Chapter 3

Extracting Spatial Relations of Extended Objects

3.1 Introduction

In most QSR approaches for HAR, humans are abstracted using single bounding boxes. However, abstracting the whole body using a single bounding box abstracts away a lot of important interaction details. For example, Figure 1-1 in Chapter 1 shows the *handshaking* activity from the UT Interaction dataset [3] with two different types of abstraction. If a single bounding box is used to abstract the human bodies involved as shown in Figure 1-1a, then the activity can best be seen as a sequence of qualitative relations expressing the following - the bodies (abstracted as bounding boxes) are far away from each other initially, then they come closer, and thereafter the bounding boxes overlap and move away from one another. However, such a description does not describe the *handshaking* activity at the desired level of fineness. The description does not include important details pertaining to the *handshaking* activity, such as the fact that hands of the two persons involved came in contact rather than the whole body. This was lost in the coarse abstraction of the interacting bodies using a single bounding box.

Human interactions are better described when the human body is viewed as a collection of parts [15]; each part abstracted through a bounding box. This is shown in Figure 1-1b, where separate bounding boxes are used to abstract different parts of the human body. If a qualitative description of such an abstraction is obtained then it would express the following - the bounding boxes corresponding to the right hands of the two interacting humans are at first far away, then they come closer and overlap, then finally they move away; the rest of the bounding

boxes remain away from each other. In contrast to the single bounding box based description, this description includes the important interaction details involving the hands of the interacting human bodies. However, such a part-based model of the human body mostly view body parts as independent entities; this is counter-intuitive to the notion of body-parts being *part* of a *whole* body.

In this chapter, we investigate the importance of abstracting human bodies as extended objects for human activity recognition in video. In the context of HAR, we define *extended objects* as a set of components, such that each component is approximated by an *axis-aligned minimum bounding rectangle*. We develop a framework for effectively representing spatial relations between extended objects, and efficiently obtaining such relations using a generic algorithm.

This chapter defines *extended objects* and discusses the shortfalls of existing models for extended objects. The Extended CORE9 framework is presented for computing topological, qualitative direction, and qualitative distance relations between a pair of extended objects. Furthermore, the results of experiments conducted using Extended CORE9 is reported along with a discussion on their significance.

Experiments are performed to show that a representation that abstracts human bodies as extended objects may lead to better representation of human activity within a video. In the experiments, we compare representations of video activities using simple bounding box abstraction of human bodies and an extended object representation of video activities. For each type of abstraction, we extract qualitative relations between human bodies and objects involved in the activity. The relations are treated as words in a bag-of-words representation of the activities. Experiments are performed using four classifiers - KNN, SVM, Naive Bayesian, and Deep Learning with a basic architecture. These experiments are conducted to show that a better classification of activities would be obtained if relations obtained using Extended CORE9 rather than relations obtained using CORE9.

3.2 Extended Objects

In literature, extended objects, which can be seen as disconnected regions, have been discussed. Extended objects have been referred to as *complex objects*, *disjoint objects*, *multi-component objects* and *composite regions* [16, 17, 75] (see Section 2.2.4 of Chapter 2). In this thesis, we define an *extended object* as a set of *axis-aligned rectangles*, where each rectangle is an approximation of a component. A *component*, in this context, is a part of some entity that can have an individual

identity. For example, the *hand* is a component of the *human body*; such components can be approximated using individual bounding rectangles. Intuitively, more the number of such bounding rectangles used to abstract a body, the more precise is the abstraction. For example, if the human body is abstracted considering the head, torso, forearm, arm, fingers, thighs, legs and feet as separate components then a more precise description of the human body is obtained compared to an abstraction that uses only head, torso, hands and legs as components.

Definition 3.1. *An extended object A consisting of m components is defined as,*

$$A = \{a_i | a_i \text{ is an axis-aligned rectangle bounding a component of } A, 1 \leq i \leq m\}$$

In any video frame there is only a finite set of objects that can be detected. Of these detected objects certain objects, such as human bodies, can be decomposed into a set of components. Each set of components describing an entity (such as a human body) can be considered an *extended object*. Any entity or object in the video frame, which can not be decomposed into its components can be considered an extended objects comprising of a singleton set of one component. Given that this thesis addresses HAR from video using extended objects, the following observation is crucial to establish the completeness of the extended object based abstraction.

Observation 1. *Given any video frame, all objects of interest can be seen as extended objects.*

3.2.1 CORE9 and Desiderata for Extended CORE9

CORE9 is a compact representation for various spatial aspects of a pair of rectangle objects [13] that was designed considering the requirements and restrictions of video analysis (see Section 2.3 of Chapter 2). As shown in Figure 2-5, objects are assumed to be single-piece, axis-aligned rectangles in CORE9. On the other hand, by definition, extended objects have multiple components. There can be two ways to handle extended objects using CORE9.

a. Approximate the whole object using a single axis-aligned rectangle: We write this as CORE9$_w$. The problem with CORE9$_w$ is it cannot distinguish between configurations shown in Figure 3-1. This is because all components of A are abstracted away as a single rectangle M_A and all components of B are abstracted as M_B.

b. Treat components as individual single-component entities approximated using separate axis-aligned rectangles; We write this as CORE9$_c$. The problem with CORE9$_c$ is it fails to recognize the relation between entities A and B as a whole. Additionally, it computes all ${}^{m+n}C_2$ relations (where m and n are the number of components in A and B respectively). All relations between components of the same object are included despite being relatively uninteresting, especially for human *interactions*. For example, in the *handshaking* activity shown in Figure 1-1, it is interesting to note the sequence of relations between one person's hand with the other person's hand rather than the relations between the person's hand and his/her own hand.

