Segment-based SVMs for Time Series Analysis

Minh Hoai Nguyen

CMU-RI-TR-12-1

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Robotics

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Version: 20 Jan 2012

Thesis Committee:
Fernando De la Torre (chair)
Martial Hebert
Carlos Guestrin
Frank Dellaert (Georgia Tech)

Abstract

Enabling computers to understand human and animal behavior has the potential to revolutionize many areas that benefit society such as clinical diagnosis, human-computer interaction, and social robotics. Critical to the understanding of human and animal behavior, and any temporally-varying phenomenon in general, is the capability to segment, classify, and cluster time series data. This thesis proposes segment-based Support Vector Machines (Seg-SVMs), a framework for supervised, weakly-supervised, and unsupervised time series analysis. Seg-SVMs outperform state-of-the-art approaches by combining three powerful ideas: energy-based structure prediction, bag-of-words representation, and maximum-margin learning. Energy-based structure prediction provides a principled mechanism for concurrent top-down recognition and bottom-up temporal localization. Bag-of-words representation provides segment-based features that tolerate misalignment errors and are computationally efficient. Maximum-margin learning, such as SVM and Structure Output SVM, has a convex learning formulation; it produces classifiers that are discriminative and less prone to over-fitting.

In this thesis, we show how Seg-SVMs outperform state-of-the-art approaches for segmenting, classifying, and clustering human and animal behavior in video and accelerometer data of varying complexity. We illustrate these benefits in the problems of facial event detection, sequence labeling of human actions, and temporal clustering of animal behavior. In addition, the Seg-SVMs framework naturally provides solutions to two novel problems: early detection of human actions and weakly-supervised discovery of discriminative events.

Acknowledgements

My life over the last few years of graduate school has been fantastic. I have many people to thank for this, and I am afraid I can only list a few here.

First I must thank my advisor, Fernando De la Torre, for the many years of mentorship. His visionary advice and constant encouragement meant a lot to me; he steered and pushed me to achieve goals that I would have never tried myself, because I was too naive and too afraid of failure. Fernando taught me that success requires practice and initial failure is unavoidable.

I want to thank the remaining members of my thesis committee: Martial Hebert, Carlos Guestrin, and Frank Dellaert, each for taking the time to discuss my research progress and provide insightful comments.

During my times in graduate school, I had the opportunity work with and learn from many great people. I would especially like to thank Tomas Simon, Jeffrey Cohn, Lorenzo Torresani, Carsten Rother, and Zhen-Zhong Lan, my collaborators in parts of this thesis.

I am grateful to many of my friends in Pittsburgh. There are too many to name, but I would particularly thank Maxim Makatchev, my officemate, who I shared many years of companionship and laughter. I enriched my life by exposing myself to his awesomeness, which goes beyond the imaginary drops of water and the motherland spirit of vodka.

I would like to express gratitude to my parents, my grandpa, and my sister for many years of unconditional love and support. Though physically distant, they were always with me when I needed them. Finally, my recent happiest moments were all with you, Huyen. Thank you for your love, trust, and understanding. Thank you for providing me with the encouragement to pursue my dreams. I am grateful to our journey so far and I am excited about our adventure ahead.

Contents

List of Figures

List of Tables

Abbreviations

AAM	**A**ctive **A**pperance **M**odel
AMOC	**A**ctivity **M**onitoring **O**perating **C**haracteristic
BoW	**B**ag **o**f **W**ords
CCCP	**C**onc**c**ave-**C**onvex **P**rocedure
CRF	**C**onditional **R**andom **F**ield
DBN	**D**ynamic **B**ayesian **N**etwork
FACS	**F**acial **A**ction **C**oding **S**ystem
HMM	**H**idden **M**arkov **M**odel
MMC	**M**aximum **M**argin **C**lustering
MMED	**M**aximum **M**argin **E**arly Event **D**etector
MMTC	**M**aximum **M**argin **T**emporal **C**lustering
MCSVM	**M**ulti-**C**lass **S**upport **V**ector **M**achine
PCA	**P**rincipal **C**omponent **A**nalysis
ROC	**R**eceiver **O**perating **C**haracteristic
SIFT	**S**cale **I**nvariant **F**eature **T**ransform
SLDS	**S**witching **L**inear **D**ynamical **S**ystem
SOSVM	**S**tructured **O**utput **S**upport **V**ector **M**achine
SVM	**S**upport **V**ector **M**achine

Symbols

$\mathbf{X}, \mathbf{X}^i, \mathbf{X}^{+i}, \mathbf{X}^{-i}$	bold uppercase $\mathbf{X}$ denotes a time series
$\mathbf{x}_t, \mathbf{x}_t^i$	the t^{th} frames of $\mathbf{X}$ and $\mathbf{X}^i$ respectively
y, y^i, y_t, y_t^i	lowercase y denotes a class/cluster label
$\mathbf{z}, \mathbf{z}^i, \mathbf{z}_t, \mathbf{z}_t^i$	bold lowercase $\mathbf{z}$ denotes a time interval,
	consisting of two scalars for the start and the end of the interval
$[s, e]$	a time interval from s to e
$(s, e]$	a time interval from $s + 1$ to e
$\mathbf{X}_{\mathbf{z}}, \mathbf{X}_{[s,e]}, \mathbf{X}_{(s,e]}$	time series segment
$\mathbf{w}, \mathbf{w}_j$	weight vectors
b	constant bias, negative of the threshold
$\xi, \xi^i, \xi_t, \xi^{+i}, \xi^{-i}$	slack variable
$\mathcal{LS}(\mathbf{X})$	set of all legitimate labeling-segmentations of time series $\mathbf{X}$
$\mathcal{X}$	set of all time series
$\mathcal{Y}$	set of all class/cluster labels
$\mathcal{Z}$	set of time intervals
$\mathcal{I}$	set of time intervals and the empty interval
l_{min}, l_{max}	minimum and maximum segment lengths
$\|\| \cdot \|\|$	L_2-norm of a vector
$\varphi(\cdot), \phi(\cdot)$	feature functions
$k(\cdot, \cdot)$	kernel function

$E(\cdot,\cdot), E(\cdot)$	energy functions
$f(\cdot,\cdot), f(\cdot)$	score functions
$g(\cdot)$	output of the time series analysis
$len(\cdot)$	length function
$\lvert\cdot\rvert$	length function
$\Delta(\cdot,\cdot)$	loss function
$\mu(\cdot)$	slack rescaling function

Chapter 1

Introduction

"History is moving statistics and statistics is frozen history."
– August Ludwig von Schlözer

Temporally-varying phenomena are all around us, from temperature and stock prices to heart rates and human behavior. An important step to understand any of these phenomena is to analyze its time series data, which are sequences of observations through time.

Time series analysis has long been an important research topic, with a history of at least 350 years [Klein, 1997]. Graunt [1662] studied the bills of mortality collected over half a century. The main tool of Graunt was the Rule of Three, $\frac{a}{b} = \frac{c}{d}$, an arithmetic technique of using three known values to solve for a fourth unknown factor in a ratio relationship. Graunt used the Rule of Three to hypothesize and verify temporal patterns. For example, Graunt noted from 1628 to 1662, 130,866 females and 139,782 males were christened. Using the Rule of Three, he simplified the gender comparisons by stating that there were thirteen women to every fourteen men. Also with this arithmetic, Graunt reduced the weekly mortality bills of 54 years into several life tables, giving probabilities of survival to each age. After the work of Graunt which analyzed ratio relationships, many other techniques such as first difference, moving average, and correlation were used for time series analysis. More recently, wavelet transform [Percival and Walden, 2000] and Kalman filter [Kalman,

1960] were invented and applied to time series analysis. These techniques, however, were developed before the age of powerful computers and affordable sensors. Most of them were designed for single, low-dimensional time series and for low-level semantic analysis such as estimating trends and computing seasonal variations. Nowadays, with the widespread availability of personal computers and affordable sensors, many more important temporally-varying phenomena can be studied. At the same time, time series data are more complex. Classical problems become more challenging. New problems emerge.

Recent methods for time series analysis are often based on extensions of dynamic Bayesian networks. This approach, however, has several limitations due to the requirement of a good hidden state model, the limited ability to model the null class, and the complicatedness and expensiveness of learning and inference.

In this thesis, we study modern time series in the context of human and animal behavior analysis. We propose segment-based SVMs (Seg-SVMs), a framework that overcomes some limitations of existing approaches for segmenting, classifying, and clustering time series. In particular, we address five important problems: event detection, sequence labeling, early event detection, discriminative event detection, and temporal clustering. Three of these problems have received little or no attention in the computer vision literature. In the following, we will describe these problems in details.

1.1 Event detection

One important problem of time series analysis is event detection, i.e., localizing and recognizing the occurrences of temporal patterns that belong to some predefined target classes. Examples of target event classes are human actions [Ke et al., 2005], sport events [Efros et al., 2003, Xu et al., 2003], and facial expressions [Bartlett et al., 2005, Lucey et al., 2006]. Figure 1.1 illustrates the task of smile detection in a video. It is important to emphasize that event detection is different from and harder than event recognition. Event detection in continuous time series involves both localization and recognition. Given a time series, a detector system must

FIGURE 1.1: *Event detection* is to localize all occurrences of an event of interest. This figure illustrates smile detection – determining when the subject starts and stops smiling.

localize the starts and the ends of target events and then recognize their classes. Event recognition systems, such as those from Yamato et al. [1992], Brand et al. [1997], Gorelick et al. [2007], Sminchisescu et al. [2005], and Laptev et al. [2008], only need to classify pre-segmented subsequences that correspond to coherent events.

Because events are fundamental components of time series, event detection is an important problem. It is a cornerstone in many applications, from video surveillance [Piciarelli et al., 2008] and earthquake detection [Roberts et al., 1989] to motion analysis [Aggarwal and Cai, 1999] and psychopathology assessment [Cohn et al., 2009].

Event detection has been extensively studied in the literature of computer vision. The most popular approach is segment classification, which first selects candidate segments and then uses a classifier to predict if the segments belong to a target event class. To select candidate segments, some methods use low level cues such as trajectories of moving objects [Liao et al., 2006, Piciarelli et al., 2008] and repetitive motions [Polana and Nelson, 1994] while other methods use the sliding window approach which considers all subwindows of certain sizes, e.g., [Efros et al., 2003, Shechtman and Irani, 2007]. To detect events of different lengths, some adopt multi-scale processing [Ke et al., 2005] while others use windows of multiple sizes [Bobick and Davis, 1996, 2001]. In the extreme case, the window size could be one, and a time series is treated as a collection of frames [Bartlett et al., 2005, Littlewort et al., 2006, Lucey et al., 2006, Tian et al., 2005]. To classify candidate segments, many pattern-recognition methods have been used, including template matching [Bobick and Davis, 1996, 2001, Polana and Nelson, 1994, Shechtman and Irani, 2007], nearest neighbor [Efros et al., 2003, Gorelick et al., 2007, Liao et al., 2006], SVMs [Cao et al., 2004, Piciarelli et al., 2008, Pittore et al., 1999], boosting [Ke et al., 2005, Laptev

and Perez, 2007, Nowozin et al., 2007, Smith et al., 2005], neural networks [Vassilakis et al., 2002], and state-space models [Andrade et al., 2006, Bobick and Wilson, 1997, Hongeng and Nevatia, 2003]. Although segment classification has been widely used for event detection, it has several limitations. First, this approach classifies each candidate segment independently; it makes myopic decisions [Wang et al., 2006] and requires post-processing (e.g., to handle overlapping detections). Second, the segment classification approach often has difficulties for accurate localization of event boundaries [Wang et al., 2006], due to the ineffective use of negative examples in training. Negative examples are segments that misalign with target events, and they are either ignored (e.g., [Bobick and Wilson, 1997, Shechtman and Irani, 2007]) or required to be disjoint from the positive training examples (e.g., [Ke et al., 2005, Laptev and Perez, 2007]). In both cases, segments that partially overlap with positive examples are not used in training; those segments, however, are candidates for inaccurate localization at test time.

In Chapter 3, we will address event detection using Segment-based SVMs (Seg-SVMs). We show how the Seg-SVMs framework leads to an algorithm that does not suffer from the aforementioned limitations of the segment classification approach.

1.2 Sequence labeling

Another important problem in time series analysis is sequence labeling, which factorizes a time series into a set of non-overlapping segments and assigns a class label to each segment. Figure 1.2 shows an example of sequence labeling: a video is labeled as a sequence of facial expressions. Sequence labeling is related to event detection and it is often used for event detection. But these two problems are different. A sequence labeling system assigns a unique semantic label to each frame, while an event detection system may assign no or multiple labels.

Sequence labeling is an important problem of time series analysis. It has been shown to be useful in a wide range of applications, from natural language processing [Rabiner, 1989] to office activity understanding [Brand and Kettnaker, 2000] and animal behavior analysis [Oh et al., 2008].

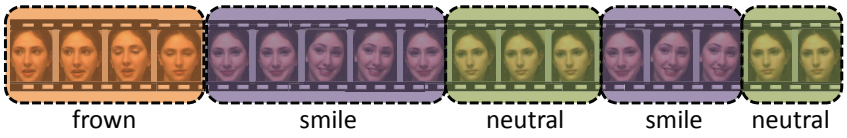


FIGURE 1.2: *Sequence labeling* factorizes a time series into a set of non-overlapping segments and recognizes their classes. In this figure, a facial video is labeled as a sequence of expressions.

Most existing techniques for sequence labeling are based on probabilistic hidden-state models, and labeling a time series is equivalent to finding the sequence of event labels that yields the highest probability. Brand and Kettnaker [2000] use Hidden Markov Models (HMMs) [Rabiner, 1989] for understanding office activities. Xu et al. [2003] use multi-layer HMMs [Rabiner, 1989] to analyze baseball and volleyball videos. Oh et al. [2008] and Fox et al. [2009] use variants of Switching Linear Dynamical Systems (SLDS) [Pavlovic and Rehg, 2000, Pavlovic et al., 2000] to analyze human and animal behavior. Chang et al. [2009], Koelstra and Pantic [2008], Shang and Chan [2009], Tong et al. [2007], Valstar and Pantic [2007] use Dynamic Bayesian Networks (DBNs) for detecting facial events, while Laxton et al. [2007] design a hierarchical structure based on DBNs to decompose complex activities. Although these generative methods have been shown to be effective in their respective scenarios, they have limited ability to model the null class (i.e., no event, unseen event, or anything that we do not have a label for) due to the large variability of the null class. Conditional Random Fields (CRFs) [Lafferty et al., 2001] are the discriminative alternatives to HMMs, and they have been successfully used for a number of applications such as detection of highlight events in soccer videos [Wang et al., 2006]. CRFs, however, cannot model long-range dependencies between labels [Sarawagi and Cohen, 2005], disabling the use of segment-level features. CRFs can be extended to account for higher-order dependencies, but the computational cost increases exponentially with the clique size. Semi-Markov CRFs [Sarawagi and Cohen, 2005] have lower computational cost, but they also require short segment lengths [Okanohara et al., 2006]. Nevertheless, CRF-based models, like HMMs or any other hidden-state model, suffer the drawbacks of needing either an explicit definition of the latent state of all frames, or the need to simultaneously learn a

state sequence and state transition model that fits the data, resulting in a high-dimensional minimization problem with typically many local minima.

In Chapter 4, we will show how Seg-SVMs can be used for sequence labeling, yielding a convex discriminative learning formulation and an efficient segmentation-labeling inference.

1.3 Early event detection

Apart from the classical problems of event detection and sequence labeling, this thesis addresses three other important problems in time series analysis, which have received little or no attention. One such problem is early event detection. A temporal event has a duration, and by early detection, we mean to detect the event as soon as possible, *after it starts but before it ends*. Figure 1.3 illustrates the early detection of a smile.

The ability to make reliable and early detection of temporal events has many potential applications in a wide range of fields, ranging from security (e.g., pandemic attack detection), environmental science (e.g., tsunami warning), to health-care (e.g., risk-of-falling detection using wearable sensors) and robotics (e.g., affective computing). As a concrete example, consider building a robot that can affectively interact with humans. Arguably, a key requirement for such a robot is its ability to accurately and rapidly detect human emotional states from facial expressions so that appropriate responses can be made in a timely manner. This requires facial events such as smiling and frowning to be detected even before they are complete; otherwise, the responses would be out of synchronization.

Despite the importance of early detection, few machine learning formulations have been explicitly developed for early detection. Most existing methods for event detection are designed for offline processing. They have a limitation for processing streaming data as they are trained to detect complete events only. But for early detection, it is necessary to recognize partial events (as illustrated in Figure 1.3), which, however, are ignored in the training process of existing event detectors.

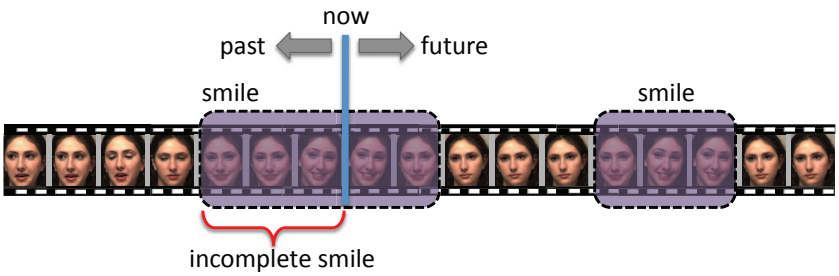


FIGURE 1.3: Can we detect a smile as soon as possible, even before it is complete? This figure shows a stream of facial video. The blue vertical bar indicates the current time; the frames on the right side of this vertical bar have not been observed yet. In this example, the subject is smiling, and the smile hasn't completed yet. The red segment is the only part of the smile that has happened, and we need to recognize it. Existing event detection methods, however, are not trained to recognize incomplete events and thus are unable to make early reliable detection. We address this problem in Chapter 5.

Little attention has been paid to early detection in the literature of computer vision. Davis and Tyagi [2006] addressed rapid recognition of human actions using the probability ratio test. This is a passive method for early detection; it assumes that a generative HMM [Rabiner, 1989] for an event class, trained in the usual way, can also generate partial events. Similarly, Ryoo [2011] took a passive approach for early recognition of human activities; he developed two variants of the bag-of-words representation to address the computational issues, not the timeliness or the accuracy, of the detection process. Previous work on early detection exists in other fields, but its applicability to computer vision is unclear. Neill et al. [2006] studied disease outbreak detection. Their approach, like online change-point detection [Adams and MacKay, 2007, Desobry et al., 2005], is based on locating points at which statistical properties change. This technique, however, cannot be applied to detect temporal events such as smiling and frowning, which must and can be detected and recognized independently of the background. Brown et al. [1992] proposed a method, based on an n-gram model, for predictive typing, i.e., predicting a word from previous words. However, it is hard to apply their method to computer vision, which does not have a well-defined language model. Early detection has also been studied in the context of spam filtering, where immediate and irreversible decisions must be made whenever an email arrives. Assuming spam messages were similar to one another, Haider et al. [2007] developed a method for detecting batches of spam messages

based on clustering. But events such as smiling or frowning cannot be detected and recognized just by observing the similarity between constituent frames, because this characteristic is neither requisite nor exclusive to our target events. It is important to distinguish between forecasting and detection. Forecasting predicts the future while detection interprets the present. For example, financial forecasting (e.g., Kim [2003], Tay and Cao [2001]) predicts the next day's stock index based on the current and past observations. This technique cannot be directly used for early event detection because it predicts the raw value of the next observation instead of recognizing the semantic class of the current and past observations. Perhaps, forecasting the future is a good first step for recognizing the present, but this two-stage approach has a disadvantage because the former may be harder than the latter. For example, it is probably easier to recognize a partial smile than to predict when it will end or how it will progress.

In Chapter 5, we will address the need of early detection and show how the Seg-SVMs framework leads to a novel learning formulation for training temporal classifiers specialized in detecting events as soon as possible.

1.4 Discriminative event detection

Another newly emerged problem in time series analysis is discriminative event detection. Given two sets of time series that correspond to two different classes, discriminative event detection aims at discovering time series segments that correspond to the differences. Figure 1.4 shows an example: given a set of facial videos of depressed people on the left and a set of normal people on the right, the goal is to automatically discover the segments that correspond to depressed moments: the behaviors that discriminate between these two sets of time series. It is important to note that discriminative event detection is a weakly supervised learning problem; examples of discriminative events are not provided in training.

Discriminative event detection is an important technique to be developed. First, the ability to discover the differences between two sets of time series has many

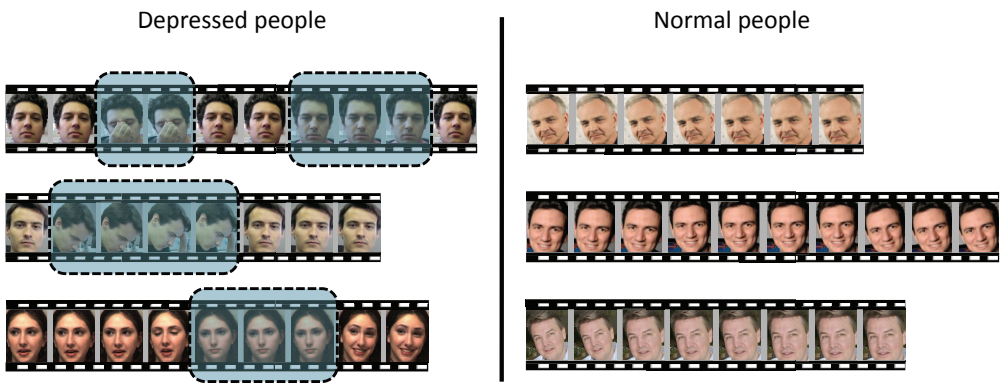


FIGURE 1.4: Discriminative event detection – localizing the segments that discriminate between two sets of time series. This figure depicts a potential application in understanding a psychological disorder. Given a set of depressed-people time series (left) and a set of normal-people time series (right), can we automatically discover the segments that correspond to the depressed behaviors? Note that these behaviors are not exhibited continuously and that many behaviors such as talking and smiling occur across both groups.

potential applications, such as finding the unique behavior patterns of psychological-disorder patients. Second, discriminative event detection can be used as a subroutine for a classification system, where the classification decision depends on whether a discriminative event can be detected. This weakly-supervised learning approach for time series classification alleviates the need for detailed human annotations; collecting detailed labels for time series data is a time-consuming procedure, which often introduces subjective biases.

Despite its foreseeable impact, discriminative event detection is an unexplored problem. The literature on weakly supervised or unsupervised localization and categorization applied to time series is fairly limited and does not address discriminative event detection. Zhong et al. [2004] detect unusual activities in videos by clustering equal-length segments extracted from the video. The segments falling in isolated clusters are classified as abnormal activities. Fanti et al. [2005] describe a system for unsupervised human motion recognition from videos. Appearance and motion cues derived from feature tracking are used to learn graphical models of actions based on triangulated graphs. Niebles et al. [2008] tackle the same problem but represent each video as a bag of video words, i.e. quantized descriptors computed at spatial-temporal interest points. An EM algorithm for topic models is then applied

to discover the latent topics corresponding to the distinct actions in the dataset. Localization is obtained by computing the maximum-a-posteriori topic of each word.

In Chapter 6, we will describe how Seg-SVMs can be used in the weakly-supervised setting for detecting discriminative events.

1.5 Temporal clustering

Another important problem that is addressed in this thesis is temporal clustering. Temporal clustering factorizes multiple time series into a set of non-overlapping segments that belong to several clusters, as illustrated in Figure 1.5. Temporal clustering is different from clustering time series (e.g., Liao [2005]), which refers to the problem of grouping pre-segmented time series that correspond to coherent events. Temporal clustering is an unsupervised problem and therefore is different from the sequence labeling problem described in Section 1.2.

Temporal clustering is useful in its own right as a self-exploratory technique or as a subroutine in more complex data-mining algorithms. It has been applied to learning taxonomies of facial behavior [Zhou et al., 2010], speaker diarization [Fox et al., 2009], discovering motion primitives [Guerra-Filho and Aloimonos, 2006, Vecchio et al., 2003], and clustering human actions in video [Turaga et al., 2009].

Temporal clustering is a relatively unexplored problem. Few algorithms exist and most of them are based on generative models such as extensions of Dynamic Bayesian Networks [Fox et al., 2009], k-means [Robards and Sunehag, 2009] and spectral clustering [Zhou et al., 2010]. These algorithms have several drawbacks due to the limited ability to model the null class, the absence of a feature selection mechanism, and the complicatedness and expensiveness (even intractability) of learning and inference.

We will address the problem of unsupervised temporal factorization in Chapter 7. We will show how the Seg-SVMs framework leads to Maximum Margin Temporal Clustering, a discriminative algorithm that simultaneously performs temporal segmentation and learns a multi-class SVM for separating temporal clusters. We

FIGURE 1.5: Temporal clustering – factorizing multiple time series into a set of non-overlapping segments that belong to several clusters. Temporal clustering is a self-exploratory technique for discovering semantic classes of events.

demonstrate our approach on several publicly available datasets and show that our method consistently matches and often surpasses the performance of state-of-the-art methods for temporal clustering.

1.6 Our contributions and approach

In this thesis, we propose Segment-based SVMs (Seg-SVMs), a machine learning framework for time series analysis. We show how the same design principles can be used to derive supervised, weakly-supervised, and unsupervised learning formulations. We address five different important problems of time series analysis: event detection, sequence labeling, early event detection, discriminative event detection, and temporal clustering. Three of these five problems have received little or no attention in the computer vision literature.

The Seg-SVMs framework combines three powerful ideas: energy-based structure prediction [LeCun et al., 2006], bag-of-words representation [Blei et al., 2003, Lewis, 1998], and maximum-margin training [Schölkopf and Smola, 2002, Taskar et al., 2003, Tsochantaridis et al., 2005, Vapnik, 1998]. The combination of these three ideas yields numerous benefits. First, we use energy-based structure prediction

(see [LeCun et al., 2006] for a tutorial) because detecting semantic events in continuous time series is inherently a structured prediction task. Given a time series, the desired output is more than a binary label indicating the presence or absence of target events. It must predict the locations of target events and their associated class labels, and energy-based structure prediction provides a principled mechanism for concurrent top-down recognition and bottom-up temporal localization. Second, the Seg-SVMs framework models temporal events with the bag-of-words representation [Lewis, 1998]. This feature representation has been successfully used for document classification [Blei et al., 2003], object recognition [Sivic et al., 2005, Zhang et al., 2001], and scene categorization [Fei-Fei and Perona, 2005]. The bag-of-words representation requires no state transition model, eliminating the need for detailed annotation and manual definition of event dynamics. This representation can model and detect events of different lengths, removing the necessity of multi-size templates or multi-sale processing. The bag-of-words representation is not as rigid as template matching or dynamic time warping; it tolerates errors in misalignment, and it is robust to the impreciseness of human annotation. Finally, our framework is based on the maximum-margin training [Schölkopf and Smola, 2002, Taskar et al., 2003, Tsochantaridis et al., 2005, Vapnik, 1998], which learns a discriminative model that maximizes the separating margin between different event classes. Maximizing the separating margin yields classifiers that are less prone to over-fitting [Vapnik, 1998]. Furthermore, the learning formulation of maximum-margin training is convex (for supervised learning), simple and extendable.

1.7 Organization of this dissertation

The rest of this dissertation is organized as follows. The next chapter provides an overview of our framework. Chapter 3 describes a supervised learning algorithm for event detection. Chapter 4 proposes a supervised algorithm for sequence labeling. Chapter 5 addresses the need of early detection and derives a novel learning formulation. The next two chapters present algorithms that require less human

annotation. Chapter 6 introduces a weakly supervised algorithm to discover discriminative events, and Chapter 7 develops an unsupervised method for temporal factorization. Chapter 8 concludes and discusses several directions for future study.

Parts of this thesis have been published [Hoai and De la Torre, 2012a, Hoai et al., 2011, Nguyen et al., 2009, 2010], one is under review for publication [Hoai and De la Torre, 2012b].

Chapter 2

The Foundation of Seg-SVMs

"The whole structure of science gradually grows,
but only as it is built upon a firm foundation of past research."
– Owen Chamberlain

The problems described in the previous chapter have similar goals. They all require factorizing a time series into a set of non-overlapping segments and providing a label to some or all segments. For event detection, the goal is to identify the segments that correspond to target events. For sequence labeling, the goal is to recognize the event class of every segment. For early event detection, the goal is to identify the segments that correspond to either complete or partial target events. For discriminative event detection, the goal is to identify the segments that distinguish between two sets of time series. And for temporal clustering, the goal is to provide the same cluster label to similar segments.

We formulate this common task as follows. Suppose there are m labels (i.e., m classes or m clusters) and let $\mathcal{Y} = \{1, \cdots, m\}$ be the set of all labels. Let $\mathcal{Z}$ be the set of all length-bounded intervals: $\mathcal{Z} = \{\mathbf{z} |\ \mathbf{z} \in \mathbb{N}^2, l_{min} \leq len(\mathbf{z}) \leq l_{max}\}$, with l_{min}, l_{max} are application specific parameters. Given a time series $\mathbf{X}$, a legitimate labeling-segmentation of $\mathbf{X}$ is a set of label-segment pairs $(y_1, \mathbf{z}_1), \cdots, (y_k, \mathbf{z}_k) \in \mathcal{Y} \times \mathcal{Z}$ of which all segments $\mathbf{z}_1, \cdots, \mathbf{z}_k$ are pairwise disjoint subintervals of $[1, len(\mathbf{X})]$, as

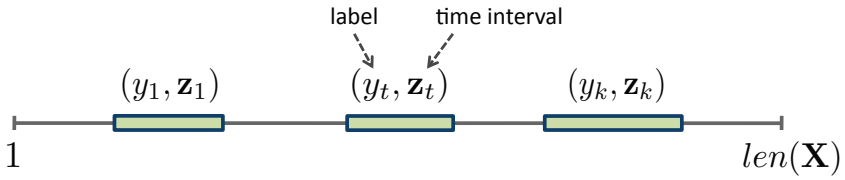


FIGURE 2.1: The common goal of our time series analysis problems – to factorize a time series into a set of non-overlapping segments and assign a class/cluster label to some or all segments. $y_t \in \{1, \cdots, m\}$ is a class/cluster label, and $\mathbf{z}_t$ consists of two scalars for the start and the end of an event.

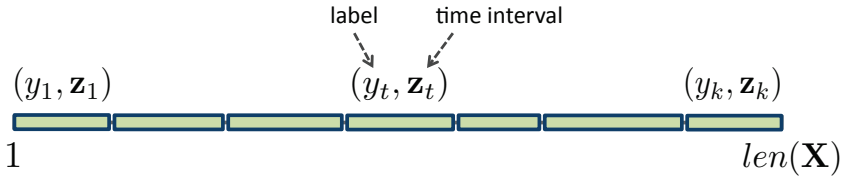


FIGURE 2.2: Some time series analysis system is required to assign a class/cluster label to every segment of a time series.

illustrated in Figure 2.1. Some additional application-specific constraints may apply, for example:

1. $k \leq k_{\max}$, an application-specific bound on the number of segments, or
2. $\mathbf{z}_1 \cup \mathbf{z}_2 \cup \cdots \cup \mathbf{z}_k = [1, len(\mathbf{X})]$, every segment of $\mathbf{X}$ must be labeled (Fig. 2.2).

