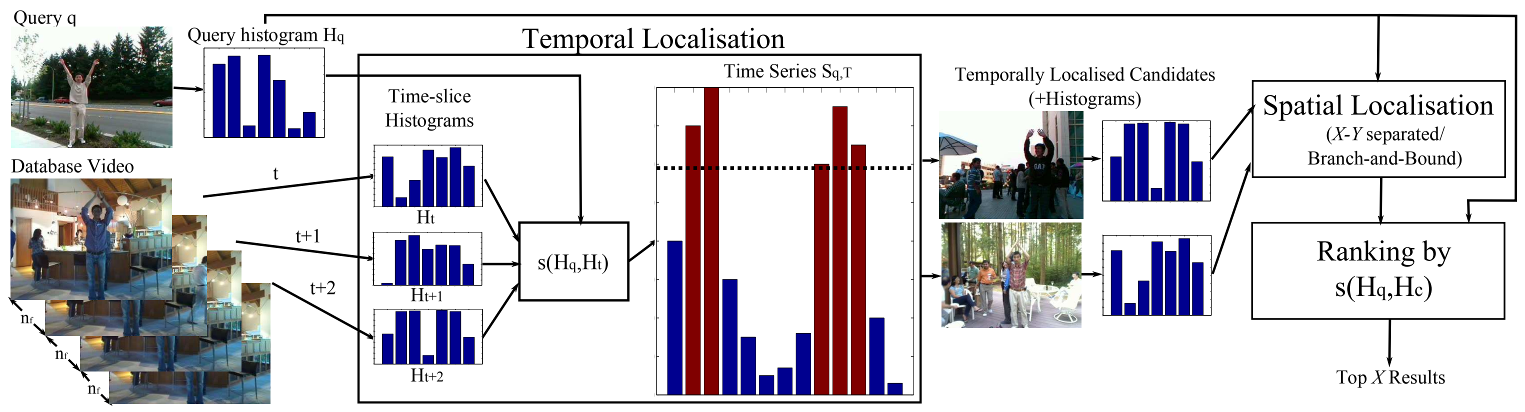


Figure 4.1: An overview of the spatio-temporal localisation and ranking aspects of the algorithm. Note that the spatial localisation part of the algorithm has been simplified here for convenience.

Figure 4.1 gives an overview of the system. The database of videos is pre-processed in batch - local features are extracted and clustered into codewords. When a user provides a query video, several steps are performed in sequence. First, the same feature extraction is applied to this video. Using these features, the system performs temporal localisation to find a large number of candidate results in the database. The candidates are refined by performing a further round of localisation – this time spatially – and then rank the candidates according to a bag-of-words model. The top X results are returned to the user. At this point, the user can choose to provide relevance feedback if necessary, to improve results. Several of the ranked videos are marked as relevant/irrelevant to the query, and this relevance feedback is used in a further search, to improve both the localisation and ranking steps.

Below, the operation of the system is given in detail. Efficiency is treated as the utmost priority, while maintaining practical accuracy; therefore the design is justified in these terms.

4.2.1 Video Representation

As described above, the video database must be pre-processed with feature extraction before a search can be performed. The system achieves this using an existing feature detector and descriptor (such as Dollar's [23], SIFT [21] and HOG3D [27]). The features are extracted at a roughly consistent rate with respect to time. Having extracted features in such a manner across the whole dataset, a combination of PCA (capturing 95% variance) and k-means clustering are applied to the descriptors, retrieving k visual codewords. The choice of k is important, as generally higher k will give better accuracy, but will also slow down retrieval, and too high k will lead to sparsity issues in the localisation algorithm.

Then, each video can be efficiently represented by the set of its features. Each feature can be represented as a tuple, $t = (x, y, t, c)$, where x, y and t represent the spatio-temporal location of the feature within its video, and c is its codeword. The process of feature extraction can often be quite costly – however, in a retrieval model this impact is minimised, as feature extraction is performed just once on the database – for subsequent searches, feature extraction is only performed on the query video.

4.2.2 Localisation

Temporal and spatial localisations are separated into linear time algorithms to decrease the search time of the algorithms. Simultaneous spatio-temporal localisation using branch and bound, such as in [116], has very high computational complexity, especially for lengthy video sequences. This work proposes that such localisation is unnecessarily complex – using local features, temporal localisation can be performed accurately, independent of spatial localisation. It can be observed that, in many multimedia video sequences (as opposed to surveillance footage) a simple human action will occupy a small proportion of the temporal domain, but a relatively large proportion of the spatial domain. Therefore, to perform an efficient search, temporal localisation is first performed to identify a relatively small number of candidate regions with respect to the size of the dataset, before performing a more complex spatial localisation operation on only these candidate regions.

Temporal Localisation

An additional step of pre-processing is performed on the database in order to facilitate fast temporal localisation. The temporal space is divided into slices of f frames, and for each time-slice, a normalised bag-of-words histogram is generated, $H_t \in H_T$. The appropriate choice of f is made empirically, and this choice is important – it should be small enough to account for short actions (approximately half the length of the shortest action in the database), but large enough so that the histograms are not overly sparse. The choice of k (the codebook vocabulary size) and f also presents a trade-off between the time efficiency and accuracy of temporal localisation during a search. In these experiments, f is set to 8, which is approximately half the length of the shortest action in the test datasets.

During a search, features are extracted from the query video and the query is represented by a single normalised bag-of-words histogram, H_q. It is important at this point to note that H_q and H_t are not directly equivalent; H_q represents the complete action, whereas each H_t will at most represent a temporal fraction of the overall action. By using the correct comparison metric, however, it is still possible to compare H_t and H_q to generate a useful value. For this the histogram intersection can be used:

$$s(H_q, H_t) = \sum_{i=1}^{k} min(H_q^i, H_t^i) \tag{4.1}$$

For retrieval tasks, however, a modification of the histogram intersection might be more appropriate:

$$s(H_q, H_t) = \sum_{i=1}^{k} \frac{min(H_q^i, H_t^i)}{H_n^i} \tag{4.2}$$

where H_n is the normalised histogram of the features across the entire database. Using H_n in this way, rarer features in the dataset – those with a presumably higher information value – have a greater weight. A major assumption of this model is that a high value for $s(H_q, H_t)$ is predictive that time-slice t contains part of action q. This is subtly different from the standard procedure, where two full-action histograms are compared. It was experimentally determined that the comparison is indeed suitable for this purpose.

Having calculated $s(H_q, H_t)$ for all $t \in T$, the system then looks for temporally adjacent regions with high values for s, which indicate potential candidate regions within each database video. A threshold value is determined individually for each scene[1] in the database, above which a time-slice is considered to be a match for the query. This threshold is set as a standard deviation above 0, and from this a set of candidate regions is generated.

In order to reduce the noisiness of these regions, two additional operations are performed. Firstly, adjacent regions are joined: if time-slice t is a match, and $t + 2$ is a match, then $t + 1$ will also be considered a match. Then singleton regions are removed: if t and $t + 2$ do not match the query, then $t + 1$ is not

[1]A scene refers here to a single contiguous camera shot.

considered a match either. Similar in concept to region dilation and erosion in 2D image segmentation, these operations significantly reduce the effect of noise (such as partial occlusions) but are relatively cheap to perform. When all these operations have been performed, the final set of candidates, termed C, is passed through to the spatial localisation step.

The complexity of the temporal search over a single video is $O(k\frac{n}{f})$ where n is the number of frames in the video, k is the number of codewords, and f is the size of the time-slice. It is postulated that the efficiency of the presented technique may be improved to a logarithmic time function by performing a coarse-to-fine hierarchical search on large-to-small time-slices, but such considerations are beyond the scope of this work.

Spatial Localisation (SL)

Once temporal localisation has identified a set of candidates, SL is linearly applied to to the temporal candidates – at this point the candidate set has already been pruned to a comparatively small subset of the total database, so spatial localisation need not be as computationally complex as temporal localisation. If maximum efficiency is desired, however, spatial localisation can be performed after ranking, only on the top X results – this results in a SL step which is constant *w.r.t.* the size of the database. For reasons of storage complexity, only minimal pre-processing of the database is performed to prepare for SL. The feature tuples retrieved in the feature extraction pre-processing step are stored in temporal order, so that once the temporal bounds of each candidate are known, the features belonging to each candidate can be swiftly retrieved.

In these experiments two methods of spatial localisation are investigated, to evaluate their accuracy and performance trade-offs. The effects of performing spatial localisation before, *and* after, the ranking step are evaluated. The former is expected to result in greater accuracy, as ranking can be more precise; however, if spatial localisation is performed after ranking, spatial localisation only needs to be performed on the top X results that are to be returned to the user. This may result in a considerable efficiency improvement.

$X - Y$ Separated SL The first approach linearly separates localisation along both spatial dimensions X and Y to minimise computational complexity, and to mitigate local feature sparsity in two dimensions – in the absence of simultaneous identical actions, it is hypothesised that this technique will achieve reasonable accuracy. The temporally localised block is split into equal slices along each dimension in turn, and a histogram H_s is generated for each slice, similar to the procedure used in temporal localisation. Then, Equation 4.2 is used to generate a set of scores r over that dimension.

At this point, the method diverges from temporal localisation, as in spatial localisation only a *single* region is required, whereas in temporal localisation multiple instances of the action are found. To find the optimal sub-region, a threshold $d = l \cdot max(r)$ is first subtracted from each of the scores in r, with l set to 0.25. Then, a maximal sub-array search is performed on r, using Kadane's algorithm [117] to perform this in $O(|r|)$.

Having determined the extent of the action along each dimension separately, this information is simply combined to determine the final bounding box.

Branch-and-Bound Simultaneous SL For potentially greater accuracy, but at the cost of higher computational complexity, it is possible to perform localisation along both spatial dimensions simultaneously. Here, the spatial extent of the candidate is divided into a number of equally sized 2D windows. Using the features, ad hoc histograms for each of these windows are generated for each temporal candidate. Using Equation 4.2, the system establishes for each 2D window whether it matches the query action; the result of this entire operation is a single low-dimensional *relevance image*, r.

Using r_i, a 2D branch-and-bound operation is performed, based on the 2D object localisation algorithm described in Lampert et al. [118], to find the optimal sub-window containing the action. Branch-and-bound, similar to the sliding window approach, is guaranteed to converge to the optimal sub-window, but its average running time is $O(xy)$ rather than $O((xy)^2)$. Similar to $X - Y$ separated localisation, it is necessary to choose a decision threshold d to subtract

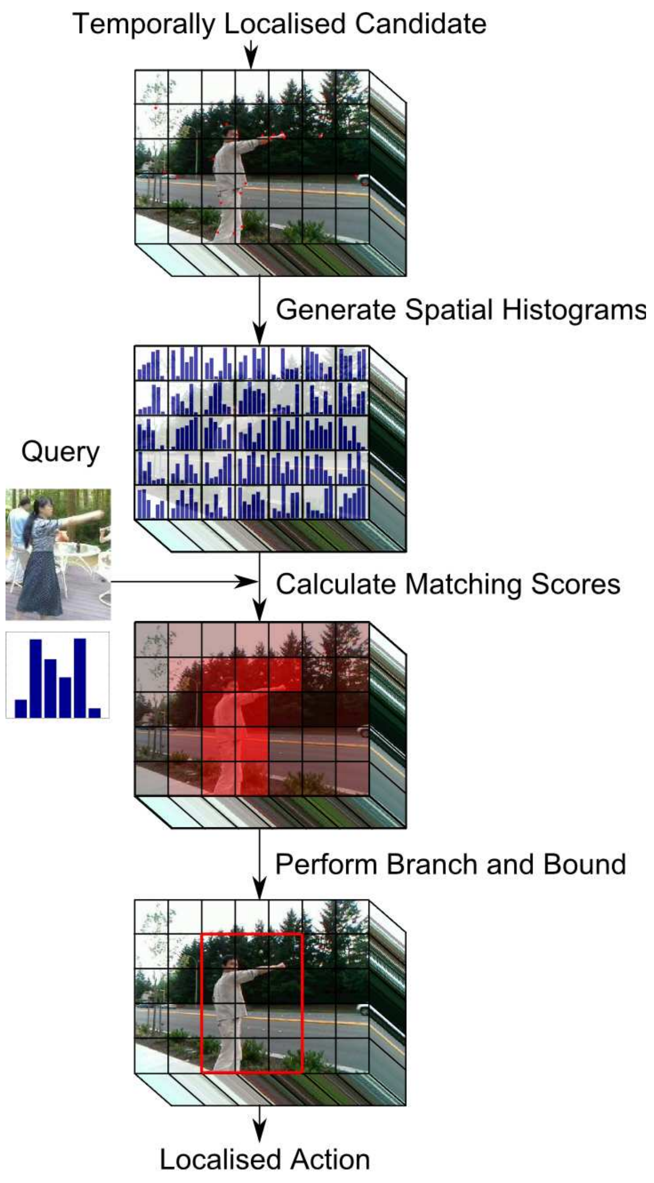

Temporally Localised Candidate
Generate Spatial Histograms
Query
Calculate Matching Scores
Perform Branch and Bound
Localised Action

Figure 4.2: An overview of how the Branch and Bound algorithm is applied for spatial localisation.

from r – branch-and-bound assumes the relevance decision threshold of each window is at 0, but r consists only of positive values. $r - d$ is passed into the branch-and-bound algorithm, and the upper bound function is set to the following:

$$\hat{f}(Y) \equiv f^{+}(y_{\cup}) + f^{-}(y_{\cap}) \tag{4.3}$$

where: $y_{\cup}$ and $y_{\cap}$ are the maximally and minimally sized rectangles within candidate set Y respectively; $f^{+}(y)$ and $f^{-}(y)$ are the sum of all positive points and sum of all negative points in rectangle y respectively. This is the same upper bound described in the Linear Classifiers section of Lampert et al. [118] – as this work details, it is possible to use positive and negative integral images to calculate Eq. 4.3 in constant time.

A diagram of the branch-and-bound spatial localisation method is shown in Figure 4.2.

4.2.3 Ranking

Having performed localisation, there is a set of candidates c_i and their bounding boxes B_{c_i}. A single feature histogram H_{c_i} is generated for each c_i, collating the features that fall within B_{c_i}; H_c for all $c_i \in C$ can then be matched against H_q using Equation 4.2. The set of scores generated by this operation provide a simple basis for ranking the localised candidates; those with higher scores are ranked first. The top X ranked candidates are returned to the user as results. The generation of H_{c_i} can be made computationally simpler using integral histograms, as seen in Ning et al.[119], though this speed-up is relatively insignificant compared to the extra storage required.

In previous systems, ranking and localisation occur simultaneously – such as in top X branch-and-bound localisation. The system presented in this chapter distinguishes itself. Because of the inclusion of this discriminative ranking step, the localisation is highly permissive – or, in other words, insensitive to false positives. The localisation generates a large number of candidates, not all of which will be matches for the query, with the expectation that most false candidates will be pruned at the ranking stage. Furthermore, good ranking should generally favour better-localised candidates, meaning that to an extent ranking can compensate for a weaker localisation step. This has allowed us to keep the localisation step computationally very simple, while still returning strong results to the user.

4.2.4 Relevance Feedback

While the process described above delivers useful results, they can be further enhanced through relevance feedback. If a user's initial search has completed, but that user is still unsatisfied with the results of the query, he/she can provide feedback about the relevance of each result to his/her search. Using a model to integrate this additional information with the original query, a more discriminative second search can be performed – this model is usually an online learning technique such as an SVM or AdaBoost.

As the goal of the system is efficiency, various relevance feedback methods are evaluated through their relative time-cost effectiveness, and settled on two effective techniques. For determining the relevance of time-slices during temporal localisation, an SVM is applied with a histogram intersection kernel (which satisfies Mercer's condition). To update the ranking of candidates, simple query expansion is applied from positive feedback only.

Also considered is applying SVM relevance feedback to the spatial localisation step of the algorithm; however, due to complexity-constraints and minimal performance improvements in the preliminary experiments, this is not reported on in the results section below.

Finally, as the system performs localisation, it is necessary to make a novel consideration related to relevance feedback. In prior retrieval systems with RF, it is straightforward for a user to mark the results for feedback, as each returned document will have a binary relevance value – in other words, it is either relevant or not. However, with localisation many of the returned results will be *partially relevant*, because many of the ostensibly relevant results will be imperfectly localised. To deal with this, it is assumed first that a user will only view a result as relevant if its bounding box overlaps sufficiently with the desired action. Once a user has decided that a result is relevant, he/she can then return it as feedback in one of two ways: 1. Adjusted: the result is modified by the user to overlap perfectly with the action, and is then returned to the system as feedback. 2. Unchanged: the result is returned as feedback with no modification to the bounding box.

If a user returns adjusted feedback, results on the next iteration should be more accurate than if he/she provides unchanged feedback. However, correcting the bounding box for adjusted feedback places a considerable onus on the user, relative to simply marking results as relevant or not. In the experiments below, both methods of returning feedback are considered to evaluate which is more practical.

4.2.5 Experiments

Setup

The MSR2 [109] and UT-interaction [111] datasets are used to show the results. These datasets are specifically designed for spatio-temporal localisation experiments, which make them well-suited for our purposes.

Each dataset was first pre-processed as follows. The datasets were scaled uniformly to 240 pixels in height (maintaining aspect ratio) and 15 frames per second, so the feature extraction procedure was identical for both. Features were extracted from each dataset at an average rate of 180 features per second, detecting features with multi-scale Dollar [23] and describing them with HOG3D [27]. The resulting features were clustered into 1000 codewords after PCA was performed to capture 95% of the features' variance. Time-slice histograms were generated over the whole dataset in batch before the main retrieval experiments; as these pre-processing steps can be performed before a retrieval search is performed, they are not included in the performance statistics. Each time slice was 10 frames in length, and the 2D spatial grid was divided into 10 by 10 pixel blocks. These parameters were chosen based on observations of the minimum length and size of the actions within the dataset.

Leave-one-out cross validation retrieval experiments were performed on each dataset in order to provide the most reliable results. Each action $a_i \in D$ was treated as the query in turn, where D is the entire dataset. The search for action a_i was performed on a subset of D, $D - v_i$, where v_i is the discrete video sequence from which a_i was extracted. $D - v_i$ is used rather than $D - a_i$, as results may be skewed in favour of other actions in the same video sequence. The results over each individual query were averaged to get the final results.

Relevance feedback, rather than being given by a real user, was simulated for experimental consistency and convenience. It was assessed whether a result would be deemed as relevant by the virtual user using the following metric:

$$L(E, G) = \frac{volume(E \cap G)}{volume(E \cup G)} \tag{4.4}$$

where E is the spatio-temporal bounding box of the estimated action, and G is the bounding box of the closest relevant action (taken from the ground truth). A result was deemed to be relevant when $L(E, G) > thres$. $thres$ was chosen to allow for an average overlap of 0.5 per localised dimension – 0.5 for temporal-only localisation, and 0.5^3 for spatio-temporal localisation – following the example set by previous works such as [116].

Typically, the performance of a retrieval algorithm can be assessed through precision/recall and top X results – however, the formulation of these metrics assume that each result has a binary relevance to the query. In the model presented here, an imperfectly localised result may have partial relevance to the query, as measured by its overlap with a relevant action. Equation 4.4was used to calculate relevance in the results.

During relevance feedback, a maximum of 5 positive and 5 negative feedback samples were given at each iteration. 5 iterations of relevance feedback were simulated, after which no further significant improvement was observed in any of the experiments.

Results

Below the results are shown and their implications are discussed.

It is clear that there is a significant difference between performance on the UT and MSR2 datasets. The significantly poorer performance on the UT dataset is ascribed primarily to: a greater number of action classes, increasing the chance of false positives and fewer examples per action class, resulting in a lower percentage of true positives. It is worth noting that relevance feedback makes a considerable impact on the performance here too, suggesting that there may be greater intraclass variability in the UT dataset compared to the MSR2 dataset.

The contribution of various methods of relevance feedback can be seen in Figures 4.3a and 4.3b. On the MSR2 dataset, by the fifth iteration, it matters little to the results whether a user returns adjusted or unchanged feedback – however on the UT dataset, adjusted feedback is considerably better than unchanged feedback. This would confirm the natural intuition that more difficult search queries require higher quality feedback samples. The results when feedback is applied to only: 1) the temporal localisation step, and 2) the ranking step, of the algorithm are also shown in these two figures. Feedback improves the accuracy of the localisation step more, but both steps work synergistically to achieve the highest performance.

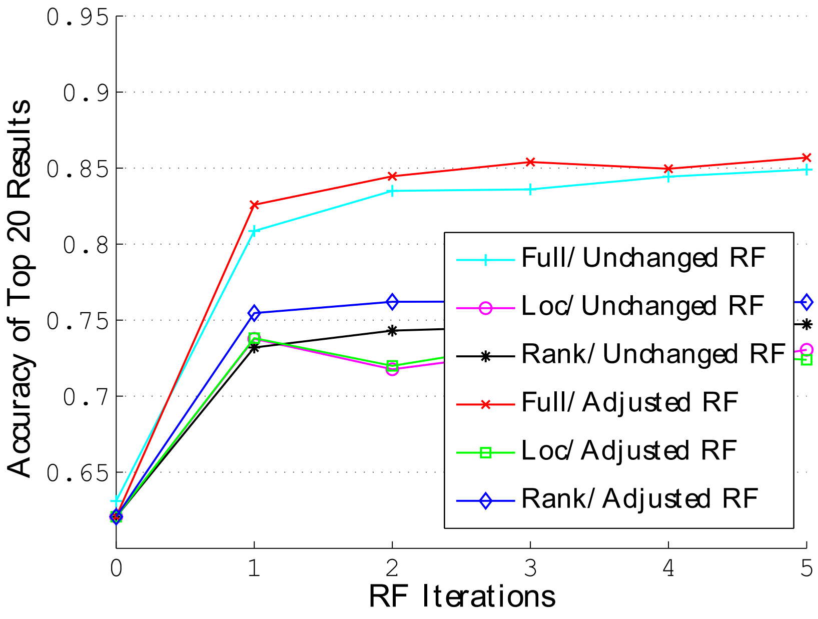
Figure 4.4 shows the performance of the modified histogram intersection (see Equation 4.2) against the ordinary histogram intersection implemented in the kernel SVM, on the MSR2 dataset. After relevance feedback, the modified intersection performs consistently better by around 1%.

Figures 4.5a and 4.5b give precision/recall curves for various levels of relevance feedback. Precision is determined as the ratio of true positives to the total number of retrieved samples. Recall is determined as the ratio of true positives to the total number of possible true positives in the whole dataset. For both datasets and all iterations, at low recall, precision is relatively high, but at higher levels of recall – beyond 10% – performance rapidly tails off. This indicates the difficulty of learning a human action from a single example. However, relevance feedback considerably improves the situation, and the tables furthermore show that only one or two iterations of relevance feedback are required to reach optimal performance. Improvements after this are negligible.

Figures 4.6a and 4.6b show the effect on accuracy of using branch-and-bound localisation, $X - Y$ separated localisation, and temporal-only localisation – for the spatial localisation methods, this is also broken down by whether spatial localisation was performed before or after ranking. Several results are clear from this. Firstly, spatial localisation appears to be a relatively trivial task – there is no significant difference in accuracy between the temporal-only and spatial localisation methods. Secondly, branch-and-bound performs considerably better than $X - Y$ separated localisation, as expected. Finally, performing spatial localisation after ranking has a small but significant impact on accuracy. Note that for the MSR2 dataset the top 20 results are considered, whereas for the UT dataset only the top 10 results are studied – retrieval results can be distorted by size of the dataset and the number of action classes, so this was partially compensated for.

In Tables 4.1 and 4.2 are shown the running times of an individual query, both before and after relevance feedback, for various methods of localisation. These are an order of magnitude better than the next best attempt to perform a search on the MSR2 dataset [116], which takes 26.7 seconds for a single query. This highlights the advantages of separating ranking, temporal localisation and spatial localisation into discrete steps. The branch-and-bound spatial localisation method was compared to $X - Y$ separated localisation, showing that the former, as expected, is much slower than the latter. This impact on runtime can be mitigated, however, by performing spatial localisation only after the ranking step, on the top X results – in Tables 4.1 and 4.2 run-times are shown both for spatial localisation performed before and after ranking, denoted in the "Loc. Order" column.

(a) MSR2



(b) UT

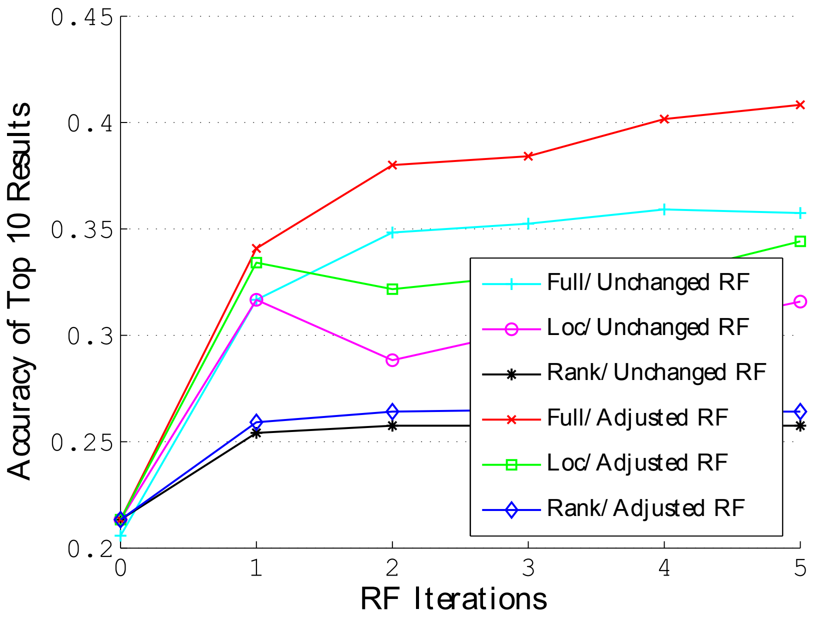


Figure 4.3: Comparison of the contribution of various relevance feedback methods.

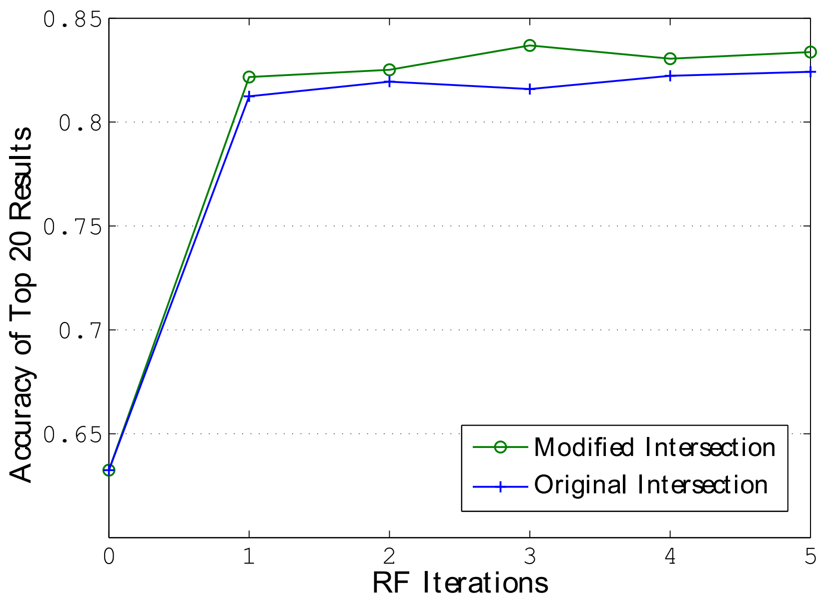


Figure 4.4: MSR2: The modified histogram intersection against the original.

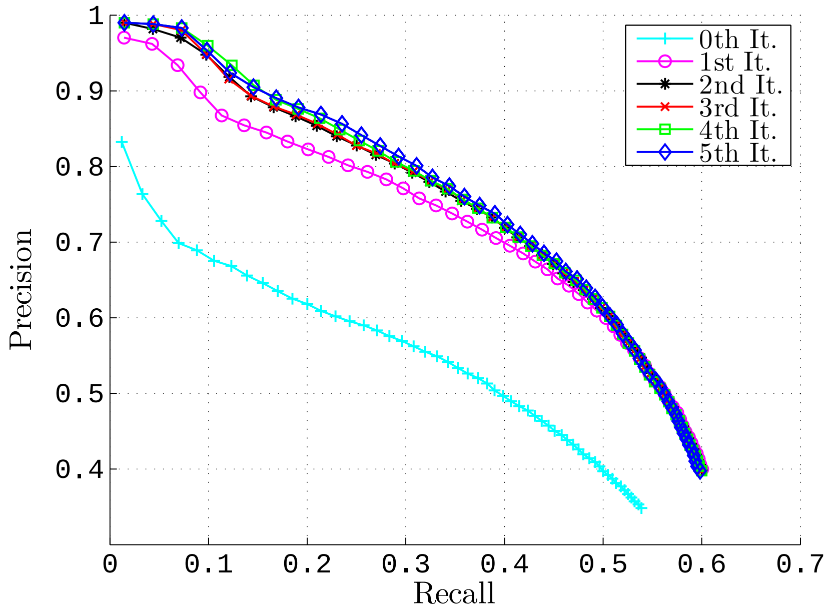
Loc. Met.	Loc Order	Time (s) (1st it)	Time (s) (RF)
Temp. Only	N/A	0.1774	1.283
Linear	Before	0.502	1.392
Linear	After	0.223	1.363
B & B	Before	0.725	1.446
B & B	After	0.247	1.304

Table 4.1: MSR2 query time costs

Loc. Met.	Loc. Order	Time (s) (1st it)	Time (s) (RF)
Temp. Only	N/A	0.099	0.635
Linear	Before	0.281	0.810
Linear	After	0.195	0.750
B & B	Before	0.906	1.303
B & B	After	0.204	0.693

Table 4.2: UT query time costs

(a) MSR2



(b) UT

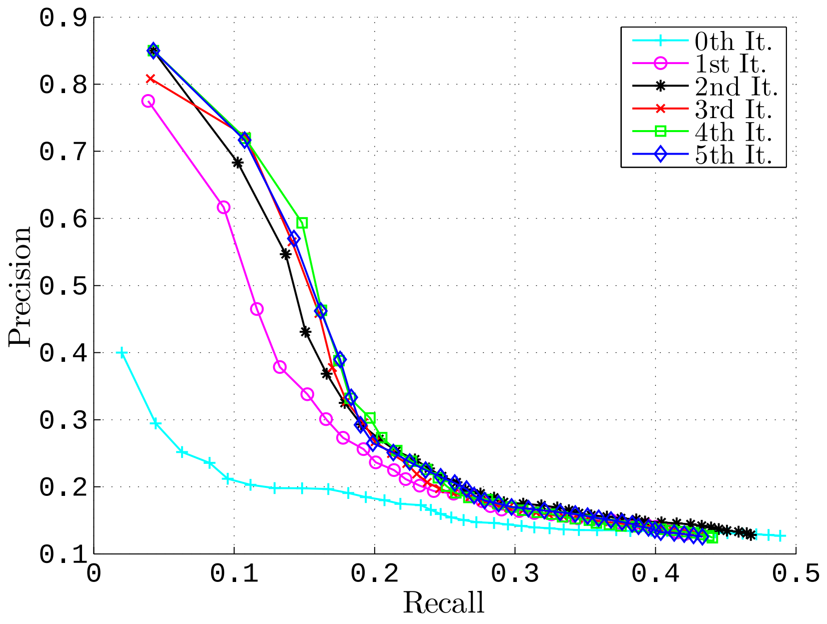
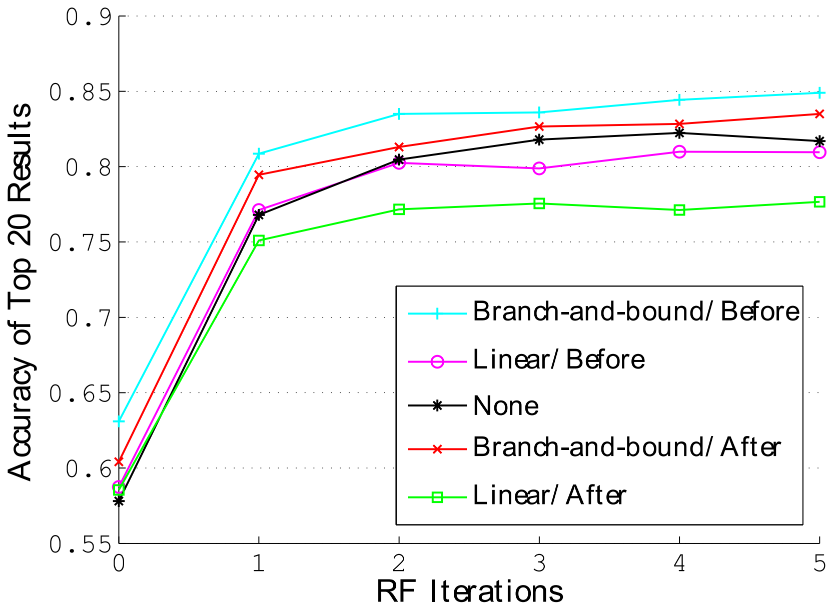


Figure 4.5: Precision/Recall after different levels of relevance feedback.

(a) MSR2



(b) UT

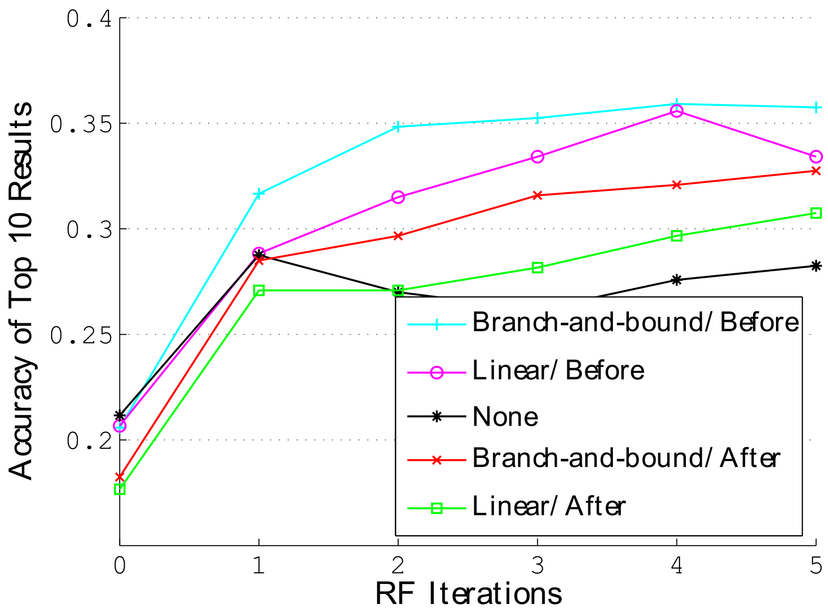


Figure 4.6: Effect on the accuracy of various spatial localisation methods, as well as temporal localisation alone.

4.3 Unsupervised temporal segmentation of human actions

In the second approach to human action localisation, an unsupervised temporal segmentation method is developed that, due to its efficiency, could have varied applications in action recognition, action localisation and action clustering – it is proposed that this method is particularly applicable to surveillance videos. First, an effective human tracker is applied to the human subject, removing any background noise, as well as compensating for translational and scale variations. Secondly, this method for unsupervised segmentation is applied to the human tracks, resulting in a series of sections of self-consistent linear motion.

The primary application of unsupervised temporal localisation of human actions is perhaps temporal *clustering* of human actions, such as in Turaga et al.[83], or as a preliminary step in a human activity retrieval system, such as that proposed in chapter 1. However, to prove the effectiveness of the segmentation method presented in this chapter more effectively, it is integrated into an action recognition system. It is proposed that short temporal segments can be formulated into more effective features for action recognition than the full action, either by using them as independent features in a model such as Bag-of-Words or Naive Bayes Nearest Neighbour (NBNN), or by treating them as states in a time-sequence model such as Hidden Markov Models. This hypothesis is proven through experimentation.

The second system presented in this chapter consists of several parts. Firstly, a poselet-based tracking method is applied to spatio-temporally localise human tracks within the video. Each human track is then split into sub-tracks at points of discontinuity in linear motion, and each sub-track is represented using a descriptor based on a modified Motion History Image. Finally each action is classified using these descriptors with Naive Bayes Nearest Neighbour. Each of these processes is described in detail below. A system diagram is shown in Figure 4.7.

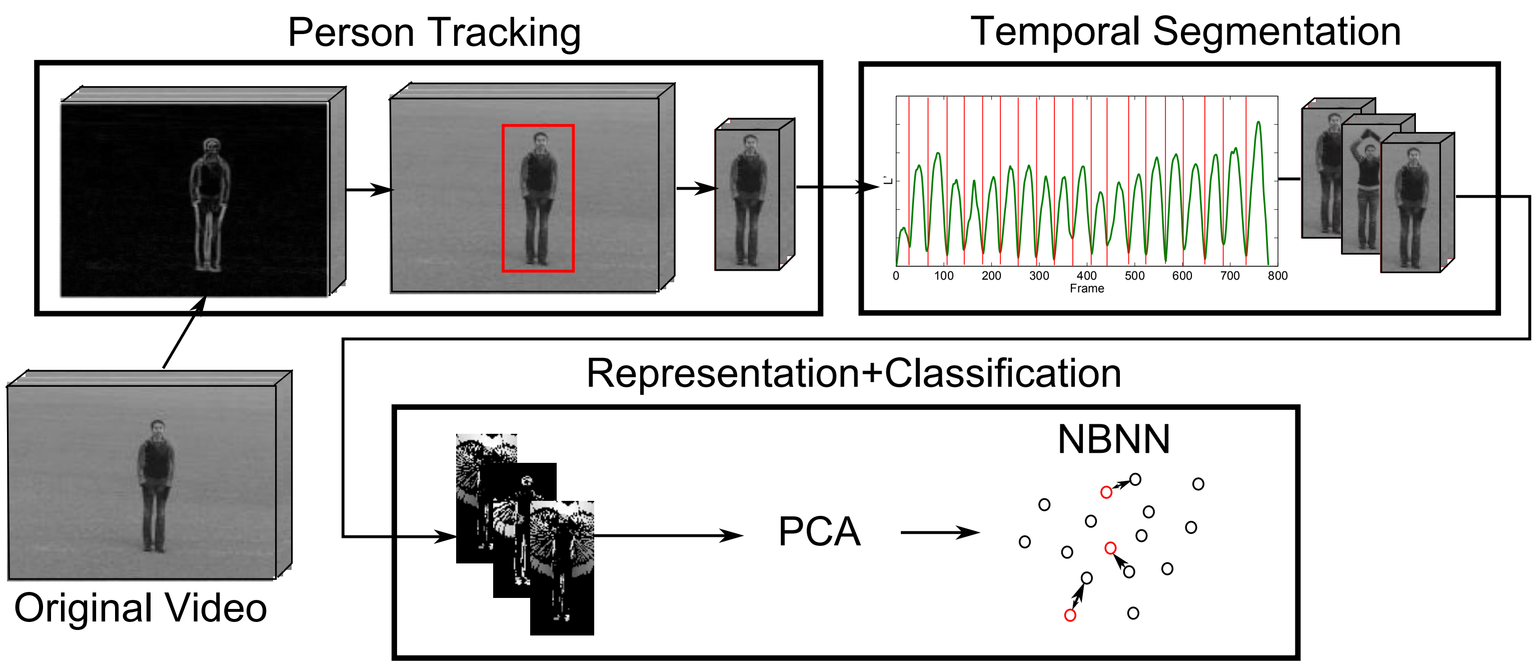


Figure 4.7: A system diagram capturing the main stages of the action recognition system.

4.3.1 Person Tracking

One of the biggest prerequisites for accurate temporal segmentation of human actions is to accurately spatially isolate the human from its surroundings. New and effective person detection methods, and tracking works such as [77] and [120], mean that it is increasingly possible to get a tight, predictable bounding box around a person in almost every frame, spatially isolating it from the rest of the scene, and allowing almost pixel-perfect alignment of the person between frames. Such tracking can mitigate background noise, translation and scale variations, as well as camera motion – this facilitates temporal segmentation based purely on the motion of the human.

For this work, a simple tracking method is implemented based on poselet detection. Poselets are a relatively recent innovation that have so far been applied largely to person detection in images, and have proven quite reliable in a variety of conditions. Here, they are applied in the context of person tracking.

The localisation method is simple. The grayscale video is first pre-processed with a 2D Gaussian filter to reduce noise at every frame. Then, a horizontal and vertical Prewitt filter is applied on every frame, summing the filtered images together to get an edge intensity image. It was found experimentally that this pre-processing improves the reliability of poselet detection, particularly in low-contrast videos. Poselet detection is then applied at every frame, using the code provided online [45].

After poselet detection, several post-processing steps are performed to establish a smooth human track. All detected poselets are thresholded by their score to remove inaccurate detections – the threshold τ is static and set empirically. As multiple actors may be present in a scene, different bounding boxes in separate frames are associated into proto-tracks using the KLT point-tracks method described in Everingham et al. [121]. Certain proto-tracks may have missed detections in some frames. This is managed using interpolation. If there is a detection gap of f frames or fewer between 2 detections, the x, y, width, and height values of the bounding boxes are linearly interpolated separately to estimate the person's position in the missed frames. If there is a gap longer than

f frames, it is assumed that the person is out-of-shot or otherwise occluded, and do not interpolate. Any very short tracks of 5 frames or fewer are discarded, as short tracks are more likely to be noisy or contain no useful motion information. Finally, the width and height of the track are increased by 20% in every frame, to ensure that the whole person is captured – this is to compensate for the poselet detector potentially missing an out-stretched arm or the feet of the actor. While this method is simple, it is found to be effective for the datasets in the below experiments. For more realistic datasets, a different solution might be considered.

4.3.2 Motion Segmentation

It is now necessary to break the human track up into temporal segments, each roughly corresponding to a single linear motion performed by the actor. In an action recognition setting, it is proposed that short temporal segments will be more rate and view invariant than complete actions, and therefore, combined with an appropriate classifier (such as AdaBoost or NBNN) they may give higher accuracy. It is also proposed that linear motion segmentation will be particularly effective in classifying repetitive or cyclical actions – the action will be split into temporal segments at the same points in the cycle, ensuring temporal alignment between each segment.

The first step to find discontinuities in linear motion is to find the most significant motion gradient of the human track. Given a human track h of static height and width, a series of difference images d is extracted: $d_i = h_{i+1} - h_i$. The optical flow is extracted from d in the x and y directions between every pair of adjacent frames, to get $optfx$ and $optfy$. To calculate $optfx$ and $optfy$ the Lucas-Kanade algorithm [122] is applied, as it provided the best efficiency and accuracy trade-off in preliminary tests. The pixels of each frame of the track are concatenated into a time-series of 1D vectors, and Principal Component Analysis (PCA) is performed on these vectors, discarding all but the first 2 principal components, resulting in $P = \{p_{1,i}p_{2,i}; i = 1, ..., t\}$, where t is the number of frames in the optical flow. It is proposed that P corresponds to the most significant motions in the video.

The next step is to look for linear discontinuities in time-series P. The R^2 statistic is a measure of how well a regression model matches the observed data, so it is possible to get a measure of linearity at every point P_m in P by calculating the R^2 statistic of a simple linear regression model on a temporal window of P, centred on P_m. In this model, $P_i, i = m - w, ..., m + w$ are the data points of the regressors p_1 and p_2, and $y = m - w, ..., m + w$ is the dependent variable, where w is a parameter defining the size of the temporal window. Let:

$$\mathbf{X} = \begin{pmatrix} 1 & p_{1,m-w} & p_{2,m-w} \\ 1 & p_{1,m-w+1} & p_{2,m-w+1} \\ \vdots & \vdots & \vdots \\ 1 & p_{1,m+w} & p_{2,m+w} \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} m-w \\ m-w+1 \\ \vdots \\ m+w \end{pmatrix} \tag{4.5}$$

Then orthogonal decomposition is applied to X in the normal fashion to get the regression co-efficients β and the predicted values f_i:

$$\mathbf{QR} = \mathbf{X}, \quad \beta = \mathbf{R}^{-1}(\mathbf{Q}^T\mathbf{y}), \quad \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{pmatrix} = \mathbf{X}\beta \tag{4.6}$$

From the observed values y_i and predicted values f_i the R^2 statistic is calculated at each time m to give a measure of how well the linear model matches the data at every data point. A higher value for R^2 corresponds to greater local linearity:

$$R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \overline{y})^2} \tag{4.7}$$

where $\overline{y}$ is the mean of the observed data. Then a degree-of-linearity time series can be obtained: $L = L_{m,w}; m = 1, ..., t$ where $L_{m,w}$ is the R^2 statistic for a linear regression around every time point m, with window size w. However, L is not sufficient for accurate temporal segmentation, as in practice the R^2 statistic is highly dependent on w. If w is too small, then L will be noisy – a single outlier could have a strong local effect. If w is too large, however, rapid changes in linear motion will be averaged out and fast actions will not be segmented correctly. One potential solution is to calculate L for every point m for the range of values of w, which is termed W, and get an average for M over all W:

$$L' = \left\{ \sum^{w \in W} sL_{m,w};\ m = 1, ..., t, s \propto w \right\} \tag{4.8}$$

Weights proportional to w are used, rather than a simple mean. This is to offset the following: as w grows larger, the average value of M tends to decrease. If a simple mean is used, therefore, larger values of w will be under-represented in L'. In the experiments $W = [2, 20]$. When the temporal window overlaps the start or the end of the track, a symmetric, mirror-reflected border is applied to get the missing values.

L' can be used directly to perform motion segmentation. The local minima in L' correspond to the break points in linear motion. Local maxima in L', alternatively, correspond roughly to the central frames of the linear motions. The effectiveness of this method can be seen in Figure 4.8, which shows the graph of L' as applied to two simple cyclical actions from the KTH dataset, handwaving and boxing. The boxing example consists of many fast, short, linear actions, with abrupt changes, whereas the handwaving example has slower, longer actions. These patterns are reflected clearly in the sinusoidal patterns of the L' graphs.

Segmentation can be performed in two ways to get overlapping temporal segments: 1) segment between the local minima of L'. In this manner, each segment contains a single linear motion. 2) Segment between the local maxima of L', so each segment contains a transition from one linear motion to another. These two groups of segments are termed S_l and S_t respectively. S_l and S_t are combined in the experiments, as both together perform better than either does alone – together, they capture more of the structure of the action.

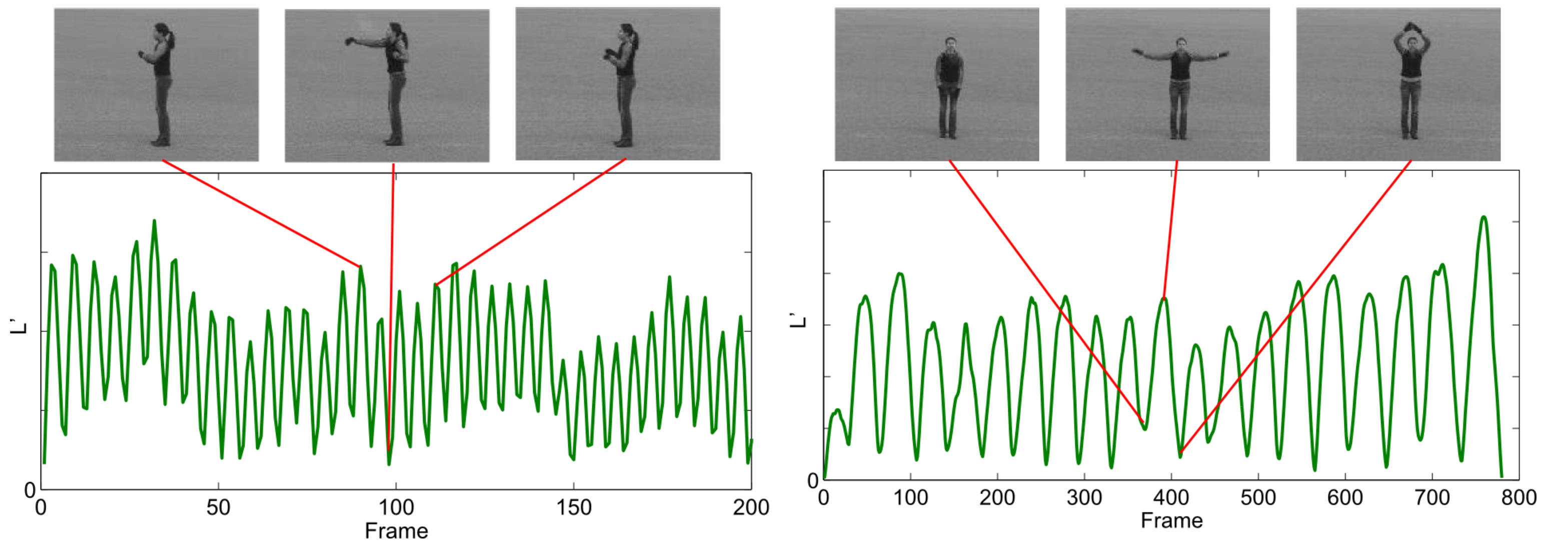


Figure 4.8: L' applied to two KTH dataset videos, one for boxing and one for handwaving. The peaks of the graph occur in the middle of a linear motion. The valleys occur as the actors' arms change direction.