Let $\mathcal{LS}(\mathbf{X})$ denote the set of all legitimate labeling-segmentations that satisfy application specific constraints. Our goal is to learn $g(\mathbf{X})$ a predictor function (e.g., event detector) that inputs a time series and outputs a legitimate labeling-segmentation corresponding to the desired output (e.g., the temporal extents and event classes of target events).

2.1 Energy-based structure prediction

We propose to find the desired output with energy-based structure prediction (see LeCun et al. [2006] for a tutorial). Energy-based structure prediction provides a principled mechanism for concurrent top-down labeling and bottom-up localization. An alternative approach is to use probabilistic models; however, probabilistic models

have two major disadvantages [LeCun et al., 2006]: i) the normalization requirement limits the choice of energy functions we can use, and ii) learning and inference may be very complicated, expensive, or even intractable.

We define $g(\mathbf{X})$ as the legitimate labeling-segmentation that yields the minimum sum of energies:

$$g(\mathbf{X}) := \underset{\{(y_t,\mathbf{z}_t)\}\in\mathcal{LS}(\mathbf{X})}{\operatorname{argmin}} \sum_t E(\mathbf{X}_{\mathbf{z}_t}, y_t). \tag{2.1}$$

Here, for a segment $\mathbf{z} = [s, e]$, $\mathbf{X}_\mathbf{z}$ denotes the segment of time series $\mathbf{X}$ extracted from time s to time e inclusive. $E(\mathbf{X}_\mathbf{z}, y)$ denotes the energy for assigning segment $\mathbf{X}_\mathbf{z}$ to label y. This energy function is defined for segments of time series, instead of for individual frames or for the entire sequence. This has several benefits. First, it reflects the goals of our problems, which are to localize temporal phenomena at the segment level. Second, it provides a model for long-term dependency of labels, and at the same time, it leads to an efficient labeling and segmentation inference. Neither frame-based nor sequence-based models have both of these properties.

The parameters of the energy function E is a set of weight vectors $\{\mathbf{w}_1, \cdots, \mathbf{w}_m\}$, one for each label class, and a scalar bias term b (i.e., negative of a threshold). The value of the energy function $E(\mathbf{X}_\mathbf{z}, y)$ depends on $\mathbf{w}_y^T\varphi(\mathbf{X}_\mathbf{z})+b$ and $\max_{y'\neq y} \mathbf{w}_{y'}^T\varphi(\mathbf{X}_\mathbf{z})+b$, with $\varphi(\mathbf{X}_\mathbf{z})$ denotes the feature vector for segment $\mathbf{X}_\mathbf{z}$. The feature function $\varphi(\cdot)$ is application specific, and in general, it can be any function that satisfies two conditions: i) the input can be time series segments of any length from l_{min} to l_{max}, and ii) the output must always be a vector of a fixed dimension. This feature function may also be implicitly defined as the feature mapping to a kernel space. In this thesis, we propose to use the Bag-of-Words (BoW) representation; more details are described in Section 2.3.

2.2 Maximum-margin training

We propose to learn $\{\mathbf{w}_1, \cdots, \mathbf{w}_m, b\}$, the parameters of the energy function $E(\cdot, \cdot)$, using maximum-margin training [Schölkopf and Smola, 2002, Vapnik, 1998]. Maximum-margin training is a state-of-the-art machine learning tool, which controls the capacity of the classifier space by optimizing the margin. Maximum-margin training permits the use of kernels. It leads to sparse solutions. It has a convex learning formulation (for supervised learning), which is simple and extendable. Given a collection of training time series $\mathbf{X}^1, \cdots, \mathbf{X}^n$, we learn $\mathbf{w}_1, \cdots, \mathbf{w}_m$ and b by optimizing:

$$\underset{\{\mathbf{w}_j\}, b, \{\xi^i\}}{\text{minimize}} \ \frac{1}{2m} \sum_{j=1}^{m} ||\mathbf{w}_j||^2 + C \sum_{i=1}^{n} \xi^i. \tag{2.2}$$

Here $\sum_{j=1}^{m} ||\mathbf{w}_j||^2$ is inversely proportional to the margin, and ξ^i is a surrogate loss of the prediction function $g(\cdot)$ on time series $\mathbf{X}^i$. This surrogate loss, and the true loss that it approximates, depends on the amount of annotation provided and several other factors. C is the parameter that controls the tradeoff for a larger margin and for a lower training loss. We will discuss this in more detail in subsequent chapters.

2.3 Bag-of-Words representation

Inspired by the success of the BoW representation [Lewis, 1998] for document classification [Blei et al., 2003], object recognition [Sivic et al., 2005, Zhang et al., 2001], and scene categorization [Fei-Fei and Perona, 2005], we consider the feature vector of a segment $\varphi(\mathbf{X}_\mathbf{z})$ as the histogram of temporal words. This representation has several benefits. It requires no state transition model, eliminating the need for detailed annotation and manual definition of event dynamics. This representation can model events of different lengths, removing the necessity of multi-size templates. BoW representation is not as rigid as template matching or dynamic time warping. It tolerates errors in misalignment, and it is robust to the impreciseness of human annotation.

The BoW representation builds a temporal codebook by applying a clustering algorithm to a set of local descriptors sampled from the training data [Leung and Malik, 2001, Sivic and Zisserman, 2003]. Each frame of a time series is associated with a local descriptor, and subsequently is represented by the ID of the corresponding codebook entry. Finally, the feature vector $\varphi(\mathbf{X_z})$ is taken as the histogram of IDs associated with the frames inside the interval $\mathbf{z}$. More formally, let $\mathbf{x}_t$ denote the local descriptor associated with the t^{th} frame of time series $\mathbf{X}$, and suppose there are d clusters (i.e., the size of temporal codebook). Let $\mathbf{a}_t \in \mathbf{R}^d$ be the indicator vector for the clustering assignment of $\mathbf{x}_t$:

$$\mathbf{a}_t = [0, \cdots, 0, 1, 0, \cdots, 0]^T. \tag{2.3}$$

All but one entries of $\mathbf{a}_t$ are 0; the u^{th} entry is 1, with u is the ID of the cluster that $\mathbf{x}_t$ is assigned to. The segment-level feature vector for time series segment $\mathbf{X}_{[s,e]}$ is defined as:

$$\varphi(\mathbf{X}_{[s,e]}) = \frac{1}{Z} \sum_{t=s}^{e} \mathbf{a}_t. \tag{2.4}$$

Here Z is the normalization factor. The feature vector is an unnormalized histogram if $Z = 1$ and a normalized histogram if $Z = len([s, e])$.

The BoW representation for a time series segment depends on local descriptors inside the segment but not their locations. However, this is different from totally ignoring the dynamics or ordering of observation values. Local descriptors are not necessarily the same as raw observation values. A local descriptor at a particular time can be some statistics over a supporting subwindow or subvolume of observation values. Some examples of local descriptors are statistics of brightness gradients and optical flows over a video subvolume (STIP [Laptev and Lindeberg, 2003] and Cuboid [Dollár et al., 2005]) and frequency-domain entropy and energy over a several-second subwindow [Bao and Intille, 2004].

Despite its simplicity, BoW representation is powerful. Furthermore, BoW representation can be extended in many ways. The rest of this section describes several particular extensions.

2.3.1 No or multiple local descriptors

The above formulation simplifies the presentation by assuming each frame is associated with a local descriptor. This is, however, not a necessary requirement. For BoW representation, a frame can be associated with zero, one, or multiple local descriptors (e.g., STIP [Laptev and Lindeberg, 2003] and Cuboid [Dollár et al., 2005]). The segment-level feature vector can still be computed using Eq. 2.4 above, with the indication vector $\mathbf{a}_t$ is the histogram of codebook IDs at frame t.

2.3.2 Soft quantization

BoW representation can be defined based on soft quantization. Instead of assigning each frame to a single cluster, a frame can be associated with multiple clusters, weighted by the proximity from the frame to the cluster centers. Segment-level feature vector can still be computed as in Eq. 2.4, but $\mathbf{a}_t$ is the proximity vector instead of a binary indication vector. In other words, let $\mathbf{c}_1, \cdots, \mathbf{c}_d$ be the cluster centers for the temporal codebook, $\mathbf{a}_t$ is defined as:

$$\mathbf{a}_t = [k(\mathbf{x}_t, \mathbf{c}_1), \cdots, k(\mathbf{x}_t, \mathbf{c}_d)]^T. \tag{2.5}$$

Here $k(\cdot, \cdot)$ is a a function measuring the similarity between two local descriptors. It is not necessary for $\mathbf{c}_1, \cdots, \mathbf{c}_d$ to be cluster centers; they can be representative vectors that are obtained using methods that are different from clustering.

2.3.3 Multiple feature types

BoW representation can be defined for different feature types. For example, suppose there are two types of local descriptors for every frame t: $\mathbf{x}_t^{(1)}$ and $\mathbf{x}_t^{(2)}$. We can build two different temporal codebooks, one for each feature type, and define cluster indication/association vectors $\mathbf{a}_t^{(1)}, \mathbf{a}_t^{(2)}$ accordingly. The segment-level feature vector can be computed using Eq. 2.4, with $\mathbf{a}_t$ is the concatenation of $\mathbf{a}_t^{(1)}$ and $\mathbf{a}_t^{(2)}$, i.e., $\mathbf{a}_t = \begin{bmatrix} \mathbf{a}_t^{(1)} \\ \mathbf{a}_t^{(2)} \end{bmatrix}$.

2.3.4 HMM-inspired feature

BoW representation can be extended to account for the interaction between pairs of consecutive frames, just like HMMs. The segment-level feature vector can be computed as before, using Eq. 2.4, with $\mathbf{a}_t$ is the concatenation of observation and interaction vectors:

$$\mathbf{a}_t = \begin{bmatrix} \mathbf{a}_t^{obs} \\ \mathbf{a}_t^{int} \end{bmatrix}. \tag{2.6}$$

Here $\mathbf{a}_t^{obs}$ and $\mathbf{a}_t^{int}$ are the observation and interaction vectors respectively. The observation vector is the $d \times 1$ indicator vector for soft quantization as defined in Eq. 2.5; this is the pseudo probability for the local descriptor to belong to a set of predefined states ($\mathbf{c}_1, \cdots, \mathbf{c}_d$, cluster centers or representative vectors). The interaction vector $\mathbf{a}_t^{int}$ is a $d^2 \times 1$ vector defined as:

$$\mathbf{a}_t^{int} = \mathbf{a}_{t-1}^{obs} \otimes \mathbf{a}_t^{obs}$$

The $((u-1)d + v)^{th}$ entry of the interaction vector is the pseudo-probability for transitioning from state v to state u at time t. The interaction vector at time t depends on the observation vectors at time t and time $t-1$.

2.3.5 Multiple event parts

BoW representation can be extended to preserve the relative order between the parts of an event. This can be achieved by breaking a time series segment into smaller subsegments and compute the BoW feature vector for each subsegments, as for spatial object [Lazebnik et al., 2006]. The feature vector for the whole segment is then the concatenation of the feature vectors of subsegments. For example, let v be the midpoint of segment $[s, e]$, we can define the segment-level feature vector as:

$$\varphi(\mathbf{X}_{[s,e]}) = \frac{1}{Z} \begin{bmatrix} \sum_{t=s}^{v} \mathbf{a}_t \\ \sum_{t=v+1}^{e} \mathbf{a}_t \end{bmatrix}. \tag{2.7}$$

Chapter 3

Supervised Learning for Event Detection

"You can't defend. You can't prevent.
The only thing you can do is detect and respond."
– Bruce Schneier

In this chapter, we describe a supervised learning algorithm for event detection in the Seg-SVMs framework. We assume the training data is fully annotated, i.e., the starts and the ends of target events in training data are provided. We also assume target events belong to a single class; thus, event recognition is unnecessary and localization is the only job of the detector (to detect events from multiple classes, we can learn a set of per-class detectors). We apply our method to detect facial Action Units (AUs) [Ekman and Friesen, 1978] in video and show its advantages over state-of-the-art approaches for AU detection.

3.1 Energy-based event detection

Our event detector is energy-based, as descried Eq. 2.1. Because there is only one class of target events, the set of labels has a single element and the energy function

$E(\mathbf{X}_\mathbf{z}, y)$ only depends on $\mathbf{X}_\mathbf{z}$. We shorten $E(\mathbf{X}_\mathbf{z}, y)$ as $E(\mathbf{X}_\mathbf{z})$ and rename $\mathbf{w}_1$ as $\mathbf{w}$ for brevity. The output of the detector $g(\cdot)$ on a time series $\mathbf{X}$ is:

$$g(\mathbf{X}) := \underset{\{\mathbf{z}_t\} \in \mathcal{LS}(\mathbf{X})}{\operatorname{argmin}} \sum_t E(\mathbf{X}_{\mathbf{z}_t}). \tag{3.1}$$

Thus the output of the event detector is a set of segments that minimizes the total sum of energies. This set of segments is possibly empty, and if it is the case, we report no detection. The energy of a segment is defined as the negative of the detection score $E(\mathbf{X}_\mathbf{z}) := -f(\mathbf{X}_\mathbf{z})$, with the detection score defined as:

$$f(\mathbf{X}_\mathbf{z}) := \mathbf{w}^T \varphi(\mathbf{X}_\mathbf{z}) + b. \tag{3.2}$$

If the energy function is given, the set of segments that minimizes the total sum of energies can be found using an efficient dynamic programming algorithm, which will be described in a subsequent chapter. We now describe the maximum-margin learning formulation.

3.2 Maximum-margin learning for event detection

This section describes the maximum-margin learning formulation for event detection. For supervised learning, this is a special case of Max-Margin Markov Networks [Taskar et al., 2003] and SOSVM [Tsochantaridis et al., 2005].

Let the training time series be $\mathbf{X}^1, \cdots, \mathbf{X}^n \in \mathcal{X}$ and their associated ground truth annotations for the occurrence of the target events be $\mathbf{z}^1, \cdots, \mathbf{z}^n$. We assume each training sequence contains at most one event of interest, as we can always break a training time series that contains several events into shorter subsequences of single events. For an ideal detector, the ground truth event $\mathbf{z}^i$ must be the segment that has the lowest energy, i.e., the highest detection score:

$$\mathbf{z}^i = \underset{\mathbf{z} \in \mathcal{Z}}{\operatorname{argmax}}\, f(\mathbf{X}^i_\mathbf{z}). \tag{3.3}$$

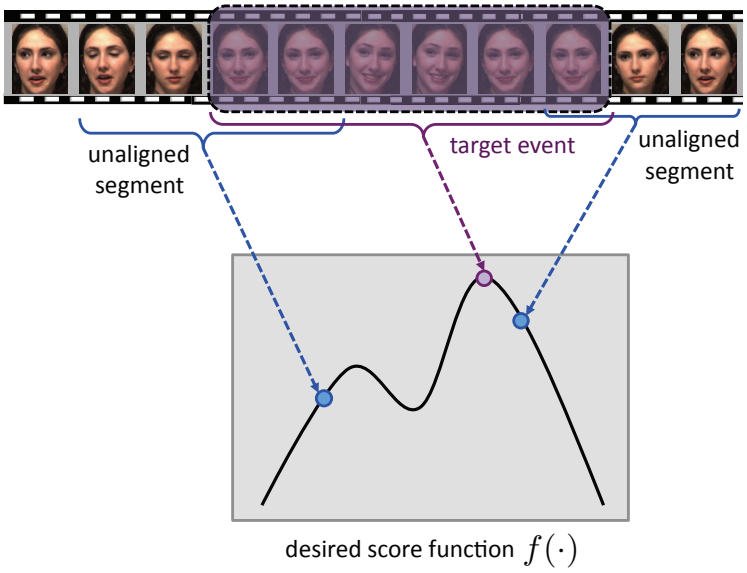


FIGURE 3.1: Desired detection function – the target event must have the highest detection score. During training, we learn the detection function by enforcing this constraint.

This is illustrated in Figure 3.1. Furthermore, the highest detection score must be positive, otherwise no detection would be reported. That requires:

$$f(\mathbf{X}^i_{\mathbf{z}^i}) > 0. \tag{3.4}$$

For the simplicity of presentation, let $\mathcal{I}$ be $\mathcal{Z} \cup \{\emptyset\}$, the set of time intervals plus the empty segment. We consider the empty segment has the detection score of zero: $f(\mathbf{X}_\emptyset) := 0$. The constraints in Eq. 3.3 and Eq. 3.4 are equivalent to:

$$\mathbf{z}^i = \operatorname*{argmax}_{\mathbf{z} \in \mathcal{I}} f(\mathbf{X}^i_{\mathbf{z}}). \tag{3.5}$$

Equivalently:

$$f(\mathbf{X}^i_{\mathbf{z}^i}) > f(\mathbf{X}^i_{\mathbf{z}}) \ \forall \mathbf{z} \in \mathcal{I}, \mathbf{z} \neq \mathbf{z}^i. \tag{3.6}$$

This constraint can be required to be well satisfied by a margin. This margin is adaptive and proportional to $\Delta(\mathbf{z}^i, \mathbf{z})$, the loss of the detector for outputting $\mathbf{z}$ when

the desired output is $\mathbf{z}^i$. The constraint for an ideal detector becomes:

$$f(\mathbf{X}^i_{\mathbf{z}^i}) \geq f(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}^i, \mathbf{z}) \ \forall \mathbf{z} \in \mathcal{I}, \mathbf{z} \neq \mathbf{z}^i. \tag{3.7}$$

This constraint forces the score of $\mathbf{X}^i_{\mathbf{z}^i}$ to exceed the score of $\mathbf{X}^i_{\mathbf{z}}$ by a margin that is equal to the loss associated the mismatch between $\mathbf{z}$ and $\mathbf{z}^i$. This loss is application dependent; it reflects the penalty for not outputting the desired output. Two examples of this loss function are: $\Delta(\mathbf{z}^i, \mathbf{z}) = 1 - \frac{len(\mathbf{z}^i \cap \mathbf{z})}{len(\mathbf{z}^i \cup \mathbf{z})}$ and $\Delta(\mathbf{z}^i, \mathbf{z}) = len(\mathbf{z}^i \setminus \mathbf{z}) + len(\mathbf{z} \setminus \mathbf{z}^i)$. In Section 3.3.5, we describe the loss function used in our experiments.

Each training time series leads to one constraint, and to learn the parameters $(\mathbf{w}, b)$ of the detector, we can maximize the margin subject to all these constraints, i.e.,

$$\underset{\mathbf{w},b}{\text{minimize}} \ \frac{1}{2}||\mathbf{w}||^2 \tag{3.8}$$

$$\text{s.t.} \ \ f(\mathbf{X}^i_{\mathbf{z}^i}) \geq f(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}^i, \mathbf{z}) \ \forall i, \forall \mathbf{z} \in \mathcal{I}. \tag{3.9}$$

As in the traditional formulation of SVM, the constraints are allowed to be violated by introducing slack variables:

$$\begin{aligned} \underset{\mathbf{w},b,\{\xi^i\}}{\text{minimize}} \ & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi^i, \\ \text{s.t.} \ \ & f(\mathbf{X}^i_{\mathbf{z}^i}) \geq f(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}^i, \mathbf{z}) - \xi^i \ \forall i, \forall \mathbf{z} \in \mathcal{I}, \\ & \xi^i \geq 0 \ \forall i. \end{aligned} \tag{3.10}$$

Here, C is the parameter controlling the trade-off between having a large margin and less constraint violation. This formulation can be viewed as a special case of Max-Margin Markov Networks [Taskar et al., 2003] and SOSVM [Tsochantaridis et al., 2005].

This optimization problem is convex, but it has an exponentially large number of constraints. A typical optimization strategy is *constraint generation* [Tsochantaridis et al., 2005] that is theoretically guaranteed to produce a global optimal solution. Constraint generation is an iterative procedure that optimizes the objective w.r.t.

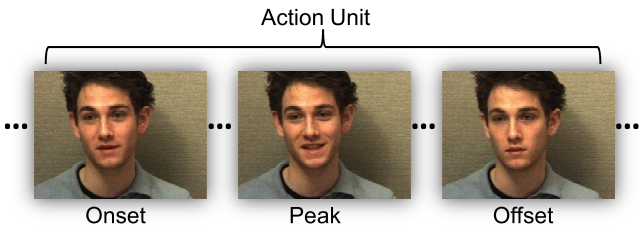


FIGURE 3.2: Left to right, evolution of an AU12 (involved in smiling), from onset, peak, to offset.

a smaller set of constraints. The constraint set is expanded at every iteration by adding the most violated constraint.

3.3 Experiments – Action Unit (AU) detection

AUs are parts of the Facial Action Coding System (FACS) [Ekman and Friesen, 1978], a comprehensive, anatomically-based system for measuring all visually discernible facial movement. FACS describes facial activity on the basis of 44 unique AUs, as well as several categories of head and eye positions and movements. Any facial event (e.g., a gesture, expression or speech component) may be a single AU or a combination of AUs. For example, the felt, or Duchenne smile, is indicated by movement of the zygomatic major (AU12, e.g., Fig. 3.2) and orbicularis oculi, pars lateralis (AU6). Because of its descriptive power, FACS has become the state-of-the-art in manual measurement of facial expression and is widely used in studies of spontaneous facial behavior. Much effort in automatic facial image analysis seeks to automatically recognize FACS action units [Littlewort et al., 2006, Pantic and Rothkrantz, 2004, Tian et al., 2005, Tong et al., 2007].

This section describes experiments on two spontaneous datasets for AU detection. Experiment 1 (Sec. 3.3.6) compares the performance of our method against state-of-the-art methods on a large dataset of FACS coded video. In Experiment 2 (Sec. 3.3.7) we evaluate the generalization performance by testing on a dataset that was not used for training.

3.3.1 Related work on AU detection

AU detection from video is a challenging computer vision and pattern recognition problem. Some of the most important challenges are to: (i) accommodate large variability of in action units across subjects; (ii) train classifiers when relatively few examples for each AU are present; (iii) recognize subtle AUs; (iv) and model the temporal dynamics of AUs, which can be highly variable.

To address some of these issues, various approaches have been proposed. Static approaches [Bartlett et al., 2005, Littlewort et al., 2006, Lucey et al., 2006, Tian et al., 2005] pose AU detection as a binary- or multi-class classification problem using different features (e.g., appearance, shape) and classifiers (e.g., Boosting, SVM). The classifiers are typically trained on a frame-by-frame basis. For a given AU, the positive class comprises a subset of frames between its onset and offset, and the negative class comprises a subset of frames labeled as neutral or other AUs. Dynamic approaches, such as modifications of dynamic Bayesian networks [Chang et al., 2009, Koelstra and Pantic, 2008, Shang and Chan, 2009, Tong et al., 2007, Valstar and Pantic, 2007] model the dynamics of the AU as transitions in a partially observed state space.

Although static and dynamic approaches have achieved high performance on most posed facial expression databases [Bartlett et al., 2005, Sun and Yin, 2008, Tian et al., 2005], accuracy tends to be much lower in studies that test on non-posed facial expressions [Bartlett et al., 2005, Littlewort et al., 2006]. Non-posed expressions are challenging. They are less stereotypic, more subtle, more likely to co-occur with other AUs, and more-often confounded by increased noise due to variation in pose, out-of-plane head motion, and co-occurring speech. They also may be more complex temporally. Segmentation into onset, one or more local peaks, and offset must be discovered.

For non-posed facial behavior, static approaches may be more susceptible to noise because independent decisions are made on each frame. Similarly, hidden state temporal models suffer the drawbacks of needing either an explicit definition of the latent state of all frames, or the need to simultaneously learn a state sequence and state

transition model that fits the data, resulting in a high-dimensional minimization problem with typically many local minima.

Our method has several benefits for AU detection: (1) all possible segments of the video may be used for training; and (2) no assumptions are required about the underlying structure of the action unit events (e.g., i.i.d.). Experimental results confirm the benefits of our approach for AU detection.

3.3.2 Datasets and AU selection

Evaluations of performance for Experiment 1 were carried out on a relatively large corpus of FACS coded video, the RU-FACS-1 [Bartlett et al., 2006] dataset. Recorded at Rutgers University, subjects were asked to either lie or tell the truth under a false opinion paradigm in interviews conducted by police and FBI members who posed around 13 questions to the subjects. These interviews resulted in 2.5 minute long continuous 30-fps video sequences containing spontaneous AUs of people of varying ethnicity and sex. Ground truth FACS coding was provided by expert coders. Data from 28 of the subjects was available for our experiments. In particular, we divided this dataset into 17 subjects for training (97000 frames) and 11 subjects for testing (67000 frames).

The AU for which we present results were selected by requiring at least 100 event occurrences in the available RU-FACS-1 data, resulting in the following set of AU: $1, 2, 12, 14, 15, 17, 24$. Additionally, to test performance on AU combinations, AU1+2 was selected due to the larger number of occurrences.

Experiment 2 tests generalization performance on the unrelated dataset *Sayette*[1]. This dataset records subjects participating in a 3-way conversation to study the effects of alcohol on social behavior. Video for 3 subjects was available to us (32000 frames). Only FACS codes for AU 6 and 12 were available.

[1]This is an in-progress data-collection.

FIGURE 3.3: AAM tracking across several frames

3.3.3 Frame-level feature extraction

This section describes the feature extraction at a frame-level. The feature representation at the segment-level is described in Section 3.3.4.

Given a video sequence, we first track the facial features using a person-specific AAM model [Matthews and Baker, 2004]. In this work, the AAM model used is composed of 66 landmarks distributed along the top of the eyebrows, the inner and outer lip outlines, the outline of the eyes, the jaw, and along the nose. Fig. 3.3 shows an example of AAM tracking of facial features in several frames from the RU-FACS-1 [Bartlett et al., 2006] video dataset.

Appearance-based features have been shown to yield good performance on many AUs [Bartlett et al., 2005, Lucey et al., 2009]. In this work we propose to use fixed-scale-and-orientation SIFT descriptors [Lowe, 1999] anchored at several points of interest at the tracked landmarks. Intuitively, the histogram of gradient orientations calculated in SIFT has the potential to capture much of the information that is described in FACS (e.g., the markedness of the naso-labial furrows, the direction and distribution of wrinkles, the slope of the eyebrows). At the same time, the SIFT descriptor has been shown to be robust to illumination changes and small errors in localization.

After the facial components have been tracked in each frame, a normalization step registers each image with respect to an average face. An affine texture transformation is applied to each image so as to warp the texture into this canonical reference frame. This normalization provides further robustness to the effects of head motion. Once the texture is warped into this fixed reference, SIFT descriptors are computed

around the outer outline of the mouth (11 points for lower face AU) and on the eyebrows (5 for upper face AU). Due to the large number of resulting features (128 by number of points), the dimensionality of the resulting feature vector was reduced using PCA to keep 95% of the energy, obtaining 261 and 126 features for lower face and upper face AU respectively.

3.3.4 Segment-level feature extraction

For segment-level feature vector, we use the soft-clustering approach defined in Eq. 2.4 and Eq. 2.5. We use non-normalized histogram but learn a bias term that scales with the segment length by appending the segment length to the feature vector, i.e., $\varphi(\mathbf{X_z}) := [\varphi(\mathbf{X_z}); len(\mathbf{z})]$. To incorporate the benefits of both statics and dynamic approaches for AU detection, $\mathbf{c}_1, \cdots, \mathbf{c}_d$ are taken as the support vectors of a frame-based SVM (*Frm-SVM*) trained to distinguish between individual positive and negative frames. This method directly improves the performance of frame-based SVM by relearning the weights to incorporate temporal constraints. To see this, consider the score function of a frame-based SVM. For a frame $\mathbf{x}_t$ of a time series $\mathbf{X}$, the SVM score is $\mathbf{v}^T\phi(\mathbf{x}_t) + b$, here $\phi(\cdot)$ is an implicit mapping of kernel $k(\cdot,\cdot)$. The representer theorem [Vapnik, 1998] states that $\mathbf{v}$ can be expressed as a linear combination of the support vectors:

$$\mathbf{v} = \sum_{j=1}^{d} \alpha_j \phi(\mathbf{c}_j). \tag{3.11}$$

Thus the SVM score for frame $\mathbf{x}_i$ is:

$$\mathbf{v}^T\varphi(\mathbf{x}_t) + b = \sum_{j=1}^{d} \alpha_j k(\mathbf{x}_t, \mathbf{c}_j) + b. \tag{3.12}$$

Meanwhile, the decision function of the proposed learning formulation is:

$$\mathbf{w}^T\varphi(\mathbf{X_z}) = \sum_{t\in\mathbf{z}} \sum_{j=1}^{d} w_j k(\mathbf{x}_t, \mathbf{c}_j) + w_{d+1} len(\mathbf{z}). \tag{3.13}$$

Observe the similarity between the decision function of frame-based SVM and the decision function of segment-based SVM, Eq. 3.12 versus Eq. 3.13. In both cases, we need to learn a weight vector that is associated with the similarity measurement between a frame and the support vectors $\{\mathbf{c}_j\}$. Furthermore, ignoring the constant threshold, the decision value of segment-based SVM is the sum of the decision values of frame-based SVM at all frames inside the segment. The key differences between frame-based SVM and this approach are: (i) frame-based SVM classifies each frame independently while this approach makes a collective decision; (ii) this approach incorporates temporal constraints during training and testing while frame-based SVM does not.

3.3.5 Setup and evaluation

We compare our method against a frame-based SVM and dynamic methods using HMM [Rabiner, 1989]. All methods use the same frame-level features described in Sec. 3.3.3.

The frame-based SVM is trained to distinguish between positive (AU) negative (non-AU) frames and uses a radial basis kernel $k(\mathbf{x}, \mathbf{z}) = \exp(-\gamma||\mathbf{x} - \mathbf{z}||^2)$.

Our method is based on soft-clustering (Sec. 3.3.4). The cluster centers are chosen to be the support vectors (SVs) of frame-based SVMs with a radial basis kernel. Because for several AUs the number of SVs can be large $(2000 - 4000)$, we apply the idea proposed by Avidan [2003] to reduce the number of SVs for faster training time and better generalization. However, instead of using a greedy algorithm for subset selection, we use LASSO regression [Tibshirani, 1996]. In our experiments, the sizes of the reduced SV sets ranges from 100 to 500 SVs. To take into account the imbalance of positive and negative frames, we penalize false negative and false positives differently and use: $\Delta(\mathbf{z}, \mathbf{z}^i) = \alpha \cdot \text{len}(\mathbf{z}^i \setminus \mathbf{z}) + \beta \cdot \text{len}(\mathbf{z} \setminus \mathbf{z}^i)$. Here α and β are penalties for false negative and false positive frames respectively.