4.3.3 Representation/Recognition

The primary application of unsupervised temporal segmentation is perhaps temporal *clustering* of human actions, such as in Turaga et al.[83], or as a preliminary step in a human activity retrieval system, such as that proposed in chapter 1. However, to prove the effectiveness of the segmentation method presented in this chapter more effectively, it is integrated into an action recognition system. It is proposed that short temporal segments can be formulated into more effective features for action recognition than the full action, either by using them as independent features in a model such as Bag-of-Words or Naive Bayes Nearest Neighbour (NBNN), or by treating them as states in a time-sequence model such as Hidden Markov Models.

This section details the representation chosen for the spatio-temporally localised motion segments in order to perform classification. A variation of the Motion History Image (MHI) is used for this purpose. The MHI is particularly compatible with this method for two reasons. First, the accurate spatial localisation given by the poselet person tracker removes all scale and translation variations from a person's motion that could potentially distort an MHI. Second, temporal segmentation mitigates spatially overlapping motions. In a full action – particularly cyclical or repetitive actions – motions may spatially overlap, but MHIs can only record the most recent motion in a pixel. If the MHI captures the full action, therefore, it will lose potentially salient information. However, as each temporal segment in this model only has a short linear motion, motion overlap within a segment is minimised, making the MHI a good choice for representation.

The variation of the MHI presented in this work is simple. Given a temporal segment v, each frame is scaled to 100x100, so every MHI will be the same size. First, a difference video is calculated: $v'_i = |v_{i+1} - vi|$. All of the pixels of v' are thresholded to get a binary video v'' – 1 is set for pixels above the threshold, 0 is set for pixels below the threshold. The threshold is set dynamically for each v' to one standard deviation above the mean pixel value. From this binary video, the MHI is calculated as per [23]. Each MHI is reduced by PCA to an N-dimensional vector, and these vectors are then used as features for classification.

At this stage, there are several possible models to use for classification, including Bag of Words for efficiency, or Hidden Markov Models to capture the temporal structure between the MHI features. In the experiments, classification is performed using NBNN due to its superior performance compared to the traditional Bag of Words [123], and its relative simplicity. Further work could investigate the use of temporal representation models in conjunction with L' segmentation.

4.3.4 Experiments

In this section the setup and results of the experiments are detailed. The goal of the experiments is to validate that the novel temporal segmentation algorithm presented here can split an action in semantically meaningful temporal segments. This is shown by demonstrating that the temporal segments generated by this work can be used to perform accurate recognition – however, more complex datasets such as the HMDB [16] or UCF YouTube [78] are not considered – these contain unusual body poses unlikely to be detected by the human tracker, and may be too noisy for the temporal segmentation to work well. It is proposed that this temporal segmentation method will find the most practical use in the unsupervised analysis of humans in surveillance videos. With this in mind, the system is applied to the KTH [2] and MSR2 [109] datasets, which, like in surveillance videos, include a static viewpoint and upright body poses.

Setup

For each experiment, each video in the datasets is first converted to grayscale, then person tracking is performed as described above. The poselet code is taken from [23], and is run using the default parameters. Temporal segmentation is performed using a variety of methods for comparison as described in §4.3.4. Each temporal segment is represented using the enhanced MHI and classification is performed using NBNN. The experimental machine has two Intel Xeon E5630s, 24GB of memory, and runs 64-bit Ubuntu. All code is written in MATLAB for simplicity.

Results

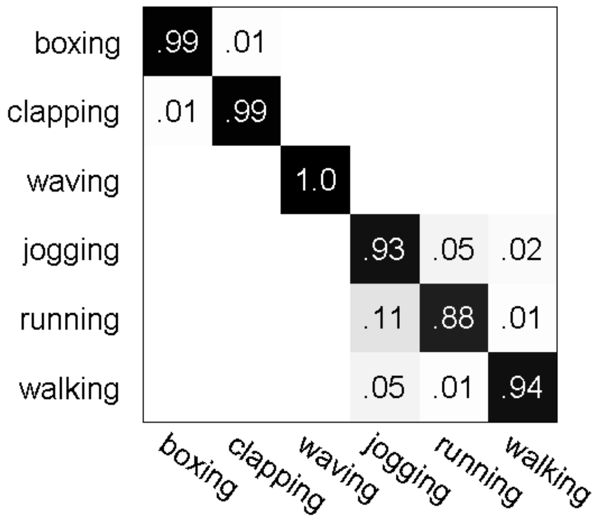


Figure 4.9: KTH dataset: action class confusion matrix

Method	Accuracy (%)
L' Seg.	**95.5**
No Seg.	89.7
Uni Seg. ($x = 10$)	92.2
Dollar et al. [23]	81.2
Zhen et al. [79]	93.3
Liu and Shah. [124]	94.2

Table 4.3: KTH dataset: comparison of various methods

The results for various methods of temporal segmentation for the KTH dataset are shown in Table 4.3. Considered methods for temporal segmentation are: L' based segmentation, uniform segmentation (each track is divided into sections of x frames, where x is chosen to optimise results) and no segmentation. It is clear from this that L' based segmentation gives a better result than any other considered temporal segmentation methods. The complete classification method presented here is also compared with several other recent works, showing that this method rivals the state-of-the-art. Figure 4.8 shows a confusion matrix of the actions - it can be seen here that the most confused actions are *jogging* and *running*, and to a lesser extent, *walking*. This is expected, as these actions appear very similar and in certain cases are difficult even for humans to distinguish.

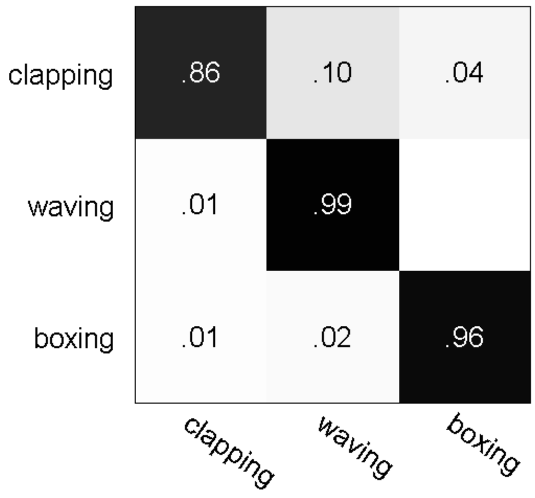


Figure 4.10: MSR2 dataset: action class confusion matrix

Method	Accuracy (%)
L' Seg.	**94.6**
No Seg.	75.4
Uni Seg. ($x = 3$)	92.1
Uni Seg. ($x = 6$)	89.7
Uni Seg. ($x = 9$)	87.2
Uni Seg. ($x = 12$)	86.2

Table 4.4: MSR2 dataset: comparison of various methods

In Table 4.4 results are shown for the MSR2 dataset, for L' based segmentation, uniform segmentation (for varying x) and no segmentation. It can be seen that the results here are similar. While it is a more challenging dataset, the temporal segmentation algorithm presented in this work gives a clear advantage over no temporal segmentation or uniform temporal segmentation. The confusion matrix is shown in Figure 4.10.

4.4 Discussion

In this chapter two approaches to human action localisation have been examined. The first approach is an exemplar-based system for finding queries in a temporally unsegmented video database, and the second is a method for temporally segmenting a video database in an unsupervised fashion.

We first address the finding of the first model. Despite the moderate success of the first model, itis only an initial example of how content-based searches can be tackled from an efficiency perspective. It is proposed that general principles of the system, such as batch pre-processing, spatio-temporal feature binning and dimensionally-sequential localisation can be combined with a wide variety of existing human action recognition/localisation techniques, and that concentrated efforts in investigating these techniques may yield further improved performance. Additionally, while the algorithm is fast enough for use on databases of even thousands of hours in length, it would not work well with online databases of millions or billions of hours – future research should concentrate on how to represent visual features so that videos of length t can be searched in better than linear time. One potential method for this includes extending this work into a temporally hierarchical model, using decreasing values of f to perform a coarse-to-fine search through the dataset, and performing indexing on the histograms at the coarser levels, potentially giving a logarithmic or constant time complexity.

In experimentation on the second model, the L' metric has been shown to be effective in splitting an action up at consistent points, and the motion segments have been used in combination with an MHI+NBNN classifier to achieve superior action recognition performance. In the future, the efficient and accurate temporal segmentation offered by L' can be applied in combination with an action localisation system, rather than used to augment an action recognition system. Splitting human tracks into linear motions may also reduce the complexity of localisation methods, as detections can be made at the motion-level rather than the frame-level. One significant area for improvement in future work would be to improve the L' metric's applicability to noisier multimedia datasets. Although it is envisaged that this system, in its current form, can already be applied to real world surveillance videos, the reliance on accurate person tracking makes the system less suitable for multimedia videos such as those found on YouTube. However, as person trackers/spatial segmentation methods grow more robust, it may be possible to extend or modify the method to work on noisier data.

Efficient localisation systems, such as the two systems presented in this chapter, are becoming increasingly relevant in today's world, as sophisticated searches are increasingly necessary to navigate the huge amounts of data. Through theoretical discussion and experimental results, basic practical applicability of both systems has been demonstrated, to the task of real-world video search and temporal segmentation of real-world human action videos. In designing both algorithms, an efficiency-first approach has been taken, resulting in faster and more effective systems than the previous state-of-the-art.

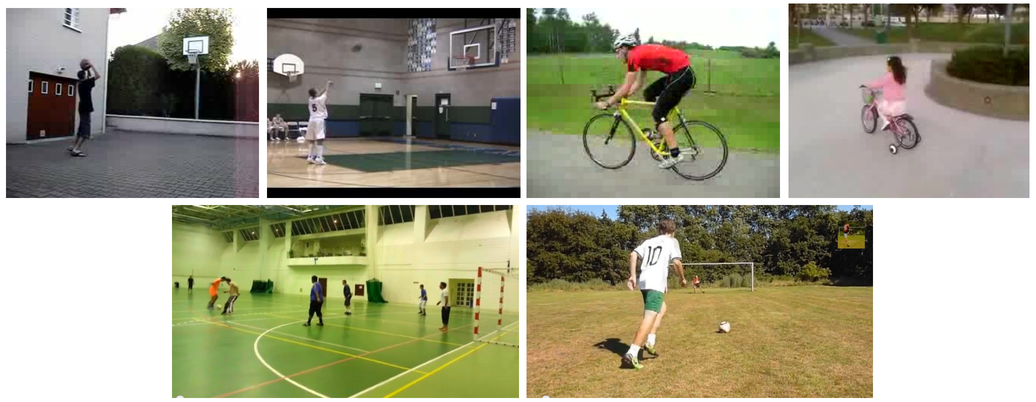
Chapter 5

Clustering Human Actions with Scene Context

5.1 Introduction

Much recent research in the field of computer vision has focused on the representation and recognition of human actions from varied sources, such as YouTube videos and Hollywood films. In these realistic videos, the actions usually have a considerable amount of context – in particular, the place it is performed in, or the object it is performed with. This context information can be integrated into an action recognition system to help disambiguate between similar classes, and thereby improve classification results, as demonstrated in Marszalek et al. [60]. If an action's scene context is recognised as a basketball court, for instance, this informs us that the action to be classified is more likely to be "playing basketball" - Figure 5.1 illustrates this concept.

While context for supervised action recognition has been explored, however, no previous research has considered using context for unsupervised human action *clustering*. This is important to consider, as accurate unsupervised and semi-supervised clustering of human actions is crucial to many practical tasks, such as automatic annotation of video databases or fast content-based video retrieval.



	Cycling	Soccer	Basketball
Indoor Court	✗	✓	✓
Field	✓	✓	✗
Driveway	✓	✗	✓

Figure 5.1: Above: various stills from videos, showing human actions happening in different scene contexts. Below: a simple model example of an action/scene relationship table. The information in this table can be directly inferred from training data in supervised learning, but in unsupervised learning it must be estimated simultaneously with the action/scene categories.

Action clustering can also assist in determining the semantic similarity of two videos' contents, which can be used, for example, to enhance the recommendation systems of video databases. However, it is not straightforward to apply existing action context research to action clustering. In existing work, labeled training data is typically available for both the actions and the context, which permits direct inference of the relationship between the action categories and their contexts – it is straightforward to construct a contextual recognition model based on this relationship. For the goal of action clustering with context, on the other hand, no training labels are provided, and so the action/context relationship cannot be learned directly. It is instead necessary to simultaneously estimate the action clustering, the context clustering, and the action/context relationship together.

In this chapter, to perform unsupervised action clustering with context, a new problem is proposed, as well as two potential approaches to this problem, The problem is defined as *dual assignment clustering*, where two clusters of a dataset as learned according to two views of the dataset, using the relationship between the views to improve both clusterings. The first solution to this problem is the heuristic Dual Assignment k-Means clustering algorithm (DAkM). In this chapter, DAkM is first described as a heuristic method that extends the original k-means algorithm. Then, the theoretical applicability of DAkM is demonstrated on synthetic data, and it is combined with a spectral representation (SDAkM) to show state-of-the-art results on several realistic human action datasets (using actions and scenes as the two views).

The rest of this chapter is structured as follows. §5.2 defines dual assignment clustering, and §5.3 details the Dual Assignment k-Means (DAkM) clustering algorithm as well as its spectral extension. Experiments on synthetic data and three realistic human action datasets are given in §5.4 for DAkM and SDAkM. The chapter concludes with a discussion of the experimental findings in §5.5.

5.2 Dual Assignment Clustering

In this section, the problem of dual assignment clustering is defined, its relation to previous clusterings works is considered, one possible multi-objective optimisation approach is described, and then the Dual Assignment k-Means algorithm (DAkM) is provided as an approximation to this optimisation.

5.2.1 Definition

The specific dual assignment clustering problem is defined as follows. We wish to cluster a set of videos into discrete groups of similar videos. We assume that there are two separate, valid clusterings of the videos: the first is based on a video's scene; the second clustering is according to the action of the video. We also assume that these two video clusterings are not independent – if the scene is known, this provides information as to the probability of the action occurring in that video. Finally, we assume that there are two views of each video – one view (derived from motion features) is generated by the action of the video, and the other view (derived from static features) is generated by the scene of the video. The aim is to produce both an action clustering and a scene clustering, estimating the relationship between actions and scenes to enhance the accuracy of both solutions.

In realistic scenarios, the relationship between actions and scenes is many-to-many. That is, a single scene can be associated with multiple actions (e.g., both cycling and walking a dog can occur in a park), and a single action can be associated with multiple scenes (e.g., basketball can be played either indoors or outdoors). Additionally, in realistic datasets certain action/scene pairs are more likely than others, and certain combinations are impossible (e.g., playing basketball in a swimming pool). The intent is to capture the full complexity of this many-to-many relationship, distinct from the one-to-one assumption implicitly made in multi-view clustering, where one action corresponds to exactly one scene.

It is possible to model this relationship using a correlation matrix, $\mathbf{R}$. $\mathbf{R}$

captures correlation information using the joint and marginal probability distributions:

$$\mathbf{R} \equiv \frac{p(A,S)}{p(A)p(S)} \tag{5.1}$$

If the labels of the actions and scenes are known, the joint distribution $p(A,S)$ can be approximated using the relative contingency table F, where each entry $F_{a,s}$ indicates the percentage of videos in the dataset containing both action a and scene s. $p(A)$ and $p(S)$ are calculated as the relative marginal frequencies of the actions and scenes in the whole dataset, represented by M_a and M_s respectively. $\mathbf{R}$ is thus calculated as:

$$\mathbf{R} = F \oslash (M_a \otimes M_s) \tag{5.2}$$

where $\oslash$ indicates a Hadamard division operation. Thus, $\mathbf{R}_{a,s} = 1$ indicates that a and s have no correlation. Similarly $\mathbf{R}_{a,s} > 1$ shows a positive correlation, and $\mathbf{R}_{a,s} < 1$, a negative correlation.

5.2.2 Relation to Previous Works

To appreciate the novelty of dual-assignment clustering, it is necessary to put it in the appropriate context with respect to existing clustering techniques. In addition to the human action clustering works outlined in §2.5, recent advances in general-purpose data clustering should also be considered. In particular, multi-view clustering [126, 127, 128, 129] and alternative clustering [130, 131] bear some similarity to dual-assignment clustering. A comparison of these concepts with dual assignment clustering is shown in Figure 5.2.

Multi-view clustering uses multiple views of the same dataset, rather than just one view, to improve clustering performance. It assumes that there is a single, true clustering of the dataset, and that the mutual information between the views can be used to find this clustering. For instance, Canonical Correlation Analysis, used in Chaudhuri et al. [128], falls under this category. It attempts to find the canonical variates – essentially latent variables – which maximally

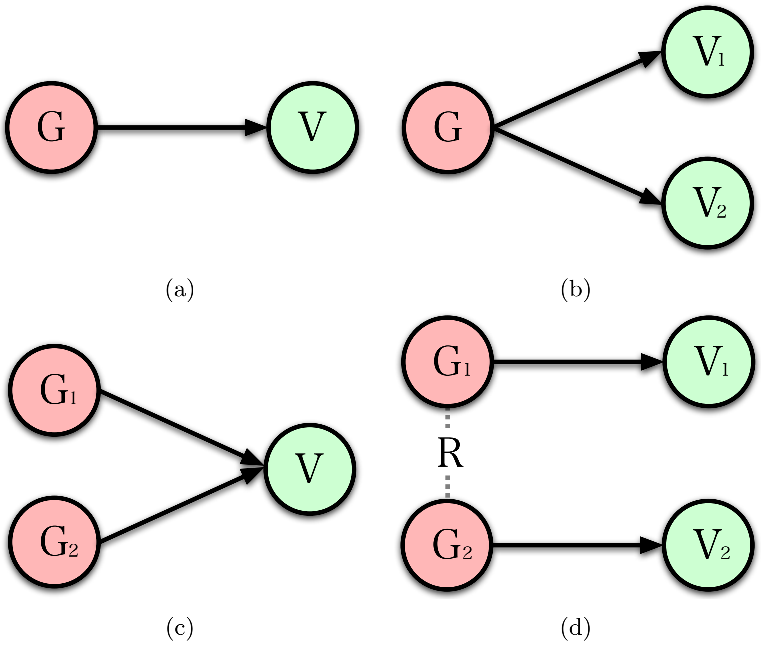


Figure 5.2: A visualisation of various clustering approaches, showing the dependence relationship between latent categorisations of the dataset, G, and the observable views on that dataset, V. (a) Ordinary Clustering. (b) Multiview Clustering for two views. (c) Alternative Clustering for two solutions. (d) Dual Assignment Clustering.

explain the shared information between two views.

Alternative clustering assumes that there are multiple valid clustering solutions for a single dataset. It then finds these multiple clustering solutions based on a single view of the dataset, maximising the optimality of each individual clustering, but also maximising the dissimilarity/orthogonality between all the clusterings.

In the dual assignment clustering algorithm it is assumed that there are two valid clusterings of the dataset (similar to alternative clustering) but there are also two views on the data (similar to multi-view clustering). Each valid clustering is associated with one view. The algorithm estimates the mutual information between the two clusterings and uses it to improve the results of both clusterings simultaneously. Interestingly, the dual assignment clustering algorithm as described in §5 can be applied as a solution for two view clustering by setting $k^{\mathbf{x}} = k^{\mathbf{y}}$, and enforcing $\mathbf{R}$ to be a generalised permutation matrix. However, it is important to note multi-view clustering is not generally equivalent to dual-assignment clustering, as it assumes that every view is generated from the same latent cluster identity (see Figure 5.2.

5.2.3 Optimisation Problem

To provide further insight into the clustering problem, let us define it as an optimisation problem. First, given a set of observations $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$, the basic k-means hard clustering algorithm optimises the following:

$$\arg\min_{\mathbf{C}} \sum_{i=1}^{n} \sum_{j=1}^{k} \mathbf{C}_{i,j} ||\mathbf{x}_i - \mu_j||^2 \tag{5.3}$$

In hard k-means clustering, $\mathbf{C}$ is a binary cluster-membership matrix, where each element $\mathbf{C}_{i,j}$ indicates whether observation $\mathbf{x}_i$ belongs to the jth cluster, and each observation belongs to only one cluster. μ_j is the jth cluster centroid.

In the dual assignment problem, the goal is to cluster two related sets of observations (or views), $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$ and $(\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n)$, into k^x and $k^{\mathbf{y}}$ sets $\mathbf{C}^{\mathbf{x}}$ and $\mathbf{C}^{\mathbf{y}}$ respectively, where corresponding pairs $\mathbf{x}_i$ and $\mathbf{y}_i$ co-occur, and there is an unknown (but non-zero) correlation between them. Here is proposed an modification to the original k-means problem, making a multi-objective optimisation problem over $\mathbf{C}^{\mathbf{x}}$ and $\mathbf{C}^{\mathbf{y}}$. The first objective function is:

$$\operatorname*{arg\,min}_{\mathbf{C}^{x},\mathbf{C}^{\mathbf{y}}} \sum_{i=1}^{n} \sum_{j=1}^{k^{\mathbf{x}}} \sum_{l=1}^{k^{\mathbf{y}}} \mathbf{C}^{x}_{i,j} ||\mathbf{x}_i - \mu^{\mathbf{x}}_j||^2 \mathbf{C}^{\mathbf{y}}_{i,l} ||\mathbf{y}_i - \mu^{\mathbf{y}}_l||^2 \tag{5.4}$$

This objective is essentially identical to that of the original k-means problem, extended for two sets of observations. It is intended to reduce the sum of distances-to-cluster-centroids for both $\mathbf{x}$ and $\mathbf{y}$. As this objective takes the product of the two distances rather than the sum, it is not necessary to account for any scale variation between $\mathbf{x}$ and $\mathbf{y}$. The second objective is:

$$\operatorname*{arg\,min}_{\mathbf{C}^{\mathbf{x}},\mathbf{C}^{\mathbf{y}}} \ -\sum_{j=1}^{k^{\mathbf{x}}} \sum_{l=1}^{k^{\mathbf{y}}} \mathbf{R}_{j,l} \ \log(\mathbf{R}_{j,l}) \tag{5.5}$$

$\mathbf{R}$ is calculated via Equation 5.2 using $\mathbf{C}^{\mathbf{x}}$ and $\mathbf{C}^{\mathbf{y}}$. Equation 5.5 roughly corresponds to maximising the mutual information between $\mathbf{C}^{\mathbf{x}}$ and $\mathbf{C}^{\mathbf{y}}$. This objective is included, as the goal is to find a sparse relationship between the clusters of $\mathbf{x}$ and those of $\mathbf{y}$. When Equation 5.5 is maximal, $\mathbf{R}$ is a uniform matrix, and no information is shared between the two clusterings – in this case, it is better to apply two individual clusterings for improved time efficiency. As $H(\mathbf{R})$ decreases, $\mathbf{R}$ approaches a one-to-one correspondence (or when $k^{\mathbf{x}} \neq k^{\mathbf{y}}$, a many-to-one correspondence) between the clusters in $\mathbf{x}$ and $\mathbf{y}$, and a great deal of information is shared between the two clusterings. If $H(\mathbf{R})$ is too low, however, $\mathbf{R}$ might be distorted by noise, or may be too sparse to accurately represent the relationship between $\mathbf{x}$ and $\mathbf{y}$, so it is balanced with the first objective. The regularisation method to balance these two objectives is detailed below.

5.3 Dual Assignment *k*-Means Algorithm (DA*k*M)

Simple Expectation-Maximization clustering initially seems an ideal solution to directly optimise the dual-assignment clustering problem above. This is similar to the method used for alternative clustering by minimising mutual information in [130]. This approach is intractable, however, due to the calculation of the second objective which introduces dependence between every row of $\mathbf{C^x}$ and $\mathbf{C^y}$. Instead, an iterative update scheme is devised that approximately optimises both objectives in Equations 5.4 and 5.5. The full method is shown in Algorithm 1.

```
Algorithm 1: Dual Assignment k-Means (DAkM)
  Data: Two sets of observations, $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$ and $(\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n)$, where
        $\mathbf{x}_i$ and $\mathbf{y}_i$ co-occur
  $k^\mathbf{x}$, $k^\mathbf{y}$, the number of clusters in each dataset
  $\lambda$, a parameter controlling the final sparsity of $\mathbf{R}$
  Result: Membership vectors $\mathbf{C^x}$ and $\mathbf{C^y}$
  begin
      $(\mathbf{C^x}, \mu^\mathbf{x}) \leftarrow$ Kmeans $(\mathbf{x}, k^\mathbf{x})$
      $(\mathbf{C^y}, \mu^\mathbf{y}) \leftarrow$ Kmeans $(\mathbf{y}, k^\mathbf{y})$
      repeat
          $\mathbf{R} \leftarrow$ UpdateRelationships($\mathbf{C^x}$,$\mathbf{C^y}$,$\lambda$)
          ($\mathbf{C^x}$,$\mathbf{C^y}$) $\leftarrow$ UpdateMemberships($\mathbf{x}$,$\mathbf{y}$,$\mu^\mathbf{x}$,$\mu^\mathbf{y}$,$\mathbf{R}$)
          $\mu^\mathbf{x} \leftarrow$ UpdateCentroids($\mathbf{x}$,$\mathbf{C^x}$,$k^\mathbf{x}$)
          $\mu^\mathbf{y} \leftarrow$ UpdateCentroids($\mathbf{y}$,$\mathbf{C^y}$,$k^\mathbf{y}$)
      until $\mathbf{C^x}$ and $\mathbf{C^y}$ don't change
  Function UpdateRelationships($\mathbf{C^x}$,$\mathbf{C^y}$,$\lambda$)
      $\mathbf{R} \leftarrow$ zeroes($k^\mathbf{x}$,$k^\mathbf{y}$), $M^\mathbf{x} \leftarrow$ zeroes($k^\mathbf{x}$) $M^\mathbf{y} \leftarrow$ zeroes($k^\mathbf{y}$)
      for $i \leftarrow 1$ to $n$ do
          $\mathbf{R}_{\mathbf{C}^\mathbf{x}_i,\mathbf{C}^\mathbf{y}_i} \leftarrow \mathbf{R}_{\mathbf{C}^\mathbf{x}_i,\mathbf{C}^\mathbf{y}_i} + \frac{1}{n}$
          $M^\mathbf{x}_{\mathbf{C}^\mathbf{x}_i} \leftarrow M^\mathbf{x}_{\mathbf{C}^\mathbf{x}_i} + \frac{1}{n}$
          $M^\mathbf{y}_{\mathbf{C}^\mathbf{y}_i} \leftarrow M^\mathbf{y}_{\mathbf{C}^\mathbf{y}_i} + \frac{1}{n}$
      $\mathbf{R} \leftarrow \mathbf{R} \oslash (M^\mathbf{x} \otimes M^\mathbf{y})$ (Eqn 5.2)
      $\mathbf{R} \leftarrow \frac{\log(1+\lambda\mathbf{R})}{\sum_{\mathbf{R}} \log(1+\lambda\mathbf{R})}$ (Eqn 5.6)
      return $\mathbf{R}$
  Function UpdateMemberships($\mathbf{x}$,$\mathbf{y}$,$\mu^\mathbf{x}$,$\mu^\mathbf{y}$,$\mathbf{R}$)
      for $i \leftarrow 1$ to $n$ do
          for $j \leftarrow 1$ to $k^\mathbf{x}$ do
              for $l \leftarrow 1$ to $k^\mathbf{y}$ do
                  dists$(j,l) \leftarrow \frac{||\mathbf{x}_i - \mu^\mathbf{x}_j|| + ||\mathbf{y}_i - \mu^\mathbf{y}_l||}{\mathbf{R}_{j,l}}$
          $(\mathbf{C}^\mathbf{x}_i, \mathbf{C}^\mathbf{y}_i) \leftarrow \arg\min_{j,l}$ dists
      return $(\mathbf{C^x}, \mathbf{C^y})$
  Function UpdateCentroids($\mathbf{d}$,$\mathbf{C}$,$k$)
      for $i \leftarrow 1$ to $k$ do
          $\mu_i \leftarrow \text{mean}(\{d | (d, c) \in \{(\mathbf{d}, \mathbf{C})\} \wedge c = i\})$
      return $\mu$
```

First, the cluster memberships of both datasets are initialised separately using the original k-means algorithm. Then, $\mathbf{R}$, $\mathbf{C^x}$, $\mathbf{C^y}$, $\mu^\mathbf{x}$ and $\mu^\mathbf{y}$ are updated iteratively in a procedure also inspired by the original k-means.

The first iterative step is to estimate $\mathbf{R}$. $\mathbf{R}$ is calculated according to Equation 5.2, and then it is normalised so all elements sum to one. Then $\mathbf{R}'$ is obtained according to Equation 5.6:

$$\mathbf{R}' = \frac{\log(1+\lambda\mathbf{R})}{\sum\limits_{\mathbf{R}} \log(1+\lambda\mathbf{R})} \tag{5.6}$$

where λ is a user-defined parameter controlling the uniformity of $\mathbf{R}'$. The reasons for using $\mathbf{R}'$ parameterised with λ, rather than $\mathbf{R}$ directly, are outlined later in this section.

The second iterative step updates the membership variables $\mathbf{C^x}$ and $\mathbf{C^y}$. For each pair of samples $(\mathbf{x}_i, \mathbf{y}_i)$, the distance to every pair of clusters $j \in [1..k^{\mathbf{x}}], l \in [1..k^{\mathbf{y}}]$ is calculated, and divided by $\mathbf{R}'_{j,l}$. Then the values of j and l that minimise the following are found:

$$\arg\min_{j,l} \frac{||\mathbf{x}_i - \mu_j^{\mathbf{x}}|| + ||\mathbf{y}_i - \mu_l^{\mathbf{y}}||}{\mathbf{R}'_{j,l}} \tag{5.7}$$

As $\mathbf{R}'$ is the divisor, more points tend to be assigned to frequent cluster-pairs, and fewer points will be assigned to rarer cluster-pairs. Over many iterations, this effect implicitly minimises the second objective – the entropy of $\mathbf{R}$ – shown in Equation 5.5.

The final step is to update the cluster means of both datasets independently. This is identical to that in ordinary k-means. The algorithm terminates when $\mathbf{C^x}$ and $\mathbf{C^y}$ stop changing.

The parameter λ acts as a regularisation parameter – it serves to balance the two objectives of the optimisation problem. For higher values of λ, $\mathbf{R}'$ will tend to have a higher entropy, a more uniform distribution, and a higher weight is placed on minimising the total distance to cluster centroids; for lower values of λ, $\mathbf{R}'$ tends to lower entropy, or high sparsity, and places more weight on utilising the mutual information between the datasets. In practical applications, higher values of λ are necessary for noisy datasets or when there is a complex relationship between $\mathbf{x}$ and $\mathbf{y}$ – if λ is too low, poor performance (below the initial baseline) will occur. However, lower values of λ result in a better clustering solution when the dataset is clean and $\mathbf{R}$ is relatively sparse. The effects of λ are considered further in the experimental section below.

There are two drawbacks to DAkM that may be improved in future work. Firstly, the computational complexity of DAkM is $O(k^{\mathbf{x}}k^{\mathbf{y}}n)$ for each iteration, where n is the number of items in the dataset. As such, DAkM is most practical for low values of $k^{\mathbf{x}}$ and $k^{\mathbf{y}}$. A hierarchical extension of DAkM could be considered as a more suitable alternative for larger cluster numbers. Secondly, as DAkM only approximates the objective functions given above, it does not provably find a local optimum with respect to each of them. In an later work it will perhaps be possible to demonstrate the convergence of a modified DAkM to a single, regularised objective function. Despite this, in the experiments below DAkM reliably terminates with a good result in fewer than 100 iterations in all cases, and shows consistently superior accuracy in comparisons.

5.3.1 Spectral Representation

The method presented above extends the original k-means algorithm. However, k-means does not always result in the best clustering results on natural data – especially when pairs of clusters are not linearly separable. This is evident in human action clustering, where Yang et al. [103] use spectral clustering and Niebles et al. [104] use natural language techniques LDA and pLSA, rather than opting for k-means. To gain the advantages of spectral clustering here, DAkM can be adapted to combine dual assignment clustering with a spectral representation. First, let us observe that step 5 of the spectral clustering algorithm in Ng et al. [132] is ordinary k-means clustering. It is therefore straightforward to perform steps 1 through 4 of the algorithm in [132] separately on two views of the dataset, and replace step 5 with DAkM to exploit the mutual information between the two spectral representations. This modified algorithm is referred to as Spectral DAkM (SDAkM), and it is applied to the human action clustering experiments in §5.4.2.

5.4 Experiments

In this section several clustering experiments are detailed. To compare experimental results to the ground truth, the clustering accuracy metric provided in [103] can be used. If each cluster c contains datapoints $x_1, .., x_n$, and each datapoint is associated with a ground truth label $l_1, .., l_n$, the label l_c of cluster c is determined to be:

$$\arg\max_{l_c} \sum_{i=1}^{n} \begin{cases} 1 & \text{if } l_c = l_i \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

Then, to get the final accuracy, it is possible to calculate the percentage of datapoints across the whole dataset that have the same label as their assigned cluster.

5.4.1 Synthetic Data Clustering

Several idealised synthetic datasets are used to demonstrate the applicability of DAkM for its intended purpose.

To generate the artificial dataset, the total number of clusters for $\mathbf{x}$ and $\mathbf{y}$ are set to ($k^{\mathbf{x}} = 12$, $k^{\mathbf{y}} = 8$). For relationships between $\mathbf{x}$ and $\mathbf{y}$, a ground truth relationship matrix $\mathbf{R}_{\text{ground}}$ is randomly generated, where an entry of 1 indicates that two clusters can co-occur, whereas an entry of 0 indicates the opposite. There is at least one positive entry per row and column. Each cluster in $\mathbf{x}$ and $\mathbf{y}$ is represented by a 2-dimensional Gaussian distribution, where the mean is chosen randomly between a range of values, the covariance is a diagonal matrix, and the entries of the covariance vary between a range of values. Then, 10000 samples are generated – for each sample, two clusters from $\mathbf{x}$ and $\mathbf{y}$ are chosen simultaneously in accordance with $\mathbf{R}_{\text{ground}}$, then two vectors are generated from the clusters' Gaussian distributions.

These synthetic data can show how various dataset properties affect the performance of DAkM – in particular, the focus is on the effect of: 1) the number of relationships in $\mathbf{R}_{\text{ground}}$, and 2) how well the clusters are separated.

Task Diff.	k-means		DAkM		Δ	
	$\mathbf{x}$	$\mathbf{y}$	$\mathbf{x}$	$\mathbf{y}$	$\mathbf{x}$	$\mathbf{y}$
H $\mathbf{x}$, H $\mathbf{y}$	56.7	60.0	59.3	62.4	2.7	2.4
H $\mathbf{x}$, E $\mathbf{y}$	56.5	92.8	63.3	95.9	6.7	3.2
E $\mathbf{x}$, E $\mathbf{y}$	83.0	92.8	90.9	97.5	7.9	4.7

Table 5.1: Performance of DAkM when view difficulty is varied.

Results are compared with ordinary k-means clustering. For all of the synthetic experiments $\lambda = 1$.

The first experiment is to look at the effects of varying the number of relationships between $\mathbf{x}$ and $\mathbf{y}$, which is achieved by controlling the number of non-zero entries in $\mathbf{R}_{\text{ground}}$. The difference in clustering performance is shown between ordinary k-means clustering and DAkM on a synthetic dataset over a range of values for $|\mathbf{R}_{\text{ground}}|$ in Figure 5.3. As expected, as $|\mathbf{R}_{\text{ground}}|$ increases, the performance improvement decreases, as there is less mutual information to exploit between $\mathbf{x}$ and $\mathbf{y}$.

Next is considered how DAkM is affected by the difficulty of the individual clustering tasks. 4 views are generated, based on how difficult they are to cluster accurately: a hard (H) $\mathbf{x}$ and hard $\mathbf{y}$, an easy (E) $\mathbf{x}$ and easy $\mathbf{y}$. In Table 5.1 the results of clustering various combinations of these views are shown.. The accuracies displayed are for k-means and DAkM. The difference between k-means and DAkM is also shown. As can be seen, DAkM results in significant improvements over ordinary k-means in all cases. This shows the robustness of DAkM: one might expect, for instance, that the noise from a more difficult task would degrade the performance of an easier task, but this is demonstrably not the case.

5.4.2 Human Action Clustering

Three real-world datasets are used for experimentation: UCF YouTube [78], UCF Sports [107], and Hollywood-2 [60], as they contain a variety of real-world actions with various scene contexts. The scene ground truths are not provided for these datasets – therefore, in the below experiments, only the action clustering

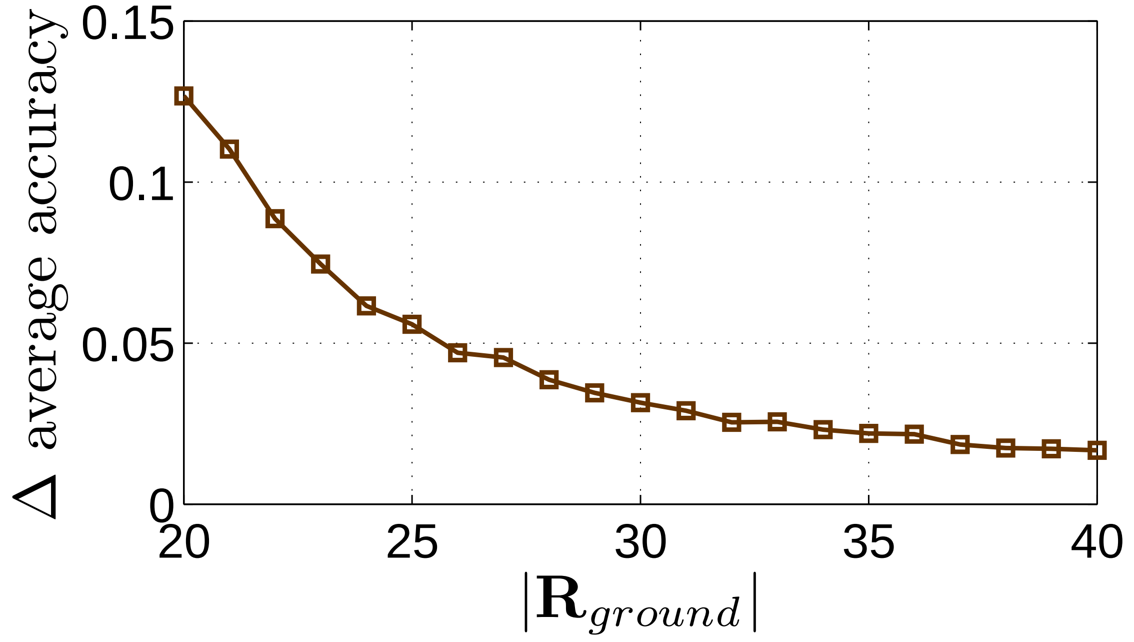


Figure 5.3: How improvement in DAkM accuracy (averaged over $\mathbf{x}$ and $\mathbf{y}$) changes with the number of relationships between $\mathbf{x}$ and $\mathbf{y}$.

accuracy is reported on. However, some qualitative results are shown for the UCF YouTube dataset scene clustering. The details of these datasets are provided in §2.6.

Dense trajectories, as presented in Wang et al. [133], are used to extract a motion representation of the actions. For this, the publicly available code is used with the default settings. Then, the features are processed for a bag-of-words type representation. PCA is performed on the features, and then k-means clustering on each of the descriptors separately (HOG, HOF, MBH, Tr), where 95% of the variance is captured from PCA and $k = 2000$ for the clustering. This results in a 8000-dimensional (4×2000) frequency histogram of motion features. For scene representations, SIFT features are appropriate[21], extracted from each video at intervals of 10 frames. Once again, PCA and k-means are performed, with $k = 2000$, resulting in a 2000-dimensional frequency histogram of static features. The histograms of both the motion and static features are normalised, as different videos can be of differing lengths, and therefore have greatly differing numbers of features.

The first step to get a spectral representation is to find the pairwise distance between all histograms in a set using the histogram intersection:

$$S(a, b) = \sum_{i=1}^{n} \min(a_i, b_i) \tag{5.9}$$

Equation 5.9 is applied to get a similarity graph for both actions and scenes. Steps 2-4 of Ng et al. [132] are applied on the actions and scenes separately to get two independent spectral representations, then DAkM is performed on the resulting vectors. k_a (the number of action clusters) is set to the number of action categories in the dataset. As the ground truths are not provided for scenes of the datasets, there is no prior knowledge of k_s, but preliminary experiments show that a relatively high number of scenes works best – $k_s = 40$ for all datasets. As all algorithms are stochastic, all experiments are run 10 times and the results averaged.

The results for motion clustering over all three datasets are summarised in Table 5.2. The following clustering methods are compared: Spectral Single-View

Dataset	Clustering Accuracy (%)				
	SSV	SCV	CMV	SD1k	SDO
YouTube	39.2	40.7	38.1	41.8	**43.9**
UCF Sports	68.0	67.2	64.2	72.9	**76.0**
Hollywood-2	35.6	32.4	31.5	**36.5**	**36.5**

Table 5.2: Clustering performance of various methods on realistic datasets.

(SSV), applying Ng et al. [132] to motion features only; Spectral Concatenated Views (SCV), applying Ng et al. [132] to a concatenated motion and static histogram; Co-Trained Spectral Multi-View (CMV), a recent multi-view clustering method presented in Kumar and Daume III [129], treating motion and static features as two views of the action; SDAkM with $\lambda = 1000$ (SD1k); SDAkM with λ set to the optimal value for each dataset individually (SDO). For SDO, the following values are considered for λ: 1, 5, 10, 50, 100, 500, 1000, 5000, 10000 and 50000.

As can be seen from the table, SDAkM with optimal λ (SDO) gives the highest accuracy on all three datasets – this is most likely because it most effectively utilises the complex relationship between actions and scenes. CMV performs far worse than the baseline clustering method (SSV) on all datasets, demonstrating that the multi-view assumption is not applicable to the relationship between motion and static features. Concatenating motion and static vectors (SCV) also negatively impacts accuracy for two of the three dataset.

SDAkM gives the greatest performance increase over the baseline on the UCF Sports dataset, which can be attributed to two properties of the dataset: a highly sparse relationship $\mathbf{R}_{ground}$, and easy-to-cluster scenes. Alternatively, the weakest performance is seen on the Hollywood-2 dataset, observing only a 0.9% increase in accuracy, even with the optimal λ. This is unsurprising: Marszalek et al. [60] used context to enhance recognition accuracy on the Hollywood-2 dataset, using training data to directly infer the relationship between actions and scenes, but only observed a 1.1% improvement over baseline performance.

The effect of varying parameter λ on the performance of SDAkM is shown in Figure 5.4 for each of the datasets. Peak performance is observed at a different

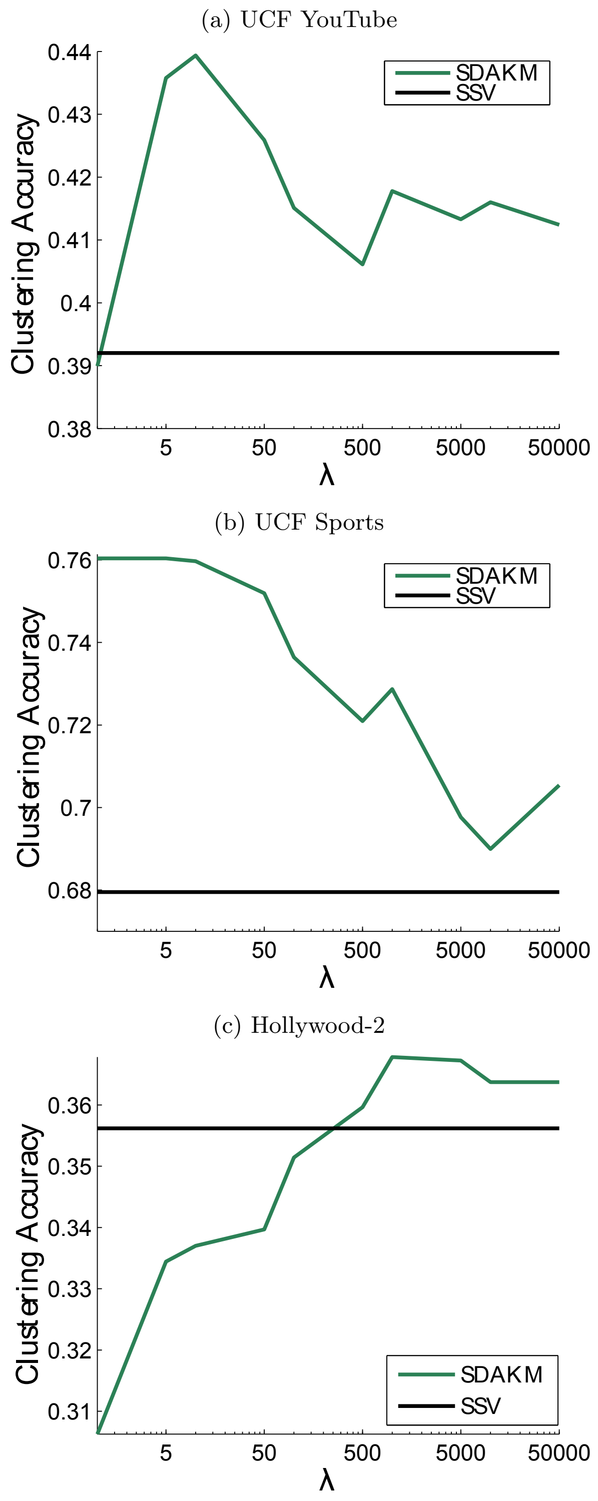


Figure 5.4: Performance of SDAkM with various λ.

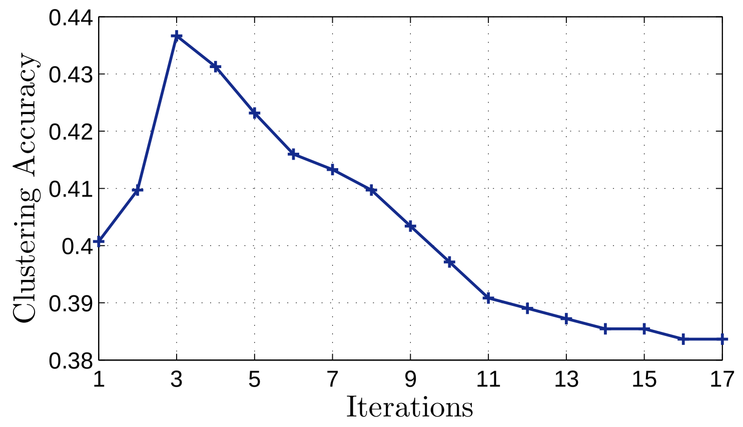


Figure 5.5: SDAkM on the YouTube dataset with $\lambda = 1$ – a high peak performance after few iterations, but eventually the algorithm terminates with poor performance.

λ for each dataset – this is expected, as the sparsity of the unknown R_{ground} is likely to differ between datasets. In future work it would therefore be beneficial to estimate the optimal value of λ automatically using the mutual information between the two views of the dataset. However, there is still a significant improvement over SSV for all datasets when $\lambda \geq 1000$.

To understand the effect of λ further, clustering is performed on the YouTube dataset when $\lambda = 1$. It is proposed that under this condition, $\mathbf{R}'$ tends to become more sparse than the true relationship between actions and scenes, which will distort the clustering results. Figure 5.5 illustrates this situation, showing the iteration-by-iteration performance of clustering the YouTube dataset with $\lambda = 1$. Iteration by iteration, $\mathbf{R}'$ grows more sparse. Initially, this results in more accurate clustering – 3.6% better than the initial solution after the third iteration – but when the algorithm terminates, it has fallen to 1.7% below the initial accuracy, suggesting that $\mathbf{R}'$ no longer reflects the true relationship between scenes and actions.

Although none of the datasets have ground truths for the scenes, Figure 5.6 provides a few examples of discovered scene categories on the UCF YouTube dataset. Several key scene categories clearly arise: shots including the sky; playing courts; fields; swimming pools; trampolines. These are each more or less commonly associated with certain actions in the dataset, as one might expect. For instance, the sky scenes typically show golf, tennis or soccer juggling, but the playing court scenes more usually show volleyball, tennis, or basketball. Furthermore, the swimming pool and trampoline scenes have a nearly one-to-one correspondence with diving and jumping respectively. It is clear in these cases how extra information from the scene category may aid in clustering the actions.

5.5 Discussion

In this chapter a new algorithm has been introduced – Dual Assignment k-Means clustering (DAkM) – for generating two clustering solutions according to two co-occurring views of a dataset. Unlike previous methods, it is suitable for use when there are two co-occurring views of a dataset, and a separate clustering solution associated with each view – similar previous clustering methods have either only been suitable to generate multiple clustering solutions from a single view (alternative clustering) or to generate a single clustering from multiple views (multi-view clustering). This chapter has demonstrated that DAkM can significantly improve clustering results on synthetic data and realistic human action/scene datasets. This performance improvement is apparent even when the clusters in both views are poorly separated, demonstrating the robustness of DAkM/SDAkM.