We compare the performance of our method with dynamic approaches using HMMs which have been used with success in the facial expression literature [Koelstra and Pantic, 2008, Valstar and Pantic, 2007]. In this experiment, we will limit ourselves

to a basic generative HMM model where the observations for each state are modeled as a Gaussian distribution using a full covariance matrix with ridge regularization (i.e., $\hat{\mathbf{\Sigma}} = \mathbf{\Sigma} + \lambda \mathbf{I}$ where $\mathbf{I}$ is the identity matrix), and consider the same feature set used for all other experiments. Two different state mappings where tried resulting in HMM2 and HMM4. HMM2 is a 2-state model, where state-0 corresponds to a neutral face (no AU present) and state-1 corresponds to frames where the AU is present. HMM4 is a 4-state model, where state-0 is mapped to neutral face frames, state-1 corresponds to AU onset frames, state-2 corresponds to peak frames, and state-3 corresponds to offset frames.

Following previous work [Bartlett et al., 2005], positive samples were taken to be frames were the AU was present, and negative samples as frames were it was not. To evaluate the performance, we report various measures: the area under the ROC, the precision-recall values, and the maximum $F1$ score. the $F1$ score is defined as: $F1 = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}$, summarizing the trade-off between high recall rates and accuracy among the predictions. In our case, the $F1$ score is a better performance measure than the more common ROC metric because the latter is designed for balanced binary classification rather than detection tasks, and fails to reflect the effect of the proportion of positive to negative samples on classification performance.

Parameter tuning is done using 3-fold subject-wise cross-validation on the training data. For the frame-based SVM, we need to tune C and γ, the scale parameter of the radial basis kernel. For our method, we need to tune C only. The kernel parameter γ of our method could also potentially be tuned, but for simplicity it was set to the same γ used for the frame-based SVM. For HMM2 and HMM4, we need to tune the the regularization parameter λ of the covariance matrix. For all methods, we choose the parameters that maximize the average cross-validation ROC area.

3.3.6 Within dataset performance

Tab. 3.1 and Tab. 3.2 show the experimental results on the RU-FACS-1 dataset. Using the ROC metric, our method appears comparable to frame-based SVM and dynamic approaches. However, using the $F1$ measure, our method consistently outperforms other approaches, achieving highest score on 7 out of 10 test cases.

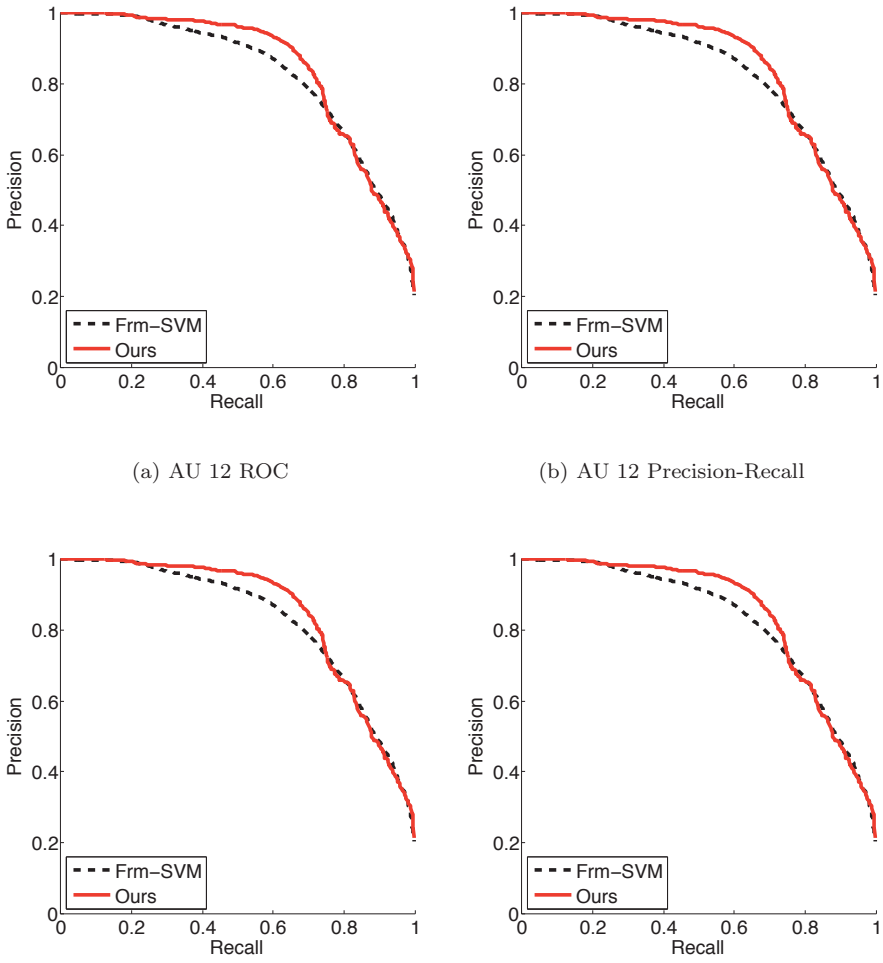
TABLE 3.1: Performance on the RU-FACS-1 dataset, ROC metric. Higher numbers indicate better performance, and best results are printed in bold.

	Area under ROC			
AU	Frm-SVM	HMM2	HMM4	Ours
1	**0.86**	0.85	0.83	**0.86**
2	0.79	0.71	0.62	**0.81**
6	0.89	**0.92**	**0.92**	0.91
12	0.94	0.94	**0.95**	0.94
14	**0.70**	**0.70**	0.69	0.68
15	**0.90**	0.86	0.85	**0.90**
17	**0.90**	0.76	0.85	0.87
24	**0.85**	0.83	0.67	0.73
1+2	0.86	0.67	0.77	**0.89**
6+12	0.95	**0.98**	**0.98**	0.96

TABLE 3.2: Performance on the RU-FACS-1 dataset, F1 metric. Higher numbers indicate better performance, and best results are printed in bold.

	Max $F1$ score			
AU	Frm-SVM	HMM2	HMM4	Ours
1	0.48	0.43	0.39	**0.59**
2	0.42	0.42	0.18	**0.56**
6	0.50	0.62	**0.63**	0.59
12	0.74	0.76	0.77	**0.78**
14	0.20	0.18	0.12	**0.27**
15	0.50	0.26	0.25	**0.59**
17	0.55	0.38	0.28	**0.56**
24	0.15	**0.18**	0.05	0.08
1+2	0.36	0.31	0.31	**0.56**
6+12	0.55	**0.64**	0.63	0.62

As noted above, the $F1$ metric is better suited for imbalanced detection tasks. Using this criterion, our method shows a substantial improvement over frame-based classification. To illustrate this point, consider Fig. 3.4 depicting the ROC and precision-recall curves of AU12 and AU15. According to the ROC metric, our method and frame-based SVM seem comparable. However, the precision-recall curves clearly



(a) AU 12 ROC

(b) AU 12 Precision-Recall

(c) AU 15 ROC

(d) AU 15 Precision-Recall

FIGURE 3.4: ROCs and precision-recall curves for AU 12 and AU 15. Although there is not a notable difference in the measured area under the ROC, precision-recall curves show a substantial improvement for our method.

show the superior performance of our method over frame-based SVM. For example, at 70% recall, the precision of frame-based SVM and our method are 0.79 and 0.87, respectively. At 50% recall for AU15, the precision of frame-based SVM is 0.48 compared to 0.67, roughly $\frac{2}{3}$ that of our method.

3.3.7 Across dataset performance

In the second experiment we compared the generalization performance of frame-based SVM and our method across datasets. Frame-based SVM and our method are trained on RU-FACS-1, and tested on Sayette, an unrelated separate dataset. Tab. 3.3 shows the ROC areas and the maximum $F1$ scores of both methods. As shown, our method consistently outperforms frame-based SVM by a large margin for all AU and their combination. Tab. 3.4 shows the precision values of both methods at two typical recall values of interest. The precision values of our method are always higher than those of frame-based SVM; in many cases the difference is as high as 50%.

TABLE 3.3: Performance on Sayette dataset. Frm-SVM and our method are trained on the RU-FACS-1 dataset which is a completely separated from Sayette.

	Area under ROC		Max $F1$ score	
AU	Frm-SVM	Ours	Frm-SVM	Ours
6	0.92	**0.94**	0.51	**0.62**
12	0.91	**0.92**	0.78	**0.79**
6+12	0.91	**0.93**	0.52	**0.61**

TABLE 3.4: Performance on Sayette dataset: precision values at recall values of interest.

	50% recall		70% recall	
AU	Frm-SVM	Ours	Frm-SVM	Ours
6	0.49	**0.60**	0.36	**0.54**
12	0.91	**0.95**	0.83	**0.87**
6+12	0.44	**0.56**	0.30	**0.53**

3.4 Summary

In this chapter, we addressed supervised learning for event detection using the Seg-SVMs framework and developed a method to detect facial Action Units (AUs) in

video. As an energy-based structure predictor, our AU detector could detect multiple target AUs simultaneously. Our detector improved frame-based SVMs by using BoW representation with soft-clustering. Our detector was trained with SOSVM, a supervised maximum-margin learning framework for structure prediction. We performed experiments on two datasets, RU-FACS-1 and Sayette, and showed the benefits of our approach compared with frame-based SVMs and HMMs, which are state-of-the-art static and dynamic approaches for AU detection. In this chapter, we trained a set of per-class detectors, assuming classes of target events can be detected independently. This approach worked well for AUs. But in many other applications, knowledge about the presence or absence of a particular event imposes a constraint on whether other events are present. In the next chapter, we will describe an algorithm that incorporates this constraint in the detection process.

Chapter 4

Supervised Learning for Sequence Labeling

"If you can't explain it simply, you don't understand it well enough."
– Albert Einstein

Using the Seg-SVMs framework, this chapter develops a supervised learning algorithm for sequence labeling, which simultaneously performs temporal segmentation and event recognition in time series. A discriminative recognition model is trained using labeled data with a multi-class SVM [Crammer and Singer, 2001] that maximizes the separating margin between classes. Once the model for all actions has been learned, simultaneous segmentation and recognition is done efficiently using dynamic programming, maximizing the SVM score of the winning class while suppressing those of the non-maximum classes.

4.1 Energy-based sequence labeling

Our goal is to factorize a time series into a sequence of events and recognize their classes. Suppose there are m classes of events. We will discuss how to learn the detectors in Section 4.2, but assume for now that the detectors $\{\mathbf{w}_j\}_{j=1}^{m}$ have been

learned. These detectors can be used independently to detect each class of target events in turn. This works well for many applications as we showed in Chapter 3 for AU detection. In many other applications, however, knowledge about the presence or absence of a particular event constrains on those of any other events, just like drinking and kissing do not occur together. This constraint can be incorporated in the detection process. If a segment $\mathbf{X_z}$ is to be detected as an event of class $\hat{y}$, it must be *confidently* recognized as class $\hat{y}$, i.e., the SVM score of class $\hat{y}$ must exceed the SVM score of any other class y by a large margin:

$$\mathbf{w}_{\hat{y}}^T \varphi(\mathbf{X_z}) \geq \mathbf{w}_y^T \varphi(\mathbf{X_z}) + 1 \ \forall y \neq \hat{y}. \tag{4.1}$$

This is equivalent to:

$$\mathbf{w}_{\hat{y}}^T \varphi(\mathbf{X_z}) \geq \max_{y \neq \hat{y}} \mathbf{w}_y^T \varphi(\mathbf{X_z}) + 1. \tag{4.2}$$

In the above constraints, $\mathbf{w}_{\hat{y}}^T \varphi(\mathbf{X_z})$ and $\mathbf{w}_y^T \varphi(\mathbf{X_z})$ are the SVM scores for assigning segment $\mathbf{X_z}$ to classes $\hat{y}$ and y respectively. We consider the energy for a segment-label pair $(\mathbf{X}_z, \hat{y})$ as a function of the recognition confidence. If $\mathbf{X_z}$ can be confidently assigned to $\hat{y}$, i.e., Constraint 4.2 is satisfied, the energy should be zero. If Constraint 4.2 is not satisfied, the energy of $(\mathbf{X}_z, \hat{y})$ is the amount of violation. Thus, the energy for a segment-label pair is defined as:

$$E(\mathbf{X_z}, \hat{y}) = \max\{\max_{y \neq \hat{y}} \mathbf{w}_y^T \varphi(\mathbf{X_z}) + 1 - \mathbf{w}_{\hat{y}} \varphi(\mathbf{X_z}), 0\}. \tag{4.3}$$

As discussed in Eq. 2.1 in Chapter 2, joint segmentation and recognition can be done by finding a legitimate segmentation that minimizes the sum of segment-label energies:

$$\underset{\{(y_t, \mathbf{z}_t)\} \in \mathcal{LS}(\mathbf{X})}{\text{minimize}} \sum_t E(\mathbf{X}_{\mathbf{z}_t}, y_t). \tag{4.4}$$

Here $\mathcal{LS}(\mathbf{X})$ is the set of all legitimate segmentation and labeling of $\mathbf{X}$ that satisfies $\mathbf{z}_1 \cup \cdots \cup \mathbf{z}_k = [1, len(\mathbf{X})]$.

What we propose is to maximize the difference between the SVM score of the winning

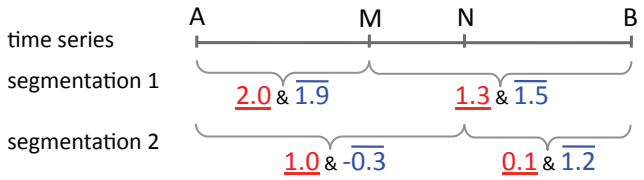


FIGURE 4.1: Which segmentation is preferred, breaking time series AB at M or N? Suppose there are only two classes, SVM scores of the first and second class for corresponding segments are printed in red and blue, respectively. Our segmentation criterion prefers to cut at N because the resulting segments can be confidently classified. This figure is best seen in color.

class y_t and that of any other class $y \neq y_t$, filtering through the Hinge loss. The idea is to seek a segmentation in which each resulting segment is assigned a class label with high confidence. This is very different from what is proposed by Shi et al. [2008], that maximizes the total SVM scores:

$$\underset{\{(y_t,\mathbf{z}_t)\}\in\mathcal{LS}(\mathbf{X})}{\text{maximize}} \sum_t \mathbf{w}_{y_t}^T \varphi(\mathbf{X}_{\mathbf{z}_t}) \tag{4.5}$$

Different from the above formulation, our segmentation criterion, Eq. (4.4), requires suppressing the non-maximum classes. To see the difference between these two criteria, consider breaking a time series AB in Figure 4.1 at either M or N. For simplicity, suppose there are only two classes, and the SVM scores of the first and second class for some segments in Figure 4.1 are in printed in underlined red and overlined blue, respectively. The segmentation criterion of Eq. (4.5) would prefer to divide AB at M because it leads to higher total SVM scores of the winning classes (total score of $3.5 = \underline{2.0} + \overline{1.5}$, $\underline{2.0}$ from segment AM and $\overline{1.5}$ from MB). On the other hand, our segmentation criterion does not prefer to cut at M because it cannot confidently classify the resulting segments. To see this, consider the segment AM, even though the SVM score of the winning class, class 1, is high, the SVM score of the alternative, class 2, is also similarly high. Our proposed criterion seeks the optimal segmentation that maximizes the difference between the SVM scores of the winning class and the next best alternative, filtering through the robust Hinge loss.

In theory, our segmentation criterion is preferred because it incorporates the constraint (4.2) in the optimization. Furthermore, as we will show in Subsection 4.2, our segmentation criterion optimizes the same objective as that of the training formulation. In Section 4.4, we will show the empirical benefits of our approach.

4.2 Maximum-margin learning for sequence labeling

We now describe how to learn $\mathbf{w}_1, \cdots, \mathbf{w}_m$, the parameters of the energy function. Given a collection of training events $\mathbf{X}^1, \cdots, \mathbf{X}^n$ with known class labels $y^1, \cdots, y^n$, we learn the parameters of the energy function to minimize the total energy while maximizing the separating margin:

$$\underset{\mathbf{w}_j}{\text{minimize}} \frac{1}{2m} \sum_{j=1}^{m} ||\mathbf{w}_j||^2 + C \sum_{i=1}^{n} E(\mathbf{X}^i, y^i). \tag{4.6}$$

Here C is the parameter controlling the trade-off between a large margin and a small total energy. This is equivalent to:

$$\underset{\mathbf{w}_j, \xi^i \geq 0}{\text{minimize}} \frac{1}{2m} \sum_{j=1}^{m} ||\mathbf{w}_j||^2 + C \sum_{i=1}^{n} \xi^i \tag{4.7}$$

$$\text{s.t. } (\mathbf{w}_{y^i} - \mathbf{w}_y)^T \varphi(\mathbf{X}^i) \geq 1 - \xi^i \ \forall i, y \neq y^i. \tag{4.8}$$

This formulation for learning $\mathbf{w}_1, \cdots, \mathbf{w}_m$ is a particular instance of multi-class SVM [Crammer and Singer, 2001].

4.3 Dynamic programming algorithm for sequence labeling

Let $s_1, \cdots, s_{k+1}$ denote the change points between $\mathbf{z}_1, \cdots, \mathbf{z}_k$, i.e., $\mathbf{z}_t = (s_t, s_{t+1}]$. See Figure 4.2 for illustration. Let $\mathbf{X}_{(s_t, s_{t+1}]}$ be $\mathbf{X}_{\mathbf{z}_t}$, the segment of time series $\mathbf{X}$, taken from frame $s_t + 1$ to frame s_{t+1} inclusive. For joint segmentation and

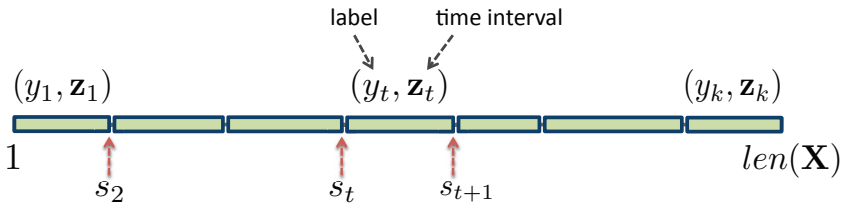


FIGURE 4.2: Joint segmentation and recognition process – we need to find the events' boundary points $s_1, \cdots, s_{k+1}$ and the class labels $y_1, \cdots, y_k$.

recognition, we need to optimize Eq. 4.3, which is equivalent to:

$$\begin{aligned} \underset{k,s_t,y_t,\xi_t \geq 0}{\text{minimize}} & \sum_{t=1}^{k} \xi_t \\ \text{s.t. } & l_{min} \leq \ s_{t+1} - s_t \leq l_{max} \ \forall t, \ s_1 = 0, s_{k+1} = len(\mathbf{X}), \\ & (\mathbf{w}_{y_t} - \mathbf{w}_y)^T \varphi(\mathbf{X}_{(s_t, s_{t+1}]}) \geq 1 - \xi_t \ \forall t, y \neq y_t. \end{aligned} \tag{4.9}$$

Given the parameters $\{\mathbf{w}_j\}_{j=1}^m$, this optimization problem can be solved using a dynamic programming algorithm, which makes two passes over the time series $\mathbf{X}$. In the forward pass, at frame u ($1 \leq u \leq len(\mathbf{X})$), it computes the best objective value for segmenting and labeling truncated time series $\mathbf{X}_{(0,u]}$ (ignoring frames from $u+1$ onward), i.e.

$$\begin{aligned} h(u) = \min_{k,s_t,y_t,\xi_t \geq 0} & \sum_{t=1}^{k} \xi_t, \\ \text{s.t. } & l_{min} \leq \ s_{t+1} - s_t \leq l_{max} \ \forall t, \ s_1 = 0, s_{k+1} = u, \\ & (\mathbf{w}_{y_t} - \mathbf{w}_y)^T \varphi(\mathbf{X}_{(s_t, s_{t+1}]}) \geq 1 - \xi_t \ \forall t, y \neq y_t. \end{aligned} \tag{4.10}$$

The forward pass computes $h(u)$, as well as $l(u)$, for $u = 1, \cdots, len(\mathbf{X})$ using the recursive formulas:

$$\begin{aligned} h(u) &= \min_{l_{min} \leq l \leq l_{max}} \{\xi(u,l) + h(u-l)\}, \\ l(u) &= \operatorname*{argmin}_{l_{min} \leq l \leq l_{max}} \{\xi(u,l) + h(u-l)\}. \end{aligned}$$

Here $\xi(u,l)$ denotes the slack value of segment $\mathbf{X}_{(u-l,u]}$, i.e.

$$\xi(u,l) = \max\{0, 1 - (\mathbf{w}_{\hat{y}} - \mathbf{w}_{\tilde{y}})^T \varphi(\mathbf{X}_{(u-l,u]})\}, \tag{4.11}$$

$$\text{where} \quad \hat{y} = \underset{y}{\operatorname{argmax}}\, \mathbf{w}_y^T \varphi(\mathbf{X}_{(u-l,u]}), \text{and} \tag{4.12}$$

$$\tilde{y} = \underset{y \neq \hat{y}}{\operatorname{argmax}}\, \mathbf{w}_y^T \varphi(\mathbf{X}_{(u-l,u]}). \tag{4.13}$$

The backward pass of the algorithm finds the best segmentation for $\mathbf{X}$, starting with $s_{k+1} = len(\mathbf{X})$ and using the backward-recursive formula:

$$s_t = s_{t+1} - l(s_{t+1}).$$

Once the optimal segmentation has been determined, the optimal assignment of class labels can be found using:

$$y_t = \underset{y}{\operatorname{argmax}}\, \mathbf{w}_y^T \varphi(\mathbf{X}_{(s_t, s_{t+1}]}).$$

The total complexity for the forward and backward passes of this dynamic programming algorithm is $O(m(l_{max} - l_{min} + 1)len(\mathbf{X}))$. This is linear in the length of the time series.

4.4 Experiments

This section describes experimental results on three standard datasets: honeybee dancing [Oh et al., 2008], Weizmann [Gorelick et al., 2007], and Hollywood [Laptev et al., 2008]. In all experiments we measured the joint segmentation-recognition performance as follows. We ran our algorithm on long video sequences to find the optimal segmentation and class labels. At that point, each frame was associated with a particular class, and the overall frame-level accuracy against the ground truth labels was calculated as the ratio between the number of agreements over the total number of frames. This evaluation criterion is different from recognition accuracy of algorithms that require pre-segmented video clips. As a consequence, our results here are not directly comparable to some published numbers in the literature [Gorelick

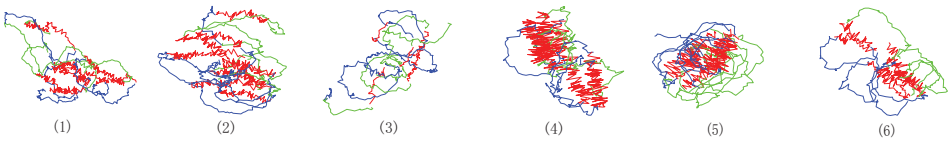


FIGURE 4.3: Honeybee dataset—trajectories of dancing bees. Each dance trajectory is the output of a vision-based tracker. The segments are color coded; red, green, and blue correspond to waggle, right-turn, and left-turn, respectively. This figure is best seen in color.

et al., 2007, Laptev et al., 2008, Satkin and Hebert, 2010]. However, where available, we included the previously reported results for reference.

4.4.1 Honeybee dataset

The honeybee dataset [Oh et al., 2008] contains video sequences of honeybees which communicate the location and distance to a food source through a dance that takes place within the hive. The dance can be decomposed into three different movement patterns: waggle, right-turn, and left-turn. During the waggle dance, the bee moves roughly in a straight line while rapidly shaking its body from left to right; the duration and orientation of this phase correspond to the distance and the orientation to the food source. At the endpoint of a waggle dance, the bee turns in a clockwise or counterclockwise direction to form a turning dance. These turning dances often shape like a capital C. The dataset consists of six video sequences with lengths 1058, 1125, 1054, 757, 609, and 814 frames, respectively.

The bees were visually tracked (Figure 4.4.a), and their locations and head angles were recorded. The 2D trajectories of the bees in six sequences are shown in Fig. 4.3. The frame-level feature vector was $[x, y, \sin(\theta), \cos(\theta)]$, where (x, y) was the 2D location of the bee and θ was the bee's head angle. Once the sequence observations were obtained, the trajectories were preprocessed as in Fox et al. [2009]. Specifically, the trajectory sequences were rotated so that the waggle dances had head angle measurements centered about zero radian. The sequences were then translated to center at $(0, 0)$, and the 2D coordinates were scaled to the $[-1, 1]$ range. Aligning the waggle dances was possible by looking at the high frequency portions of the head angle measurements. Following the suggestion of Oh et al. [2008], the data was

smoothed using Gaussian FIR pulse-shaping filter with 0.5dB bandwidth-symbol time. Figure 4.4.b shows the correlation between the feature vectors and the labels. Since the lengths of original waggle, right-turn, and left-turn sequences are quite long, we further broke them down into smaller subsequences (maximum length 13) to increase the number of training instances.

Following [Altun et al., 2003] and inspired by HMMs, we propose to use two types of features, interactions between the observation vectors and the set of predefined states as well as the transition between states of neighboring frames:

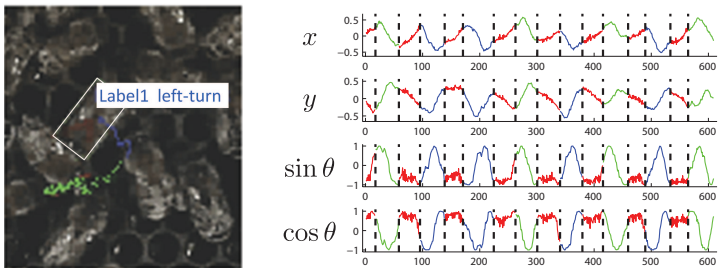
$$\varphi(\mathbf{X}_\mathbf{z}) = \sum_{p \in \mathbf{z}} \begin{bmatrix} \boldsymbol{\phi}^{obs}(\mathbf{X}_p) \\ \boldsymbol{\phi}^{int}(\mathbf{X}_p) \end{bmatrix}. \tag{4.14}$$

Here $\phi^{obs}(\mathbf{X}_p)$ and $\phi^{int}(\mathbf{X}_p)$ are the observation and interaction feature vectors, respectively. These feature vectors are computed as follows. First we build a dictionary of temporal words by clustering the raw feature vectors from the time series in the dataset. Let $\mathbf{c}_1, \cdots, \mathbf{c}_r$ denote the set of clustering centroids. We consider $\phi^{obs}(\mathbf{X}_p)$ as a $r \times 1$ vector with the i^{th} entry is $\phi_i^{obs}(\mathbf{X}_p) = \mu \exp(-\gamma ||\mathbf{X}_p - \mathbf{c}_i||^2)$. Intuitively, the i^{th} entry of observation vector is the pseudo-probability that $\mathbf{X}_p$ belongs to state i, which is proportional to how close $\mathbf{X}_p$ to the cluster centroid $\mathbf{c}_i$. The scale factor μ is chosen such that the sum of the entries of $\phi^{obs}(\mathbf{X}_p)$ is one. The interaction feature vector $\phi^{int}(\mathbf{X}_p)$ is defined as a $r^2 \times 1$ vector, with:

$$\phi^{int}_{(u-1)r+v}(\mathbf{X}_p) = \phi^{obs}_u(\mathbf{X}_p)\phi^{obs}_v(\mathbf{X}_{p-1}) \ \forall u, v = 1, \cdots, r.$$

The above quantity is the pseudo-probability for transitioning from state v to state u at time p. The interaction feature vector depends on both the observation vectors of the frame $\mathbf{X}_p$ and the preceding frame $\mathbf{X}_{p-1}$. In our experiment, we set $r = 15$.

Following Fox et al. [2009], Oh et al. [2008], we adopted the leave-one-out evaluation strategy: training on five sequences and testing on the left-out sequence. Table 4.1 displays the experimental results of our method along with three state-of-the-art methods. SLDS and PS-SLDS [Oh et al., 2008] are switching linear dynamical system and parametric segmental switching linear dynamical system, respectively. HDP-HMM [Fox et al., 2009] is the method combining hierarchical Dirichlet process



(a) (b)

FIGURE 4.4: a) Visual tracking: green + blue trajectory and the bounding box for tracking. b) plots of the frame-level features $[x, y, \sin(\theta), \cos(\theta)]$. Red, green, and blue correspond to waggle, right-turn, and left-turn, respectively. This is best seen in color.

TABLE 4.1: Frame-level accuracy (%) on honeybee dataset. Our method achieved similar and sometimes better results than state-of-the-art methods [Fox et al., 2009, Oh et al., 2008]. Averaged over all six sequences, our method yielded the best result.

Sequence	1	2	3	4	5	6	Mean
SLDS [Oh et al., 2008]	74.0	86.1	81.3	**93.4**	90.2	90.4	85.9
PS-SLDS [Oh et al., 2008]	75.9	92.4	**83.1**	**93.4**	90.4	91.0	87.7
HDP-HMM [Fox et al., 2009]	55.0	86.3	81.7	89.0	**92.4**	89.6	83.3
MaxScoreSeg	82.2	85.3	75.0	87.5	88.8	88.0	84.5
Ours	**85.9**	**92.6**	81.3	92.3	90.6	**93.1**	**89.3**

prior and HMM. Although all methods are supervised learning, the setting of HDP-HMM is slightly different from those of the others. HDP-HMM requires knowing the testing sequences (without labels) at training time. We also implemented MaxScoreSeg (c.f., Shi et al. [2008]), a variant of our proposed algorithm, that performed temporal segmentation by maximizing the total SVM scores (Eq. 4.5) instead of maximizing the assignment confidence (Eq. 4.4). The reported numbers in Table 4.1 are frame-level accuracy (%) measuring the joint segmentation-recognition performance as described at the beginning of Section 4.4. As can be seen, our method achieved similar or better results than state-of-the-art methods on all sequences, and it had the best overall performance. Figure 4.5 displays side-by-side comparison of the prediction result and the human-labeled ground truth.