Further work on DAkM might focus on determining λ automatically, which can perhaps be calculated mathematically as a function of the mutual information between the two views of the dataset. Also, while the algorithm presented here is restricted, for complexity reasons, to considering dual-assignments only, in future multiple-assignment clustering could also be considered.

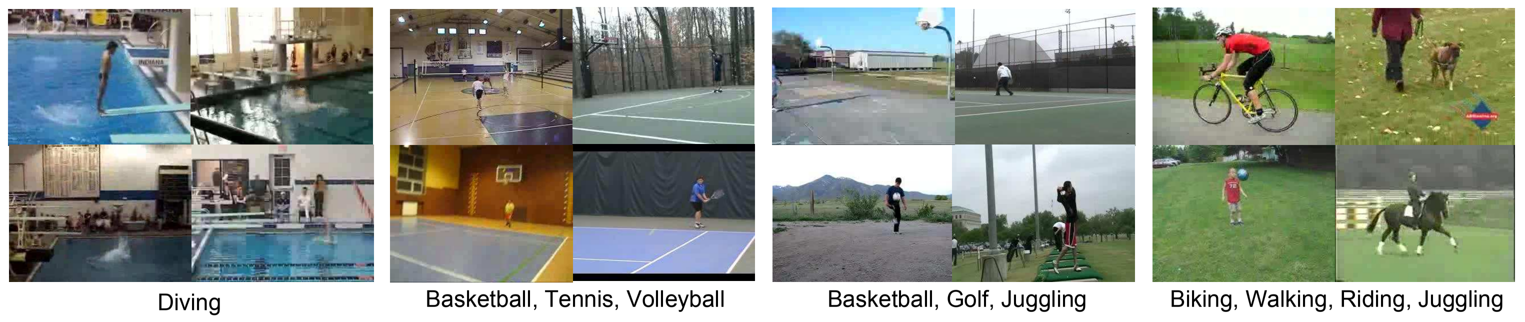

Diving
Basketball, Tennis, Volleyball
Basketball, Golf, Juggling
Biking, Walking, Riding, Juggling

Figure 5.6: Examples of scene categories discovered in the UCF YouTube dataset, and their most strongly associated actions below.

Chapter 6

Dual-Assignment Clustering of Single Actions with Multiple Objects

6.1 Introduction

As the previous chapter has shown, DAkM and the closely related SDAkM are fast, heuristic algorithms. However, they have some specific drawbacks which make them inapplicable to certain scenarios. Firstly, as previously mentioned, they have no theoretical guarantee of convergence to a local minimum with respect to the proposed objective functions (even though they *tend* towards a local minimum.) Secondly, they are only applicable to situations where each item in the dataset has exactly two views, and that each of these views has a single topic source – this does not account for situations where each view might have multiple topic sources (such as in Latent Dirichlet Allocation), or where there might be more than two views.

The rest of this chapter is structured as follows: §6.2 introduces the concept of graphical models for dual assignment clustering, and specifically details the MOSAC model, then experiments comparing MOSAC and SDAkM on a cooking

Variable	Definition
L	The number of mixture components
V	The number of datapoints/observations
ζ	The mixture weights of each component
$(A_1, ..., A_V)$	The component of each datapoint
$(\Xi_1, ..., \Xi_V)$	The datapoints/observations
$(\mu_1, ..., \mu_L)$	The mean of each component
$(\sigma_1^2, ..., \sigma_L^2)$	The variance of each component

Table 6.1: Variable definitions for GMM

dataset are shown in §6.6.

6.2 Graphical models for Dual-Assignment Clustering

(N.B.: In this section non-standard notation is used to denote the random variables of Gaussian Mixture Models and Latent Dirichlet Allocation – these can be seen in Figures 6.1 and 6.3. This is to avoid confusion due to overlapping symbols when the final combined model, MOSAC, is introduced, as in Figure 6.4).

For scenarios where (S)DAkM is inadequate, a better alternative might be a graphical model. By capturing the conditional dependencies between the variables, graphical models can capture complex structures of data. This graphical models can be arbitrarily complex, but probabilistic inference of the unobserved variables in the model can be done using various methods, including MCMC simulations for particularly complex models. Graphical models are easily visualised as directed graphs, hence their name – a node represents a variable or a parameter, and an arrow connecting two nodes shows a parent/child conditional dependence between them.

To illustrate how a graphical model can be used for dual-assignment clustering, it is first useful to describe how a univariate Gaussian Mixture Model (GMM) is constructed. In Figure 6.1, a visualisation of a GMM is shown, using plate notation. The meaning of the various variables in this figure are shown in Table

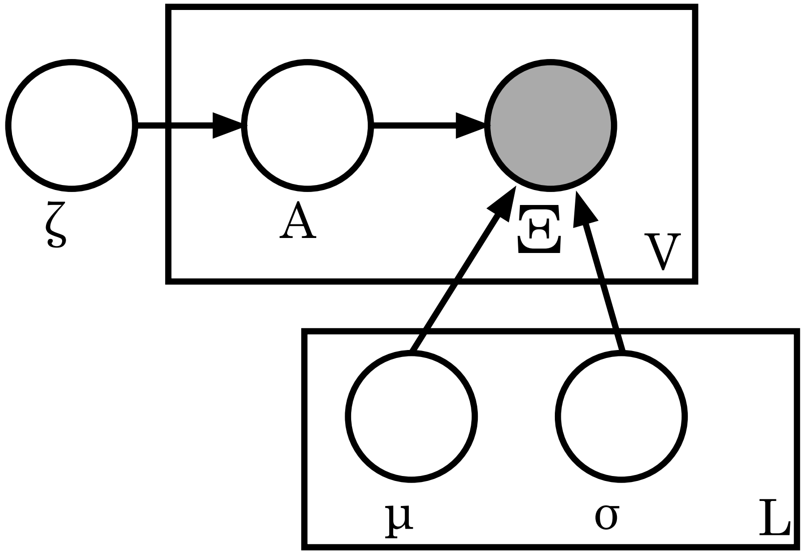


Figure 6.1: A GMM visualised with plate notation. Evidence nodes are shaded.

6.1. To generate a dataset using this GMM, the following steps are followed:

1. For each sample v, $1 \leq v \leq V$ in the dataset:

   (a) Choose mixture component A_v with $A_v \sim Categorical(\zeta)$.

   (b) Generate sample point $\Xi_v \sim \mathcal{N}(\mu_{A_v}, \sigma^2_{A_v})$

The unobserved variables of a GMM can be approximately inferred using a variety of methods. Most commonly, Expectation-Maximization (EM) is used to infer GMMs, as it is accurate and efficient. MCMC simulation can also be used, such as the Metropolis-Hastings sampler, or a collapsed Gibbs sampler [134] – for the purposes this work, however, EM is sufficient.

The basic GMM is now extended to show how it can be applied for dual-assignment clustering. In Figure 6.2 a more complex graphical model is shown. This can be interpreted as two connected GMMs – the mixing component sets y and z are the two clustering solutions, and they are connected by a conditional dependence, $p(y|z)$. The generation of a point in this model is similar to that of two individual single GMMs, but the Dirichlet prior for y is chosen from the value of z. Using this Dual-Assignment GMM (DAGMM), the full generation of a dataset has the following steps:

1. For each sample v, $1 \leq v \leq V$ in the dataset:

   (a) Choose mixture component A_v with $A_v \sim Categorical(\zeta_1)$.

   (b) Generate sample point $\Xi_{1v} \sim \mathcal{N}(\mu_{1A_v}, \sigma^2_{1A_v})$

   (c) Choose mixture component S_v with $S_v \sim Categorical(\zeta_{2A_v})$.

   (d) Generate sample point $\Xi_{2v} \sim \mathcal{N}(\mu_{2S_v}, \sigma^2_{2S_v})$

In some cases, a DAGMM might give slightly better accuracy, just as an ordinary GMM can provide better clustering accuracy than the k-means algorithm in certain cases. Additionally, there exists more theoretical grounding for the algorithmic convergence of this graphical model than there does for the convergence of DAkM. However, these marginal benefits may not be worth it, as they come at the cost of significantly increased complexity for inference, both

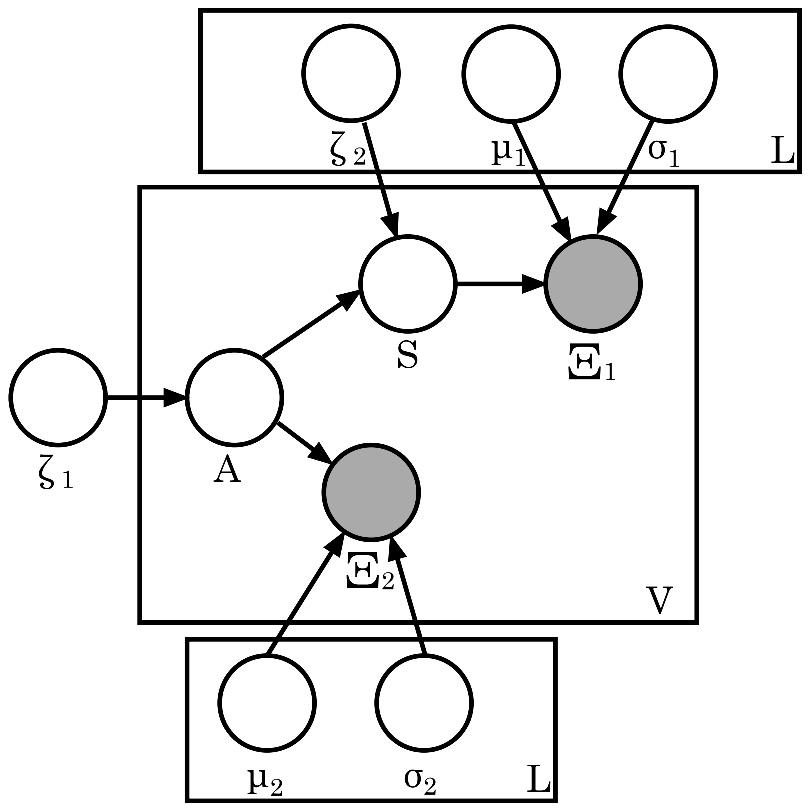


Figure 6.2: A Dual-Assignment GMM visualised with plate notation. Evidence nodes are shaded.

Variable	Definition
K	Number of topics
V	Number of documents (or videos, scenes)
$(N_1, ..., N_V)$	Number of words in each document
α	Topic distribution prior
$(\theta_1, ..., \theta_V)$	Dirichlet topic distribution for each document
$(Z_{11}, ..., Z_{VN})$	Topic for each word in each document
$(W_{11}, ..., W_{VN})$	The specific words observed for each document
β	Word distribution prior
$(\varphi_1, ..., \varphi_K)$	Dirichlet word distribution for each topic

Table 6.2: Variable definitions for Latent Dirichlet Allocation

in time and space, whether by EM or MCMC simulation. Additionally, the asymmetric conditional relationship $p(y|z)$ between the two component sets may adversely affect results, depending on the inference method used.

6.3 Graphical models for the Dual-Assignment Clustering of Multiple Objects

While a DAGMM has limited utility, there are certain scenarios where a more complex graphical model may usefully extend dual-assignment clustering. Consider a kitchen scenario, where a person prepares several dishes, interacting with a variety of utensils, ingredients and containers. Each single action might be associated with several objects – for instance, when a person performs the action "frying", they might be interacting with a frying pan, an egg and a wooden spoon simultaneously. This is problematic for the DAkM or DAGMM models, which both assume the two views are generated from a single source topic, rather than multiple topics.

To solve this kitchen scenario, it is necessary to first introduce the Latent Dirichlet Allocation (LDA)[125] model. As shown in [135], it is possible to use LDA to account for the multiple objects contributing to a single Bag-of-Words scene vector. An LDA is shown in Figure 6.3, and the variables are given in Table 6.2.

To generate a set of "documents" (or videos, or scenes) using LDA, the

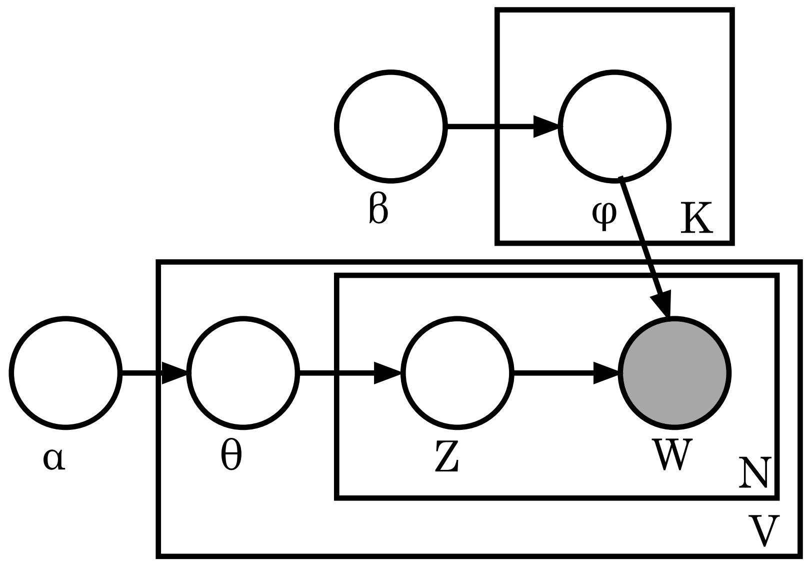


Figure 6.3: Latent Dirichlet Allocation visualised with plate notation. Evidence nodes are shaded.

Variable	Definition
L	The number of actions
V	The number of videos
ζ	The mixture weights of each action
$(A_1, ..., A_V)$	The action associated with each datapoint
$(\Xi_1, ..., \Xi_V)$	The action datapoints/observations
$(\mu_1, ..., \mu_L)$	The mean of each action datapoint
$(\sigma_1^2, ..., \sigma_L^2)$	The variance of each action datapoint
K	Number of scene topics
$(N_1, ..., N_V)$	Number of scene words in each document
α	Scene topic distribution prior
$(\theta_1, ..., \theta_V)$	Dirichlet scene topic distribution for each document
$(Z_{11}, ..., Z_{VN})$	Scene topic for each scene word in each document
$(W_{11}, ..., W_{VN})$	The specific scene words observed for each document
β	Scene word distribution prior
$(\varphi_1, ..., \varphi_K)$	Dirichlet scene word distribution for each scene topic

Table 6.3: Variable definitions for MOSAC

following steps are followed:

1. For each topic k:
   (a) Choose word distribution $\varphi_k \sim Dir(\beta)$
2. For each document v in the dataset:
   (a) Choose topic distribution for document v: $\theta_v \sim Dir(\alpha)$
   (b) For each word i in document v:
      i. Choose a topic Z_{vi} from θ_v: $Z_{vi} \sim Categorical(\theta_v)$
      ii. Choose a word W_{vi} from $\varphi_{Z_{vi}}$: $W_{vi} \sim Categorical(\varphi_{Z_{vi}}$

In this setting, each "document" (scene vector) can be generated from multiple "topics" (objects). The distribution of the topics within a document is captured as a Dirichlet distribution. This topic model has proven popular in textual analysis, but also in scene categorization, as shown in [135] and [136].

Now the necessary background has been introduced, it is possible to demonstrate how a GMM and LDA can be combined to perform Dual-Assignment clustering on both single actions and multiple objects. This graphical model, termed *Multiple Object Single Action Clustering* (MOSAC), is shown in Figure

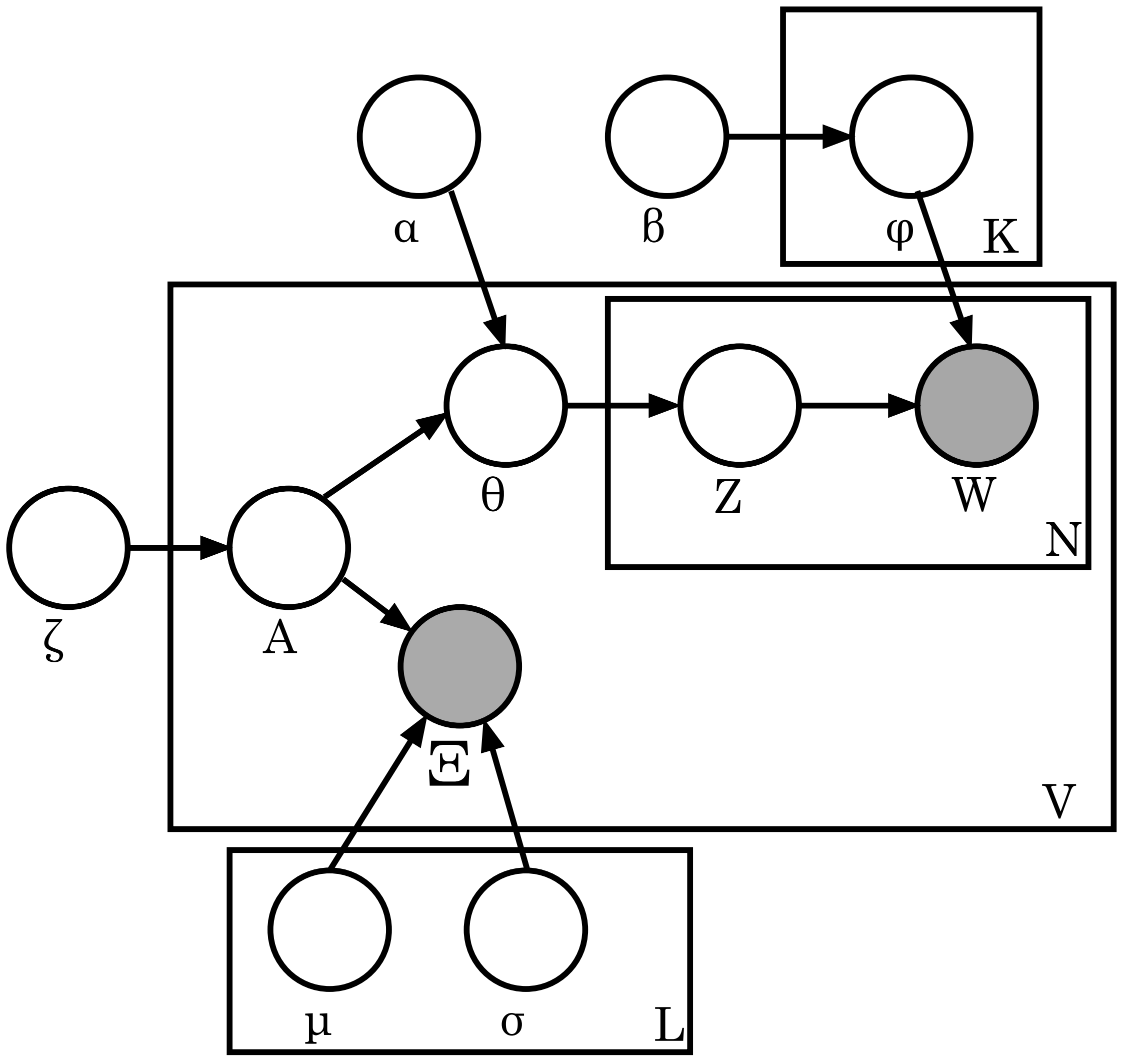


Figure 6.4: MOSAC visualised with plate notation. Evidence nodes are shaded.

6.4. Its variables are defined in Table 6.3. As can be seen from the lower half of the figure, action categories A are the membership components, and of a GMM-like structure (with mixing components, means and variances defined by ζ, μ and σ^2 respectively). Random variable θ captures the distribution of the scene objects as a Dirichlet distribution, *conditioned on* α_A. Z, X and ϕ are the same as in LDA. The generative process of MOSAC is as follows:

1. For each scene topic k:
    (a) Choose scene word distribution $\varphi_k \sim Dir(\beta)$
2. For each document v, $1 \leq v \leq V$ in the dataset:
    (a) Choose mixture component A_v with $A_v \sim Categorical(\zeta)$.
    (b) Generate sample point $\Xi_v \sim \mathcal{N}(\mu_{A_v}, \sigma^2_{A_v})$
    (c) Choose topic distribution for document v: $\theta_v \sim Dir(\alpha_{A_v})$
    (d) For each word i, $1 \leq i \leq N_v$ in document v:
        i. Choose a topic Z_{vi} from θ_v: $Z_{vi} \sim Categorical(\theta_v)$
        ii. Choose a word W_{vi} from $\varphi_{Z_{vi}}$: $W_{vi} \sim Categorical(\varphi_{Z_{vi}}$

To perform dual-assignment clustering with MOSAC, it is necessary to maximise $p(A_v|x_v, \Theta, \theta_v)$ for the actions, and $p(\theta_v|x_v, A_v, \Theta)$ for the scenes where Θ is the set of all parameters in the model. Inference of these values can be done using a nested Expectation-Maximisation algorithm which combines EM for GMMs and a modified Variational EM for LDA collections.

6.4 Nested Expectation-Maximisation

The algorithm is composed of two main components – namely, a GMM EM component, and a nested LDA Variational EM component. These components are partly derived from previous work, such as in Redner et al. [33] and Blei et al. [125] respectively, but also require some modification to be combined properly in the MOSAC framework. Intuitively, the LDA EM algorithm is *nested within* the GMM EM algorithm – the rest of this section clarifies this idea. The idea of nesting EM algorithms within one another is not entirely new (see [137]), but is relatively under-explored. The full algorithm for performing inference on MOSAC is found in Algorithm 2.

6.4.1 Nested LDA Variational EM

First, the LDA Variational EM component is described. The original algorithm consists of two steps:

- Expectation: calculate variational components, in this work termed γ_v and ρ_v, for each video v using an iterative algorithm.
- Maximisation: Update model parameters α and β such that the log likelihood of the model is maximised:

$$\arg\max_{\alpha,\beta} \sum_{v=1}^{|x|} logp(x_v|\alpha,\beta) \tag{6.1}$$

In MOSAC, the structure of the LDA component is similar to an ordinary LDA model, and roughly the same algorithm can be implemented. The components θ, α, Z, W, phi and β in MOSAC (See Figure 6.4) all roughly correspond to their analogous components in LDA.

The primary distinction is that in LDA α is a T-dimensional vector: T is the number of topics – however, in MOSAC α is a $T \times k_a$ matrix. A different column of α is chosen for each video v according to the action of the video, A_v. This requires only a straightforward modification to the Expectation step, which is calculated on a per-video basis. The Maximisation update step is also trivially updated. To calculate the update of α, the Newton-Raphson method is applied using the steps outlined in Appendix A.4.2 of [125] – the log likelihood with respect to α_c is similar, but not identical:

$$L_{[\alpha_c]} = \sum_{d=1}^{M} A_{dc}(log\Gamma(\sum_{j=1}^{k} \alpha_{jc}) \quad - \sum_{i=1}^{k} log\Gamma(\alpha_{ic}) + \sum_{i=1}^{k}((\alpha_{ic} - 1)(\Psi(\gamma_{di}) - \Psi(\sum_{j=1}^{k} \gamma_{dj})))) \tag{6.2}$$

where A_{dc} is a binary matrix indicating whether document d belongs to action c – all rows sum to 1. The derivative and second partial derivative extend from this modification trivially, and so the Newton-Raphson method as described in Appendix A.2 of [125] can be applied. The update step for β remains unchanged from the original LDA.

6.4.2 Nested GMM EM

Given the following:

- V datapoints $(x_1, ..., x_V)$,
- Binary indicator variables $(A_1, ..., A_{k_a})$ determining which of k_a actions each datapoint x_v belongs to,
- Membership weight matrix $w_{vi} = p(A_{vi} = 1|x_v, \Theta)$ indicating the likelihood of datapoint x_v belonging to action A_i,
- Mixture component vector ζ of length k_a, where entry ζ_i indicates the likelihood of a random datapoint x belonging to action A_i,
- Means μ_i and variances σ_i^2 for each action A_i,

the basic GMM Expectation-Maximisation algorithm is as follows:

- Expectation: compute w_{vi} for $1 \leq v \leq V$ and $1 \leq i \leq k_a$, according to:

$$w_{vi} = \frac{\mathcal{N}(x_v; \mu_i, \sigma_i^2) \cdot \zeta_i}{\sum_{m=1}^{k_a} \mathcal{N}(x_v; \mu_m, \sigma_m^2) \cdot \zeta_m} \tag{6.3}$$

where $\mathcal{N}(x; \mu, \sigma^2)$ is the pdf of a Gaussian distribution at point x.