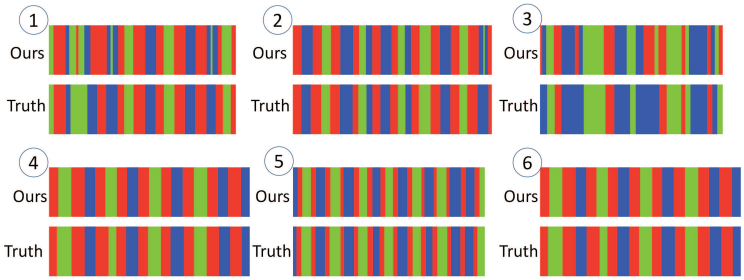
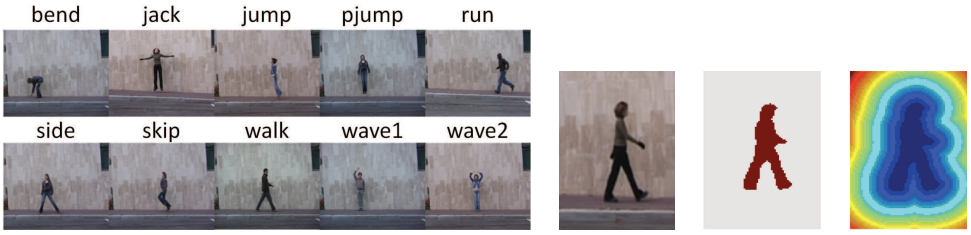


FIGURE 4.5: Automatic segmentation-recognition versus human-labeled ground truth. The segments are color coded; red, green, and blue correspond to waggle, right-turn, and left-turn, respectively. This figure is best seen in color.



(a) (b) (c) (d)

FIGURE 4.6: Weizmann dataset. (a): typical frames. (b)-(d): how frame-level features are computed; (b) is an original frame, (c) is the binary mask, and (d) is the Euclidean distance transform.

4.4.2 Weizmann dataset

The Weizmann dataset contains 90 video sequences (180×144 pixels, deinterlaced 50fps) of 9 people, each performing 10 actions: bend, jumping-jack (or shortly jack), jump-forward-on-two-legs (jump), jump-in-place-on-two-legs (pjump), run, gallop-sideways (side), skip, walk, wave-one-hand (wave1), and wave-two-hands (wave2). Figure 4.6(a) displays several typical frames extracted from the dataset. Each video sequence in this dataset only consists of a single action.

To evaluate the segmentation and recognition performance of our method, we performed experiments on longer video sequences which were created by concatenating existing single-action sequences. Specifically, we created 9 long sequences, each composed of 10 videos for 10 different actions (each original video samples was used only once). Following Gorelick et al. [2007], we extracted binary masks (Figure 4.6(c))

TABLE 4.2: Performance on Weizmann dataset, confusion matrix for segmentation and recognition of 10 different actions at frame level. The number at row R and column C is the proportion of R class which is classified as C class. For example, 3% of the **wave1** frames is misclassified as **wave2** class. The average accuracy is 87.7%.

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	.85	.08	.05	.01	.00	.01	.00	.00	.00	.00
jack	.00	.93	.00	.00	.04	.00	.01	.00	.01	.01
jump	.00	.01	.88	.06	.04	.00	.00	.00	.00	.01
pjump	.00	.01	.04	.85	.02	.00	.00	.00	.08	.00
run	.00	.00	.03	.00	.93	.00	.00	.01	.03	.00
side	.00	.03	.00	.03	.00	.90	.00	.01	.00	.03
skip	.00	.00	.02	.00	.05	.00	.77	.03	.00	.13
walk	.00	.00	.08	.00	.00	.00	.00	.88	.00	.04
wave1	.00	.00	.00	.00	.01	.00	.03	.00	.93	.03
wave2	.00	.02	.02	.00	.00	.00	.08	.02	.01	.85

and computed Euclidean distance transform (Figure 4.6(d)) for frame-level features. We built a codebook of temporal words with 100 clusters using k-means. As in the experiment for honeybee dataset, we measured the leave-one-out segmentation and recognition performance. Table 4.2 shows the confusion matrix for segmentation and recognition of 10 actions. Our method yielded the average accuracy of 87.7%, aggregated over 9 sequences and 20 runs. Gorelick et al. [2007] reported the recognition result of 97.8%. Unfortunately, their result and ours are not directly comparable. Their method required pre-segmented video sequences and only measured the recognition performance. The variant of our method, MaxScoreSeg [Shi et al., 2008], that performed temporal segmentation by maximizing the total SVM scores (Eq. 4.5) obtained the average accuracy of 69.7%. This relatively low accuracy is due to the mismatch between the segmentation criterion and the training objective, as explained in Section 4.1.

To evaluate the performance of the proposed method in the presence of the null

TABLE 4.3: Weizmann dataset with the null class. Confusion matrix for segmentation and recognition of five different actions: bend, jack, jump, pjump, and run. The null class is the combination of all other classes. The average accuracy is 93.3%.

	bend	jack	jump	pjump	run	Null
bend	.96	.01	.01	.00	.00	.01
jack	.00	.97	.00	.01	.00	.02
jump	.00	.00	.88	.06	.04	.02
pjump	.00	.00	.01	.98	.00	.01
run	.00	.00	.01	.00	.91	.08
Null	.01	.03	.00	.03	.03	.90

class, background clutter with large variability, we repeated the experiment considering the last five classes of actions (side, skip, walk, wave1, and wave2) as the null class. Table 4.3 shows the confusion matrix for five actions and the null class. Our method yielded the average accuracy of 93.3%, compared with 77.9% of MaxScore-Seg. Figure 4.7 displays side-by-side comparison of the prediction result and the human-labeled ground truth. Except for several cases, the majority of error occurs at the boundaries between actions. Error at the boundaries does not necessarily indicate the flaw of our method as human labels are often imperfect [Satkin and Hebert, 2010].

4.4.3 Hollywood dataset

Hollywood dataset contains video samples of human action from 32 movies. Each sample is labeled with one of eight action classes: AnswerPhone, HugPerson, Kiss, SitDown, SitUp, GetOutCar, HandShake, and StandUp. The dataset is divided into two disjoint subsets; the training set contains video clips from 12 movies and the testing set contains the remaining clips. The total number of video samples in the training and testing sets are 219 and 211, respectively. Here we selected the first

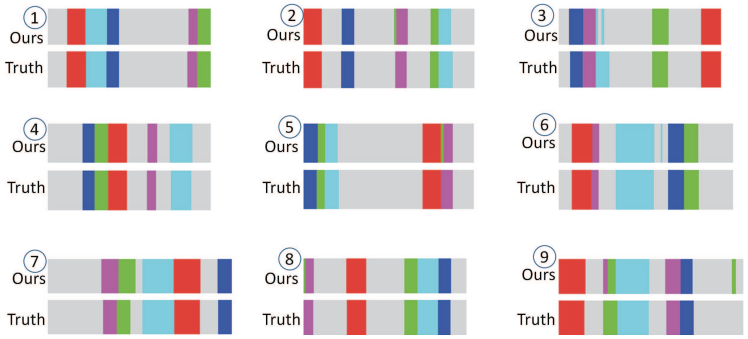


FIGURE 4.7: Automatic segmentation-recognition versus human-labeled ground truth for Weizmann dataset. The segments are color coded; red, cyan, magenta, blue, green, and gray correspond to bend, jack, jump, pjump, run, and null classes, respectively. This figure is best seen in color.

FIGURE 4.8: Typical frames from the Hollywood dataset.

four classes as actions to be recognized, and the others were considered as parts of the null class.

Following Laptev et al. [2008], we detected space-time interest points and described them using histogram of oriented (spatial) gradients (HOG). Features belong to the same frame were combined together. A codebook of temporal words with 100 clusters was constructed using k-means quantization. To evaluate the joint segmentation and recognition performance, we created 30 long testing sequences by concatenating eight randomly selected original video samples. The evaluation criterion was based on frame-level accuracy as described at the beginning of Section 4.4. Our method achieved the average accuracy of 42.24% (averaged over 30 sequences, repeated with 50 runs). As a reference, Laptev et al. [2008] reported the average recognition result of 27% on this dataset with the same HOG features. Unfortunately, their result

TABLE 4.4: Hollywood dataset—confusion matrix for AnswerPhone (AP), HugPerson (HP), Kiss (KS), SitDown (SD), and the null class (all other actions). The average accuracy is 42.24%.

	AP	HP	KS	SD	Null
AP	.35	.14	.13	.22	.16
HP	.08	.34	.20	.17	.22
KS	.08	.10	.51	.11	.21
SD	.09	.06	.14	.45	.27
Null	.11	.07	.17	.19	.47

and ours are not directly comparable since their method required pre-segmented video sequences and only measured the recognition performance. Furthermore, the number of action classes in two experiments are different.

4.5 Summary

This chapter described a novel algorithm for simultaneous temporal segmentation and recognition of temporal events, which used the proposed Seg-SVMs framework. The recognition model was trained discriminatively using multi-class SVM, while segmentation inference was done efficiently with dynamic programming. This algorithm provides a principled technique for time series segmentation and event recognition. Experimental validation on several human action datasets showed the competitiveness of our algorithm against state-of-the-art methods. Though the proposed method yielded encouraging results on standard datasets, its reliance on fully labeled data for training inevitably limits its applicability to small training sets with few event classes. In Chapter 7, we will remove this reliance on labeled data and develop an unsupervised alternative.

Chapter 5

Supervised Learning for Early Event Detection

"You may delay, but time will not."
– Benjamin Franklin

This chapter addresses the need for early detection of temporal events using the Seg-SVMs framework. This need arises in a wide spectrum of applications ranging from disease outbreak detection to security and robotics applications. We derive Max-Margin Early Event Detectors (MMED), a novel formulation for training event detectors that recognize partial events, enabling early detection. MMED is based on SOSVM [Tsochantaridis et al., 2005], but extends it to accommodate the nature of sequential data. In particular, we simulate the sequential frame-by-frame data arrival for training time series and learn an event detector that correctly classifies partially observed sequences. Fig. 5.1 illustrates the main idea behind MMED: partial events are simulated and used as positive training examples. It is important to emphasize that we train a *single* event detector to recognize *all* partial events. But MMED does more than augmenting the set of training examples. It trains a detector to localize the temporal extent of a target event, even when the target event has yet completed. This requires monotonicity of the detection function with respect to the inclusion relationship between partial events; the detection score (confidence)

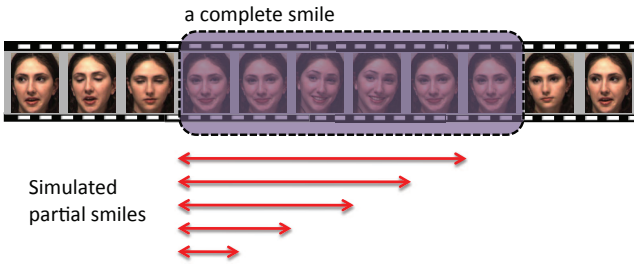


FIGURE 5.1: We simulate the sequential arrival of training data and use partial events as positive training examples. In this figure, the red segments indicate the temporal extents of the partial events.

of a partial event cannot exceed the score of an encompassing partial event. MMED provides a principled mechanism to achieve this monotonicity, which cannot be assured by a naive solution that simply augments the set of training examples.

The learning formulation of MMED is a constrained quadratic optimization problem. This formulation is theoretically justified. In Sec. 5.2.2, we discuss two ways for quantifying the loss of a detector on streaming data. We prove, in both cases, the objective of the learning formulation is to minimize an upper bound of the true loss on the training data.

MMED has numerous benefits. First, MMED inherits the advantages of SOSVM, including its convex learning formulation and its ability for accurate localization of event boundaries. Second, MMED, specifically designed for early detection, is superior to SOSVM and other competing methods with respect to the timeliness of the detection. Experiments on datasets of varying complexity, from synthetic data and sign language to facial expression and human action, showed that our method often made faster detection while maintaining comparable or even better accuracy.

To the best of our knowledge, in the literature of computer vision, this is the first learning formulation that is explicitly designed for early event detection.

5.1 Energy-based early event detection

Early event detection requires realtime processing, and therefore, target events must be detected sequentially. We propose a detection mechanism as follows. The detector reads from a stream of data and keeps a sequence of observations in its memory. It continuously monitors for the happening of a target event. If a target event is detected, the temporal extent of the event is returned. If a target event is recognized complete, the detector's memory is cleared and the process recurs to detect the upcoming target event. Thus, at every single time step, the detector needs to detect at most one target event.

Our early event detector is based on an energy model that is similar to the model for offline detection, but it detects one event at a time. As in Chapter 3, we assume target events belong to a single class and use $E(\mathbf{X}_\mathbf{z})$ in place of $E(\mathbf{X}_\mathbf{z}, y)$ for brevity. Let $\mathbf{X}$ be the time series correspond to the sequence of observations in the detector's memory. We find the segment of $\mathbf{X}$ that yields the minimum energy:

$$\mathbf{z}^* := \operatorname*{argmin}_{\mathbf{z} \in \mathcal{Z}} E(\mathbf{X}_\mathbf{z}). \tag{5.1}$$

We report $\mathbf{z}^*$ as a partial or complete event of interest if the minimum energy is negative and report no detection otherwise. Hence, the output of the detector on $\mathbf{X}$ is defined as:

$$g(\mathbf{X}) := \begin{cases} \mathbf{z}^* & \text{if } E(\mathbf{X}_{\mathbf{z}^*}) < 0 \\ \emptyset & \text{otherwise} \end{cases} \tag{5.2}$$

As for offline detection, Chapter 3, the energy of a segment is defined as the negative of the detection score $E(\mathbf{X}_\mathbf{z}) := -f(\mathbf{X}_\mathbf{z})$, with the detection score defined as:

$$f(\mathbf{X}_\mathbf{z}) := \mathbf{w}^T \varphi(\mathbf{X}_\mathbf{z}) + b. \tag{5.3}$$

Recall $\mathcal{I}$ is $\mathcal{Z} \cup \{\emptyset\}$ and the detection score of an empty segment is zero, $f(\mathbf{X}_\emptyset) = 0$. Thus the output of the detector on $\mathbf{X}$ can be conveniently expressed as:

$$g(\mathbf{X}) = \operatorname*{argmax}_{\mathbf{z} \in \mathcal{I}} f(\mathbf{X}_\mathbf{z}). \tag{5.4}$$

5.2 Maximum-margin learning for early event detection

As shown in Section 3.2, SOSVM was used to train a detector to detect complete events. SOSVM, however, does not train detectors to recognize partial events. Consequently, using this method for early detection would lead to unreliable decisions as we will illustrate in the experimental section. This section presents a novel learning formulation that extends SOSVM to overcome this limitation.

5.2.1 Learning with simulated sequential data

Let the training time series be $\mathbf{X}^1, \cdots, \mathbf{X}^n \in \mathcal{X}$ and their associated ground truth annotations for the occurrence of target events be $\mathbf{z}^1, \cdots, \mathbf{z}^n$. Here we assume each training sequence contains at most one target event (we can always break a training sequence of several events into shorter sequences of single events). To support early detection of events in time series data, we propose to use partial events as positive training examples (Fig. 5.1). In particular, we simulate the sequential arrival of training data as follows. Suppose the length of $\mathbf{X}^i$ is l_i. For each time $t = 1, \cdots, l_i$, let $\mathbf{z}_t^i$ be the part of event $\mathbf{z}^i$ that has already happened, i.e., $\mathbf{z}_t^i = \mathbf{z}^i \cap [1, t]$, which is possibly empty. Ideally, we want the output of the detector on time series $\mathbf{X}^i$ at time t is the partial event $\mathbf{z}_t^i$, i.e.,

$$\mathbf{z}_t^i = g(\mathbf{X}_{[1,t]}^i). \tag{5.5}$$

Here, $\mathbf{X}_{[1,t]}^i$ is the subsequence of $\mathbf{X}^i$ from frame 1 to frame t, and $g(\mathbf{X}_{[1,t]}^i)$ is the output of the detector on this subsequence. Substitute Eq. 5.4 into the above, we

get an equivalent constraint:

$$\mathbf{z}_t^i = \underset{\mathbf{z}\in\mathcal{I},\mathbf{z}\subset[1,t]}{\operatorname{argmax}} f(\mathbf{X}_{\mathbf{z}}^i). \tag{5.6}$$

To understand the differences between the requirement of early event detection and that of offline event detection, compare the above constraint and the constraint in Eq. 3.5, which, for convenience, is reprinted below:

$$\mathbf{z}^i = \underset{\mathbf{z}\in\mathcal{I}}{\operatorname{argmax}} f(\mathbf{X}_{\mathbf{z}}^i). \tag{5.7}$$

There are key differences between Eq. 5.6 and Eq. 5.7. The Left Hand Side (LHS) of Eq. 5.7 is the complete event, while the LHS of Eq. 5.6 is the partial event at a particular time t. The Right Hand Side (RHS) of Eq. 5.7 is the output of the detector on the entire sequence $\mathbf{X}^i$ while the RHS of Eq. 5.6 is the output of the detector on the partially observed sequence $\mathbf{X}_{[1,t]}^i$, from frame 1 to frame t.

Eq. 5.6 is equivalent to:

$$f(\mathbf{X}_{\mathbf{z}_t^i}^i) \geq f(\mathbf{X}_{\mathbf{z}}^i)\ \forall \mathbf{z}\in\mathcal{I}, \mathbf{z}\subset[1,t]. \tag{5.8}$$

This constraint requires the score of the partial event $\mathbf{z}_t^i$ to be bigger than the score of any other time series segment $\mathbf{z}$ which has been seen in the past, $\mathbf{z}\subset[1,t]$. This is illustrated in Fig. 5.2. Note that the score of the partial event is not required to be bigger than the score of a segment in the future.

As for offline event detection, we enforce the above constraint to be well satisfied by a margin. This margin is adaptive and proportional to $\Delta(\mathbf{z}_t^i, \mathbf{z})$, the loss of the detector for outputting $\mathbf{z}$ when the desired output is $\mathbf{z}_t^i$. Hence, the desired constraint is:

$$f(\mathbf{X}_{\mathbf{z}_t^i}^i) \geq f(\mathbf{X}_{\mathbf{z}}^i) + \Delta(\mathbf{z}_t^i, \mathbf{z})\ \forall \mathbf{z}\in\mathcal{I}, \mathbf{z}\subset[1,t]. \tag{5.9}$$

This constraint should be enforced for all $t = 1, \cdots, l_i$, and each training time series leads to a set of these constraints. To learn the parameters $(\mathbf{w}, b)$ for early detection, we can maximize the margin subject to all these constraints, i.e.,

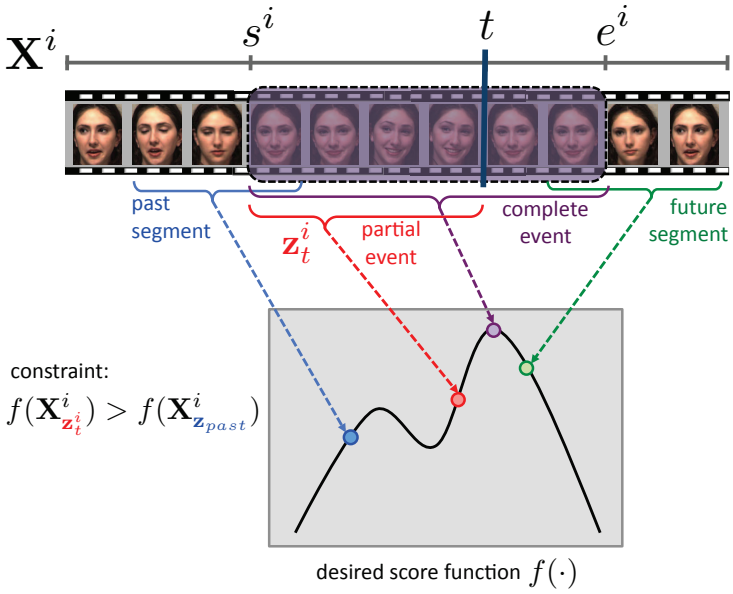


FIGURE 5.2: From desire to constraint. The desired score function for early event detection: the complete event must have highest detection score, and the detection score of a partial event must be bigger than that of any segment that ends before the partial event. To learn this function, we explicitly consider partial events during training. At time t, the score of the truncated event (red segment) is required to be bigger than the score of any segment in the past (e.g., blue segment); however, it is not required to be bigger than the score of any future segment (e.g., green segment). This figure is best seen in color.

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \ & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \ & f(\mathbf{X}^i_{\mathbf{z}_t^i}) \geq f(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}_t^i, \mathbf{z}) \\ & \forall i, \forall t = 1, \cdots, l_i, \forall \mathbf{z} \in \mathcal{I}, \mathbf{z} \subset [1, t]. \end{aligned} \tag{5.10}$$

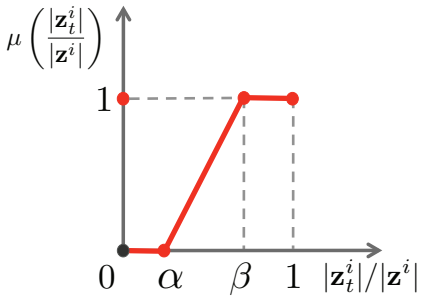


FIGURE 5.3: μ – a function to weigh the importance of partially observed events. Here 0 and 1 correspond to the total absence and full completion of the event of interest, respectively. $\mu(0) = \mu(1)$ emphasizes that true rejection is as important as true detection of the complete event. This is best seen in color.

As in the formulation of SOSVM, the constraints are allowed to be violated by introducing slack variables, and we obtain the following learning formulation:

$$\underset{\mathbf{w},b,\{\xi^i\}}{\text{minimize}} \ \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi^i, \tag{5.11}$$

$$\text{s.t. } f(\mathbf{X}^i_{\mathbf{z}_t^i}) \geq f(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}_t^i, \mathbf{z}) - \frac{\xi^i}{\mu\left(\frac{|\mathbf{z}_t^i|}{|\mathbf{z}^i|}\right)}$$

$$\forall i, \forall t = 1, \cdots, l_i, \forall \mathbf{z} \in \mathcal{I}, \mathbf{z} \subset [1, t], \tag{5.12}$$

$$\xi^i \geq 0 \ \forall i. \tag{5.13}$$

In the above formulation, $|\cdot|$ denotes the length function, and $\mu\left(\frac{|\mathbf{z}_t^i|}{|\mathbf{z}^i|}\right)$ is a function of the proportion of the event that has occurred at time t. $\mu\left(\frac{|\mathbf{z}_t^i|}{|\mathbf{z}^i|}\right)$ is a slack variable rescaling factor and should correlate with the importance of rightly detecting at time t whether the event $\mathbf{z}^i$ has happened. $\mu(\cdot)$ can be any arbitrary non-negative function, and in general, it should be a non-decreasing function in $(0, 1]$. In our experiments, we find the piece-wise linear function as depicted in Fig. 5.3 is a reasonable choice. There, α and β are tunable parameters; $\mu(0) = \mu(1)$ emphasizes that *true rejection* when the event has not started is as important as *true detection* when the event has completed. $\Delta(\mathbf{z}_t^i, \mathbf{z})$ is the loss function, for quantifying the loss associated with outputting $\mathbf{z}$ at time t when the true truncated event is $\mathbf{z}_t^i$. A possible and popular loss function is: $\Delta(\mathbf{z}_t^i, \mathbf{z}) = 1 - \frac{2|\mathbf{z}_t^i \cap \mathbf{z}|}{|\mathbf{z}_t^i| + |\mathbf{z}|}$ if $\mathbf{z}_t^i \neq \mathbf{z}$ and 0 otherwise.

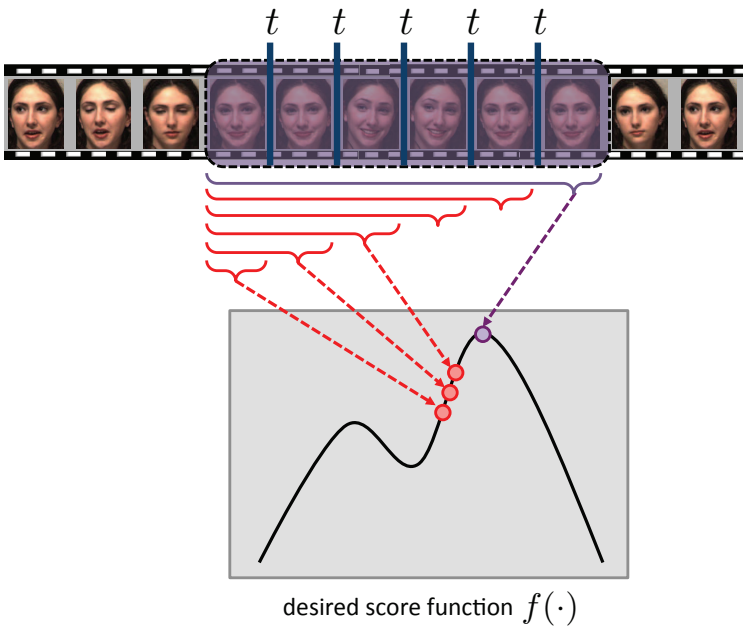


FIGURE 5.4: Monotonicity requirement – the detection score (confidence) of a partial event cannot exceed the score of an encompassing partial event. MMED provides a principled mechanism to achieve this monotonicity.

This learning formulation is an extension of SOSVM. From this formulation, we obtain SOSVM by not simulating the sequential arrival of training data, i.e., to set $t = l_i$ instead of $t = 1, \cdots, l_i$ in Constraint (5.12). The key idea of MMED is to learn a single detector to recognize all partial events. But our method does more than augmenting the set of training examples; it provides a principled mechanism for enforcing monotonicity with respect to the inclusion relationship between partial events, as illustrated in Figure 5.4. This monotonicity requirement cannot be assured by a naive solution that simply augments the set of training examples.

For a better understanding of Constraint (5.12), let us break it into three cases: i) $t < s^i$; ii) $t \geq s^i$, $\mathbf{z} = \emptyset$; iii) $t \geq s^i$, $\mathbf{z} \neq \emptyset$. Constraint (5.12) is the combination of

the following constraints:

$$\mathbf{w}^T\varphi(\mathbf{X}^i_{\mathbf{z}}) + b \leq -1 + \xi^i/\mu(0)\ \forall i, \forall \mathbf{z} \subset [1, s^i), \mathbf{z} \neq \emptyset, \tag{5.14}$$

$$\mathbf{w}^T\varphi(\mathbf{X}^i_{\mathbf{z}^i_t}) + b \geq 1 - \xi^i/\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)\ \forall i, \forall t \geq s^i, \tag{5.15}$$

$$\mathbf{w}^T\varphi(\mathbf{X}^i_{\mathbf{z}^i_t}) \geq \mathbf{w}^T\varphi(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}^i_t, \mathbf{z}) - \xi^i/\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)$$
$$\forall i, \forall t \geq s^i, \forall \mathbf{z} \subset [1, t], \mathbf{z} \neq \emptyset. \tag{5.16}$$

Cases (i), (ii), (iii) lead to Constraints (5.14), (5.15), (5.16), respectively. To see this, recall $f(\mathbf{X}_{\mathbf{z}}) = \mathbf{w}^T\varphi(\mathbf{X}_{\mathbf{z}}) + b$ if $\mathbf{z} \neq \emptyset$ and 0 otherwise. Furthermore, recall $\mathbf{z}^i_t = \emptyset$ for $t < s^i$ and $\Delta(\mathbf{z}^i_t, \mathbf{z}) = 1$ if $\mathbf{z}^i_t$ is different from $\mathbf{z}$ and either of them is empty. Constraint (5.14) prevents false detection when the event has not started. Constraint (5.15) requires successful recognition of truncated events. Constraint (5.16) trains the detector to localize the temporal extent of the events.

The proposed learning formulation Eq. (5.11) is convex, but it contains a large set of constraints. Following Tsochantaridis et al. [2005], we propose to use constraint generation in optimization, i.e., we maintain a smaller subset of constraints and iteratively update it by adding the most violated ones. Constraint generation is guaranteed to converge to the global minimum. In our experiments described in Sec. 5.3, this usually converges within 20 iterations.

5.2.2 Loss function and empirical risk minimization

In Sec. 5.2.1, we have proposed a formulation for training early event detectors. This section provides further discussion on what exactly is being optimized. First, we briefly review the loss of SOSVM and its surrogate empirical risk. We then describe two general approaches for quantifying the loss of a detector on streaming data. In both cases, what Eq. (5.11) minimizes is an upper bound on the loss.

As previously explained, $\Delta(\mathbf{z}, \hat{\mathbf{z}})$ is the function that quantifies the loss associated with a prediction $\hat{\mathbf{z}}$, if the true output value is $\mathbf{z}$. Thus, in the setting of

offline detection, the loss of a detector g on a sequence-event pair $(\mathbf{X}, \mathbf{z})$ is quantified as $\Delta(\mathbf{z}, g(\mathbf{X}))$. Suppose the sequence-event pairs $(\mathbf{X}, \mathbf{z})$ are generated according to some distribution $P(\mathbf{X}, \mathbf{z})$, the loss of the detector g is $\mathcal{R}^{\Delta}_{true}(g) = \int_{\mathcal{X}\times\mathcal{I}} \Delta(\mathbf{z}, g(\mathbf{X}))dP(\mathbf{X}, \mathbf{z})$. However, P is unknown so the performance of g is described by the empirical risk on the training data $\{(\mathbf{X}^i, \mathbf{z}^i)\}$, assuming they are generated i.i.d according to P. The empirical risk is $\mathcal{R}^{\Delta}_{emp}(g) = \frac{1}{n}\sum_{i=1}^{n} \Delta(\mathbf{z}^i, g(\mathbf{X}^i))$. It has been shown that SOSVM [Tsochantaridis et al., 2005] minimizes an upper bound on the empirical risk $\mathcal{R}^{\Delta}_{emp}$. In other words, if $\{\xi^{*1}, \cdots, \xi^{*n}\}$ is the optimal solution of the slack variables in Eq. 5.11, then $\frac{1}{n}\sum_{i=1}^{n} \xi^{i*}$ is an upper bound on the empirical risk $\mathcal{R}^{\Delta}_{emp}$.

Due to the nature of continual evaluation, quantifying the loss of an online detector on streaming data requires aggregating the losses evaluated throughout the course of the data sequence. Let us consider the loss associated with a prediction $\mathbf{z} = g(\mathbf{X}^i_{[1,t]})$ for time series $\mathbf{X}^i$ at time t as $\Delta(\mathbf{z}^i_t, \mathbf{z})\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)$. Here $\Delta(\mathbf{z}^i_t, \mathbf{z})$ accounts for the difference between the output $\mathbf{z}$ and true truncated event $\mathbf{z}^i_t$. $\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)$ is the scaling factor; it depends on how much the temporal event $\mathbf{z}^i$ has happened. Two possible ways for aggregating these loss quantities is to use the maximum or the average of $\{\Delta(\mathbf{z}^i_t, g(\mathbf{X}^i_{[1,t]}))\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)\}$. They lead to two different empirical risk functions for a set of training time series:

$$\mathcal{R}^{\Delta,\mu}_{max}(g) = \frac{1}{n}\sum_{i=1}^{n} \max_t \left\{\Delta(\mathbf{z}^i_t, g(\mathbf{X}^i_{[1,t]}))\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)\right\}, \tag{5.17}$$

$$\mathcal{R}^{\Delta,\mu}_{ave}(g) = \frac{1}{n}\sum_{i=1}^{n} \operatorname*{mean}_t \left\{\Delta(\mathbf{z}^i_t, g(\mathbf{X}^i_{[1,t]}))\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)\right\}. \tag{5.18}$$

In the following, we state and prove a proposition that establishes that the learning formulation given in Eq. 5.11 minimizes an upper bound of the above two empirical risk functions.