- Maximisation: update ζ, μ and σ^2, with $1 \leq i \leq k_a$ according to:

$$\zeta_i = \frac{\sum_{v=1}^{V} w_{vi}}{V} \tag{6.4}$$

$$\mu_i = \frac{1}{\zeta_i} \sum_{v=1}^{V} w_{vi} \cdot x_v \tag{6.5}$$

$$\sigma_i^2 = \frac{1}{\zeta_i} \sum_{v=1}^{V} w_{vi} \cdot (x_v - \mu_i)(x_v - \mu_i)^T \tag{6.6}$$

To use a GMM for hard clustering, the above EM algorithm is performed until convergence, and then each datapoint x_i is assigned to a cluster according to the maximum weight in the ith row of w.

In MOSAC, ζ,A,σ^2,μ and Ξ roughly approximate an ordinary GMM model. The only difference is that, in the expectation step, the update of A_{vi} includes a new term incorporating λ_v and α_i from LDA above:

$$w_{vi} = \frac{\mathcal{N}(x_v; \mu_i, \sigma_i^2) \cdot d(\theta_v; \alpha_i) \cdot \zeta_i}{\sum_{m=1}^{k_a} \mathcal{N}(x_v; \mu_m, \sigma_m^2) \cdot d(\theta_v; \alpha_m) \cdot \zeta_m} \tag{6.7}$$

Where $d(\theta; \alpha)$ denotes the Dirichlet distribution pdf at point θ with concentration parameters α.

6.4.3 Full MOSAC Inference

Now that the main two components of the MOSAC inference algorithm are defined, they can be combined into a complete algorithm. This is illustrated in Algorithm 2.

The first initialisation step of the algorithm is to perform k-means on the action representations of the videos – the clusters obtained from this process can give a good initial estimate of μ and σ^2. The second initialisation step is to calculate the parameters of ordinary GMM on the actions to further refine the estimate of μ and σ^2, and to get an initial estimate for ζ. Performing these steps greatly reduces the time to convergence.

After initialisation, the nested EM loop begins. First, variational EM is first performed on the scene features as per §6.4.1 – in the first iteration, each column

Algorithm 2: MOSAC Nested EM Inference

```
   Data:
   $X$ - a motion feature histogram representation of a video dataset.
   $Y$ - a static feature histogram representation of a video dataset.
   $k_a$,$k_s$ - the number of action categories and scene topics respectively.
   $\alpha_0$,$\beta_0$,$\zeta_0$ - initial parameter values.
   Result: $A$ - A matrix of binary indicator variables indicating which videos
           belong to which actions.
1  Perform $k$-means clustering on $X$ to get initial membership weights $w$
2  Set initial $\mu$ and $\sigma^2$ according to Eqns 6.5 and 6.6
3  Perform ordinary GMM EM according to [125] to get initial $w$,$\mu$, $\sigma^2$ and $\zeta$
4  while Not Converged do
5      GMM Maximisation:
6          LDA Expectation:
7              Calculate $\gamma_v$ and $\rho_v$ for each document (see Figure 6 of [125])
8          LDA Maximisation:
9              Update $\beta$ according to Eqn 9 of [125]
10             Update $\alpha_c$ for each action $A_c$ independently, using
               Newton-Raphson method, with Eqn 6.2 and first/second order
               derivatives
11     GMM Expectation:
12         Recalculate $w$ according to Eqn 6.7
13 Set $A_{vi}$ to 1 if $w_{vi}$ is the maximum value of row $v$ of $w$, 0 otherwise
```

of α is set uniformly to 1. This converges to obtain the values for λ and α for each video and each action respectively. Then, the maximization step of §6.4.2 is run to update μ, σ^2 and ζ. Finally, the expectation step of §6.4.2 is run to update A. This procedure is iterated until convergence – this is defined as the point at which the aggregate change in A between iterations falls below a set threshold.

Intuitively, MOSAC could be considered similar to running ordinary GMM on the action representation, and ordinary LDA on the scene representation. However, these two models *share information* probabilistically through the modified α and w update steps.

Perhaps the most obvious drawback to the algorithm is that it is potentially very time-expensive, as the inner EM algorithm needs to fully converge for every outer loop of the whole inference algorithm – and LDA EM optimisation in particular can take a long time to converge. To mitigate this, it is possible to limit the inner loop to a low, set number of iterations. Indeed, it has been determined experimentally that the algorithm still runs with acceptable performance even when the inner LDA EM is limited to a single iteration.

6.5 Representing a Multiple-Object Dataset

While the MOSAC algorithm is potentially well-suited to multiple-object single-action datasets, it is still necessary to represent the dataset appropriately for it to work well, and the representation depends highly on the specific dataset being analysed. Here, the MPII Cooking Activities dataset [112] is proposed as a testbed for MOSAC - see Figure 2.7 for example scenes. This dataset records several actors performing several cooking activities, from a fixed perspective, high definition camera. The background is static, making it straightforward to isolate the actor and any object from the rest of the scene using background subtraction. The ground truth associated with the dataset specifies the temporal extent of each action, which is used to break up the dataset into the individual actions before representation.

To represent the actions for MOSAC, a straightforward Bag-of-Words model is generated from dense trajectory features, and then a spectral representation is calculated from the resulting action vectors. This is identical to the method used in SDAkM above. The spectral representation given is Euclidean and therefore the GMM-like structure in MOSAC can be used to model the distribution of the action vectors.

Representing the objects well in MPII is more of a challenge. The first step is to perform standard background subtraction to eliminate all extraneous information – as the camera is fixed, the background is calculated simply as the modal pixel values over the whole of each video. The background image is subtracted from a frame. Each pixel of the resultant difference frame is thresholded to determine if it belongs to the background. The resulting binary image shows which pixels belong to the foreground. Simple erosion/dilation operations are then applied to clean up the binary image, and finally the binary image is used to mask the original frame. This isolates the actor and the objects that are involved in the cooking process.

Having extracted foreground pixels, a simple approach to extract the object descriptors would be to calculate SIFT features on several images from the action, keeping only SIFT features on the foreground pixels. However, this is not ideal, as many of the objects detected in the scene might be irrelevant to the action being performed. Instead, body pose detection is applied to the frames using the method in [112], giving the position of the person's hands in each frame.

Skin tone classification of the pixels is performed using the method described in [138] in a radius around the hands to determine which pixels belong to the hands. Finally, to extract the features of the objects being used in the action, dense SIFT features are extracted centred on all foreground pixels in a radius around the hand, excluding pixels that belong to the hand itself. A Bag-of-Words model can then be created from these SIFT features in the usual manner, using PCA and k-means clustering, giving a complete representation of the scene of an action.

Note that while in the simple SDAkM algorithm problem of §5.4.2, all the SIFT features are generated from a single *scene*, MOSAC makes the assumption that each SIFT feature could belong to a different object, or *topic*. Because of this, the scene SIFT feature space cannot be transformed into a linear one via spectral embedding, as in §5.3.1. However, this is not necessary, as the Latent Dirichlet Allocation branch of the MOSAC model can already capture non-linear dependencies in the scene variables without an explicit spectral transformation of the scene space.

6.6 Cooking Activity Experiments

In this section MOSAC's effectiveness on the MPII cooking activities dataset is examined experimentally.

6.6.1 Dataset

In order to perform experiments on MOSAC, a subset of the original MPII Cooking Activities dataset [112] is used. Some of the action classes in the original dataset are deemed unsuitable for MOSAC experiments according to the following two criteria:

- An action class must be defined only by its motion – not by its scene context. Certain actions in the dataset (e.g. *open/close fridge*, *open/close drawer*, *put on cutting-board*) cannot be identified from the action data alone, but also require knowledge of the scene context. If these action classes were included, it would artificially inflate the improvement in accuracy that MOSAC shows over ordinary clustering methods.

- Each included action class must have more than thirty examples in the dataset – certain actions, such as *puree*, *mix*, and *stamp*, have too few examples for good clustering.

The resulting subset of the dataset has the 17 following classes: *screw on*, *screw off peel*, *stir*, *pour*, *wash objects*, *strew*, *spice*, *grate*, *shake*, *throw in garbage*, *cut apart*, *cut dice*, *cut off ends*, *cut out inside*, *cut slices*, *cut stripes*. It is proposed that this subset provides a fair evaluation of the MOSAC algorithm for the reasons outlined above.

6.6.2 Setup

The setup for the extraction of motion features is identical to that in §5.4.2 above, and static SIFT features are extracted according to §6.5. A Bag-of-Words representation is generated for both action and static features (using the process described in §5.4.2) and Ng et al.'s [132] spectral embedding is calculated on the action representation only. The static representation is left in Bag-of-Words form so that it is compatible with topic models such as LDA and MOSAC.

Most of the experimental work, including the processing of the dataset, is performed in MATLAB. MOSAC inference, however, is not suitable for

Method	Accuracy (%)
k-means	36.56
DAkM	39.76
MOSAC	**40.82**

Table 6.4: Action clustering results on the MPII dataset using various techniques

implementation in MATLAB due to its heavily iterative nature, so is written in C++ instead.

6.6.3 Results

Table 6.4 shows the performance of clustering on the action representations, using various methods. As can be seen from the performance of k-means, if the scene context of the actions is ignored, performance is relatively poor, due to the fine-grained nature of the actions in the dataset.

Figure 6.5 shows the effect of varying the number of scene topics on the accuracy of the action clustering. There is a peak at 11 topics, and increasing the number of topics beyond this has little effect on accuracy – it is therefore preferable to set the number of topics relatively low, because the extra computational cost of finding more topics yields no better results.

As with the DAkM/SDAkM experiments above, the MPII cooking activities dataset was unfortunately not annotated with scene data to allow the quantitative analysis of scene clustering accuracy, so these results were not included.

6.7 Discussion

This chapter introduced the concept of MOSAC - Multiple-Object Single-Action Clustering. MOSAC is designed to perform dual-assignment clustering even when one of the views has multiple topics, and can be better captured with a topic model. Its performance over topic models that don't take into account contextual information (such as LDA) has been proven experimentally.

Despite the positive findings, this chapter has only presented an initial exploration of applying complex probabilistic graphical models to the problem

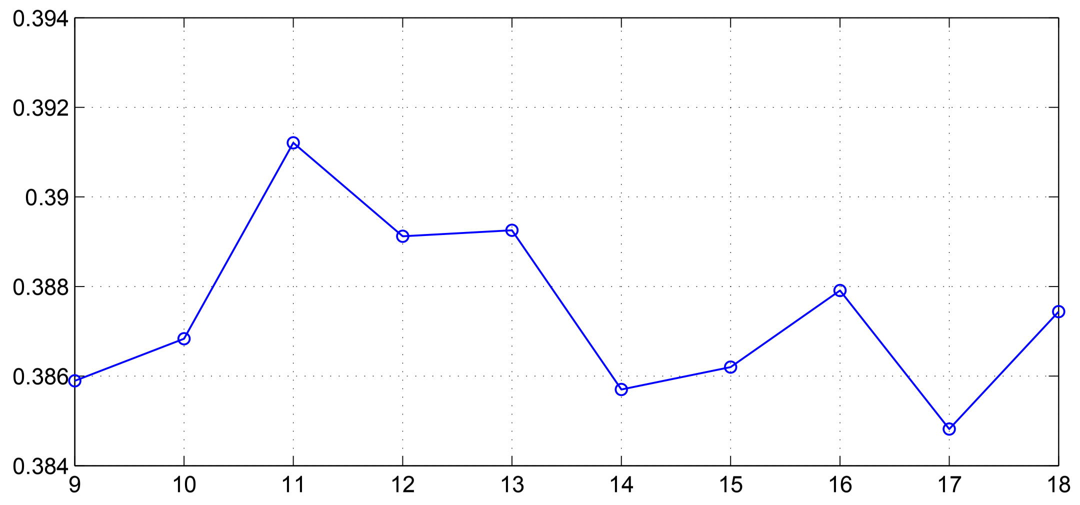


Figure 6.5: How the action clustering performance of MOSAC on the MPII dataset varies with the number of scene topics

of dual-assignment clustering. There are many questions that remain to be answered. Firstly, it may be possible to use different inference methods, such as Gibbs sampling, rather than the proposed Nested EM algorithm, to get better clustering results. Additionally, MOSAC could be modified to perform dual-assignment clustering where even *both* of the views have multiple topics – this could be achieved by extending the model to have two LDA-like branches. With the recent popularity of Big Data, both MOSAC and DAkM (from the previous chapter) should be investigated for their applicability to larger datasets, especially through the improvement of their computational complexities with respect to the number of scene and action categories.

Chapter 7

Multi-Spectral Representation for Human Actions

7.1 Introduction

Graph-based algorithms are a powerful way of exploiting the underlying structure of a dataset to improve the performance of unsupervised and semi-supervised tasks. To illustrate this, consider three of the most successful graph-based methods: spectral clustering [132], which can be used to find unusually structured clusters; manifold ranking, which has been applied to information retrieval tasks with great success [88]; and Laplacian Eigenmaps (LE) [139], which are applied to dimensionality reduction. In general, by using graph-based methods, it is possible to uncover the latent structure of a high-dimensional dataset, thereby improving the accuracy of unsupervised and semi-supervised learning tasks.

The first step of these graph-based methods is to generate an affinity matrix, W, which represents the affinity between every pair of points in dataset X. For bag-of-features (BoF) histograms (which are the focus of this chapter), it is possible to use a heat kernel applied to the χ^2 distance between every pair of points $x_i, x_j \in X$:

$$W_{ij} = exp\left(-\frac{\chi^2(x_i, x_j)}{\sigma^2}\right) \tag{7.1}$$

Alternatively, the histogram intersection could be used. After W is generated, various further operations are performed on W to get the final result. In certain methods, such as LE [139], W is made sparse using kNN or ϵ neighbourhoods, but in others it is fully connected.

Nonetheless, all graph-based methods for representation share the same flaw: when W is generated from X, there is significant information loss from the original feature space – only a single affinity value is generated for every pair of points. The information loss is particularly severe for small datasets (each row of W is therefore low-dimensional) or when the dataset's original feature space has a high dimensionality.

On high-dimensional data such as histograms, a single graph, generated using a single affinity metric, is not often sufficient to capture the full structure present in the original feature space. When representing realistic images or videos, there may be multiple statistically independent components within the histogram – then, a single graph would not be able to distinguish between these components. Instead, it is suggested that multiple graphs should be constructed, each corresponding to a different component of the original images or videos.

This chapter presents a novel method that generates multiple graphs from independent subsets of the feature space. First, multiple graphs are found by partitioning the feature space into several mutually independent subspaces, then generating a different affinity matrix from each subspace. Then, a spectral embedding method is performed on each subspace. Finally, the embeddings are scaled and concatenated together, resulting in a single representation for each datapoint. This representation is referred to as the Feature Grouped Spectral Multigraph (FGSM). It is expected that FGSM will result in minimal information loss from the original feature space compared to ordinary spectral embedding methods. Through experimentation on several human action datasets, it is demonstrated that FGSM gives superior results compared to the state-of-the-art algorithms for clustering, retrieval and recognition tasks.

The rest of this chapter is structured as follows. §7.2 describes the theory and implementation details of the Feature Grouped Spectral Multigraph. §7.3 details various experiments on clustering, retrieval, and recognition, and §7.4 concludes with a discussion of the chapter's findings.

7.2 Multigraph Representation

When applying FGSM to a dataset, the original feature space of the dataset should have two properties: 1) the feature space must be high dimensional, and 2) the feature set must be divisible into several disjoint subsets with high independence between all the subsets and high dependence within the subsets. It is proposed that these properties apply to the histogram representation of videos due to the locality of the features – each histogram bin is primarily associated with a different component of the original video. This concept is illustrated in Figure 7.1.

In ordinary graph-based learning methods, much of the information from the original feature space will be lost in the creation of the affinity graph. FGSM, however, overcomes this issue by finding multiple independent views (subspaces) of the original representation and generating a separate affinity graph for each view.

The full algorithm for FGSM is shown in Algorithm 3.

7.2.1 Feature Grouping

The first step is to extract several mutually independent subspaces from the original feature space. This can be achieved by spectrally clustering features on an affinity graph of Hilbert-Schmidt Independence Criterion (HSIC) values, calculated between every pair of features.

HSIC captures all *non-linear* dependencies between two random variables x and y, as described in Gretton et al. [140], so long as the associated reproducing kernel Hilbert spaces are universal. It is more suitable for this work's purposes

Algorithm 3: FGSM – Multigraph Representation

```
   Data:
   $X$ - a histogram representation of a dataset
   $m$ - the number of feature subspaces to find
   $k$ - the number of eigenvectors per feature subspace
   Result: $Y$ - a multigraph representation of the dataset
 1 Calculate HSIC affinity matrix between pairs of columns of $X$, where
   $W_{jk} = tr((L_j^T L_k)(L_k^T L_j))$ (Eqn 7.3)
 2 Spectrally cluster $W$ according to Ng et al. [132] to find $m$ feature clusters:
   $C_1..C_m$
 3 Define functions $P_1..P_m$ to project $X$ into feature subspaces according to
   $C_1..C_m$
 4 for $i \leftarrow 1$ to $m$ do
 5     Calculate $T \leftarrow P_i(X)$
 6     Calculate $W_{jk} \leftarrow \sum(min(T_j, T_k))$
 7     Calculate $S \leftarrow D^{-1/2} L D^{-1/2}$, where $L \leftarrow D - W$ and $D$ is a diagonal
       matrix where $D_{ll}$ is set as the sum of values of the $l$th row of $W$
 8     Find first $k$ eigenvectors $e_1..e_k$ of $S$, concatenate them columnwise:
       $E_i \leftarrow [e_1..e_k]$
 9     Normalise rows of $E_i$ to sum to 1
10     Find $\lambda_i$ as the mean distance between rows in $M_i$: $\lambda_i \leftarrow \sigma(dists(E_i))$
11 Concatenate scaled $E_1..E_m$ columnwise to get $Y$:
   $Y \leftarrow [(\lambda_1^{-1} E_1)..(\lambda_m^{-1} E_m)]$
```

than other independence measures, such as the correlation co-efficient, which only capture linear dependencies. To demonstrate that it is a true independence measure, Gretton et al. show it equals zero if and only if x and y are independent. It can be empirically estimated from a finite number of (x_i, y_i) tuples by the following:

$$\rho_h(x, y) = \frac{1}{(1-n)^2} tr(HK_x HK_y) \tag{7.2}$$

where $H_{ij} = \delta_{ij} - n^{-1}$, K_x and K_y are the outer products of vectors x and y respectively, and n is the number of samples. Calculating K_x and K_y, however, takes $O(n^2)$ time and space, which is highly expensive for larger datasets, so incomplete Cholesky decomposition is used to find L_x and L_y, such that K_x and K_y can be approximated as $K'_x = L_x L_x^T$ and $K'_y = L_y L_y^T$. The approximate HSIC can then be calculated using the following:

$$\rho_h(x, y) = tr((L_x^T L_y)(L_y^T L_x)) \tag{7.3}$$

This completes in $O(nf^2)$ time, where f is the chosen number of columns in L. On very large datasets, HSIC estimation can be made more efficient by sparsely sampling the original population. Such sampling can be done with acceptable loss of accuracy, as the estimated HSIC approaches the true HSIC at speed $\frac{1}{\sqrt{n}}$.

To perform feature grouping, ρ_h is calculated on every pair of features i and j in the original feature space of dataset $\mathbf{x}$, resulting in affinity graph $W_{ij} = \rho_h(\mathbf{x}_i, \mathbf{x}_j)$. Spectral clustering is performed on W according to Ng et al. [132] to find m disjoint feature subspaces, $\mathbf{s}_1, .., \mathbf{s}_m \subset \mathbf{x}$. A large range of values for m give good results, as shown in experiments below, so this choice is not crucial. Nonetheless, $m \geq 20$ typically achieves the best results.

7.2.2 Multigraph Spectral Embedding

Having obtained m disjoint subspaces, it is possible to find m separate embeddings of the dataset according to each subspace. For each subspace s_m, an affinity graph W_m is constructed using:

$$W_{m,ij} = \sum_{d_m} (min(P_m(x_i), P_m(x_j))) \tag{7.4}$$

where $P_m(x)$ is a function that maps x to the mth, d_m-dimensional subspace. Rather than using a kNN-neighbourhood or a ϵ-neighbourhood graph, as typically used in Laplacian Eigenmaps, W is constructed as a fully connected graph as in Ng et al.[132]. The choice to use a fully connected graph is made empirically – in preliminary experiments, a fully connected graph gave better results than a kNN neighbourhood graph for any k.

Spectral embedding is then performed on W_m as per steps 2-4 of Ng et al. to find a spectral embedding. These steps are:

1. Find $L = D^{-1/2} W D^{-1/2}$, where D is a diagonal matrix where D_{ii} is set as the sum of values of the ith row of W.

2. Find the k highest eigenvectors of L, $e_1, .., e_k$, and construct a matrix E columnwise as $[e_1..e_k]$.

3. Normalise E so each row sums to 1.

It is notable that this process also differs from Laplacian Eigenmaps, because of step 3, instead follows Ng et al.[132]. The unit normalisation is important to reduce the scale variation between the m separate embeddings. The optimal choice of k is likely to vary between spectral embeddings – however, for simplicity a single k is chosen that is uniform across all embeddings. Future work might show improved performance heuristically choosing an individual k per embedding.

The final step to generate the FGSM is to aggregate the m embeddings. This can be simply and naively achieved by concatenating all $E_1, .., E_m$ columnwise: $X = [E_1..E_m]$. Then, row i of X is an $m \times k$ length vector describing sample x_i. While this scheme works well, however, further performance increases can be achieved by scaling each embedding appropriately before aggregation. The Euclidean distance is calculated between every pair of rows in E_i and it is used to find λ_i thus:

$$\begin{aligned} dists_{i,jk} &= ||e_{i,j} - e_{i,k}||^2 \qquad (7.5) \\ \lambda_i &= \sigma(dists_i) \qquad (7.6) \end{aligned}$$

where $\sigma(x)$ is the standard deviation of the values in x. Then, to get the final representation, scale each $E_1, .., E_m$ with $\lambda_1, ..., \lambda_m$ and concatenate columnwise: $X = [(\lambda_1^{-1} E_1)..(\lambda_m^{-1} E_m)]$. As a result, each embedding is scaled to have a total distance variation of 1.

Out-of-sample extension is not considered in this chapter, which could be necessary when performing time critical tasks such as recognition or retrieval. However, the Nyström approximation could be easily applied to each embedding separately, as in [141], to achieve out-of-sample extension.

7.3 Experiments

In this section FGSM is applied to various machine learning problems to demonstrate its applicability to real-world machine learning tasks. specifically consider several realistic human action datasets, although in future work FGSM could also be applied to image datasets.

7.3.1 Setup

The datasets used are the UCF YouTube, UCF Sports, HOHA2 and Olympic Sports datasets. These are described in §2.6.

To extract a motion representation of the actions, the publicly available code for dense trajectory feature extraction is used as presented in Wang et al. [133], with the default settings of the software. This results in a series of local features, each feature represented with 4 descriptors (HOG, HOF, MBH, Tr). PCA, then k-means, are performed on each of the 4 descriptors in turn. For k-means, $k = 4000$. A separate histogram is generated for each descriptor, and then the histograms are aggregated together. The result is a 16000-bin histogram per video. When performing FGSM, features are grouped in 30 subspaces (i.e., $m = 30$) and extracted 40 eigenvectors for each manifold ($k = 40$).

The method for clustering features given in this chapter is compared to the diffusion map method presented in Liu et al. [142], which is used for finding semantic words in bag-of-words models. To create a method for comparison, steps 1-3 of Algorithm 3 are replaced with the representation and clustering algorithm given in section 4 of [142]. The parameters for diffusion maps are optimised empirically. In the experimental results below, this hybrid algorithm is referred to as DM (for Diffusion Maps), and it is compared to FGSM both for clustering and retrieval below.

7.3.2 Clustering

First action clustering is considered. Given an action dataset with k action classes, the goal is to find k disjoint subsets of the action dataset so that each

Dataset	Clustering Accuracy (%)			
	SC1	SC2	DM	FGSM
YouTube	22.3	39.2	39.9	**42.6**
UCF Sports	32.6	68.0	27.6	**70.8**
Hollywood-2	18.2	33.6	34.5	**38.6**
Ol. Sports	23.1	39.7	40.3	**42.8**

Table 7.1: Clustering performance of various methods on each dataset.

subset contains only one action class. The FGSM representation can improve clustering on human actions by finding a strongly representative low-dimensional embedding of the original histograms.

To measure a clustering algorithm's performance in this chapter, the same performance metric is used as in [103]. If each cluster c contains datapoints $x_1, .., x_n$, and each datapoint is associated with a ground truth label $l_1, .., l_n$, the label l_c of cluster c is determined to be:

$$\arg\max_{l_c} \sum_{i=1}^{n} \begin{cases} 1 & \text{if } l_c = l_i \\ 0 & \text{otherwise} \end{cases} \tag{7.7}$$

The accuracy is then percentage of data points across the whole dataset that have the same label as their assigned cluster.

The results of various clustering methods are in Table 7.1. SC1 and SC2 are the methods presented in Shi and Malik [143] and Ng et al. [132], respectively. For each of these, the affinity matrix W is generated using the histogram intersection in Equation 7.4. For FGSM, the FGSM representation is applied to the dataset, followed by ordinary k-means clustering using the Euclidean distance. Each clustering algorithm is run for 100 trials and the mean accuracy over all results is shown.

As can be seen from the table, FGSM achieves superior results to any of the compared spectral clustering methods on all four datasets. As stated above, this is likely because a single graph is unable to capture all of the information in the original feature space, and because DM's feature clustering is not as accurate as ours. The UCF Sports dataset results are improved the least by FGSM (2.8% over ordinary spectral clustering) whereas in the Hollywood-2 dataset a 5%

accuracy boost is observed. The scale of the improvement appears to be related to the size of the dataset – the larger the dataset, the larger the improvement that FGSM gives. This intuitively makes sense, as spectral embedding methods tend to be more accurate for larger populations. Performing the initial feature grouping may also be more accurate on larger datasets.

7.3.3 Content-based Retrieval with Relevance Feedback

Next, content-based video retrieval (CBVR) is considered. This is recently a popular research field, although the bulk of retrieval work is on images rather than videos. Typically CBVR is aimed at improving the accuracy of multimedia search engines.

The formal aim of CBVR is as follows: given a query video, rank the videos in a video database according to their relevance to the query, and return the most relevant videos. Once a query has been submitted and the results returned, a user can give relevance feedback to the system, by marking each result item as "positive" or "negative", indicating which results are related to the query or not. The CBVR system can incorporate this relevance feedback to perform a further query, and return improved results.

As discussed in the literature review, recent works [144, 145] have shown how manifold ranking can be made efficient enough for practical use on retrieval tasks. However, the rankings are generated from a single graph, so FGSM is likely to outperform manifold ranking.

Performance is compared between histogram intersection ranking (HI), χ^2 distance ranking (CD), manifold ranking (MR), Liu et al's method [142] (DM) and FGSM ranking (FGSM) in Table 7.2, showing retrieval performance before and after a single round of relevance feedback. To test, each video in the dataset is set as the query in turn, performing retrieval on the remaining videos. The percentage of relevant videos in the top 20 results is determined, and this is averaged over all the queries.

To simulate relevance feedback, several of the top 20 results are marked as positive or negative according to the ground truth of the dataset, and the query is

Dataset	RF	Accuracy of Top 20 Results (%)				
		HI	CD	MR	DM	FGSM
YouTube	B	53.2	52.4	56.4	56.6	**63.5**
	A	74.4	74.4	75.2	75.9	**82.8**
UCF Sp.	B	38.5	38.3	39.5	17.9	**41.5**
	A	48.0	48.6	49.0	28.5	**51.7**
HW-2	B	25.4	25.2	28.1	26.2	**30.4**
	A	41.0	41.1	42.7	33.9	**46.0**
Ol. Sp.	B	35.2	34.7	37.7	39.4	**41.5**
	A	51.9	51.4	51.4	54.4	**59.6**

Table 7.2: Retrieval performance of various methods on each dataset, before (B) and after (A) relevance feedback (RF).

rerun. For HI, CD, and FGSM relevance feedback is incorporated using a kernel SVM. For HI, the histogram intersection kernel is used; for CD, the χ^2 distance; for FGSM, the RBF kernel. To incorporate positive/negative relevance feedback in manifold ranking, scheme 1 is used as presented in [88], setting $\gamma = 0.25$.

As shown in the table, FGSM performs well for retrieval, especially after relevance feedback. A multigraph representation confers an advantage over the traditional manifold ranking (MR) method and DM, and gives the best performance for all four datasets. The histogram intersection and χ^2 distance both result in poor performance, as they do not leverage the underlying structure of the data. Figure 7.2 also shows the precision-recall curves for each dataset, making clear the advantage of FGSM.

Finally, in Figure 7.3 is shown the effects of varying parameter m on retrieval performance, compared against baseline manifold ranking performance. As can be seen from the figure, performance is vastly increased even when $m = 2$, and continues to rise until about $m = 30$ for all datasets. Performance is weakest compared to manifold ranking on the UCF Sports dataset, perhaps due to the small number of videos in that dataset.

7.3.4 Recognition

The final experiment is to perform fully-supervised action recognition using FGSM. Here, FGSM is not expected to outperform the state-of-the-art. Spectral

Dataset	Recognition Accuracy (%)			
	Orig	STP	LE	FGSM
YouTube	84.1	85.4	74.6	**89.0**
UCF Sports	88.0	**89.1**	67.6	87.7
Hollywood-2	58.2	**59.9**	45.8	58.2
Ol. Sports	74.1	**77.2**	57.5	74.6

Table 7.3: Recognition performance of various methods on each dataset.

embedding methods such as FGSM perform well on unsupervised and semi-supervised tasks because they make use of latent structural information in the unlabeled portion of the dataset. When performing fully supervised action recognition, however, all of the training data are labeled – it is not necessary to find the latent structure of fully labeled data. Instead, a discriminative classifier such as a kernel SVM can use all of the data to accurately model the separating hyperplane between classes even on the original feature space. If a spectral embedding method is used in conjunction with a fully-supervised SVM, much information will be lost from the original feature space, which will make an optimal hyperplane between classes *harder* to find.

So instead of outperforming the state-of-the-art, it is only intended to show that FGSM does not result in significant *loss* of recognition accuracy compared to the original representation, thus demonstrating that FGSM retains all important components from the original feature space.

In Table 7.3, FGSM is compared against the state-of-the-art human action recognition work in Wang et al. [43]. Orig is the dense trajectory histogram method presented in [43], using a multi-channel χ^2 kernel SVM for classification; STP is the spatio-temporal pyramid representation in [43] using a multi-channel χ^2 kernel SVM for classification; LE applies Laplacian Eigenmaps to the histogram and uses an RBF kernel SVM for classification; FGSM is the multigraph representation presented in this chapter. To evaluate classification with FGSM, a kernel SVM is applied to the FGSM representation of a dataset - determined empirically that an RBF kernel works better than a linear or quadratic kernel. For each dataset, the same experimental setup is used as that provided in Wang et al. [43]

As can be seen, on recognition tasks FGSM consistently outperforms Laplacian Eigenmaps, and performs similarly to the original histogram representation Orig. This illustrates that FGSM preserves the underlying structure of the dataset far better than a single spectral embedding. For one of the datasets – YouTube – FGSM even surpasses the state-of-the-art results in Wang et al., which is a surprising result, requiring further investigation. STP achieves the best results on the other three datasets, as it takes into account the spatio-temporal structure of the videos – in future, it may be possible to achieve even better results by combining STP with FGSM.

7.4 Discussion

In this chapter a new method is introduced for representing multimedia data – particularly human actions – for improved accuracy in clustering, retrieval and recognition tasks. Based on previous works on spectral embedding, several spectral embeddings were generated on separate subspaces of the original feature space. It was postulated that this would maximise the retained information from the original feature space. Through comprehensive experiments on four datasets, it has been demonstrated that the new representation – FGSM – can surpass the state-of-the-art for clustering and retrieval/relevance feedback tasks on all datasets, and can also surpass the state-of-the-art recognition accuracy on certain datasets.

Further work might consider how this representation performs on a greater variety of multimedia datasets, such as object image datasets, and larger-scale datasets such as HMDB51 - especially as spectral embedding becomes more accurate on larger populations. It would also be worthwhile to implement out-of-sample extension using the Nyström approximation, to improve the method's performance on retrieval tasks. Finally, works such as Xiang and Gong [146] have presented improvements to the basic spectral embedding method, which may also be combined with FGSM to enhance its performance.

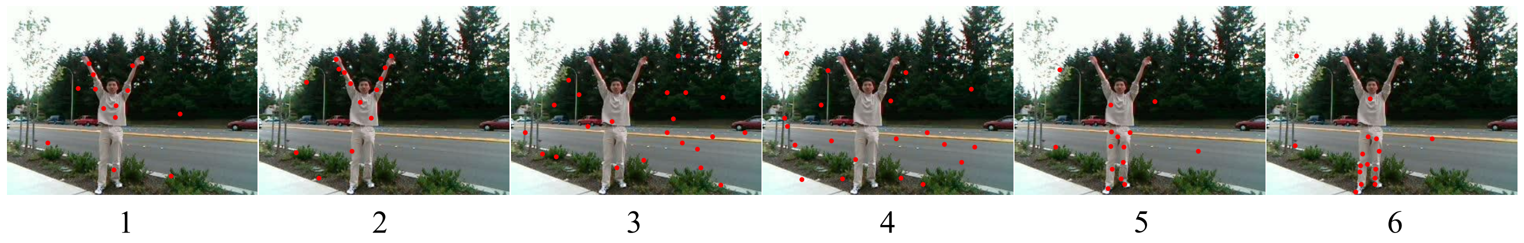


Figure 7.1: Certain BoF histogram bins may be associated more with one component of the video than any other. In this simplified 6-bin example, bins 1 and 2 are strongly associated with the upper body, bins 3 and 4 are strongly associated with the background, and bins 5 and 6 are strongly associated with the lower body. The high separability of the three components would make this histogram ideal for FGSM.

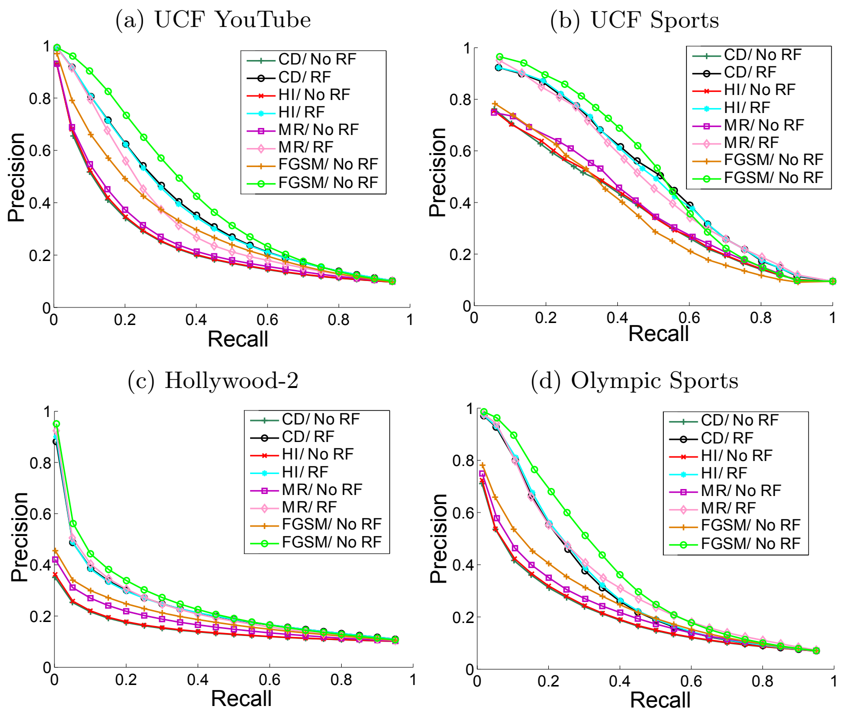


Figure 7.2: Precision/recall curves for video retrieval on various datasets.

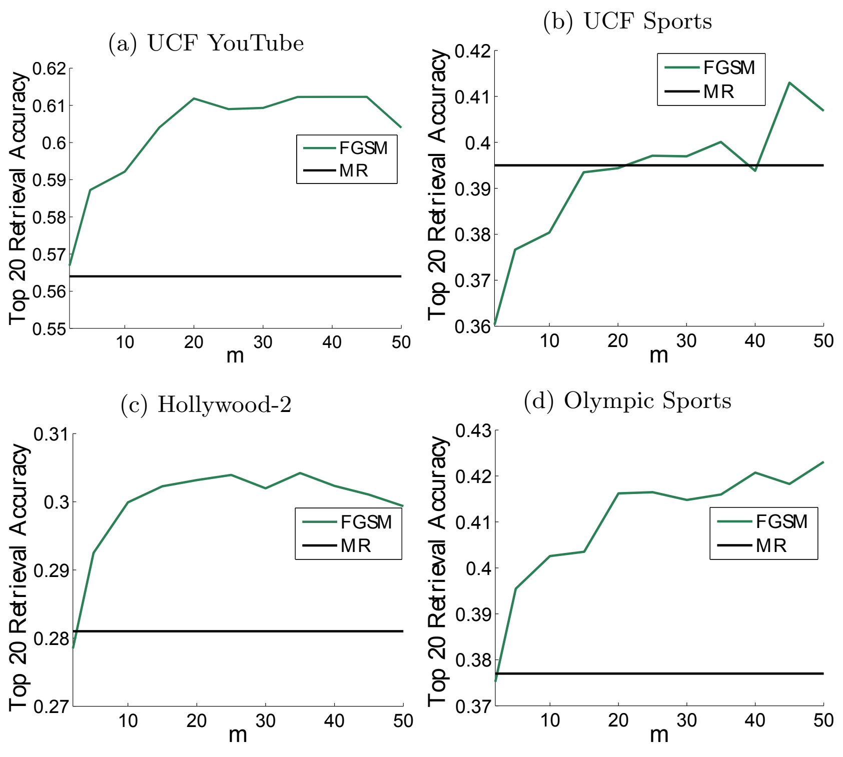


Figure 7.3: The effect of varying parameter m on retrieval performance in the top 20 results, versus manifold ranking baseline.

Chapter 8

Conclusions

This chapter summarises the findings of this thesis, tying together the various topics of work, and provides some direction for future work.

8.1 Discussion

The goal of this thesis, as provided in chapter 1, has been to devise methods for performing unsupervised and semi-supervised analysis of human actions, with a particular focus on providing groundwork for later work on practical systems such as human activity retrieval. In this section, several conclusions are drawn and related back to the original purpose of the thesis.

8.1.1 Action Retrieval

Firstly, it has been shown possible to perform human action retrieval with good accuracy, even on complex datasets such as the Hollywood 2 [3]. Further, the first approach introduced in chapter 4 can perform retrieval *with* localisation in at most linear time with respect to the length of the dataset while still maintaining a practical level of accuracy – far superior to previous works such as [116], which take at least quadratic time. The various experiments detailed in chapter 3 cover representation, ranking and feedback methods for action retrieval. It was shown that a relatively simple representation tends to work best for retrieval.

It is suggested that this is because more complex schemes like spatio-temporal pyramids require more training data to work effectively, but in a retrieval system only the query and feedback can be used to train the ranker.

8.1.2 Action Localisation and Temporal Segmentation

The first approach from chapter 4, while simple, performed efficiently and accurately on the datasets – by separating the temporal dimension from the spatial dimensions, great efficiency improvements were observed without significantly affecting the accuracy of the algorithm. The second approach performed fully unsupervised temporal localisation (segmentation) of the human actions. This is an important problem to solve, as purely supervised temporal localisation of actions is infeasible to apply to large datasets, due to the cost of acquiring the necessary training data. By temporally segmenting human actions into semantic chunks, it will become possible to perform higher level analysis of video content, such as performing activity retrieval or activity recognition. The work in this thesis has made a solid step in this direction, detailing an algorithm that segments human motion based on points of linear discontinuity, and proving that it is possible to create a practical recognition system based on unsupervised temporal segmentation.

8.1.3 Contextual Clustering

Dual-assignment clustering, an entirely novel clustering problem, has been introduced in chapter 5. Two algorithms to solve to this problem, DAkM and MOSAC, have been described, in chapters 5 and 6 respectively, and have been used to cluster actions with scene/object contexts. This is a considerable innovation – the context of human actions have been shown for some time to have a significant impact on the correct classification of actions, but no one has considered applying this in an unsupervised context. The simultaneous estimation problem posed is difficult to overcome, but experiments have shown both for the DAkM and MOSAC approach that dual-assignment clustering can consistently beat existing clustering methods on action-with-context problems – even similar algorithms such as multi-view clustering algorithms perform more poorly than DAkM.

8.1.4 Multigraph Representations

The novel multigraph representation provided in chapter 7 has been shown superior to existing methods for action representation when performing unsupervised or semi-supervised learning tasks. Ordinary spectral embedding methods are highly lossy, because they are dependent on a single similarity measure between every pair of datapoints – this can lose some of the subtle similarities or differences which might be important in, for instance, retrieval with relevance feedback. Intuitively, this representation preserves more important information than a single graph representation, as is evidenced by the strong results in chapter 7. The consequences of this finding may extend beyond human action representation, and may also be useful in image or document retrieval.

8.2 Future Work

There are many ways that the work presented herein could be extended and improved.

Perhaps the most promising future work is the combination of all of these chapters into a unified human activity retrieval system, as proposed in chapter 1. The work in chapters 3, 4 and 7 provide groundwork for the retrieval of atomic actions of such a system. Chapter 4 additionally provides a way to extract these atomic actions from a longer, temporally unsegmented dataset. Chapters 5 and 6 allows atomic actions to be represented as clusters. What remains is to devise a way to describe actions at a semantic level, extract this semantic description from a query video, and use it to assist in a full *activity* search rather than an *action* search. Various activity recognition techniques, such as that in [75] could be used as inspiration for this work. The main drawback of this direction is that in order to test such a system, a new, large dataset will need to be created, as no existing dataset is sufficient for an experimental evaluation of an activity retrieval system.

Each of the individual chapters can also be extended in various ways:

- Works extending Chapters 3 and 7 could consider how well the Fisher Vector representation [32] performs on human action retrieval. This promising technique has been used with great success on various other similar tasks such as image classification and should extend well into human action retrieval.

- Works extending Chapter 4 could focus on how to properly represent and index human actions for a logarithmic or even constant time human action search in unsegmented human action datasets. While the work presented here is already a significant improvement in computational complexity, practical systems for real-world use must be insensitive to the size of the database involved – to reason why, consider that YouTube currently has many millions of hours worth of video to search, and this database grows at over 100 hours per minute. It would be infeasible to search such a database in anything slower than logarithmic time, with low coefficients. This is important for surveillance systems too. Additionally, both approaches in this chapter should consider more complex datasets, and how to extend the methods to perform unsupervised *hierarchical* localisation, such as that in [83]. This could provide a more semantic level localisation and segmentation, which would be useful for a full *activity* retrieval or recognition system.

- Works extending Chapter 5 and 6 could take various directions. SDAKM and MOSAC could be experimented with in a variety of different domains – human actions are only one of many potential applications. Dual-assignment clustering could be extended to tri-assignment or n-assignment clustering. Other clustering algorithms, such as agglomerative clustering, could potentially be extended to the dual-assignment paradigm.

Glossary

ABRS-SVM Asymmetric Bagging Random Subsampling Support Vector Machine. A **relevance feedback** technique used in order to overcome several deficiencies associated with using an ordinary **Support Vector Machine** for relevance feedback.

Active Learning A technique used in retrieval systems, similar to **Relevance Feedback**. Unlike in relevance feedback, the items that the user should provide feedback on are chosen by an algorithm in order to maximise their expected information value..

Bag of Words In this thesis, an action representation model based on local features. The local features are extracted then clustering is performed on the features to group the features into codewords. A single histogram of codewords is used to represent each action in a dataset.

BoW See **Bag of Words**.

Branch-and-Bound In this thesis, refers to a method for localisation that can be used in conjunction with local features to localise in $O(xy)$ time, rather than $O(x^2y^2)$ time for a sliding window approach.

CBIR See **Content-based Information Retrieval**.

Classification A fully supervised machine learning task where each item in a dataset is assigned to one of k classes..

Clustering, Action Clustering applied to human actions.

Codebook A full mapping of local features to codewords.

Codeword A single quantised descriptor in the Bag of Words model. If the Bag of Words model involves some sort of feature clustering, such as k-means, then each cluster is equivalent to a codeword.

Content-based Information Retrieval A search technique, whereby a multimedia database is searched by comparing the *content* of the multimedia items to the contents of a query item. This is in contrast to keyword-based information retrieval, which searches a multimedia database based on the textual metadata attached to the multimedia items.

Context In this thesis, context refers to information in an image/video that is indirectly related to the subject of interest, and that can potentially provide clues as to the identity of the subject.

DAKM See **Dual-Assignment K-means Clustering**.

Dimensionality Reduction A machine learning task, where a set of high-dimensional vectors describing a dataset are reduced to a set of lower-dimensional vectors, by exploiting redundancies in the data.

Dual-Assignment K-means Clustering A novel technique introduced in this thesis in order to perform **Action Clustering** using **Context** to improve accuracy..

Expectation-Maximisation An optimisation technique used to perform inference in certain probabilistic models, such as LDA and GMMs.

Feature Description The second part of **feature extraction**, after a feature has been detected, where the feature patch is vectorised using a technique such as **HOG** or **HOF**.

Feature Detection The first part of **feature extraction**, whereby salient or discriminative features are localised within an image/video.

Feature Extraction The technique of finding and describing **local features** in an image or video in order to represent the image/video for machine learning tasks.

Feature Grouped Spectral Multigraph A new technique for **Action Representation** introduced in this thesis that exploits the **Spectral** properties of an action dataset in order to give good results on **unsupervised** and **semi-supervised** tasks.

FGSM See **Feature Grouped Spectral Multigraph**.

Gaussian Mixture Model A probabilistic graphical model that represents the data as being generated from a set of independent Gaussian distributions.

Global Feature As opposed to a local feature which is only concerned with a small patch of the image/video, a global feature is a description of an image/video which takes into account the appearance of the overall image/video.

GMM See **Gaussian Mixture Model**.

Ground Truth Information, for instance class labels or localised (x, y) coordinates, associated with each instance in a training dataset, containing information required for performing fully supervised classification.

Hidden Markov Model A graphical model used to model time series data – used in particular for speech and action recognition. Exploits the Markov assumption that a state at $t + 1$ is conditionally independent of the state at $t - 1$, given state t .

Histogram of Oriented Flow A technique commonly used for feature description, where the optical flow in the local feature is binned into several orientations and represented as a histogram.

Histogram of Oriented Gradients A technique commonly used for feature description, where the gradients in the local feature are binned into several orientations and represented as a histogram.

HMM See **Hidden Markov Model**.

HOF See **Histogram of Oriented Flow**.

HOG See **Histogram of Oriented Gradients**.

k Nearest Neighbours A basic classification technique for multiple classes that is computationally expensive for large training sets.

Kadane's Algorithm The canonical best solution to the Maximal Subarray problem. The goal is to find the contiguous subarray within a larger array of numbers that sums to the maximum possible total. Related to **branch-and-bound**.

Keyframe A frame of significance extracted from a video. Extraction of keyframes is used as opposed to extracting every frame in order to reduce information processing requirements. .

K-means Clustering A simple **unsupervised clustering** technique that finds K clusters in a given dataset..

kNN See **k Nearest Neighbours**.

Label See **Ground Truth**.

Latent Dirichlet Allocation An unsupervised graphical model originating from natural language processing. Given a set of text documents, LDA can discover the topics contained within each document, and the words associated with each topic. LDA has also been extended to scene understanding.

LDA See **Latent Dirichlet Allocation**.

Local Feature A discriminative part of a image/video extracted from a small 2D or 3D patch in an image or video respectively, used to build up a representation of the image/video.

Localisation, Action The task of determining the position of an action in space and/or time.

Manifold Ranking A technique for content-based information retrieval, that relies on **spectral** analysis of the database to provide good results.

MOSAC See **Multiple Object Single Action Clustering**.

Motion History Image A primitive global feature method for describing a human action, based on extracting a human silhouette from the video.

Multiple Object Single Action Clustering An extension of the Dual-Assignment K-means technique to account for **Context** with multiple topics..

Naive Bayes Nearest Neighbour A computationally complex method for **classification** using a set of **local features**.

NBNN See **Naive Bayes Nearest Neighbour**.

Occlusion The obstruction from view of a person/object of interest in a image/video. Can be partial (only part of the person/object is blocked from view) or full.

PCA See **Principal Components Analysis**.

Pose Estimation In this thesis, refers to the estimation of a person's pose from an image/video, specified in terms of the joints of their body.

Poselets A method for human detection from images, even in the case of significant partial **occlusion** or in unusual poses.

Principal Components Analysis In this thesis, a type of **dimensionality reduction**. Each of the output dimensions is generated as a linear combination of the original dimensions..

Probabilistic Graphical Model A type of statistical model that captures the probabilistic conditional dependencies between random variables in a graph format.

Recognition, Action Classification applied to human actions. See **Classification**.