Proposition: Denote by $\boldsymbol{\xi}^*(g)$ the optimal solution of the slack variables in Eq. 5.11 for a given detector g, then $\frac{1}{n}\sum_{i=1}^{n} \xi^{i*}$ is an upper bound on the empirical risks $\mathcal{R}^{\Delta,\mu}_{max}(g)$ and $\mathcal{R}^{\Delta,\mu}_{ave}(g)$.

Proof: Consider Constraint (5.12) with $\mathbf{z} = g(\mathbf{X}^i_{[1,t]})$ and together with the fact that $f(\mathbf{X}^i_{g(\mathbf{X}^i_{[1,t]})}) \geq f(\mathbf{X}^i_{\mathbf{z}^i_t})$, we have

$$\xi^{i*} \geq \Delta(\mathbf{z}^i_t, g(\mathbf{X}^i_{[1,t]}))\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right) \ \forall t. \tag{5.19}$$

Thus

$$\xi^{i*} \geq \max_t\{\Delta(\mathbf{z}^i_t, g(\mathbf{X}^i_{[1,t]}))\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)\}. \tag{5.20}$$

Hence

$$\frac{1}{n}\sum_{i=1}^{n}\xi^{i*} \geq \mathcal{R}^{\Delta,\mu}_{max}(g) \geq \mathcal{R}^{\Delta,\mu}_{ave}(g). \tag{5.21}$$

This completes the proof of the proposition.

5.2.3 Discussion – slack variable rescaling versus margin rescaling

This section describes an alternative formulation to Eq. 5.11 and then discusses the advantages of of using Eq. 5.11.

Recall in Eq. 5.11, we use $\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)$ to rescale the slack variable ξ^i to weight the importance of rightly detecting the partial event at time t. An alternative approach is the rescale the margin $\Delta(\mathbf{z}^i_t, \mathbf{z})$, which leads to the following formulation:

$$\underset{\mathbf{w},b,\{\xi^i\}}{\text{minimize}} \ \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi^i, \tag{5.22}$$

$$\text{s.t. } f(\mathbf{X}^i_{\mathbf{z}^i_t}) \geq f(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}^i_t, \mathbf{z})\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right) - \xi^i$$

$$\forall i, \forall t = 1, \cdots, l_i, \forall \mathbf{z} \in \mathcal{I}, \mathbf{z} \subset [1,t], \tag{5.23}$$

$$\xi^i \geq 0 \ \forall i. \tag{5.24}$$

It is possible to use the above formulation for early event detection. However, this formulation has a disadvantage compared with the formulation proposed in Eq 5.11. To see this disadvantage, consider the difference between these two formulations, which lies at their constraints, Constraint (5.12) versus Constraint (5.23). Consider these two constraints for a particular time series $\mathbf{X}^i$ and at a particular time t. Both constraints adjust the original constraint, $f(\mathbf{X}^i_{\mathbf{z}^i_t}) \geq f(\mathbf{X}^i_{\mathbf{z}}) + \Delta(\mathbf{z}^i_t, \mathbf{z})$, based on the importance for recognizing the partial event at time t. The former reweigh the original constraint, while the latter reweigh the margin. In reality, not every event can be detected as soon as a small fraction of the event occurs; therefore, it is important to reweigh the constraint and even to deactivate it. This can be achieved using the former constraint, but not the latter. For example, the former allows us to deactivate the constraint by setting the scaling factor $\mu\left(\frac{|\mathbf{z}^i_t|}{|\mathbf{z}^i|}\right)$ to 0, while the latter does not.

5.3 Experiments

This section describes our experiments on several publicly available datasets of varying complexity.

5.3.1 Evaluation criteria

This section describes the criteria for evaluating the accuracy and timeliness of detectors. We use the area under the ROC curve for accuracy comparison, $F1$-score for evaluating localization quality, and Normalized Time to Detection (NTtoD) for benchmarking the timeliness of detection.

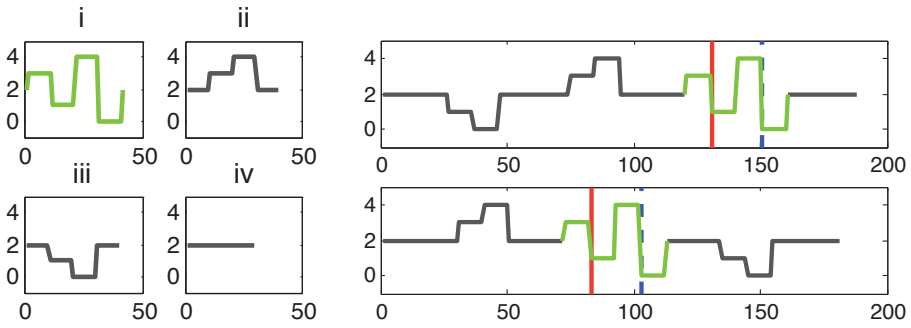
ROC area: Consider testing a detector on a set of time series. The False Positive Rate (FPR) of the detector is defined as the fraction of time series that the detector fires before the event of interest starts. The True Positive Rate (TPR) is defined as the fraction of time series that the detector fires during the event of interest. A detector typically has a detection threshold that can be adjusted to trade off high TPR for low FPR and vise versa. By varying this detection threshold, we can generate the ROC curve which is the function of TPR against FPR. We use the area under the ROC for evaluating the detector accuracy.

AMOC curve: To evaluate the timeliness of detection we use Normalized Time to Detection (NTtoD) which is defined as follows. Given a testing time series with the event of interest occurs from s to e. Suppose the detector starts to fire at time t. For a successful detection, $s \leq t \leq e$, we define the NTtoD as the fraction of event that has occurred, i.e., $\frac{t-s+1}{e-s+1}$. NTtoD is defined as 0 for a false detection ($t < s$) and ∞ for a false rejection ($t > e$). By adjusting the detection threshold, one can achieve smaller NTtoD at the cost of higher FPR and vice versa. For a complete characteristic picture, we vary the detection thresh hold and plot the curve of NToD versus FPR. This is referred as the Activity Monitoring Operating Curve (AMOC) [Fawcett and Provost, 1999].

F1-score curve: The ROC and AMOC curves, however, do not provide a measure for how well the detector can localize the event of interest. For this purpose, we propose to use the $F1$-scores. Consider running a detector on a times series. At time t the detector output the segment $\mathbf{z}$ (empty segment for no detection) while the ground truth (possibly) truncated event is $\mathbf{z}^*$. The $F1$-score is defined as the harmonic mean of precision and recall values: $F1 := 2.\frac{Precision.Recall}{Precision+Recall}$, with $Precision := \frac{|\mathbf{z}\cap\mathbf{z}^*|}{|\mathbf{z}|}$ and $Recall := \frac{|\mathbf{z}\cap\mathbf{z}^*|}{|\mathbf{z}^*|}$. For a new test time series, we can simulate the sequential arrival of data and record the $F1$-scores as the event of interest unroll from 0% to 100%. We refer to this as the *F1-score curve*.

5.3.2 Synthetic data

We first validated the performance of MMED on a synthetically generated dataset of 200 time series, each contained one instance of the event of interest, signal 5.5(a).i, and several instances of other events, signals 5.5(a).ii–iv. Some examples of these time series are shown in Fig. 5.5(b). We randomly split the data into training and testing subsets of equal sizes. During testing we simulated the sequential arrival of data and recorded the moment that MMED started to detect the start of the event of interest. With 100% precision, MMED detected the event when it had completed 27.5% of the event. For comparison, SOSVM required observing 77.5% of the event for a positive detection. Examples of testing time series and results are depicted in Fig. 5.5(b). The events of interest are drawn in green and the solid vertical red



(a) (b)

FIGURE 5.5: Synthetic data experiment. (a): time series were created by concatenating the event of interest (i) and several instances of other events (ii)–(iv). (b): examples of testing time series; the solid vertical red lines mark the moments that our method starts to detect the happening of the event of interest while the dash blue lines are the results of SOSVM. This figure is best seen in color.

lines mark the moments that our method started to detect the happening of these events. The dash vertical blue lines are the results of SOSVM. Notably, this result reveals an interesting capability of MMED. For the time series in this experiment, the change in signal values from 3 to 1 is exclusive to the target events. MMED was trained to recognize partial events, it implicitly discovered this unique behavior, and it detected the target events as soon as this behavior occurred. In this experiment, we represented each time series segment by the L_2-normalized histogram of signal values in the segment (normalized to have unit norm). We used linear SVM with $C = 1000, \alpha = 0, \beta = 1$.

5.3.3 Auslan dataset – Australian sign language

This section describes our experiments on a publicly available dataset [Kadous, 2002] that contains 95 Auslan signs, each with 27 examples. The signs were captured from a native signer using position trackers and instrumented gloves; the location of two hands, the orientation of the palms, and the bending of the fingers were recorded. We considered detecting the sentence "I love you" in monologues obtained by concatenating multiple signs. In particular, each monologue contained an I-love-you sentence which was preceded and succeeded by 15 random signs. The

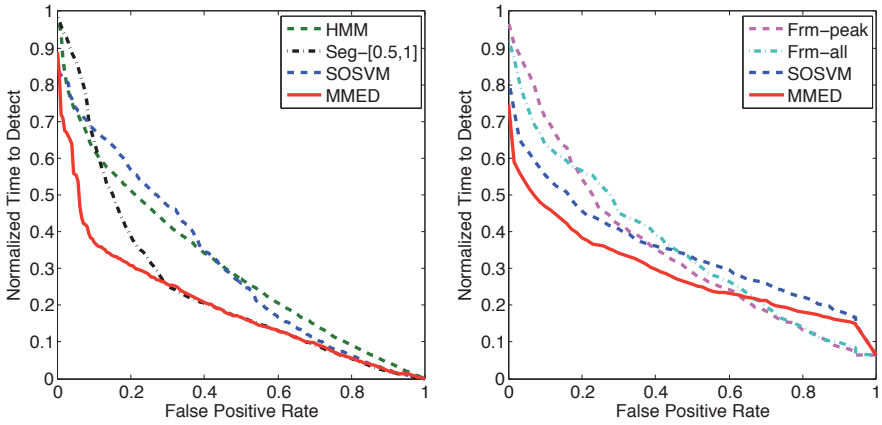
I-love-you sentence was ordered concatenation of random samples of three signs: "I", "love", and "you". We created 100 training and 200 testing monologues from disjoint sets of sign samples; the first 15 examples of each sign were used to create training monologues while the last 12 examples were used for testing monologues. The average lengths and standard deviations of the monologues and the I-love-you sentences were 1836 ± 38 and 158 ± 6 respectively.

Previous work [Kadous, 2002] reported high recognition performance on this dataset using Hidden Markov Models (HMMs) [Rabiner, 1989]. Following their success, we implemented a continuous density HMM for I-love-you sentences. Our HMM implementation consisted of 10 states, each was a mixture of 4 Gaussians. To use the HMM for detection, we adopted a sliding window approach; the window size was fixed to the average length of the I-love-you sentences.

Inspired by the high recognition rate of HMM, we constructed feature representation for SVM-based detectors (SOSVM and MMED) as follows. We first trained a Gaussian Mixture Model of 20 Gaussians for the frames extracted from the I-love-you sentences. Each frame was then associated with a 20×1 log-likelihood vector. We retained the top three values of this vector, zeroing out the other values, to create a frame-level feature representation. This is the soft quantization approach. To compute the feature vector for a given window, we divided the window into two roughly equal halves, the mean feature vector of each half was then calculated, and the concatenation of these mean vectors was then used as the feature representation of the window.

A naive strategy for early detection is to use truncated events as positive examples. For comparison, we implemented *Seg-[0.5,1]*, a binary SVM that used the first halves of the I-love-you sentences in addition to the full sentences as positive training examples. Negative training examples were random segments that had no overlapping with the I-love-you sentences.

We repeated our experiment 10 times and recorded the average performance. Regarding the detection accuracy, all methods except SVM-[0.5,1] performed similarly well. The ROC areas for HMM, SVM-[0.5,1], SOSVM, and MMED were 0.97, 0.92, 0.99, and 0.99, respectively. However, when comparing the timeliness of detection,



(a) Auslan, AMOC (b) CK+, AMOC

FIGURE 5.6: AMOC curves on Auslan and CK+ datasets; at the same false positive rate, MMED detects target events sooner than the other methods. This figure is best seen in color.

MMED outperformed the others by a large margin. For example, at 10% false positive rate, our method detected the I-love-you sentence when it observed the first 37% of the sentence. At the same false positive rate, the best alternative method required seeing 62% of the sentence. The full AMOC curves are depicted in Fig. 5.6(a). In this experiment, we used linear SVM with $C = 1, \alpha = 0.25, \beta = 1$.

5.3.4 Extended Cohn-Kanade dataset – facial expression

The Extended Cohn-Kanade dataset (CK+) [Lucey et al., 2010] contains 327 facial image sequences from 123 subjects performing one of seven discrete emotions: anger, contempt, disgust, fear, happy, sadness, and surprise. Each of the sequences contains images from onset (neutral frame) to peak expression (last frame). We considered the task of detecting negative emotions: anger, disgust, fear, and sadness.

We used the same representation as Lucey et al. [2010], where each frame uses the canonical normalized appearance feature, referred as CAPP in Lucey et al. [2010]. For comparison purposes, we implemented two frame-based SVMs: *Frm-peak* was trained on peak frames of the training sequences while *Frm-all* was trained using all frames between the onset and offset of the facial action. Frame-based SVMs can

be used for detection by classifying individual frames. In contrast, SOSVM and MMED are segment-based. Since a facial expression is a deviation of the neutral face, we represented each segment of an emotion sequence by the difference between the end frame and the start frame. Even though the start frame was not necessary a neutral face, this representation led to good recognition results.

We randomly divided the data into disjoint training and testing subsets. The training set contained 200 sequences with equal numbers of positive and negative examples. For reliable results, we repeated our experiment 20 times and recorded the average performance. Regarding the detection accuracy, segment-based SVMs outperformed frame-based SVMs. The ROC areas (mean and standard deviation) for Frm-peak, Frm-all, SOSVM, MMED are 0.82 ± 0.02, 0.84 ± 0.03, 0.96 ± 0.01, and 0.97 ± 0.01, respectively. Comparing the timeliness of detection, our method was significantly better than the others, especially at low false positive rate which is what we care about. For example, at 10% false positive rate, Frm-peak, Frm-all, SOSVM, and MMED can detect the expression when it completes 71%, 64%, 55%, and 47% respectively. Fig. 5.6(b) plots the AMOC curves, and Fig. 5.7 displays some qualitative results. In this experiment, we used a linear SVM with $C = 1000, \alpha = 0, \beta = 0.5$.

5.3.5 Weizmann dataset – human action

As described in Section 4.4.2, the Weizmann dataset contains 90 video sequences of 9 people, each performing 10 actions. Each video sequence in this dataset only consists of a single action. To measure the accuracy and timeliness of detection, we performed experiments on longer video sequences which were created by concatenating existing single-action sequences. We computed frame-level features and built a codebook of 100 temporal words as explained in Section 4.4.2.

Each action class took turn to be the subject of detection. We created 9 long video sequences, each composed of 10 videos of the same person and had the event of interest at the end of the sequence. We performed leave-one-out cross validation; each cross validation fold trained the event detector on 8 sequences and tested it on the leave-out sequence. For the testing sequence, we computed the normalized

FIGURE 5.7: Disgust (a) and fear (b) detection on CK+ dataset. From left to right of each sequence are the onset frame, the frame at which MMED fires, the frame at which SOSM fires, and the peak frame. The number in each image is the corresponding NTtoD.

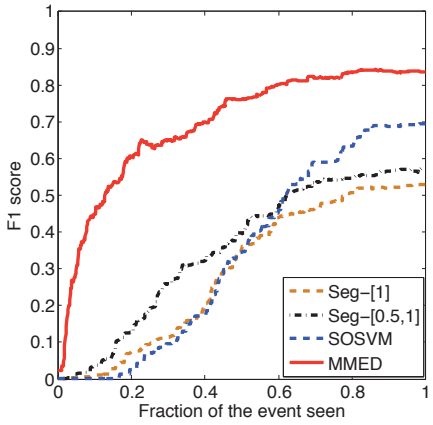


FIGURE 5.8: F1-score curves on Weizmann dataset; MMED provides better localization for the event of interest, especially when the fraction of the event observed is small. This figure is best seen in color.

time to detection at 0% false positive rate. This false positive rate was achieved by raising the threshold for detection so that the detector would not fire before the event started. We calculated the median normalized time to detection across 9 cross validation folds and averaged these median values across 10 action classes; the resulting values for Seg-[1], Seg-[0.5,1], SOSVM, MMED are 0.16, 0.23, 0.16, and 0.10

respectively. Here Seg-[1] was a segment-based SVM, trained to classify the segments corresponding to the complete action of interest. Seg-[0.5,1] was similar to Seg-[1], but used the first halves of the action of interest as additional positive examples. For each testing sequence, we also generated a F1-score curve as described in Sec. 5.3.1. Fig. 5.8 displays the $F1$-score curves of all methods, averaged across different actions and different cross-validation folds. MMED significantly outperformed the other methods. The superiority of MMED over SOSVM was especially large when the fraction of the event observed so far was small. This was because MMED was trained to detect truncated events while SOSVM was not. Though also trained with truncated events, Seg-[0.5,1] performed relatively poor because it was not optimized to produce correct temporal extend of the event. In this experiment, we used the linear SVM with $C = 1000, \alpha = 0, \beta = 1$.

5.4 Summary

This chapter addressed early event detection, a relatively unexplored problem in the computer vision literature. We used Seg-SVMs to develop MMED, a temporal classifier specialized in detecting events as soon as possible. Moreover, MMED provides localization for the temporal extent of the event in case it has begun. MMED is based on SOSVM, but extends it to anticipate streaming data. During training, we simulate the sequential arrival of data and train a detector to recognize incomplete events. It is important to emphasize that we train a *single* event detector to recognize *all* partial events. This contrasts to an approach that trains multiple detectors, which lacks a principled mechanism for integrating multiple detection outputs. Our method is particularly suitable for events which cannot be reliably detected by classifying individual frames; detecting this type of events requires pooling information from a supporting window. Experiments on datasets of varying complexity, from synthetic data, sign language to facial expression and human action, showed that our method often made faster detection while maintaining comparable or even better accuracy. Furthermore, our method provided better localization for the target

event, especially when the fraction of the event seen so far was small. In this chapter, we illustrated the benefits of our approach in the context of human behavior analysis, but our approach can be applied to many other domains.

Chapter 6

Weakly Supervised Learning for Discriminative Event Detection

"God grant me the serenity to accept the things I cannot change;
courage to change the things I can;
and wisdom to know the difference."
– Reinhold Niebuhr

So far, our event detectors are trained on a large collection of examples manually annotated with the temporal locations of target events. The reliance on time-consuming human labeling effectively limits the application of this approach to problems involving few event categories. Furthermore, the human selection of the event locations introduces arbitrary biases (e.g., in terms of event boundaries) which may be suboptimal for training the detector.

In this chapter, we use Seg-SVMs to develop a novel method for learning a discriminative event detector from examples annotated with binary labels indicating the presence of target events, but *not* their temporal locations. During training, our method simultaneously localizes the most discriminative set of temporal segments and learn an SVM to detect them, as illustrated in Fig. 6.1. We apply our method to video and accelerometer data and discover discriminative patterns. We extend

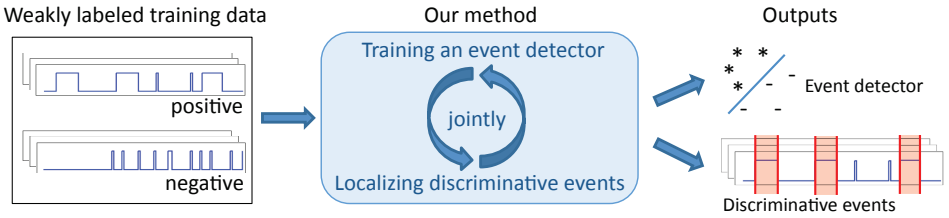


FIGURE 6.1: A framework for simultaneous localization of discriminative events and training a detector to detect them.

our method to the spatial domain to discover objects that discriminate between two image classes. We use the results of discriminative detection for classification and achieve quantitative results similar and in many cases superior to those obtained with full supervision.

6.1 Energy-based discriminative detection

This section describes an energy-based model for discriminative detection. Assume there are two classes of time series, called positive and negative (we will discuss the extension to the multi-class case in Section 6.3). Assume there is a class of events that are unique to the positive class; each positive time series contains one or more such event while negative time series contain no such event. Our goal is to discover these discriminative events. We propose an energy-based model to discover these events. Since there is one class of target events, we use $E(\mathbf{X}_\mathbf{z})$ instead of $E(\mathbf{X}_\mathbf{z}, y)$ to denote the energy of a segment, as in Chapter 3 and Chapter 5. We will return to discuss how to learn the energy function $E(\cdot)$ and a detection threshold b, but assume for now that the energy function and this threshold have been learned. The energy function is used to detect a discriminative event in an unseen time series $\mathbf{X}$ as follows. First, we find the segment that has the minimum energy:

$$\mathbf{z}^* = \underset{\mathbf{z} \in \mathcal{Z}}{\operatorname{argmin}} \, E(\mathbf{X}_\mathbf{z}). \tag{6.1}$$

If $E(\mathbf{X}_{\mathbf{z}^*}) < b$, we report $\mathbf{z}^*$ as the discriminative event and classify $\mathbf{X}$ as positive. Otherwise, we declare no detection and classify $\mathbf{X}$ as negative.

In the above, we assume the set of discriminative events are unique to the positive class. For time series data, however, this clear-cut separation between positive and negative classes does not always exist. In many cases, some negative time series also contain some instances of such events, and what separate the positive class from the negative class is the number of event occurrences. Thus, we propose to make a weaker assumption and address a more general problem. The assumption is each positive time series contains at least $\bar{k}$ events while each negative time series contains fewer than $\bar{k}$ events. $\bar{k}$ is an application-specific parameter. For $\bar{k} = 1$, we get back the problem of clear separation between positive and negative classes. Our goal is to localize this set of events in a time series and also use it for classification. Let $\mathcal{LS}(\mathbf{X})$ be the set of all legitimate segmentations of $\mathbf{X}$ with $\bar{k}$ or fewer segments; we refer to such a segmentation as a $\bar{k}$-segmentation. We propose to use an energy-based model to achieve this goal as follows. Given an unseen testing time series $\mathbf{X}$, we first find the $\bar{k}$-segmentation (i.e., $\bar{k}$ or fewer segments) of $\mathbf{X}$ that minimizes the total sum of energies.

$$\{\mathbf{z}_t^*\} := \underset{\{\mathbf{z}_t\} \in \mathcal{LS}(\mathbf{X})}{\operatorname{argmin}} \sum_t E(\mathbf{X}_{\mathbf{z}_t}). \tag{6.2}$$

If the total sum of energies is smaller than the threshold, i.e., $\sum E(\mathbf{X}_{\mathbf{z}_t^*}) < b$, we report $\{\mathbf{z}_t^*\}$ as the set of events that discriminate the positive class from the negative class, and we classify $\mathbf{X}$ as a positive time series. If the total sum of energies is not smaller than the threshold, we classify $\mathbf{X}$ as a negative time series.

The energy of a segment, $E(\mathbf{X}_{\mathbf{z}})$, is taken as $-\mathbf{w}^T\varphi(\mathbf{X}_{\mathbf{z}})$.

6.2 Maximum-margin learning for discriminative detection

6.2.1 The learning objective

Given a set of positive training time series $\{\mathbf{X}^{+i}|i = 1, \cdots, n^+\}$ and a set of negative training time series $\{\mathbf{X}^{-i}|i = 1, \cdots, n^-\}$, we learn an SVM for joint detection and

classification by solving the following constrained optimization:

$$\underset{\mathbf{w},b}{\text{minimize}}\ \frac{1}{2}||\mathbf{w}||^2, \tag{6.3}$$

$$\text{s.t.}\quad \min_{\{\mathbf{z}_t\}\in\mathcal{LS}(\mathbf{X}^{+i})} \sum_t E(\mathbf{X}^{+i}_{\mathbf{z}_t}) \leq b-1\ \forall i \in \{1,\cdots,n^+\}, \tag{6.4}$$

$$\min_{\{\mathbf{z}_t\}\in\mathcal{LS}(\mathbf{X}^{-i})} \sum_t E(\mathbf{X}^{-i}_{\mathbf{z}_t}) \geq b+1\ \forall i \in \{1,\cdots,n^-\}. \tag{6.5}$$

The constraints appearing in this objective reflect how we use the energy function for detecting discriminative events, which was described in the previous section. These constraints state that each positive time series must contain at least one set of $\bar{k}$-or-fewer intervals classified as positive, and that *all* sets of $\bar{k}$-or-fewer intervals in each negative time series must be classified as negative. The goal is then to maximize the margin subject to these constraints. By optimizing this problem we obtain the parameters $\mathbf{w}$ of the energy function and the threshold b.

For a $\bar{k}$-segmentation, $\mathbf{p}$, $\mathbf{p}=\{\mathbf{z}_t\}\in\mathcal{LS}(\mathbf{X})$, let $\phi(\mathbf{X},\mathbf{p})$ denote $\sum_t \varphi(\mathbf{X}_{\mathbf{z}_t})$. Use this compact notation and recall that $E(\mathbf{X}_{\mathbf{z}_t}) = -\mathbf{w}^T\varphi(\mathbf{X}_{\mathbf{z}_t})$, the learning formulation in Eq. 6.3 is equivalent to:

$$\underset{\mathbf{w},b}{\text{minimize}}\ \frac{1}{2}||\mathbf{w}||^2, \tag{6.6}$$

$$\text{s.t.}\quad \max_{\mathbf{p}\in\mathcal{LS}(\mathbf{X}^{+i})} \mathbf{w}^T\phi(\mathbf{X}^{+i},\mathbf{p}) + b \geq 1\ \forall i \in \{1,\cdots,n^+\}, \tag{6.7}$$

$$\max_{\mathbf{p}\in\mathcal{LS}(\mathbf{X}^{-i})} \mathbf{w}^T\phi(\mathbf{X}^{-i},\mathbf{p}) + b \leq -1\ \forall i \in \{1,\cdots,n^-\}. \tag{6.8}$$

As in the traditional formulation of SVM, the constraints are allowed to be violated by introducing slack variables:

$$\begin{aligned}
\underset{\mathbf{w},b,\{\xi^{+i}\},\{\xi^{-i}\}}{\text{minimize}} \ & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n^+}\xi^{+i} + C\sum_{i=1}^{n^-}\xi^{-i}, \qquad (6.9)\\
\text{s.t.} \ & \max_{\mathbf{p}\in\mathcal{LS}(\mathbf{X}^{+i})} \mathbf{w}^T\phi(\mathbf{X}^{+i},\mathbf{p}) + b \geq 1 - \xi^{+i} \ \forall i \in \{1,\cdots,n^+\},\\
& \max_{\mathbf{p}\in\mathcal{LS}(\mathbf{X}^{-i})} \mathbf{w}^T\phi(\mathbf{X}^{-i},\mathbf{p}) + b \leq -1 + \xi^{-i} \ \ \forall i \in \{1,\cdots,n^-\},\\
& \xi^{+i} \geq 0 \ \forall i \in \{1,\cdots,n^+\},\\
& \xi^{-i} \geq 0 \ \forall i \in \{1,\cdots,n^-\}.
\end{aligned}$$

Here, C is the parameter controlling the trade-off between having a large margin and less constraint violation. This formulation is a particular instance of multiple instance learning [Andrews et al., 2003, Dietterich et al., 1997].

6.2.2 Optimization

Our objective is non-convex. We propose optimization via a coordinate descent approach that alternates between optimizing the objective w.r.t. parameters $(\mathbf{w}, b, \{\xi^{+i}\}, \{\xi^{-i}\})$ and finding the set of $\bar{k}$ or fewer intervals of positive time series $\{\mathbf{X}^{+i}\}$ that maximize the SVM scores. Let $obj(\mathbf{w}, b, \{\xi^{+i}\}, \{\xi^{-i}\})$ denote $\frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi^{+i} + C\sum_i \xi^{-i}$, the optimization objective. The optimization procedure for $obj(\mathbf{w}, b, \{\xi^{+i}\}, \{\xi^{-i}\})$ is provided in Algorithm 1 below.

The iterative process of the above algorithm is a special case of Concave-Convex procedure (CCCP) [Smola et al., 2005, Yuille and Rangarajan, 2002]. CCCP has been proved theoretically to converge to a critical point. It has been shown empirically to be an effective and efficient optimization procedure in the context of maximum margin clustering [Zhao et al., 2008] and structural SVMs with latent variables [Yu and Joachims, 2009].

Every iteration of Algorithm 1 requires optimizing the objective w.r.t. parameters $\mathbf{w}, b$, $\{\xi^{+i}\}$, $\{\xi^{-i}\}$ while fixing the candidate set of discriminative events of positive time series $\{\mathbf{X}^{+i}\}$ (Eq. 6.10). Although this is a convex optimization problem,

Algorithm 1 The optimization procedure for (6.9)

1: Initialize sets of discriminative events for positive time series $\{\hat{\mathbf{p}}^{+i}\}_{i=1}^{n^+}$.
2: $obj := +\infty$
3: **repeat**
4: $cur\_obj := obj$
5: Optimize for SVM parameters:

$$\hat{\mathbf{w}}, \hat{b}, \hat{\xi}^{+i}, \hat{\xi}^{-i} := \underset{\mathbf{w}, b, \{\xi^{+i}\}, \{\xi^{-i}\}}{\operatorname{argmin}} obj(\mathbf{w}, b, \{\xi^{+i}\}, \{\xi^{-i}\}) \tag{6.10}$$

$$\text{s.t. } \mathbf{w}^T \phi(\mathbf{X}^{+i}, \hat{\mathbf{p}}^{+i}) + b \geq 1 - \xi^{+i} \ \forall i,$$

$$\max_{\mathbf{p} \in \mathcal{LS}(\mathbf{X}^{-i})} \mathbf{w}^T \phi(\mathbf{X}^{-i}, \mathbf{p}) + b \leq -1 + \xi^{-i} \ \forall i, \tag{6.11}$$

$$\xi^{+i} \geq 0, \xi^{-i} \geq 0 \ \forall i.$$

6: Update the objective:

$$obj := obj(\hat{\mathbf{w}}, \hat{b}, \hat{\xi}^{+i}, \hat{\xi}^{-i}).$$

7: Find the set of $\bar{k}$ or fewer intervals of positive time series that maximize SVM scores:

$$\hat{\mathbf{p}}^{+i} := \underset{\mathbf{p} \in \mathcal{LS}(\mathbf{X}^{+i})}{\operatorname{argmax}} \mathbf{w}^T \phi(\mathbf{X}^{+i}, \mathbf{p}) \ \forall i. \tag{6.12}$$

8: **until** $cur\_obj - obj < \epsilon$ *//convergence*

the cardinality of the set of $\bar{k}$ or fewer intervals is very large. Therefore, special treatment is required for constraints (6.11). We use *constraint generation* (i.e., the cutting plane algorithm) to handle these constraints [Tsochantaridis et al., 2005]. Algorithm 2 outlines this optimization procedure.

Each iteration of Algorithm 2 minimizes a convex quadratic function subject to manageable-size sets of linear constraints $\mathcal{P}^{-i}$ (Line 3). These sets of constraints are updated by adding the most violated constraints at every step (Line 7). The algorithm terminates when the total constraint violation is smaller than a threshold (as also used by Zhao et al. [2008]). Algorithm 2 is guaranteed to find the global optimum of (6.10). Like the Simplex algorithm, constraint generation has exponential running time in the worst case; however, it often works well in practice.

Algorithm 2 The optimization procedure for (6.10)

1: $\mathcal{P}^{-i} := \emptyset \ \forall i$.
2: **repeat**
3: Optimize the quadratic program :

$$\begin{aligned}\hat{\mathbf{w}}, \hat{b}, \hat{\xi}^{+i}, \hat{\xi}^{-i} := \underset{\mathbf{w}, b, \{\xi^{+i}\}, \{\xi^{-i}\}}{\operatorname{argmin}} \ & obj(\mathbf{w}, b, \{\xi^{+i}\}, \{\xi^{-i}\}) \\ \text{s.t. } & \mathbf{w}^T \phi(\mathbf{X}^{+i}, \hat{\mathbf{p}}^{+i}) + b \geq 1 - \xi^{+i} \ \forall i, \\ & \mathbf{w}^T \phi(\mathbf{X}^{-i}, \mathbf{p}) + b \leq -1 + \xi^{-i} \ \forall i, \forall \mathbf{p} \in \mathcal{P}^{-i} \\ & \xi^{+i} \geq 0, \xi^{-i} \geq 0 \ \forall i.\end{aligned}$$

4: $tv := 0$. *//total violation*
5: **for all** $i \in \{1, \cdots, n^-\}$ **do**
6: Find the most violated constraints:

$$\hat{\mathbf{p}}^{-i} := \underset{\mathbf{p} \in \mathcal{LS}(\mathbf{X}^{-i})}{\operatorname{argmax}} \ \hat{\mathbf{w}}^T \phi(\mathbf{X}^{-i}, \mathbf{p}) \tag{6.13}$$

7: $\mathcal{P}^{-i} := \mathcal{P}^{-i} \cup \{\hat{\mathbf{p}}^{-i}\}$
8: $tv := tv + \min\{\hat{\mathbf{w}}^T \phi(\mathbf{X}^{-i}, \hat{\mathbf{p}}^{-i}) + \hat{b} - (-1 + \hat{\xi}^{-i}), 0\}$
9: **until** $tv < \delta$ *//total violation is negligible*

6.3 Multi-class extension

We now extend our formulation to handle multiple classes. Assume we are given a set of training time series $\{\mathbf{X}^i | i = 1, \cdots, n\}$ with corresponding class labels $\{y^i | i = 1, \cdots, n\}$. The label $y^i \in \{1, \cdots, m\}$ indicates that the time series $\mathbf{X}^i$ contains target events of category y^i. We learn an SVM for joint detection and classification by solving the following constrained optimization:

$$\begin{aligned}\underset{\{\mathbf{w}_j\}, \{\xi^i\}}{\text{minimize}} \ & \frac{1}{2m} \sum_{j=1}^{m} ||\mathbf{w}_j||^2 + C \sum_{i=1}^{n} \xi^i \\ \text{s.t. } & \max_{\mathbf{p} \in \mathcal{LS}(\mathbf{X}^i)} \mathbf{w}_{y^i}^T \phi(\mathbf{X}^i, \mathbf{p}) \geq \max_{\mathbf{p} \in \mathcal{LS}(\mathbf{X}^i)} \mathbf{w}_y^T \phi(\mathbf{X}^i, \mathbf{p}) + 1 - \xi^i \ \forall i \forall y \neq y^i, \\ & \xi^i \geq 0 \ \forall i.\end{aligned} \tag{6.14}$$

The constraints appearing in this objective state that for each time series $\mathbf{X}^i$, the detector of the correct class (y^i) should output a classification score higher than those produced by the detectors of the other classes. Here, $\{\xi^i\}$ are slack variables, and C is the parameter controlling the trade-off between having larger margin and less constraint violation. The goal is then to maximize the margin subject to these constrains. By optimizing this problem we obtain a multi-class SVM, i.e. parameters $(\mathbf{w}_1, \cdots, \mathbf{w}_m)$, that can be used for detection and categorization. Given a new testing time series $\mathbf{X}$, detection and categorization are done as follows. First, we find the category $\hat{y}$ and $\bar{k}$-segmentation $\hat{\mathbf{p}} \in \mathcal{LS}(\mathbf{X})$ yielding the maximum SVM score:

$$\hat{y}, \hat{\mathbf{p}} = \underset{y \in \mathcal{Y}, \mathbf{p} \in \mathcal{LS}(\mathbf{X})}{\operatorname{argmax}} \mathbf{w}_y^T \phi(\mathbf{X}, \mathbf{p}). \tag{6.15}$$

We report $\hat{\mathbf{p}}$ as the detected events of category $\hat{y}$ for time series $\mathbf{X}$.

6.4 Feature representation and localization algorithm

The above optimization requires at each iteration to localize the set of $\bar{k}$ or fewer intervals maximizing the SVM score in each time series (Eqs. 6.12 & 6.13). Thus, we need a very fast localization procedure. In the this section we describe a representation of temporal signals and a novel efficient algorithm to address this challenge.

6.4.1 Feature representation

Time series can be represented by descriptors computed at spatial-temporal interest points [Dollár et al., 2005, Laptev and Lindeberg, 2003, Niebles et al., 2008]. Sample descriptors from training data can be clustered to create a visual-temporal vocabulary [Dollár et al., 2005]. Subsequently, each descriptor is represented by the ID of the corresponding vocabulary entry and the frame number at which the point is detected. Given a segment $\mathbf{z}$ of a time series $\mathbf{X}$, we consider the feature vector $\varphi(\mathbf{X}_\mathbf{z})$ as the histogram of visual-temporal words associated with interest points

in $\mathbf{z}$. Thus, for a $\bar{k}$-segmentation $\mathbf{p} \in \mathcal{LS}(\mathbf{X})$, the feature vector $\phi(\mathbf{X}, \mathbf{p})$ is the histogram of visual-temporal words associated with interest points in $\mathbf{p}$.

Let C_i denote the set of words occurring at frame i. Let $a_i = \sum_{c \in C_i} w_c$ if C_i is non-empty, and $a_i = 0$ otherwise. a_i is the weighted sum of words occurring in frame i where word c is weighted by SVM weight w_c. From these definitions it follows that $\mathbf{w}^T \phi(\mathbf{X}, \mathbf{p}) = \sum_{i \in \mathbf{p}} a_i$. For fast localization of discriminative patterns in time series we need an algorithm to efficiently find the $\bar{k}$-segmentation maximizing the SVM score $\mathbf{w}^T \phi(\mathbf{X}, \mathbf{p})$. Indeed, this optimization can be solved globally in a very efficient way. The following section describes the algorithm. In the appendix, we prove the optimality of the solution produced by this algorithm.

6.4.2 An efficient localization algorithm

Let n be the length of the time signal and $\mathcal{I} = \{[s, e] : 1 \leq s \leq e \leq n\} \cup \{\emptyset\}$ be the set of all subintervals of $[1, n]$. For a subset $S \subseteq \{1, \cdots, n\}$, let $h(S) = \sum_{i \in S} a_i$. Maximization of $\mathbf{w}^T \phi(\mathbf{X}, \mathbf{p})$ is equivalent to:

$$\underset{\mathbf{z}_1, \ldots, \mathbf{z}_{\bar{k}} \in \mathcal{I}}{\text{maximize}} \sum_{j=1}^{\bar{k}} h(\mathbf{z}_j) \text{ s.t. } \mathbf{z}_i \cap \mathbf{z}_j = \emptyset \ \forall i \neq j. \tag{6.16}$$

This problem can be optimized very efficiently using Algorithm 3 presented below.

Algorithm 3 Find best $\bar{k}$ disjoint intervals that optimize (6.16)

Input: $a_1, \cdots, a_n$, $\bar{k} \geq 1$.
Output: a set $\mathcal{Z}^{\bar{k}}$ of best $\bar{k}$ disjoint intervals.

1: $\mathcal{Z}^0 := \emptyset$.
2: **for** $m = 0$ to $\bar{k} - 1$ **do**
3: $\quad J_1 := \arg\max_{J \in \mathcal{I}} h(J)$ s.t. $J \cap S = \emptyset \ \forall S \in \mathcal{Z}^m$.
4: $\quad J_2 := \arg\max_{J \in \mathcal{I}} -h(J)$ s.t. $J \subset S \in \mathcal{Z}^m$.
5: $\quad$ **if** $h(J_1) \geq -h(J_2)$ **then**
6: $\quad\quad \mathcal{Z}^{m+1} := \mathcal{Z}^m \cup \{J_1\}$
7: $\quad$ **else**
8: $\quad\quad$ Let $S \in \mathcal{Z}^m : J_2 \subset S$. S is divided into three disjoint intervals: $S = S^- \cup J_2 \cup S^+$.
9: $\quad\quad \mathcal{Z}^{m+1} := (\mathcal{Z}^m - \{S\}) \cup \{S^-, S^+\}$

This algorithm progressively finds the set of m intervals (possibly empty) that maximize (6.16) for $m = 1, \cdots, \bar{k}$. Given the optimal set of m intervals, the optimal set of $m+1$ intervals is obtained as follows. First, find the interval J_1 that has maximum score $h(J_1)$ among the intervals that do not overlap with any currently selected interval (line 3). Second, locate J_2, the worst subinterval of all currently selected intervals, i.e. the subinterval with lowest score $h(J_2)$ (line 4). Finally, the optimal set of $m+1$ intervals is constructed by executing either of the following two operations, depending on which one leads to the higher objective:

1. Add J_1 to the optimal set of m intervals (line 6);
2. Break the interval of which J_2 is a subinterval into three intervals and remove J_2 (line 9).

Algorithm 3 assumes J_1 and J_2 can be found efficiently. This is indeed the case. We now describe the procedure for finding J_1. The procedure for finding J_2 is similar.

Let $\overline{\mathcal{Z}^m}$ denote the relative complement of $\mathcal{Z}^m$ in $[1, n]$, i.e., $\overline{\mathcal{Z}^m}$ is the set of intervals such that the "union" of the intervals in $\mathcal{Z}^m$ and $\overline{\mathcal{Z}^m}$ is the interval $[1, n]$. Since $\mathcal{Z}^m$ has at most m elements, $\overline{\mathcal{Z}^m}$ has at most $m+1$ elements. Since J_1 does not intersect with any interval in $\mathcal{Z}^m$, it must be a subinterval of an interval of $\overline{\mathcal{Z}^m}$. Thus, we can find J_1 as $J_1 = \arg\max_{S \in \overline{\mathcal{Z}^m}} h(J^S)$ where:

$$J^S = \arg\max_{J \subseteq S} h(J). \tag{6.17}$$

Eq. (6.17) is a basic operation that is needed to be performed repeatedly: finding a subinterval of an interval that maximizes the sum of elements in that subinterval. This operation can be performed by Algorithm 4 below with running time complexity $\mathbf{O}(n)$.

Note that the result of executing (6.17) can be cached; we do not need to recompute J^S for many S at each iteration of Algorithm 3. Thus the total running complexity of Algorithm 3 is $\mathbf{O}(n\bar{k})$. Algorithm 3 guarantees to produce a globally optimal solution for (6.16) (see Appendix A).

Algorithm 4 Find the best subinterval

Input: $a_1, \cdots, a_n$, an interval $[l, u] \subset [1, n]$.
Output: $[sl, su] \subset [l, u]$ with maximum sum of elements.

```
b_0 := 0.
for m = 1 to n do
    b_m := b_{m-1} + a_m. //compute integral image
[sl, su] := [0, 0]; val := 0. //empty subinterval
m̂ := l − 1. //index for minimum element so far
for m = l to u do
    if b_m − b_m̂ > val then
        [sl, su] := [m̂ + 1; m]; val := b_m − b_m̂
    else if b_m < b_m̂ then
        m̂ := m. //keep track of the minimum element
```

6.5 Experiments

This section describes our experiments on several time series datasets.

6.5.1 A synthetic example

The data in this evaluation consists of 800 artificially generated examples of binary time series (400 positive and 400 negative). Some examples are shown in Fig. 6.2. Each positive example contains three long segments of fixed length with value 1. We refer to these as the foreground segments. Note that the end of a foreground segment may meet the beginning of another one, thus creating a longer foreground segment (see e.g. the bottom left signal of Fig. 6.2). The locations of the foreground segments are randomly distributed. Each negative example contains fewer than three foreground segments. Both positive and negative data are artificially degraded to simulate measurement noise: with a certain probability, zero energy values are flipped to have value 1. The temporal length of each signal is 100 and the length of each foreground segment is 10. We split the data into separate training and testing sets, each containing 400 examples (200 positive, 200 negative).

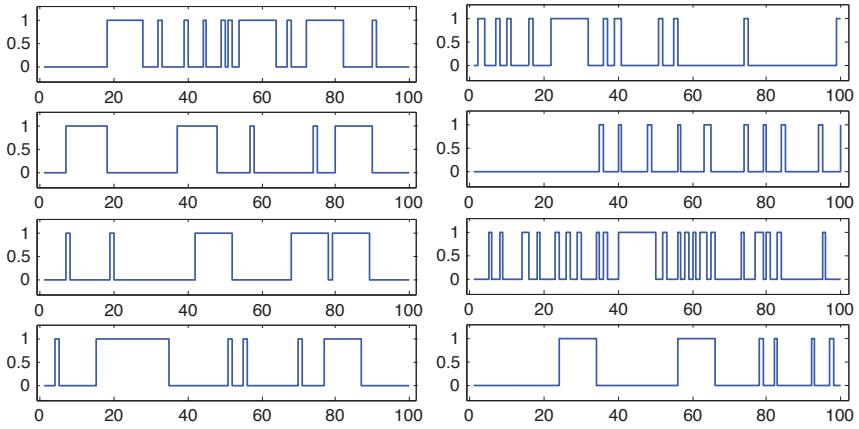


FIGURE 6.2: What distinguishes the time series on the left from the ones on the right? Left: positive examples, each containing three long segments with value 1 at random locations. Right: negative examples, each containing fewer than three long segments with value 1. All signals are perturbed with measurement noise corresponding to spikes with value 1 at random locations.

We evaluated the ability of our algorithm to discover automatically the discriminative segments in these weakly-labeled examples. We trained our localization-classification SVM by learning $\bar{k}$-segmentations for values of $\bar{k}$ ranging from 1 to 20. Note that the algorithm has no knowledge of the length or the type of the pattern distinguishing the two classes. Figure 6.3 summarizes the performance of our approach. *Glob-SVM*, traditional SVM based on the statistics of the whole signals, yields an accuracy rate of 66.5%. Our approach provides much better accuracy than Glob-SVM. Note that the performance of our method is relatively insensitive to the choice of $\bar{k}$, the number of discriminative time-intervals used for classification. It achieves 100% accuracy when the number of intervals are in the range 3 to 7; it works relatively well for other settings. When $\bar{k} = 1$, our method achieves the accuracy of only 77%; this reaffirms the need of using multiple intervals. When the value of $\bar{k}$ is too big, our algorithm essentially uses the statistics of the whole signals for classification, and it behaves like Glob-SVM. In practice, we can use cross validation to choose the appropriate number of segments.

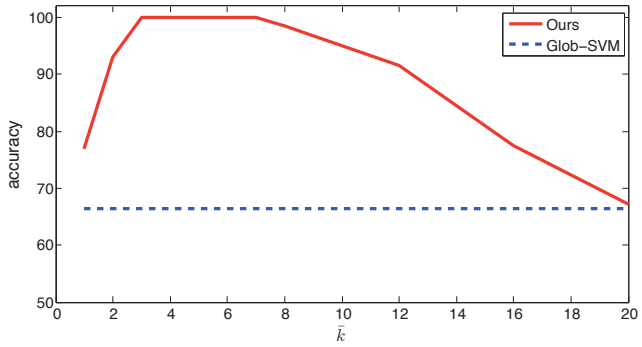


FIGURE 6.3: Classification performance on synthetic time series. For our method, we show the accuracy obtained using different values of $\bar{k}$, the maximum number of discriminative time intervals allowed. Here *Glob-SVM*, traditional SVM based on the global statistics of the signals, yields an accuracy rate of 66.5%, which does not depend on $\bar{k}$

6.5.2 Discriminative localization in human motion

For a qualitative evaluation, we collected some accelerometer readings of human walking activity. A 40Hz 3-axis accelerometer was attached to the left arm of a subject, and we collected a training set of 10 negative and 15 positive time series, respectively. The negative samples recorded the normal walking activity of the subject, while in each positive sample, the subject walked but fell twice during the course the activity. Each time series contains 2000 frames; at 40Hz, this corresponds to 50 seconds. Some examples of the time series in this dataset are shown in Fig. 6.4.

We obtained a temporal codebook of 20 clusters using k-means on frame-level accelerometer vectors. Subsequently, each frame was represented by the ID of the cluster that it belonged to. We trained our algorithm and localized $\bar{k}$-segmentations with values of $\bar{k}$ varying from 1 to 10. In Fig. 6.5, we show the qualitative results for discriminative localization in several time series that were not used in training. The proposed method correctly discovered the discriminative segments (falling events) for a wide range of $\bar{k}$ values.

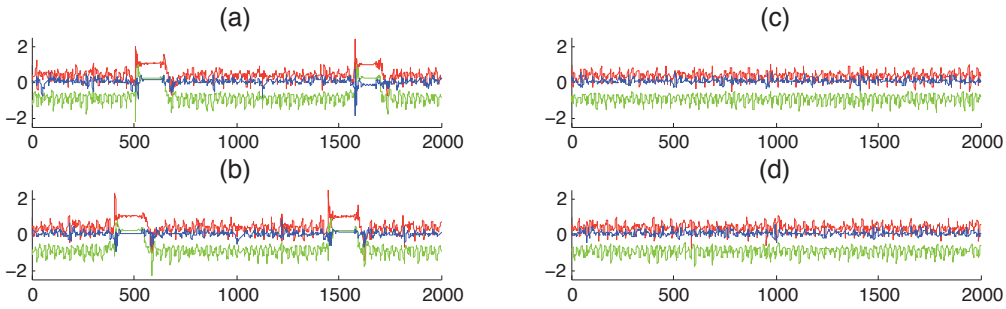


FIGURE 6.4: Examples of accelerometer readings of human activity. Red, green, blue correspond to three channels of a triaxial accelerometer. Negative samples (c, d) recorded normal walking activity while positive samples (a, b) included the falling events.

6.5.3 Mouse behavior

We now describe an experiment of mouse behavior recognition performed on a publicly available dataset[1]. This collection contains videos corresponding to five distinct mouse behaviors: drinking, eating, exploring, grooming, and sleeping. There are seven groups of videos, corresponding to seven distinct recording sessions. Because of the limited amount of data, performance is estimated using leave-one-group-out cross validation. This is the same evaluation methodology used by Dollár et al. [2005]. Fig. 6.6 shows some representative frames of the clips. Please refer to Dollár et al. [2005] for further details about this dataset.

We represented each video clip as a set of *cuboids* [Dollár et al., 2005] which were spatial-temporal local descriptors. From each video we extracted cuboids at interest points computed using the cuboid detector [Dollár et al., 2005]. To these descriptors we added cuboids computed at random locations in order to yield a total of 2500 points for each video (this augmentation of points was done to cancel out effects due to differing sequence lengths). A library of 50 cuboid prototypes was created by clustering cuboids sampled from training data using k-means. Subsequently, each cuboid was represented by the ID of the closest prototype and the frame number at which the cuboid was extracted. We trained our algorithm with values of $\bar{k}$ varying from 1 to 3. Here we report the performance obtained with the best setting for each class.

[1] http://vision.ucsd.edu/∼pdollar/research/research.html

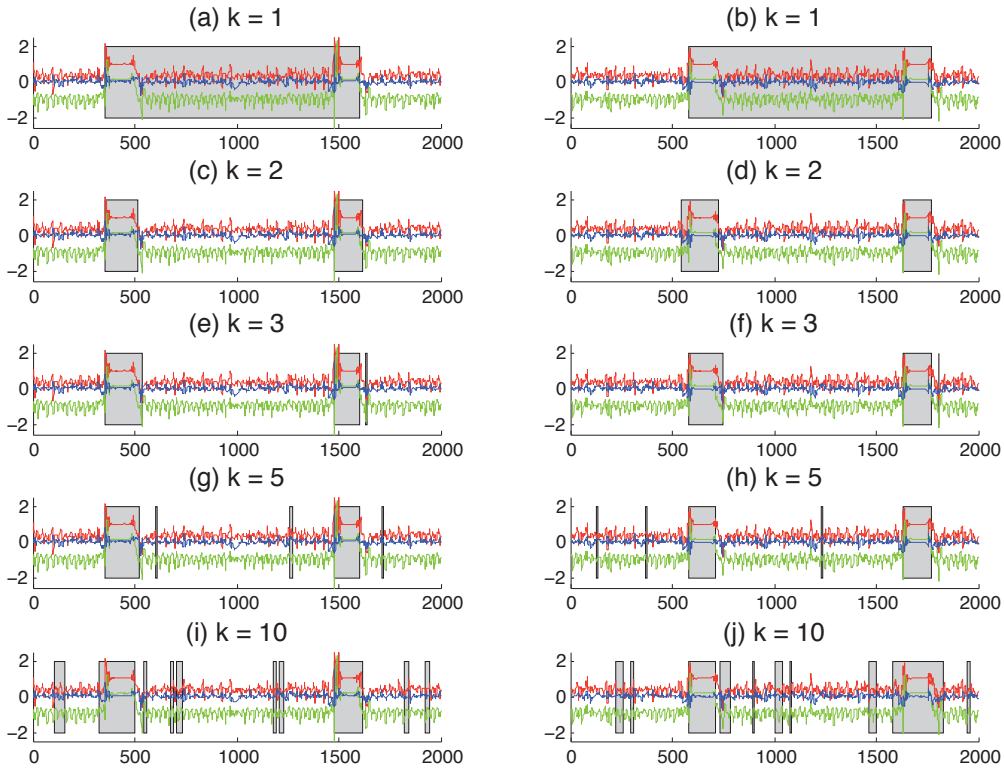


FIGURE 6.5: Discriminative localization in human motion analysis. This figure shows two examples of testing time series and the results for different values of $\bar{k}$, the number of segments in $\bar{k}$-segmentations. The left sub-figures (a, c, e, g, i) show the same time series, while the right subfigures (b, d, f, h, j) depict another time series. $\bar{k}$ is 1, 2, 3, 5, 10 for (a, b), (c, d), (e, f), (g, h), and (i, j) respectively. Our method successfully discovers the discriminative patterns (falling events) for a wide range of $\bar{k}$ values.

A performance comparison is shown in Table 6.1. The second column shows the result reported by Dollár et al. [2005] using a 1-nearest neighbor classifier on histograms containing only words computed at spatial-temporal interest points. *Glob-NN* is the result obtained with the same method applied to histograms including also random points. *Glob-SVM* is the traditional SVM approach in which each video is represented by the histogram of words over the entire clip. The performance is measured using the F1 score which is defined as:

$$F1 = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}. \tag{6.18}$$

Here we use this measure of performance instead of the ROC metric because the

FIGURE 6.6: Example frames from the mouse videos.

TABLE 6.1: F1 scores: detection performance of several algorithms. Higher F1 scores indicate better performance.

Action	Dollár et al. [2005]	Glob-NN	Glob-SVM	Ours
Drink	0.63	0.58	0.63	**0.67**
Eat	**0.92**	0.87	0.91	0.91
Explore	0.80	0.79	**0.85**	**0.85**
Groom	0.37	0.23	0.44	**0.54**
Sleep	0.88	0.95	**0.99**	**0.99**

latter is designed for binary classification rather than detection tasks [Agarwal et al., 2004]. Our method achieves the best $F1$ score on all but one action.

6.5.4 Multi-class categorization of cooking activity

This section explores the use of accelerometers for activity classification in the context of cooking and preparing recipes in an unstructured environment. We performed our experiments on the Carnegie Mellon University Multimodal Activity (CMU-MMAC) database [De la Torre et al., 2008]. This collection contains multimodal measures of human subjects performing tasks involved in cooking five different recipes: brownies, scrambled eggs, pizza, salad, and sandwich. Fig. 6.7a shows an example of the data collection process, a subject is cooking scrambled eggs in a fully operable kitchen. Although the database contains multimodal measures (video, audio, motion capture, bodymedia, RFID, eWatch, IMUs), we only used the accelerometer readings from the five wired Inertial Measurement Units (IMUs). These 125Hz accelerometers are triaxial and attached to the waist and the limbs of

(a) (b)

FIGURE 6.7: CMU-MMAC dataset. (a): data collection in action, a subject is cooking scrambled egg in a fully operable kitchen. (b): locations of five wired Inertial Measurement Units (IMUs); the accelerometer readings of these IMUs are used for experiments in Section 6.5.4

.

the subjects as shown in Fig. 6.7b. We used the main dataset[2] which contains data of 39 subjects. We arbitrarily divided the data into disjoint training and testing subsets: subjects with odd IDs were used for training and subjects with even IDs were reserved for testing. The training and testing subsets contained 89 and 80 samples respectively.

Previous work in the literature [Bao and Intille, 2004] has achieved high accuracy using acceleration data for classifying repetitive human activities such as walking, running, and bicycling. However, CMU-MMAC dataset is far more challenging because it was captured in an unstructured environment and the subjects were minimally instructed. As a consequent, how a recipe was cooked varied greatly from one subject to another. Moreover, the course of food preparation and recipe cooking contains a series of actions, and most of them are not repetitive. Many actions such as walking, opening the fridge, and turning on the oven are common

[2]http://kitchen.cs.cmu.edu/main.php

for most recipes. More discriminative actions such as opening a brownie bag or cracking an egg are often buried in a long chain of actions.

We adopted the feature representation proposed by Bao and Intille [2004]. In particular, we computed a feature vector every second. To compute the feature vector at a specific time, we obtained a surrounding window of 1000 frames; at 125Hz, this corresponds to 8 seconds. Mean, frequency-domain energy, frequency-domain entropy, and correlation features were extracted from this supporting window, as described in Bao and Intille [2004]. Every second of a time series was therefore associated with a feature vector of 150 dimensions. The attributes of these features vectors were scale-normalized to have maximum magnitude of 1. These normalized feature vectors were clustered using k-means to obtain a codebook of 50 temporal words. Subsequently, each second of the accelerometer data was represented by the ID of the closest temporal word. Because the amount of time to prepare and cook different recipes might differ, the histogram feature vector for a time series (either computed globally or on the foreground segments) was normalized by the length of the time series.

We implemented the multi-class categorization approach described in Section 6.3 combining with the multi-event localization method of Section 6.4. In our implementation, $\bar{k}$, the number of time-intervals of $\bar{k}$-segmentations, was set to 5. Table 6.2 displays the confusion matrix of this proposed method for categorizing five different recipes using accelerometer data. The mean accuracy is 52.2%. This is significantly higher than the mean accuracy of Glob-SVM which is 42.4%, as shown in Figure 6.8. The expected accuracy of a random classifier is 20%.

6.6 Extension to images

Our algorithm can be extended to the spatial domain, to discover image regions that discriminate between two classes of images. This can be achieved by using the exact learning formulation given in Eq. 6.9. However, $\mathbf{X}^{+i}$ and $\mathbf{X}^{-i}$ are images instead of time series, and $\mathcal{LS}(\mathbf{X})$ is the set of all subwindows of image $\mathbf{X}$. This section describes some experiments on object localization and image classification.

TABLE 6.2: Results on CMU-MMAC dataset: confusion matrix of the proposed method for five different recipes. The mean accuracy is 52.2%, compared with 42.4% from the traditional SVM. A random classifier would yield an expected accuracy of 20%.

	Brownie	Egg	Pizza	Salad	Sandwich
Brownie	68.8	6.2	6.2	0.0	18.8
Egg	25.0	31.2	12.5	12.5	18.8
Pizza	11.8	5.9	47.1	17.6	17.6
Salad	5.9	11.8	23.5	35.3	23.5
Sandwich	0.0	7.1	0.0	14.3	78.6

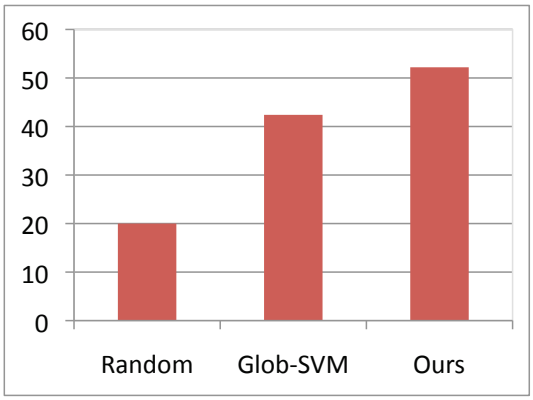


FIGURE 6.8: The mean accuracies on CMU-MMAC dataset – our method significantly outperforms Glob-SVM.

6.6.1 Experiments on car and face datasets

This subsection presents evaluations on two image collections. The first experiment was performed on CMU Face Images, a publicly available dataset from the UCI machine learning repository[3]. This database contains 624 face images of 20 people with different expressions and poses. The subjects wear sunglasses in roughly half

[3] http://archive.ics.uci.edu/ml/datasets/CMU+Face+Images

(a)

(b)

FIGURE 6.9: Examples taken from (a) the CMU Face Images and (b) the street scene dataset.

of the images. Our classification task was to distinguish between the faces with sunglasses and the faces without sunglasses. Some image examples from the database are given in Fig. 6.9(a). We divided this image collection into disjoint training and testing subsets. Images of the first 8 people were used for training while images of the last 12 people were reserved for testing. Altogether, we had 254 training images (126 with glasses and 128 without glasses) and 370 testing images (185 examples for both the positive and the negative class).