Relevance Feedback A technique used in retrieval systems, in which a user can offer feedback on how relevant the retrieval system's initial results were, and this feedback is used to iteratively improve the accuracy of the retrieval system.

Representation, Action In this thesis, refers to a vectorised representation of a human action that is conductive to machine learning, as opposed to its initial 3D pixel representation.

Retrieval, Action Information Retrieval applied to human actions. See **Content-based Information Retrieval**.

SDAKM See **Spectral Dual-Assignment K-means Clustering**.

Segmentation Determination of the boundaries (either spatial boundaries or temporal boundaries) that separate a person, object, action, etc, from its surroundings. In this thesis, temporal segmentation is considered.

Semi-supervised Learning A type of machine learning where the training data includes a mixture of labeled and unlabeled samples. For instance, see **Retrieval, Action**..

Spatio-temporal Referring to both space and time. In this thesis, space typically refers to a 2D frame of a video, and time refers to a 3rd temporal dimension of the video..

Spectral In this thesis, spectral refers to the use of the spectrum of the similarity matrix of a dataset. Most commonly used to refer to Spectral Clustering.

Spectral Dual-Assignment K-means Clustering A spectral extension of **Dual-Assignment K-means Clustering** to apply it to noisy/irregularly clustered data.

Supervised Learning A type of machine learning that makes use of training data where all the training data is given **Ground Truth Labels**.

Support Vector Machine A classification technique (for two classes) that works by specifying a hyperplane in the feature space that maximally separates the two classes .

SVM See **Support Vector Machine**.

Unsupervised Learning A type of machine learning performed on unlabeled data, that somehow attempts to exploit the underlying structure for some task. Examples include **Clustering, Action** and **Dimensionality Reduction**.

Bibliography

[1] J. Choi, W. J. Jeon, and S.-C. Lee, "Spatio-temporal pyramid matching for sports videos," in *ACM SIGMM International Conference on Multimedia Information Retrieval*, pp. 291–297, 2008.

[2] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing Human Actions: A Local SVM Approach," in *Proceedings of the International Conference on Pattern Recognition*, vol. 3, pp. 32–36, 2004.

[3] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning Realistic Human Actions From Movies," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[4] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.

[5] J. W. Davis and A. F. Bobick, "The Representation and Recognition of Human Movement Using Temporal Templates," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, p. 928, 1997.

[6] J. Niebles, B. Han, and L. Fei-Fei, "Efficient Extraction of Human Motion Volumes by Tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 655–662, IEEE, 2010.

[7] J. Yamato, J. Ohya, and K. Ishii, "Recognizing Human Action in Time-Sequential Images using Hidden Markov Model," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 379–385, 1992.

[8] X. Feng and P. Perona, "Human Action Recognition By Sequence of Movelet Codewords," *International Symposium on 3D Data Processing Visualization and Transmission*, p. 717, 2002.

[9] D. Weinland, E. Boyer, and R. Ronfard, "Action Recognition from Arbitrary Views using 3D Exemplars," in *IEEE International Conference on Computer Vision*, pp. 1–7, 2007.

[10] N. Oliver and E. Horvitz, "Layered representations for human activity recognition," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3–8, 2002.

[11] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh, "Activity Recognition and Abnormality Detection with the Switching Hidden Semi-Markov Model," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 838–845, 2005.

[12] H. Ikeda, M. Maeda, N. Kato, and H. Kashimura, "Classification of human actions using face and hands detection," in *Proceedings of ACM Multimedia*, pp. 484–487, 2004.

[13] M. Andriluka, S. Roth, and B. Schiele, "Monocular 3d pose estimation and tracking by detection," *Computer*, no. 2, 2010.

[14] G. Junxia, D. Xiaoqing, W. Shengjin, and W. Youshou, "Full body tracking-based human action recognition," in *Proceedings of the International Conference on Pattern Recognition*, pp. 1–4, Ieee, Dec. 2008.

[15] L. Shao, X. Zhen, D. Tao, and X. Li, "Spatio-temporal laplacian pyramid coding for action recognition," *IEEE Transactions on Cybernetics*, vol. 44, no. 6, pp. 817–827, 2014.

[16] H. Kuehne and H. Poggio, "HMDB: A Large Video Database for Human Motion Recognition," in *IEEE International Conference on Computer Vision*, 2011.

[17] G. Johansson, "Visual perception of biological motion and a model for its analysis," *Attention, Perception & Psychophysics*, 1973.

[18] I. Laptev, "On Space-Time Interest Points," *International Journal of Computer Vision*, vol. 64, no. 2-3, pp. 107–123, 2005.

[19] K. Mikolajczyk, "An affine invariant interest point detector," *Proceedings of the European Conference on Computer Vision*, 2002.

[20] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, pp. 761–767, Sept. 2004.

[21] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[22] P. Scovanner, S. Ali, and M. Shah, "A 3-Dimensional SIFT Descriptor and its Application to Action Recognition," in *Proceedings of ACM Multimedia*, pp. 357–360, 2007.

[23] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior Recognition via Sparse Spatio-Temporal Features," *Proceedings of the IEEE Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 65–72, 2005.

[24] H. Ning, Y. Hu, and T. Huang, "Searching Human Behaviors Using Spatial-Temporal Words," in *Proceedings of the IEEE International Conference on Image Processing*, pp. 337–340, 2007.

[25] M.-y. Chen and A. Hauptmann, "MoSIFT : Recognizing Human Actions in Surveillance Videos," tech. rep., Carnegie Mellon University, 2009.

[26] H. Shabani, D. a. Clausi, and J. S. Zelek, “Towards a Robust Spatio-Temporal Interest Point Detection for Human Action Recognition,” in *Proceedings of the Canadian Conference on Computer and Robot Vision*, pp. 237–243, Ieee, May 2009.

[27] A. Kläser, M. Marszałek, and C. Schmid, “A Spatio-Temporal Descriptor Based on 3D-Gradients,” in *Proceedings of the British Machine Vision Conference*, pp. 995–1004, 2008.

[28] X. He, “Locality Preserving Projections,” in *Advances in Neural Information Processing Systems*, 2003.

[29] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, pp. 2323–2326, 2000.

[30] L. Zhang, P. Zhu, Q. Hu, and D. Zhang, “A Linear Subspace Learning Approach via Sparse Coding,” in *IEEE International Conference on Computer Vision*, vol. 1, Springer, Sept. 2011.

[31] O. Boiman, E. Shechtman, and M. Irani, “In Defense of Nearest-Neighbor Based Image Classification,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[32] F. Perronnin, J. Sánchez, and T. Mensink, “Improving the fisher kernel for large-scale image classification,” in *Computer Vision–ECCV 2010*, pp. 143–156, Springer, 2010.

[33] R. A. Redner and H. F. Walker, “Mixture Densities, Maximum Likelihood and the Em Algorithm,” *SIAM Review*, vol. 26, no. 2, pp. 195–239, 1984.

[34] K. Grauman and T. Darrell, “The pyramid match kernel: Discriminative classification with sets of image features,” in *IEEE International Conference on Computer Vision*, pp. 1458–1465, 2005.

[35] K. Grauman and T. Darrell, “Approximate correspondences in high dimensions,” in *Advances in Neural Information Processing Systems*, 2006.

[36] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2169–2178, 2006.

[37] S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 509–522, Apr. 2002.

[38] M. Körtgen, G. Park, and M. Novotni, “3D shape matching with 3D shape contexts,” in *Central European Seminar on Computer Graphics*, 2003.

[39] M. Grundmann and F. Meier, “3d shape context and distance transform for action recognition,” in *Proceedings of the International Conference on Pattern Recognition*, pp. 1–4, Dec. 2008.

[40] L. Shao and Y. Du, "Spatio-temporal Shape Contexts for Human Action Retrieval," in *Proceedings of the International Workshop on Interactive Multimedia for Consumer Electronics*, pp. 43–50, 2009.

[41] M. Bregonzio, S. Gong, and T. Xiang, "Recognising action as clouds of space-time interest points," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1948–1955, 2009.

[42] E. Shechtman and M. Irani, "Space-Time Behavior Based Correlation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 405–412, 2005.

[43] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International Journal of Computer Vision*, vol. 103, pp. 60–79, 2013.

[44] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1627–1645, Sept. 2010.

[45] L. Bourdev and J. Malik, "Poselets: Body part detectors trained using 3d human pose annotations," in *IEEE International Conference on Computer Vision*, 2009.

[46] M. Raptis, I. Kokkinos, and S. Soatto, "Discovering discriminative action parts from mid-level video representations," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1242–1249, June 2012.

[47] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *IEEE International Conference on Computer Vision*, pp. 1–8, 2007.

[48] M.-J. Escobar, G. Masson, T. Vieville, and P. Kornprobst, "Action Recognition Using a Bio-Inspired Feedforward Spiking Network," *International Journal of Computer Vision*, pp. 284–301, 2009.

[49] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[50] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," in *Proceedings of the International Conference on Machine Learning*, pp. 495–502, Citeseer, 2010.

[51] M. Baccouche, F. Mamalet, C. Wolf, and C. Garcia, "Sequential Deep Learning for Human Action Recognition," in *International Workshop on Human Behavior Understanding*, pp. 29–39, 2011.

[52] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014.

[53] D. Weinland, R. Ronfard, and E. Boyer, "Free Viewpoint Action Recognition Using Motion History Volumes," *Computer Vision and Image Understanding*, vol. 104, no. 2, pp. 249–257, 2006.

[54] P. Yan and S. Khan, "Learning 4d action feature models for arbitrary view action recognition," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–7, June 2008.

[55] B. Peng, G. Qian, and S. Rajko, "View-invariant full-body gesture recognition via multilinear analysis of voxel data," in *International Conference on Distributed Smart Cameras*, pp. 1–8, IEEE, Aug. 2009.

[56] P. Turaga, A. Veeraraghavan, and R. Chellappa, "Statistical analysis on Stiefel and Grassmann manifolds with applications in computer vision," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2008.

[57] R. Cilla, M. Patricio, A. Berlanga, and J. Molina, "Multicamera Action Recognition with Canonical Correlation Analysis and Discriminative Sequence Classification," *Foundations on Natural and Artificial Computation*, pp. 491–500, 2011.

[58] M. A. Naiel and M. M. Abdelwahab, "Multi-view human action recognition system employing 2DPCA," *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 2011.

[59] G. Srivastava, H. Iwaki, J. Park, and A. Kak, "Distributed and lightweight multi-camera human activity classification," in *International Conference on Distributed Smart Cameras*, pp. 1–8, IEEE, 2009.

[60] M. Marszalek, I. Laptev, and C. Schmid, "Actions in context," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2929–2936, IEEE, June 2009.

[61] Y. Jiang, Z. Li, and S. Chang, "Modeling Scene and Object Contexts for Human Action Retrieval with Few Examples," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 99, pp. 1–1, 2011.

[62] D. Han, L. Bo, and C. Sminchisescu, "Selection and context for action recognition," in *IEEE International Conference on Computer Vision*, pp. 1933–1940, 2009.

[63] M. Everingham, J. Sivic, and A. Zisserman, "Taking the bite out of automated naming of characters in TV video," *Image and Vision Computing*, vol. 27, pp. 545–559, Apr. 2009.

[64] F. Schroff, A. Criminisi, and A. Zisserman, "Harvesting Image Databases from the Web," in *IEEE International Conference on Computer Vision*, 2007.

[65] N. Ikizler-Cinbis and S. Sclaroff, "Object, scene and actions: combining multiple features for human action recognition," in *Proceedings of the European Conference on Computer Vision*, pp. 494–507, 2010.

[66] A. Prest, C. Schmid, and V. Ferrari, "Weakly supervised learning of interactions between humans and objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 601–614, 2012.

[67] J. Liu, B. Kuipers, and S. Savarese, "Recognizing human actions by attributes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3337–3344, 2011.

[68] J. Li, S. Gong, and T. Xiang, "Learning behavioural context," *International Journal of Computer Vision*, vol. 97, no. 3, pp. 276–304, 2012.

[69] A. Nefian, L. Liang, X. Pi, and L. Xiaoxiang, "A coupled HMM for audio-visual speech recognition," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 2013–2016, 2002.

[70] Q. Wu, Z. Wang, F. Deng, and D. D. Feng, "Realistic Human Action Recognition with Audio Context," *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications*, pp. 288–293, Dec. 2010.

[71] L. N. Abdullah and S. A. M. Noah, "Integrating Audio Visual Data for Human Action Detection," in *Proceedings of the International Conference on Computer Graphics, Imaging and Visualisation*, pp. 242–246, Aug. 2008.

[72] J. Sullivan and S. Carlsson, "Recognizing and Tracking Human Actions," in *Proceedings of the European Conference on Computer Vision*, 2002.

[73] A. Kläser, M. Marszalek, C. Schmid, and A. Zisserman, "Human Focused Action Localization in Video," in *International Workshop on Sign, Gesture, Activity*, 2010.

[74] T. H. Thi, J. Zhang, L. Cheng, L. Wang, and S. Satoh, "Human Action Recognition and Localization in Video Using Structured Learning of Local Space-Time Features," in *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance*, pp. 204–211, 2010.

[75] M. Ryoo and J. Aggarwal, "Spatio-temporal Relationship Match: Video Structure Comparison for Recognition of Complex Human Activities," in *IEEE International Conference on Computer Vision*, pp. 1593–1600, 2009.

[76] A. Oikonomopoulos, I. Patras, and M. Pantic, "Spatiotemporal Localization and Categorization of Human Actions in Unsegmented Image Sequences," *IEEE Transactions on Image Processing*, vol. 20, pp. 1126–1140, Apr. 2011.

[77] A. Roshan Zamir, A. Dehghan, and M. Shah, "Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs," in *Proceedings of the European Conference on Computer Vision*, pp. 343–356, 2012.

[78] J. Liu, J. Luo, and M. Shah, "Recognizing Realistic Actions from Videos "in the Wild"," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1996–2003, June 2009.

[79] X. Zhen, L. Shao, D. Tao, and X. Li, “Embedding motion and structure features for human action recognition,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 7, pp. 1182–1190, 2013.

[80] S. Ji, W. Xu, M. Yang, and K. Yu, “3d convolutional neural networks for human action recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[81] T. Xiang and S. Gong, “Activity based surveillance video content modelling,” *Pattern Recognition*, vol. 41, pp. 2309–2326, July 2008.

[82] F. Zhou, F. De la Torre, and J. Hodgins, “Hierarchical aligned cluster analysis for temporal clustering of human motion,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 3, pp. 582–596, 2013.

[83] P. Turaga, A. Veeraraghavan, and R. Chellappa, “Unsupervised view and rate invariant clustering of video sequences,” *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 353 – 371, 2009.

[84] A. López-Méndez, J. Gall, J. Casas, and L. van Gool, “Metric learning from poses for temporal clustering of human motion,” in *Proceedings of the British Machine Vision Conference*, pp. 49.1–49.12, 2012.

[85] F. Arman, R. Depommier, A. Hsu, and M.-Y. Chiu, “Content-based Browsing of Video Sequences,” in *Proceedings of ACM Multimedia*, pp. 97–103, 1994.

[86] H. J. Zhang, J. Wu, D. Zhong, and S. Smoliar, “An Integrated System for Content-based Video Retrieval and Browsing,” *Pattern Recognition*, vol. 30, no. 4, pp. 643–658, 1997.

[87] R. Jin and L. Shao, “Retrieving Human Actions Using Spatio-temporal Features and Relevance Feedback,” in *Multimedia Interaction and Intelligent User Interfaces* (L. Shao, C. Shan, J. Luo, and M. Etoh, eds.), Springer-Verlag, Sept. 2010.

[88] J. He, M. Li, H. jiang Zhang, H. Tong, and C. Zhang, “Manifold-ranking based image retrieval,” in *In ACM Multimedia*, pp. 9–16, 2004.

[89] R. Rahmani, S. A. Goldman, H. Zhang, J. Krettek, and J. E. Fritts, “Localized Content Based Image Retrieval,” in *ACM SIGMM International Conference on Multimedia Information Retrieval*, pp. 227–236, 2005.

[90] D. Zhang, F. Wang, Z. Shi, and C. Zhang, “Interactive Localized Content Based Image Retrieval With Multiple-Instance Active Learning,” *Pattern Recognition*, vol. 43, no. 2, pp. 478–484, 2010.

[91] H. J. Seo and P. Milanfar, “Action recognition from one example.,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, pp. 867–82, May 2011.

[92] G. Salton, *The SMART Retrieval System—Experiments in Automatic Document Processing.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1971.

[93] S. Tong and E. Chang, "Support Vector Machine Active Learning for Image Retrieval," in *Proceedings of ACM Multimedia*, pp. 107–118, 2001.

[94] P. Hong, Q. Tian, and T. S. Huang, "Incorporate Support Vector Machines to Content-Based Image Retrieval with Relevance Feedback," in *Proceedings of the IEEE International Conference on Image Processing*, vol. 3, pp. 750–753, 2000.

[95] D. Tao, X. Tang, X. Li, and X. Wu, "Asymmetric Bagging and Random Subspace for Support Vector Machines-Based Relevance Feedback in Image Retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1088–1099, 2006.

[96] Z. Zhang, R. Ji, H. Yao, P. Xu, and J. Wang, "Random Sampling SVM Based Soft Query Expansion for Image Retrieval," in *International Conference on Image and Graphics*, pp. 805–809, 2007.

[97] W. Bian and D. Tao, "Biased Discriminant Euclidean Embedding for Content-Based Image Retrieval," *IEEE Transactions on Image Processing*, pp. 545–554, 2010.

[98] X. Tian, D. Tao, X.-S. Hua, and X. Wu, "Active Reranking for Web Image Search," *IEEE Transactions on Image Processing*, pp. 805–820, 2010.

[99] D. Tao, X. Li, and S. J. Maybank, "Negative Samples Analysis in Relevance Feedback," *IEEE Transactions on Knowledge and Data Engineering*, pp. 568–580, 2007.

[100] R. Yan, A. Hauptmann, and R. Jin, "Negative pseudo-relevance feedback in content-based video retrieval," in *Proceedings of ACM Multimedia*, pp. 343–346, ACM, 2003.

[101] C. Zhang, S. Member, and T. Chen, "An active learning framework for content-based information retrieval," *Proceedings of ACM Multimedia*, vol. 4, pp. 260–268, June 2002.

[102] S. Tong and E. Chang, "Support Vector Machine Active Learning for Image Retrieval," in *Proceedings of ACM Multimedia*, pp. 107–118, 2001.

[103] Y. Yang, I. Saleemi, and M. Shah, "Discovering motion primitives for unsupervised grouping and one-shot learning of human actions, gestures, and expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1635–1648, 2013.

[104] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words," *International Journal of Computer Vision*, vol. 79, pp. 299–318, Mar. 2008.

[105] Y. Wang, H. Jiang, M. S. Drew, Z.-N. Li, and G. Mori, "Unsupervised discovery of action classes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1654–1661, 2006.

[106] T. Hospedales, S. Gong, and T. Xiang, "Video behaviour mining using a dynamic topic model," *International Journal of Computer Vision*, vol. 98, pp. 303–323, July 2012.

[107] M. D. Rodriguez, J. Ahmed, and M. Shah, "Action MACH: a spatio-temporal maximum average correlation height filter for action recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[108] "http://server.cs.ucf.edu/vision/data.html."

[109] J. Yuan, Z. Liu, and Y. Wu, "Discriminative video pattern search for efficient action detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1728–1743, 2011.

[110] H. Wang and C. Schmid, "Action Recognition with Improved Trajectories," in *IEEE International Conference on Computer Vision*, pp. 3551–3558, Dec. 2013.

[111] M. S. Ryoo and J. K. Aggarwal, "Ut-interaction dataset." ICPR contest on Semantic Description of Human Activities (SDHA), `http://cvrc.ece.utexas.edu/SDHA2010/Human_Interaction.html`, 2010.

[112] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele, "A database for fine grained activity detection of cooking activities," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012.

[113] L. Shao and R. Mattivi, "Feature Detector and Descriptor Evaluation in Human Action Recognition," in *Proceedings of the ACM International Conference on Image and Video Retrieval*, pp. 477–484, 2010.

[114] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.

[115] S. Jones, L. Shao, and K. Du, "Active learning for human action retrieval using query pool selection," *Neurocomputing*, pp. 89–96, 2014.

[116] G. Yu, J. Yuan, and Z. Liu, "Unsupervised Random Forest Indexing for Fast Action Search," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 865–872, 2011.

[117] J. Bentley, "Programming pearls: Algorithm design techniques," *Commun. ACM*, vol. 27, pp. 865–873, Sept. 1984.

[118] C. H. Lampert, M. B. Blaschko, and T. Hofmann, "Efficient subwindow search: A branch and bound framework for object localization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 2129–2142, Dec 2009.

[119] H. Ning, T. X. Han, D. B. Walther, M. Liu, and T. S. Huang, "Hierarchical space-time model enabling efficient search for human actions," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, pp. 808–820, June 2009.

[120] J. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proceedings of the European Conference on Computer Vision*, vol. 7575, pp. 702–715, 2012.

[121] M. Everingham, J. Sivic, and A. Zisserman, ““hello! my name is... buffy” – automatic naming of characters in tv video,” in *Proceedings of the British Machine Vision Conference*, 2006.

[122] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proc. 7th Int. Joint Conf. Artificial intelligence*, pp. 674–679, 1981.

[123] O. Boiman, E. Shechtman, and M. Irani, “In Defense of Nearest-Neighbor Based Image Classification,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[124] J. Liu and M. Shah, “Learning human actions via information maximization,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[125] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, Mar. 2003.

[126] S. Bickel and T. Scheffer, “Multi-view clustering,” in *International Conference on Data Mining*, pp. 19–26, 2004.

[127] C. Chen, Y.-C. Gong, and Y. Tian, “Kck-means: A clustering method based on kernel canonical correlation analysis,” in *International Conference on Computational Science*, pp. 995–1004, 2008.

[128] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan, “Multi-view clustering via canonical correlation analysis,” in *Proceedings of the International Conference on Machine Learning*, pp. 129–136, 2009.

[129] A. Kumar and H. Daume III, “A co-training approach for multi-view spectral clustering.,” in *Proceedings of the International Conference on Machine Learning*, pp. 393–400, 2011.

[130] X. H. Dang and J. Bailey, “Generation of alternative clusterings using the cami approach,” in *SIAM International Conference on Data Mining*, pp. 118–129, 2010.

[131] D. Niu, J. G. Dy, and M. I. Jordan, “Multiple non-redundant spectral clustering views.,” in *Proceedings of the International Conference on Machine Learning*, pp. 831–838, 2010.

[132] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” in *Advances in Neural Information Processing Systems*, pp. 849–856, 2001.

[133] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, “Action Recognition by Dense Trajectories,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3169–3176, 2011.

[134] K. P. Murphy, *Machine learning: a probabilistic perspective.* Cambridge, MA: The MIT Press, 2012.

[135] X. Wang and E. Grimson, “Spatial latent dirichlet allocation,” in *Advances in Neural Information Processing Systems*, 2007.

[136] L. Cao and L. Fei-Fei, "Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes," in *IEEE International Conference on Computer Vision*, pp. 1–8, IEEE, 2007.

[137] D. A. V. Dyk, "Nesting em algorithms for computational efficiency," *Statistical Sinica*, pp. 203–225, 2000.

[138] M. Jones and J. Rehg, "Statistical color models with application to skin detection," *International Journal of Computer Vision*, vol. 46, no. 1, pp. 81–96, 2002.

[139] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, pp. 1373–1396, 2002.

[140] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with hilbert-schmidt norms," in *Proc. Algorithmic Learning Theory*, pp. 63–77, Springer-Verlag, 2005.

[141] A. Talwalkar, S. Kumar, and H. A. Rowley, "Large-scale manifold learning," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[142] J. Liu, Y. Yang, I. Saleemi, and M. Shah, "Learning semantic features for action recognition via diffusion maps," *Computer Vision and Image Understanding*, vol. 116, no. 3, pp. 361–377, 2012.

[143] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905, 1997.

[144] B. Xu, J. Bu, C. Chen, D. Cai, X. He, W. Liu, and J. Luo, "Efficient manifold ranking for image retrieval," in *ACM SIGIR Conf. Research and Development in Information Retrieval*, pp. 525–534, 2011.

[145] J. He, M. Li, H. Zhang, H. Tong, and C. Zhang, "Generalized manifold-ranking-based image retrieval," *IEEE Trans. Image Processing*, vol. 15, no. 10, pp. 3170–3177, 2006.

[146] T. Xiang and S. Gong, "Spectral clustering with eigenvector selection," *Pattern Recognition*, vol. 41, pp. 1012–1029, Mar. 2008.