The second experiment was performed on a dataset collected by us. Our collection contains 400 images of street scenes. Half of the images contain cars and half of them do not. This is a challenging dataset because the appearance of the cars in the images varies in shape, size, grayscale intensity, and location. Furthermore, the cars occupy only a small portion of the images and may be partially occluded by other objects. Some examples of images from this dataset are shown in Fig. 6.9(b). Given the limited amount of examples available, we applied 4-fold cross validation to obtain an estimate of the performance.

Each image was represented by a set of 10,000 local SIFT descriptors [Lowe, 2004] selected at random locations and scales. The descriptors were quantized using a dictionary of 1,000 visual words obtained by applying hierarchical k-means [Nistér and Stewénius, 2006] to 100,000 training descriptors.

TABLE 6.3: Comparison results on the CMU Face and car datasets. *Glob-NN*: 10 nearest neighbor approach [Nistér and Stewénius, 2006]. *Glob-SVM*: SVM using global statistics. *Seg-SVM-FS* [Lampert et al., 2008] requires bounding boxes of foreground objects during training. Our method is significantly better than the others, and it outperforms even the algorithm using strongly labeled data.

Dataset	Measure	Glob-NN	Glob-SVM	Seg-SVM-FS	Ours
Faces	Acc. (%)	80.11	82.97	86.79	**90.0**
	ROC Area	n/a	0.90	0.94	**0.96**
Cars	Acc. (%)	77.5	80.75	81.44	**84.0**
	ROC Area	n/a	0.86	0.88	**0.90**

In order to speed up the learning, an upper constraint on the rectangle size was imposed. In the first experiment, as the image size is 120×128 and the sizes of sunglasses are relative small, we restricted the height and width of permissible rectangles to not exceed 30 and 50 pixels, respectively. Similarly, for the second experiment, we constrained permissible rectangles to have height and width no larger than 300 and 500 pixels, respectively (c.f. image size of 600×800).

We compared our approach to several competing methods. *Glob-SVM* denotes a traditional SVM approach in which each image is represented by the histogram of the words in the whole image. *Glob-NN* is the method of Nistér and Stewénius [2006] in the implementation of Vedaldi and Fulkerson [2008]. It uses a 10-nearest neighbor classifier. We also benchmarked our method against *Seg-SVM-FS* [Lampert et al., 2008], a fully supervised method requiring ground truth subwindows during training (Seg stands for segment and FS stands for fully supervised). *Seg-SVM-FS* trains an SVM using ground truth bounding boxes as positive examples and ten random rectangles from each negative image for negative data.

Table 6.3 shows the classification performance measured using both the accuracy rates and the areas under the ROCs. Note that our approach outperforms not only *Glob-SVM* and *Glob-NN* (which are based on global statistics), but also *Seg-SVM-FS*, which is a fully supervised method requiring the bounding boxes of the objects during training. This suggests that the boxes tightly enclosing the objects of interest are not always the most discriminative regions.

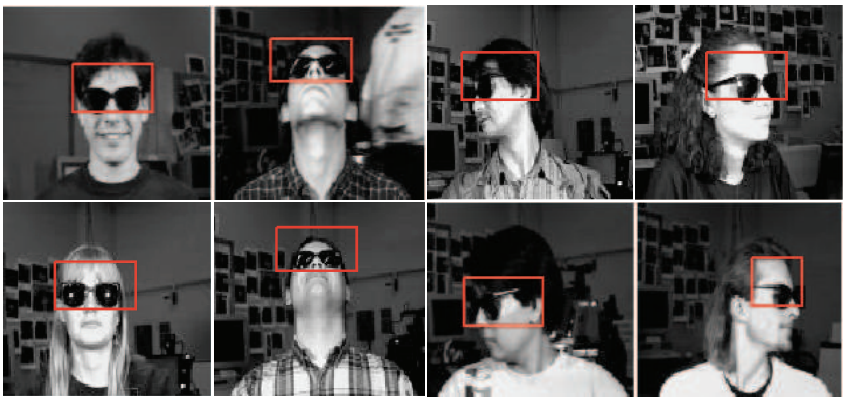


FIGURE 6.10: Localization of sunglasses on test images.

Our method automatically localizes the subwindows that are most discriminative for classification. Fig. 6.10 shows discriminative detection on a few face testing examples. Sunglasses are the distinguishing elements between positive and negative classes. Our algorithm successfully discovers such regions and exploits them to improve the classification performance. Fig. 6.11 shows some examples of car localization. Parts of the road below the cars tend to be included in the detection output. This suggests that the appearance of roads is a contextual indication for the presence of cars. Fig. 6.12 displays several difficult cases where our method does not provide good localization of the objects.

Glob-SVM, *Seg-SVM-FS*, and our proposed method require tuning of a single parameter, C, controlling the trade-off between a large margin and less constraint violation. This parameter was tuned using 4-fold cross validation on training data. The parameter sweeping was done exactly in the same fashion for all algorithms. Optimizing (6.9) was an iterative procedure, where each iteration involved solving a convex quadratic programming problem. Our implementation[4] used `CVX`, a package for specifying and solving convex programs Grant and Boyd [2008a,b]. We also used `Ilog Cplex`[5] for quadratic programming. We found that our algorithm generally converged within 100 iterations of coordinate descent.

[4]http://www.andrew.cmu.edu/user/minhhoan/downloads.html
[5]http://www-01.ibm.com/software/integration/optimization/ cplex-optimizer/

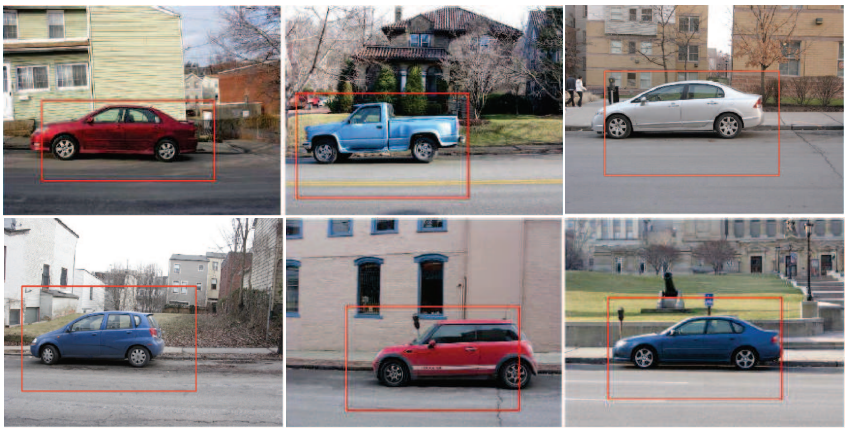


FIGURE 6.11: Localization of cars on test images. Note how the road below the cars is partially included in the detection output. This indicates that the appearance of road serves as a contextual indication for the presence of cars.

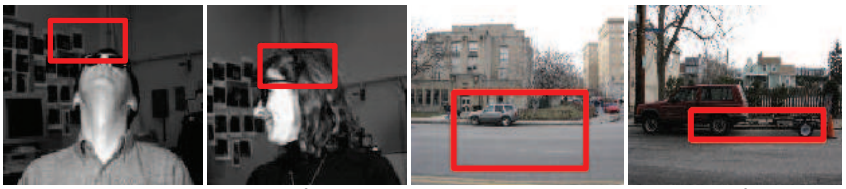


FIGURE 6.12: Difficult cases for localization. a, b: sunglasses are not clearly visible in the images. c: the foreground object is very small. d: misdetection due to the presence of the trailer wheel.

6.6.2 Experiments on Caltech-4

This subsection describes an experiment on the publicly available[6] Caltech-4 dataset. This collection contains images of different categories: airplanes_side, cars_brad, faces, motorbikes_side, and background clutter. We consider binary classification tasks where the goal is to distinguish one of the four object classes (airplanes_side, cars_brad, faces, and motorbikes_side) from the background clutter class. In this experiment, we randomly sampled a set of 100 images from each class for training. The set of the remaining images was split into equal-size testing and validation sets. The validation data was used for parameter tuning.

[6] http://www.robots.ox.ac.uk/∼vgg/data3.html

TABLE 6.4: Results of binary classification between each of the four classes of Caltech-4 and the background clutter class. *Glob-NN*, nearest neighbor approach [Nistér and Stewénius, 2006]. *GlobS-VM*: traditional SVM using global statistics. *Seg-SVM-FS* [Lampert et al., 2008] is the SVM method that require strongly labeled data during training. Results of *Seg-SVM-FS* for the Cars class is displayed as n/a because of the unavailability of ground truth annotation.

Class	Measure	Glob-NN	Glob-SVM	Seg-SVM-FS	Ours
Airplanes	Acc. (%)	89.74	**96.05**	89.40	**96.05**
	ROC Area	n/a	**0.99**	0.95	**0.99**
Cars	Acc. (%)	94.93	98.17	n/a	**98.28**
	ROC Area	n/a	**1.00**	n/a	**1.00**
Faces	Acc. (%)	59.83	88.70	86.78	**89.57**
	ROC Area	n/a	**0.95**	0.91	**0.95**
Motorbikes	Acc. (%)	76.80	**88.99**	84.67	87.81
	ROC Area	n/a	**0.95**	0.92	0.94

Table 6.4 shows the results of this experiment. *Seg-SVM-FS*, a method that requires bounding boxes of the foreground objects for training, does not perform as well as *Glob-SVM* which is based on global statistics from the whole image. This result suggests that contextual information is very important for classification tasks on this dataset. Indeed, it is easy to verify by visual inspection that the image backgrounds here often provide very strong categorization cues (see e.g. the almost constant background of the face images). As a result our method cannot provide any significant advantage on this dataset. However note that, unlike *Seg-SVM-FS*, our joint localization and classification does not harm the classification performance as our algorithm automatically learns the importance of contextual information and uses large subwindows for recognition.

6.7 Summary

In this chapter, we used the Seg-SVMs framework to develop a novel algorithm for discriminative detection and classification from weakly labeled time series. Discriminative detection was done using energy-based structure prediction, which sought

a set of subsegments that minimizes the sum of energies; this was performed efficiently using the algorithm proposed in Subsection 6.4.2. To learn the energy function for discriminative detection, we derived a maximum-margin learning formulation, which was based on multiple instance learning. We further extended our method to the spatial domain for discriminative object detection. We showed that the joint learning of the discriminative regions and of the region-based classifiers led to categorization accuracy superior to the performance obtained with supervised methods relying on costly human ground truth data.

Chapter 7

Unsupervised Learning for Temporal Clustering

"All truths are easy to understand once they are discovered;
the point is to discover them."
– Galileo Galilei

In this chapter, we show how to use Seg-SVMs to develop an unsupervised learning method for temporal factorization. This method is in contrast to the ones described in previous chapters which require fully or weakly annotated data. This unsupervised learning method is based on temporal clustering, which factorizes multiple time series into a set of non-overlapping segments that belong to several temporal clusters. It simultaneously determines the start and the end of each segment, and learns a multi-class SVM to separate temporal clusters. Fig. 7.1 illustrates the key idea of our method: divide each time series into a set of disjoint segments such that each segment belongs to a cluster and the cluster separability is maximum using the SVM margin as the measure of separability. Experiments on clustering human actions and bee dancing motions show that our method consistently matches and often surpasses the performance of state-of-the-art methods for temporal clustering.

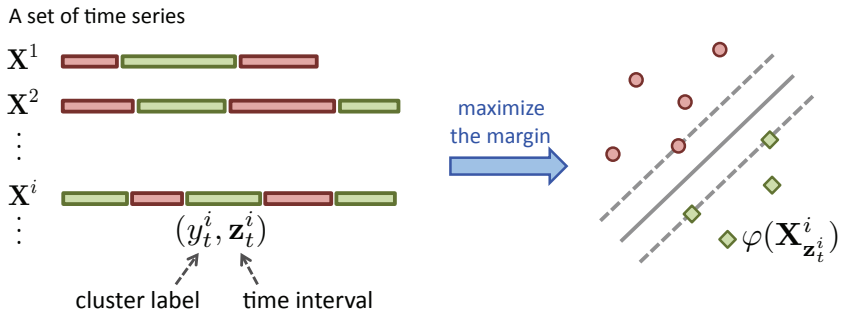


FIGURE 7.1: Temporal clustering: time series are partitioned into segments $\{\mathbf{z}_t^i\}$ and similar segments are grouped into classes (i.e., assigning a cluster label y_t^i to each segment $\mathbf{z}_t^i$). The objective is to maximize the margin for the separation between clusters. Though this figure only illustrates the case of two classes, our method is multi-class.

7.1 Energy-based temporal factorization

Our energy-based model for unsupervised factorization is the same as for the sequence labeling problem of Chapter 4. Given a time series $\mathbf{X}$, let $\mathcal{LS}(\mathbf{X})$ be the set of all legitimate segmentation and labeling of $\mathbf{X}$ of which the union of all segments is the entire time series $\mathbf{X}$. We perform joint segmentation and clustering by minimizing the sum of energies:

$$\underset{\{(y_t,\mathbf{z}_t)\}\in\mathcal{LS}(\mathbf{X})}{\text{minimize}} \sum_t E(\mathbf{X}_{\mathbf{z}_t}, y_t). \tag{7.1}$$

The energy of a segment-label pair is defined as:

$$E(\mathbf{X}_\mathbf{z}, y) = \max\{\max_{y'\neq y} \mathbf{w}_{y'}^T \varphi(\mathbf{X}_\mathbf{z}) + 1 - \mathbf{w}_y \varphi(\mathbf{X}_\mathbf{z}), 0\}. \tag{7.2}$$

Here $\mathbf{w}_1, \cdots, \mathbf{w}_m$ are parameter vectors for m clusters. We return to discuss how these parameters are learned in the next section. The above energy function reflects our desire that: if a segment $\mathbf{X}_\mathbf{z}$ is assigned to cluster y, the assignment must be confidently made, i.e., the assignment score of cluster y must exceed the assignment score of any other cluster y' by a large margin:

$$\mathbf{w}_y^T \varphi(\mathbf{X}_\mathbf{z}) \geq \mathbf{w}_{y'}^T \varphi(\mathbf{X}_\mathbf{z}) + 1 \ \forall y' \neq y. \tag{7.3}$$

7.2 Maximum-margin learning for temporal clustering

In Chapter 4, we presented an algorithm that used multi-class SVM for supervised learning. For unsupervised learning, we proposed to use Maximum Margin Clustering (MMC) [Xu et al., 2004, Zhao et al., 2008], which is unsupervised SVM. MMC, however, suffers from the problem of cluster degeneration, even in the presence of the cluster balancing constraint. To address this limitation, we propose to replace the current balancing constraint by another that better regulates the cluster sizes. Furthermore, we extend the formulation to temporal clustering.

7.2.1 Multi-class MMC

MMC [Xu et al., 2004] is a discriminative clustering algorithm that seeks a binary partition of the data to maximize the classification margin of SVMs. Xu and Schuurmans [2005], Zhao et al. [2008] further extended MMC for the multi-class case. Given a set of data points $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$, multi-class MMC simultaneously finds the maximum margin hyperplanes $\mathbf{w}_1, \cdots, \mathbf{w}_m \in \mathbb{R}^d$ and the best cluster labels $y_1, \cdots, y_n \in \{1, \cdots, m\}$ by optimizing:

$$\underset{\mathbf{w}_j, y_i, \xi_i \geq 0}{\text{minimize}} \ \frac{1}{2m} \sum_{j=1}^{m} ||\mathbf{w}_j||^2 + C \sum_{i=1}^{n} \xi_i, \tag{7.4}$$

$$\text{s.t.} \ \ \mathbf{w}_{y_i}^T \mathbf{x}_i - \mathbf{w}_y^T \mathbf{x}_i \geq 1 - \xi_i \ \forall i, y \neq y_i, \tag{7.5}$$

$$-\lambda \leq (\mathbf{w}_j - \mathbf{w}_{j'})^T \sum_{i=1}^{n} \mathbf{x}_i \leq \lambda \ \forall j, j'. \tag{7.6}$$

Here $\mathbf{w}_y^T \mathbf{x}_i$ is the confidence score for assigning data point $\mathbf{x}_i$ to cluster y. Constraint (7.5) requires $\mathbf{x}_i$ to belong to cluster y_i with relatively high confidence, higher than that of any other cluster by a margin. $\{\xi_i\}$ are slack variables which allow for penalized constraint violation, and C is the parameter controlling the trade-off between having a larger margin and having less constraint violation. Constraint (7.6) is added aiming to attain the balance between clusters.

The above MMC formulation has an inherent problem of a discriminative clustering method which is cluster degeneration, i.e., many clusters are empty. MMC requires every pair of clusters to be well separated by a margin. Thus every pair of clusters leads to a constraint on the maximum size of the margin. As a result, MMC is biased towards a model with fewer number of clusters because less effort for separation is required. In the extreme case, MMC would create a single cluster if Constraint (7.6) is not used, and therefore Constraint (7.6) is added to balance the cluster sizes. Here λ is a tunable parameter of the balancing constraint. However, in practice, it only works well if the number of allowable clusters is two, $m = 2$. For $m > 2$, cluster degeneration still occurs very often. Furthermore, Constraint (7.6) is not translation invariant. If the data is centralized at the origin, i.e. $\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i = \mathbf{0}$, the constraint has no effect and becomes redundant. In the next subsection we propose a modification to the MMC formulation to address this issue.

7.2.2 Membership requirement MMC

This section proposes Membership Requirement Maximum Margin Clustering (MRMMC), a modification to the MMC formulation to address the issue of cluster degeneration:

$$\underset{\substack{\mathbf{w}_j, y_i \\ \xi_i \geq 0, \beta_j \geq 0}}{\text{minimize}} \ \frac{1}{2m}\sum_{j=1}^{m}||\mathbf{w}_j||^2 + C\sum_{i=1}^{n}\xi_i + C_2\sum_{j=1}^{m}\beta_j, \tag{7.7}$$

$$\text{s.t. } \forall i : \mathbf{w}_{y_i}^T\mathbf{x}_i - \mathbf{w}_y^T\mathbf{x}_i \geq 1 - \xi_i \ \forall y \neq y_i, \tag{7.8}$$

$$\forall j : \exists\ l \text{ different indexes } i\text{'}s : \mathbf{w}_j^T\mathbf{x}_i - \mathbf{w}_{j'}^T\mathbf{x}_i \geq 1 - \beta_j \ \forall j' \neq j. \tag{7.9}$$

The difference between MRMMC and the original MMC formulation lies at Constraint (7.9). In the essence, this is a soft constraint for requiring each cluster to have at least l members; β_j's are slack variables that allow for penalized constraint violation. This new formulation has several advantages over the original one, as will be shown in the experimental section.

We propose to optimize the above using block coordinate descent, which alternates between two steps: i) fixing $\{\mathbf{w}_j\}$, optimizes Eq. 7.7 over $\{y_i\}$, $\{\xi_i\}$, $\{\beta_j\}$, and the l members $\mathbf{x}_i$'s for each cluster j; ii) fixing $\{y_i\}$ and the l members $\mathbf{x}_i$'s for each cluster j, optimizes Eq. 7.7 over $\{\mathbf{w}_j\}$, $\{\xi_i\}$, and $\{\beta_j\}$. This optimization algorithm is simple to implement and is guaranteed convergent. It is effective when combining with multiple restarts, as will be shown in the experiment section.

7.2.3 Maximum-margin temporal clustering

This section describes Maximum Margin Temporal Clustering (MMTC), an extension of MRMMC for temporal segmentation and clustering.

Given a collection of time series $\mathbf{X}^1, \cdots, \mathbf{X}^n$, MMTC divides each time series into a set of disjoint segments such that the separation between clusters of the segments is maximum. In other words, we would like to find $\{(y_t^i, \mathbf{z}_t^i)\}$, a legitimate segmentation and labeling of time series $\mathbf{X}^i$, that lead to maximum cluster separation:

$$\underset{\substack{\mathbf{w}_j,(y_t^i,\mathbf{z}_t^i)\in\mathcal{LS}(\mathbf{X}^i)\\ \xi_t^i\geq 0,\beta_j\geq 0}}{\text{minimize}} \ \frac{1}{2m}\sum_{j=1}^{m}||\mathbf{w}_j||^2 + C\sum_{i=1}^{n}\sum_{t=1}^{k_i}\xi_t^i + C_2\sum_{j=1}^{m}\beta_j, \tag{7.10}$$

$$\text{s.t. } \forall i,t : (\mathbf{w}_{y_t^i} - \mathbf{w}_y)^T\varphi(\mathbf{X}^i_{\mathbf{z}_t^i}) \geq 1 - \xi_t^i \ \forall y \neq y_t^i, \tag{7.11}$$

$$\forall j : \exists\ l \text{ pairs } (i,t) : (\mathbf{w}_j^T - \mathbf{w}_{j'}^T)\varphi(\mathbf{X}^i_{\mathbf{z}_t^i}) \geq 1 - \beta_j \ \forall j' \neq j. \tag{7.12}$$

Here $\mathbf{w}_y^T\varphi(\mathbf{X}^i_{\mathbf{z}_t^i})$ is the confidence score for assigning segment $\mathbf{X}^i_{\mathbf{z}_t^i}$ to cluster y. Constraint (7.11) requires segment $\mathbf{X}^i_{\mathbf{z}_t^i}$ to belong to cluster y_t^i with relatively high confidence, higher than that of any other cluster by a margin. $\{\xi_t^i\}$ are slack variables which allow for penalized constraint violation, and C is the parameter controlling the trade-off between large margin and less constraint violation. Constraint (7.12) requires each cluster to have at least l members; this is also a soft constraint as slack variables $\{\beta_j\}$ are used.

For unnormalized BoW feature, we have the additive property:

$$\varphi(\mathbf{X}^i_{\mathbf{z}^i_t}) = \sum_{p \in \mathbf{z}^i_t} \varphi(\mathbf{X}^i_p). \tag{7.13}$$

Given Eq. (7.13), the left hand side of Constraint (7.12) is:

$$(\mathbf{w}_j^T - \mathbf{w}_{j'}^T)\varphi(\mathbf{X}^i_{\mathbf{z}^i_t}) = (\mathbf{w}_j^T - \mathbf{w}_{j'}^T) \underset{p \in \mathbf{z}^i_t}{\text{mean}}\{\varphi(\mathbf{X}^i_p)\} len(\mathbf{z}^i_t). \tag{7.14}$$

For tractable optimization, we approximate the mean of $\{\varphi(\mathbf{X}^i_p)\}$ by a particular instance $\varphi(\mathbf{X}^i_q)$ and $len(\mathbf{z}^i_t)$ by $l_{max}/2$. Constraint (7.12) is then approximated by:

$$\forall j : \exists\ l' \text{ index pairs } (i,q) : (\mathbf{w}_j^T - \mathbf{w}_{j'}^T)\varphi(\mathbf{X}^i_q)\frac{l_{max}}{2} \geq 1 - \beta_j\ \forall j' \neq j. \tag{7.15}$$

Roughly speaking, Constraint (7.12) requires each cluster to have at least l segments, while Constraint (7.15) requires each cluster to have at least l' frames, with $l' = \frac{l_{max}}{2} l$. Both constraints regulate the cluster sizes by putting requirements on the cluster parameters $\mathbf{w}_j$. However, the latter does not depend on the segmentation.

The above problem can be solved using block coordinate descent that alternates between the following two procedures:

(A) Given the current segmentation, update the clustering model, i.e., fixing $\{\mathbf{z}^i_t\}$, optimizing (7.10) w.r.t. $\{y^i_t\}$, $\{\mathbf{w}_j\}$, $\{\xi^i_t\}$, and $\{\beta_j\}$.

(B) Given the current clustering model, update the segmentation and cluster labels, i.e., fixing $\{\mathbf{w}_j\}$, optimizing (7.10) w.r.t. $\{(y^i_t, \mathbf{z}^i_t)\}$, and $\{\xi^i_t\}$.

Note that $\{y^i_t\}$ and $\{\xi^i_t\}$ are optimized in both procedures. Procedure (A) performs MMC on a defined set of temporal segments. Procedure (B) updates the segmentation and cluster labels while fixing the weight vectors of the clustering model. Procedure (B) can be optimized efficiently using the dynamic programming algorithm described in Section 4.3.

7.3 Experiments

This section describes two sets of experiments. In the first set of experiments, we compare the performance of MRMMC against MMC and other clustering algorithms to illustrate the problem of unbalanced cluster. In the second set of experiments we compare the performance of MMTC to state-of-the-art algorithms for the TC problem on several time series datasets.

Our method has several parameters, and we found our algorithm robust to the selection of these parameters. We set up the slack parameters C and C_2 to 1 in our experiments. For the experiments in 7.3.1, we set $l = \frac{n}{3m}$ where n is the number of training samples and m is the number of classes. Similarly, for experiments in 7.3.2, we set $l' = \frac{\sum n_i}{3m}$ where $\sum n_i$ is the total lengths of all sequences and m is the number of classes.

7.3.1 Clustering performance of MRMMC

We validated the performance of MRMMC on publicly available datasets from the UCI repository[1]. This repository contains many datasets, but not many of them have more than several classes and contain no categorical or missing attributes. We selected the datasets that were used in the experiments of Zhao et al. [2008] and added several ones to create a collection of datasets with diversified numbers of classes. In particular, we used Wine, Glass, Segmentation, Digits, and Letters. We compared our method against the MMC formulation of Zhao et al. [2008] and k-means.

In our experiments, we set the number of clusters equal to the true number of classes. To measure clustering accuracy, we followed the strategy used by Xu et al. [2004], Zhao et al. [2008], where we first took a set of labeled data, removed the labels and ran the clustering algorithms. We then found the best one-to-one association between the resulting clusters and the ground truth clusters. Finally, we reported the percentage of correct assignment. This is referred as *purity* in information

[1] http://archive.ics.uci.edu/ml/

TABLE 7.1: Clustering accuracies (%) of k-means, MMC [Zhao et al., 2008], and MRMMC on UCI datasets. For each dataset, results within 1% of the maximum value are printed in bold. The second column lists the numbers of classes.

Dataset	m	k-means	MMC	MRMMC
Digit 3,8	2	94.7	**96.6**	**96.6**
Digit 1,7	2	**100**	**100**	**100**
Wine	3	**95.8**	**95.6**	**96.3**
Digit 1,2,7,9	4	87.4	**90.4**	**90.5**
Digit 0,6,8,9	4	94.8	94.5	**97.6**
Glass	6	43.5	46.1	**48.8**
Segmentation	7	59.0	40.0	**66.1**
Digit 0-9	10	79.2	36.5	**85.1**
Letter a-j	10	**42.6**	28.6	**43.0**
Letter a-z	26	27.3	10.9	**33.8**

theoretic measures [Meila, 2007, Tuytelaars et al., 2009]. Initialization was done similarly for all methods. For each method and a dataset, we first ran the algorithm with 10 random initializations on 1/10 of the dataset. We used the output of the run with lowest energy to initialize the final run of the algorithm on the full dataset. Table 7.1 displays the experimental results. As can be seen, our method consistently outperforms other clustering algorithms. The MMC formulation by Zhao et al. [2008] yields similar results to ours when the number of classes is two or three. However, when the number of classes is higher, MMC performance is significantly worse than ours; this is due to the problem of cluster degeneration.

7.3.2 Segmentation-clustering experiments

This section describes experimental results on several time series datasets. In all experiments we measured the joint segmentation-clustering performance as follows. We ran our algorithm to obtain a segmentation and cluster labels. At that point, each frame was associated with a particular cluster, and we found the best cluster-to-class association between the resulting clusters and the ground truth classes. The overall frame-level accuracy was calculated as the percentage of agreement; this is referred as *purity* in information theoretic measures [Meila, 2007, Tuytelaars et al.,

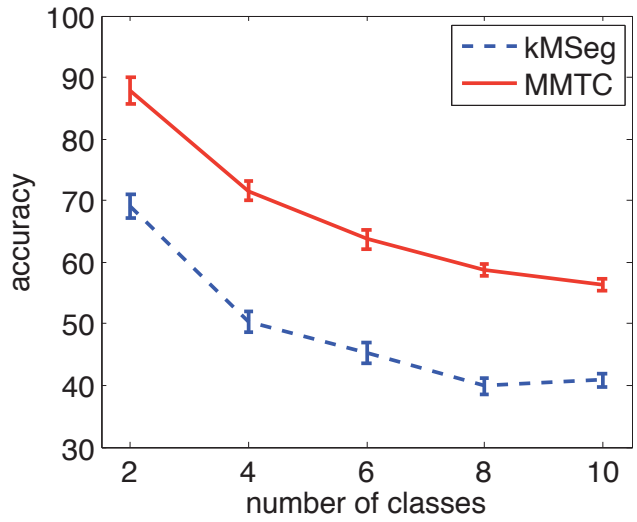


FIGURE 7.2: Segmentation-clustering accuracy as a function of the number of classes. MMTC outperforms kMSeg.

2009]. For comparison, we implemented kMSeg [Robards and Sunehag, 2009] a generative counterpart of MMTC in which MRMMC is replaced by k-means.

7.3.2.1 Weizmann dataset

As described in Section 4.4.2, the Weizmann dataset contains 90 video sequences of 9 people, each performing 10 actions. We extracted binary masks and computed Euclidean distance transform for frame-level features. We built a codebook of temporal words with 100 clusters using k-means, and the segment-level feature vector was the histogram of temporal words in the segment.

Fig. 7.2 plots the frame-level accuracies as a function of the number of classes. We computed the frame-level accuracy for m classes ($2 \leq m \leq 10$) as follows. We randomly chose m classes out of 10 actions and concatenated video sequences of those actions (with random ordering) to form a long video sequence. We ran MMTC and kMSeg and reported the frame level accuracies as explained at the beginning of Sec. 7.3.2. We repeated the experiment with 30 runs; the mean and standard error curves are plotted in Fig. 7.2. As can be seen, MMTC outperformed kMSeg. In this experiment, the desired number of clusters was set to the true number of classes.

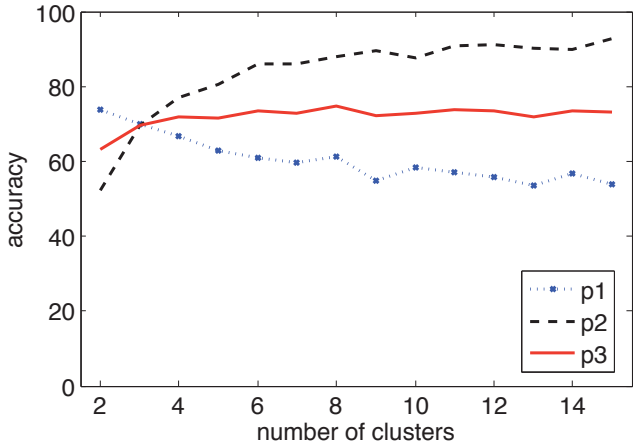


FIGURE 7.3: Sensitivity analysis – accuracy values when the desired number of clusters varies around 10, the true number of classes.

The above experiment assumed the true number of classes was known, but this might not be the case in reality. For sensitivity analysis, we performed an experiment where we fixed the number of true classes but varied the desired number of clusters. For this experiment, the evaluation criterion given at the beginning of Sec. 7.3.2 could not be applied because there was no one-to-one mapping between the resulting clusters and the ground truth classes. We instead used different performance criteria which were based on the two principles: i) two frames that belong to the same class should be assigned to the same cluster; and ii) two frames that belong to different classes should be assigned to different clusters. Formally speaking, consider all pairs of same-class video frames, let p_1 be the percentage of pairs of which both frames were assigned to the same cluster. Consider all pairs of different-class video frames, let p_2 be the percentage of pairs of which two frames were assigned to different clusters. Let p_3 be the average of these two values $p_3 = (p_1 + p_2)/2$, which summarizes the clustering performance. These criteria are referred as pair-counting measures [Meila, 2007]. Fig. 7.3 plots these values; the true number of classes is 10 while the desired number of clusters varies from 2 to 15. As the number of clusters increases, p_1 decreases while p_2 increases. However, the summarized value p_3 is not so sensitive to the desired number of clusters.

TABLE 7.2: Joint segmentation-clustering accuracy (%) on the honeybee dataset. HDP-HMM-US results were published by Fox et al. [2009]. MMTC and kMSeg results are averaged over 20 runs; the standard errors are also shown. Results within 1% of the maximum values are displayed in bold. Our method achieves the best or close to the best result on five out of six sequences, and it has the highest average accuracy.

Sequence	1	2	3	4	5	6	Mean
HDP-HMM-US	45.0	42.7	47.3	88.1	**92.5**	**88.2**	67.3
kMSeg	**51.5±.01**	50.1±.15	46.7±.12	**91.0±.07**	**91.7±.07**	84.7±2.27	69.3±.45
MMTC	**51.0±.56**	**66.6±2.39**	**48.3±.25**	**91.6±.16**	91.2±.02	**88.8±.07**	**72.9±.57**

7.3.2.2 Honeybee dance dataset

As described in Section 4.4.1, the honeybee dataset [Oh et al., 2008] contains video sequences of dancing honeybees. The bees were visually tracked, and their locations and head angles were recorded. The frame-level feature vector was $[vx, vy, \sin(v\theta), \cos(v\theta)]$, where (vx, vy) was the velocity vector and $v\theta$ was the angular velocity of the bee's head angle. The segment-level feature vector combines observation and interaction features as described in Section 4.4.1.

Tab. 7.2 displays the experimental results of MMTC, kMSeg, and HDP-HMM-US [Fox et al., 2009] the state-of-the-art unsupervised method for this dataset. HDP-HMM-US is a non-parametric method combining hierarchical Dirichlet process prior and a switching linear dynamical system. The reported numbers in Tab. 7.2 are frame-level accuracy (%) measuring the joint segmentation-clustering performance as described at the beginning of Sec. 7.3.2. For MMTC and kMSeg, we show both the averages and standard errors of the results over 20 runs. For each honeybee sequence, results within 1% of the maximum value are printed in bold. MMTC achieves the best or close to the best performance on five out of six sequences, and it has the highest overall accuracy. For several sequences, the results of our method are close to those of the supervised methods, Table 4.1. Fig. 7.4 displays side-by-side comparison of the prediction result and the human-labeled ground truth. In this experiment, the coordinate descent optimization algorithm of MMTC required 34 iterations on average (for convergence).

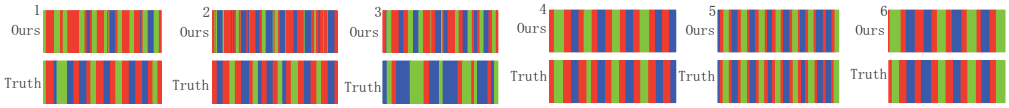


FIGURE 7.4: MMTC results versus human-labeled ground truth. Segments are color coded; red, green, blue correspond to waggle, right-turn, left-turn, respectively. This figure is best seen in color.

7.4 Summary

This chapter proposed MMTC, a novel Seg-SVMs algorithm for simultaneous segmentation and clustering of time series. Clustering was performed using temporal extensions of MMC for learning discriminative patterns whereas the inference over the segments was done with dynamic programming. Experiments on several real datasets in the context of human activity and honeybee dancing showed that our discriminative clustering often led to segmentation-clustering accuracy superior to the performance obtained with generative methods. Although the results presented in the chapter exceeded state-of-the-art algorithms', there are several open research problems that need to be addressed in future work. First, currently, the number of clusters is assumed to be known. In order to automatically select the optimal number of clusters, criteria similar to Akaike Information Criterion or Minimum Description Length could be added to the MMTC formulation. Second, MMTC is susceptible to local minima, and although random initialization with multiple restarts worked well, better initialization strategies or convex approximations to the problem will be worth exploring in future work.

Chapter 8

Discussion and Conclusion

"Now this is not the end. It is not even the beginning of the end.
But it is, perhaps, the end of the beginning."
– Winston Churchill

We have presented Seg-SVMs, a segment-based framework for time series analysis, and demonstrated its benefits in understanding various types of human and animal behavior, from facial expression, hand gesture, human action to bee dance and mouse activity. The development of the Seg-SVMs framework was driven by the importance of detecting some temporal events of interest; these events of interest may be the actions that belong to a predefined class, the activities that discriminate between two different behaviors, or the motions that are repeatedly performed. Our framework was designed to model and detect these events, which are typically complex and buried in long chains of observations.

Although the current framework is applicable to and effective for a wide range of important problems, it has limitations. This chapter describes several ways to improve and extend the current framework.

8.1 Limitation and Future Directions

8.1.1 Probabilistic Interpretation

Throughout this thesis, we have proposed to use energy-based structure prediction for time series analysis. We have shown that energy-based structure prediction provides a principled mechanism for concurrent top-down labeling and bottom-up localization. An alternative approach is to use probabilistic models, which, however, have several major disadvantages [LeCun et al., 2006]: i) the normalization requirement limits the choice of energy functions we can use, and ii) learning and inference may be very complicated, expensive, or even intractable. Furthermore, probabilistic models are not as flexible as energy-based models. In this dissertation, we have shown the flexibility of energy-based models for incorporating additional constraints to address novel applications. In contrast, it is unclear how to extend probabilistic models to satisfy new demands. Take early event detection as an example, for early detection, it is necessary for the detector to recognize partial events. For an energy-based model, this can be achieved by requiring that the energy of a partial event is lower than the energy of any past segment. For a probabilistic model, it is unclear how to extend the learning formulation to train a detector to detect partial events.

Energy-based models, however, have a disadvantage. They do not provide a probability estimate, which is sometimes necessary. For the problems described in this dissertation, it is merely necessary that the time series analysis system gives the lowest energy to the correct answer; the energy of the correct answer is irrelevant, as long as it is lower than the energies of other answers. However, the output of time series analysis must sometimes be combined with that of another system, fed into the input of another system, or presented to a human decision maker. But energies are uncalibrated (i.e., measured in arbitrary units), and therefore, combining separately trained energy-based models is not straightforward. Calibrating energies to permit such combinations can be done in a number of ways such as Platt scaling or Gibbs distribution fitting. These methods, however, do not guarantee that the calibrated energies are good probability estimates.

8.1.2 Verification of what is discovered

We have demonstrated the ability of Seg-SVMs for discovering discriminative and similar events, and in general, discovery ability is a crucial requirement for time series analysis. However, it remains unclear how to verify what we discover, especially for a discriminative method like Seg-SVMs. One possible solution is to use annotated data as we did in Chapter 7, but annotated data is not always available for verification. Another possible solution is to integrate time series analysis into a bigger system and measure the performance of whole system (e.g., in Chapter 6, we used classification performance to benchmark discriminative detection). Another possible direction is to derive a solution that is analogous to what has been done for generative probabilistic models. For a generative probabilistic model, one can measure the fitness of the model in terms of probabilities. For a discriminative model, we can possibly measure the degree of separation between different classes. But this direction has not been well understood yet. It is a good subject for future study.

8.1.3 Constraint satisfaction

Seg-SVMs train a system for time series analysis by solving a constrained optimization problem: the objective of the optimization is to maximize the margin while the constraints are derived from the requirements for an ideal system. In general, however, the set of constraints might be too stringent and no ideal system exists. In this thesis, we allow for constraint violation by introducing slack variables and then penalizing for the slackness. But the effects of non-satisfying constraints remain unclear, especially with respect to the tradeoff between different types of requirements. Consider early detection as a concrete example, the decisions need to be both reliable and timely. But not every event can be detected reliably and early, even reliably alone. In this case, would it make sense to address early detection? Would reliability is sacrificed for earliness? In this dissertation, we have shown that the obtained detectors can make faster decisions while maintaining the same or even better level of reliability. However, in general, there is no prior theoretical guarantee that adding the timeliness constraints would not lower the reliability of

the detectors. This limitation of the current framework is a good direction for future study.

8.1.4 Inter-segment dependency

One limitation of Seg-SVMs is the ignorance of inter-segment dependency. Seg-SVMs exploit within-segment causality constraint that the presence or absence of a particular event constrains on those of any other events. Seg-SVMs assume that the label of a segment can be recognized by classifying the constituent frames; the frames outside the segment and the labels of other segments are irrelevant. In this dissertation, we have shown the effectiveness of Seg-SVMs for a number of time series analysis problems. However, in many domains, it might be beneficial to consider inter-segment dependency (e.g., hand shaking is often followed by greeting, getting in a car must be preceded by opening a car door). As such, a direction for future study is to extend the current framework to account for inter-segment dependency.

8.1.5 Improvement with non-linear kernel

A possible improvement is to use a non-linear kernel for measuring similarity between time series segments. Non-linear kernels such as Intersection or Chi-square kernels have been shown to outperform the linear kernel in scene categorization and object detection. In this thesis, however, we deliberately avoided non-linear kernels due to the implicitness of their feature maps. This implicitness prevents the use of constraint generation in the optimization of SOSVMs. This has long been a limitation of SOSVMs. However, recent work from Vedaldi and Zisserman [2010] shed some light on a solution to this problem. They showed that non-linear kernels can be approximated by some explicit feature maps. Thus, a non-linear SOSVM can be approximated by a linear SOSVM in a transformed space, and the existing optimization procedure with constraint generation can be used. This approach is worth exploring in future work.

8.1.6 Optimization

Another future direction is to investigate a better optimization strategy for weakly supervised and unsupervised learning algorithms that have non-convex formulations. Even though in our experiments, random initialization with multiple restarts worked well, better initialization strategies (e.g., self-space learning [Kumar et al., 2010]) or convex approximations (e.g., Semi-Definite Programming relaxation [Xu et al., 2004]) to the problem will be worth exploring in future work.

8.1.7 Beyond time series

Although the Seg-SVMs framework was developed for time series analysis, many ideas presented in this dissertation can be extended to the spatial domain. The weakly supervised learning algorithm can be used to discover discriminative image regions, as shown in Chapter 6. The active approach for training early event detectors can be generalized to detection of truncated objects. This would be in contrast to the passive approach of Vedaldi and Zisserman [2009]. In Chapter 4, we showed segmentation with non-maxima suppression worked better than maximizing the SVM scores. This idea can be investigated for object detection.

In this thesis, we addressed event localization in time. This satisfied the goals of the applications described in this dissertation, but it may not suffice for applications in which events can happen at the same temporal locations but at different spatial locations. A direction for future work is to extend this framework for detecting spatio-temporal events, which requires localization in both time and space.

8.2 Conclusion

We presented segment-based SVMs (Seg-SVMs), a framework for time series analysis. Seg-SVMs were developed on three ideas: energy-based structure prediction, bag-of-words representation, and maximum-margin training. We used Seg-SVMs to address five important problems, three of which have received little or no attention

in the computer vision literature. Specifically, we proposed fully-supervised learning algorithms for event detection, sequence labeling, and early event detection. We introduced a weakly-supervised learning algorithm for discovering discriminative events and an unsupervised learning algorithm for temporal factorization. We performed experiments on datasets of varying complexity and showed the advantages of our algorithms over competing approaches. In this thesis, we demonstrated the benefits of our framework for human and animal behavior understanding, but we believe it can be applied to many other domains.

Appendix A

Global Optimality of Algorithm 3

Algorithm 3 guarantees to produce a globally optimal solution for (6.16). Even stronger, the set $\mathcal{Z}^m = \{I_1^m, \cdots, I_m^m\}$ produced by the algorithm is the set of best m intervals that maximize (6.16). This section sketches a proof by induction.

+) $m = 1$, this can be easily verified.
+) Suppose $\mathcal{Z}^m$ is the set of best m intervals that maximize (6.16). We now prove that $\mathcal{Z}^{m+1}$ is optimal for $m+1$ intervals. Assume the contrary, $\mathcal{Z}^{m+1}$ is not optimal for $m+1$ intervals. There exist disjoint intervals $T_1, \cdots, T_{m+1}$ such that:

$$\sum_{i=1}^{m+1} h(T_i) > \sum_{i=1}^{m+1} h(I_i^{m+1}). \tag{A.1}$$

Because the way we construct $\mathcal{Z}^{m+1}$ from $\mathcal{Z}^m$, we have:

$$\sum_{i=1}^{m+1} h(I_i^{m+1}) = \sum_{i=1}^{m} h(I_i^m) + \max\{h(J_1), -h(J_2)\},$$

$$\text{where } J_1 = \arg\max_{J \in \mathcal{I}} h(J) \text{ s.t. } J \cap I_i^m = \emptyset \ \forall i, \tag{A.2}$$

$$J_2 = \arg\max_{J \in \mathcal{I}} -h(J) \text{ s.t. } J \subset I_i^m \text{ for an } i. \tag{A.3}$$

This, together with (A.1), leads to:

$$\max\{h(J_1), -h(J_2)\} < \sum_{i=1}^{m+1} h(T_i) - \sum_{i=1}^{m} h(I_i^m). \tag{A.4}$$

Consider the overlapping between $T_1, \cdots, T_{m+1}$ and $I_1^m, \cdots, I_m^m$, there are two cases.

• *Case 1:* $\exists j : T_j \cap I_i^m = \emptyset \ \forall i$. In this case, we have:

$$h(T_j) \leq h(J_1) < \sum_{i=1}^{m+1} h(T_i) - \sum_{i=1}^{m} h(I_i^m), \tag{A.5}$$

$$\Rightarrow \sum_{i=1}^{m} h(I_i^m) < \sum_{i=\overline{1,m+1}, i\neq j} h(T_i). \tag{A.6}$$

This contradicts with the assumption that $\{I_1^m, \cdots, I_m^m\}$ is the set of best m intervals that maximize (6.16).

• *Case 2:* $\forall j, \exists i : T_j \cap I_i^m \neq \emptyset$. Since there are $m+1$ T_j's, and there are only m I_i^m's, there must exist one i s.t. I_i^m intersects with at least two of T_j's. Suppose l, l_1, l_2 are indexes s.t. $T_{l_1} \cap I_l^m \neq \emptyset$ and $T_{l_2} \cap I_l^m \neq \emptyset$. Furthermore, suppose T_{l_1}, T_{l_2} are consecutive intervals of T_j's (T_{l_1} precedes T_{l_2} and there is no T_j in between). Let $T_{l_1} = [t_{l_1}^-, t_{l_1}^+]$, $T_{l_2} = [t_{l_2}^-, t_{l_2}^+]$. Consider the interval $T = [t_{l_1}^+ + 1, t_{l_2}^- - 1]$. Because $T_{l_1} \cap I_l^m \neq \emptyset$ and $T_{l_2} \cap I_l^m \neq \emptyset$, T must be a subinterval of I_l^m, i.e. $T \subset I_l^m$. Hence

$$-h(T) \leq -h(J_2) < \sum_{i=1}^{m+1} h(T_i) - \sum_{i=1}^{m} h(I_i^m), \tag{A.7}$$

$$\Rightarrow \sum_{i=1}^{m} h(I_i^m) < h(T) + \sum_{i=1}^{m+1} h(T_i), \tag{A.8}$$

$$\Rightarrow \sum_{i=1}^{m} h(I_i^m) < h(\underbrace{T_{l_1} \cup T \cup T_{l_2}}_{\text{an interval}}) + \sum_{i\neq l_1, l_2} h(T_i). \tag{A.9}$$

This contradicts with the assumption that $\{I_1^m, \cdots, I_m^m\}$ is the best set of m intervals that maximize (6.16).

Since both cases lead to a contradiction, $\mathcal{Z}^{m+1}$ must be the best set of $m+1$ intervals that maximize (6.16). This completes the proof. □

Bibliography

R. Adams and D. MacKay. Bayesian online changepoint detection. Technical report, University of Cambridge, 2007.

S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:1475–1490, 2004.

J. Aggarwal and Q. Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999.

Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines. In *International Conference on Machine Learning*, 2003.

E. Andrade, S. Blunsden, and R. Fisher. Modelling crowd scenes for event detection. In *International Conference on Pattern Recognition*, 2006.

S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Neural Information Processing Systems*, 2003.

S. Avidan. Subset selection for efficient SVM tracking. In *Computer Vision and Pattern Recognition*, 2003.

L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *International Conference on Pervasive Computing*, 2004.

M. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Recognizing facial expression: Machine learning and application to spontaneous behavior. In *Computer Vision and Pattern Recognition*, 2005.

M. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Automatic recognition of facial actions in spontaneous expressions. *Journal of Multimedia*, 1(6): 22–35, 2006.

D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(4–5):993–1022, 2003.

A. F. Bobick and J. Davis. Real-time recognition of activity using temporal templates. In *Proceedings of IEEE Workshop on Applications on Computer Vision*, 1996.

A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

A. F. Bobick and A. D. Wilson. A state-based technique for the summarization and recognition of gesture. *Transactions on Pattern Analysis and Machine Intelligence*, 19(12): 1325–1337, 1997.

M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):844–851, 2000.

M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *Proceedings of IEEE Conference of Computer Vision and Pattern Recognition*, 1997.

P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4), 1992.

D. B. D. Cao, O. Masoud, and N. Papanikolopoulos. Online motion classification using support vector machines. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2004.

K. Chang, T. Liu, and S. Lai. Learning partially-observed hidden conditional random fields for facial expression recognition. In *Computer Vision and Pattern Recognition*, 2009.

J. Cohn, T. Simon, I. Matthews, Y. Yang, M. H. Nguyen, M. Tejera, F. Zhou, and F. De la Torre. Detecting depression from facial actions and vocal prosody. In *Proceedings of International Conference on Affective Computing and Intelligent Interaction*, 2009.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

J. Davis and A. Tyagi. Minimal-latency human action recognition using reliable-inference. *Image and Vision Computing*, 24(5):455–472, 2006.

F. De la Torre, J. Hodgins, J. Montano, S. Valcarcel, and J. Macey. Guide to the Carnegie Mellon University multimodal activity (CMU-MMAC) database. Technical Report CMU-RI-TR-08-22, Robotics Institute, Carnegie Mellon University, 2008.

F. Desobry, M. Davy, and C. Doncarli. An online kernel change detection algorithm. *IEEE Transactions on Signal Processing*, 53(8):2961–2974, 2005.

T. Dietterich, R. Lathrop, and T. Lozano-Pérez. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1–2):31–71, 1997.

P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *ICCV Workshop on Visual Surveillance & Performance Evaluation of Tracking and Surveillance*, 2005.

A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Proceedings of International Conference on Computer Vision*, 2003.

P. Ekman and W. Friesen. Facial action coding system: A technique for the measurement of facial movement. *Consulting Psychologists Press*, 1978.

C. Fanti, L. Zelnik-Manor, and P. Perona. Hybrid models for human motion recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

T. Fawcett and F. Provost. Activity monitoring: Noticing interesting changes in behavior. In *Proceedings of the SIGKDD Conference on Knowledge Discovery and Data Mining*, 1999.

L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *Proceedings of IEEE Conference of Computer Vision and Pattern Recognition*, 2005.

E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky. Nonparametric Bayesian learning of switching linear dynamical systems. In *Advances in Neural Information Processing Systems*. 2009.

L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, 2007.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming (web page & software). http://stanford.edu/∼boyd/cvx, Oct. 2008a.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control (a tribute to M. Vidyasagar), Lecture Notes in Control and Information Sciences*, pages 95–110. Springer, 2008b.

J. Graunt. *Natural and Political Observations Made Upon the Bills of Mortality*. John Martin and James Allestry, 1662.

G. Guerra-Filho and Y. Aloimonos. Understanding visuo-motor primitives for motion synthesis and analysis. *Computer Animation and Virtual Worlds*, 17(3-4), 2006.

P. Haider, U. Brefeld, and T. Scheffer. Supervised clustering of streaming data for email batch detection. In *International Conference on Machine Learning*, 2007.

M. Hoai and F. De la Torre. Maximum margin temporal clustering. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, 2012a.

M. Hoai and F. De la Torre. Max-margin early event detectors. In *Under review for the IEEE Conference on Computer Vision and Pattern Recognition*, 2012b.

M. Hoai, Z.-Z. Lan, and F. De la Torre. Joint segmentation and classification of human actions in video. In *Proceedings of IEEE Conference of Computer Vision and Pattern Recognition*, 2011.

S. Hongeng and R. Nevatia. Large-scale event detection using semi-hidden Markov models. In *International Conference on Computer Vision*, 2003.

M. Kadous. *Temporal classification: Extending the classification paradigm to multivariate time series*. PhD thesis, 2002.

R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(Series D):35–45, 1960.

Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. In *Proceedings of International Conference on Computer Vision*, 2005.

K.-J. Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

J. L. Klein. *Statistical Vision in Time: a History of Time Series Analysis*. Cambridge University Press, Cambridge, UK, 1997.

S. Koelstra and M. Pantic. Non-rigid registration using free-form deformations for recognition of facial actions and their temporal dynamics. In *International Conference on Automatic Face and Gesture Recognition*, 2008.

M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, 2010.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: object localization by efficient subwindow search. In *Computer Vision and Pattern Recognition*, 2008.

I. Laptev and T. Lindeberg. Space-time interest points. In *International Conference on Computer Vision*, 2003.

I. Laptev and P. Perez. Retrieving actions in movies. In *Proceedings of International Conference on Computer Vision*, 2007.

I. Laptev, M. Marsza, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition*, 2008.

B. Laxton, J. Lim, and D. Kriegman. Leveraging temporal, contextual and ordering constraints for recognizing complex activities in video. In *Computer Vision and Pattern Recognition*, 2007.

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*, 2006.

Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F.-J. Huang. A tutorial on energy-based learning. In G. Bakir, T. Hofman, B. Schölkopf, A. Smola, and B. Taskar, editors, *Predicting Structured Data*. MIT Press, 2006.

T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.

D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of European Conference on Machine Learning*, 1998.

H.-Y. M. Liao, D.-Y. Chen, C.-W. Su, and H.-R. Tyan. Real-time event detection and its application to surveillance systems. In *International Symposium on Circuits and Systems*, 2006.

T. W. Liao. Clustering of time series data – a survey. *Pattern Recognition*, 38(11):1857–1874, 2005.

G. Littlewort, M. Bartlett, I. Fasel, J. Susskind, and J. Movellan. Dynamics of facial expression extracted automatically from video. *Image and Vision Computing*, 24(6): 615–625, 2006.

D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of International Conference on Computer Vision*, 1999.

D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

P. Lucey, J. F. Cohn, S. Lucey, S. Sridharan, and K. M. Prkachin. Automatically detecting pain using facial actions. *Proceedings of International Conference on Affective Computing and Intelligent Interaction*, 2009.

P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended Cohn-Kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *CVPR Workshop on Human Communicative Behavior Analysis*, 2010.

S. Lucey, I. Matthews, C. Hu, Z. Ambadar, F. De la Torre, and J. Cohn. AAM derived face representations for robust facial action recognition. In *International Conference on Automatic Face and Gesture Recognition*, 2006.

I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):1573–1405, 2004.

M. Meila. Comparing clusterings – an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895, 2007.

D. Neill, A. Moore, and G. Cooper. A Bayesian spatial scan statistic. In *Advances in Neural Information Processing Systems*. 2006.

M. H. Nguyen, L. Torresani, F. De la Torre, and C. Rother. Weakly supervised discriminative localization and classification: a joint learning process. In *Proceedings of International Conference on Computer Vision*, 2009.

M. H. Nguyen, T. Simon, F. De la Torre, and J. Cohn. Action unit detection with segment-based SVMs. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3):299–318, 2008.

D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

S. Nowozin, G. Bakir, and K. Tsuda. Discriminative subsequence mining for action classification. In *International Conference on Computer Vision*, 2007.

S. M. Oh, J. M. Rehg, T. Balch, and F. Dellaert. Learning and inferring motion patterns using parametric segmental switching linear dynamic systems. *International Journal of Computer Vision*, 77(1–3):103–124, 2008.

D. Okanohara, Y. Miyao, Y. Tsuruoka, and J. Tsujii. Improving the scalability of semi-Markov conditional random fields for named entity recognition. In *Proceedings of International Conference on Computational Linguistics*, 2006.

M. Pantic and L. Rothkrantz. Facial action recognition for facial expression analysis from static face images. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(3):1449–1461, 2004.

V. Pavlovic and J. M. Rehg. Impact of dynamic model learning on classification of human motion. In *Proceedings of IEEE Conference of Computer Vision and Pattern Recognition*, 2000.

V. Pavlovic, J. M. Rehg, and J. MacCormick. Learning switching linear models of human motion. In *Advances in Neural Information Processing Systems*, 2000.

D. B. Percival and A. T. Walden. *Wavelet Methods for Time Series Analysis*. Cambridge University Press, 2000.

C. Piciarelli, C. Micheloni, and G. L. Foresti. Trajectory-based anomalous event detection. *IEEE Transactions on Circuits and System for Video Technology*, 18(11):1544–1554, 2008.

M. Pittore, C. Basso, and A. Verri. Representing and recognizing visual dynamic events with support vector machines. In *Proceedings of International Conference on Image Analysis and Processing*, 1999.

R. Polana and R. Nelson. Low level recognition of human motion (or how to get your man without finding his body parts). In *Proceedings of IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, 1994.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

M. Robards and P. Sunehag. Semi-Markov kMeans clustering and activity recognition from body-worn sensors. In *International Conference on Data Mining*, 2009.

R. G. Roberts, A. Christoffersson, and F. Cassidy. Real-time event detection, phase identification and source location estimation using single station three-component seismic data. *Geophysical Journal*, 97:471–480, 1989.

M. Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *Proceedings of International Conference on Computer Vision*, 2011.

S. Sarawagi and W. Cohen. Semi-Markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, 2005.

S. Satkin and M. Hebert. Modeling the temporal extent of actions. In *European Conference on Computer Vision*, 2010.

B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and beyond*. MIT Press, Cambridge, MA, 2002.

L. Shang and K. Chan. Nonparametric discriminant HMM and application to facial expression recognition. In *Conference on Computer Vision and Pattern Recognition*, 2009.

E. Shechtman and M. Irani. Space-time behavior based correlation –or– how to tell if two underlying motion fields are similar without computing them? *Transactions on Pattern Analysis and Machine Intelligence*, 29(11):2045–2056, 2007.

Q. Shi, L. Wang, L. Cheng, and A. Smola. Discriminative human action segmentation and recognition using semi-Markov model. In *Computer Vision and Pattern Recognition*, 2008.

J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision*, 2003.

J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering objects and their location in images. In *Proceedings of International Conference on Computer Vision*, 2005.

C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Conditional models for contextual human motion recognition. In *International Conference on Computer Vision*, 2005.

P. Smith, N. da Vitoria Lobo, and M. Shah. Temporal boost for event recognition. In *Proceedings of International Conference on Computer Vision*, 2005.

A. J. Smola, S. V. N. Vishwanathan, and T. Hofmann. Kernel methods for missing variables. In *International Workshop on Artificial Intelligence and Statistics*, 2005.

Y. Sun and L. Yin. Facial expression recognition based on 3D dynamic range model sequences. In *European Conference on Computer Vision*, 2008.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*. 2003.

F. E. Tay and L. Cao. Application of support vector machines in financial time series forecasting. *The International Journal of Management Science*, 29(4):309–317, 2001.

Y. Tian, J. F. Cohn, and T. Kanade. Facial expression analysis. In S. Z. Li and A. K. Jain, editors, *Handbook of face recognition*. New York, New York: Springer, 2005.

R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society, Series B*, 58(267–288), 1996.

Y. Tong, W. Liao, and Q. Ji. Facial action unit recognition by exploiting their dynamic and semantic relationships. *Transactions on Pattern Analysis and Machine Intelligence*, 29 (10):1683–1699, 2007.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

P. Turaga, A. Veeraraghavan, and R. Chellappa. Unsupervised view and rate invariant clustering of video sequences. *Computer Vision and Image Understanding*, 113(2), 2009.

T. Tuytelaars, C. H. Lampert, M. B. Blaschko, and W. Buntine. Unsupervised object discovery: A comparison. *International Journal of Computer Vision*, 88(2):284–302, 2009.

M. Valstar and M. Pantic. Combined support vector machines and hidden markov models for modeling facial action temporal dynamics. In *ICCV Workshop on Human Computer Interaction*, 2007.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, NY, 1998.

H. Vassilakis, A. J. Howell, and H. Buxton. Comparison of feedforward (TDRBF) and generative (TDRGBN) network for gesture based control. In *Proceedings of Revised Papers From the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction*, 2002.

D. D. Vecchio, R. M. Murray, and P. Perona. Decomposition of human motion into dynamics-based primitives with application to drawing tasks. *Automatica*, 39(12), 2003.

A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

A. Vedaldi and A. Zisserman. Structured output regression for detection with partial truncation. In *Proceedings of Neural Information Processing Systems*, 2009.

A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *Computer Vision and Pattern Recognition*, 2010.

T. Wang, J. Li, Q. Diao, W. Hu, Y. Zhang, and C. Dulong. Semantic event detection using conditional random fields. In *CVPR Workshop*, 2006.

G. Xu, Y.-F. Ma, H.-J. Zhang, and S. Yang. A HMM based semantic analysis framework for sports game event detection. *International Conference on Image Processing*, 2003.

L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *AAAI Conference on Artificial Intelligence*, 2005.

L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems*. 2004.

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*, 1992.

C.-N. J. Yu and T. Joachims. Learning structural SVMs with latent variables. In *International Conference on Machine Learning*, 2009.

A. L. Yuille and A. Rangarajan. The concave-convex procedure (CCCP). In *Neural Information Processing Systems*, 2002.

J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *International Journal of Computer Vision*, 73(2):213–238, 2001.

B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *International Conference on Machine Learning*, 2008.

H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

F. Zhou, F. De la Torre, and J. F. Cohn. Unsupervised discovery of facial events. In *Computer Vision and Pattern Recognition*, 2010.