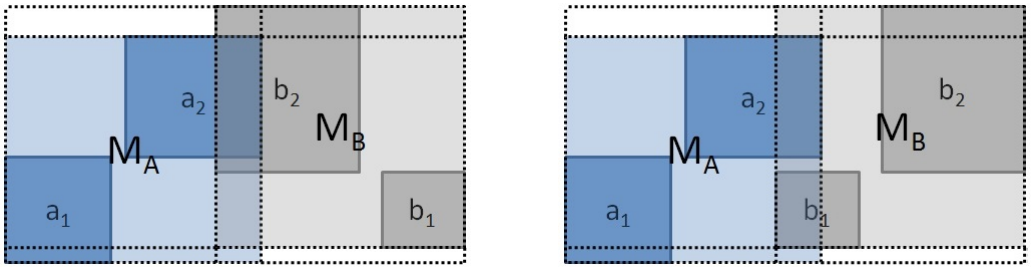


Figure 3-1: Two configurations that would be indistinguishable to CORE9$_w$

Due to the difficulties of representing extended objects and computing the relations between them efficiently using CORE9, we propose Extended CORE9. Within Extended CORE9, the relation between a pair of extended objects is defined using *component relations* and *whole relations*. Further we show that a recursive algorithm using geometric reasoning can reduce the number of computations while extracting qualitative spatial relations of two extended objects. Extended CORE9 is capable of computing *topological*, *directional* and *distance* relations between a pair of *extended objects*.

3.3 Extended CORE9

To express binary spatial relation between two extended objects, two types of relations are used: *component* relations and *whole* relations. This is close to the two-level description of topological relations for *composite* regions introduced in [16].

- *Component* relations are the relations between components of one extended object and the components of the other extended object, i.e. inter-object

component-component relations. This is similar to the *detailed level* relations defined in [16].

- *Whole* relation is the general relationship between the extended objects, computed using the *component* relations. This is similar to the *coarse level* relations in [16].

However, the work described in [16] is a formal model for expressing topological relations between multi-component composite regions where each component is a closed connected point set. While the formulation in this thesis follows a similar model, we further propose methods for efficiently obtaining not only topological but also qualitative directional and distance relations between a pair of extended objects. We assume each component of the extended object is an axis-aligned rectangle. This is because most tracking information is given as coordinates of rectangles within a video frame. More importantly, such an assumption allows us to use geometric reasoning for opportunistically inferring multiple qualitative relations with fewer computations. Axis-aligned rectangles also require simpler geometric computations for finding *overlap* and *intersection* of components, compared to other enclosing structures such as, *circle* or *oriented rectangles*. For the sake of convenience, hereafter the term *rectangle* will be used to mean *axis-aligned rectangle*, unless stated otherwise.

Let us consider a pair of extended objects, say A and B. A has m components that are abstracted using rectangles, $a_1, a_2, ..., a_m$. Similarly B has n components that are abstracted using $b_1, b_2, ..., b_n$. To extract binary spatial information between A and B, we first obtain the *minimum bounding rectangles* (MBRs) of A and B. The MBR of a set of rectangles, MBR$(a_1, a_2, ..., a_n)$, is defined as the rectangle with the smallest area enclosing all the rectangles. In Figure 3-1, M_A is MBR(A), where extended object $A = \{a_1, a_2\}$; similarly M_B is MBR(B).

In Extended CORE9, we obtain cores for MBRs of extended objects A and B, in a manner similar to CORE9 [13] (see Section 2.3 of Chapter 2). The nine cores for the extended objects are obtained by extending the endpoints of the corresponding MBRs in both directions parallel to the axes (as shown in Figure 3-1). However for extended objects, instead of the *state information* defined in CORE9, we compute an *extended state information* that allows computation of several component relations at a time; a detailed discussion of this is given in Section 3.3.1. The *extended state information* tells us which components of the extended objects A and B overlap a core and is defined as follows:

Definition 3.2. *The* ***extended state information for core$_{xy}$(A, B)****,* $x, y \in \{1, 2, 3\}$ *of MBR(A) and MBR(B), is denoted by* $\sigma_{xy}(A, B)$ *and is computed as follows,*

$$\sigma_{xy}(A,B) = \begin{cases} \{a_i | a_i \cap core_{xy}(A,B) \neq \phi\} \cup & \\ \qquad \{b_j | b_j \cap core_{xy}(A,B) \neq \phi\} & \text{if } core_{xy}(A,B) \neq NULL \\ \square & \text{if } core_{xy}(A,B) = NULL \end{cases} \tag{3.1}$$

$\sigma_{xy}(A, B)$ *is* ϕ *if no components of A or B intersect with* $core_{xy}(A, B)$*.*

Definition 3.3. *The* ***extended state information of a pair of objects****, A and B, is denoted by* $\sigma(A, B)$ *and is computed as follows,*

$$\sigma(A,B) = \begin{bmatrix} \sigma_{1,3}(A,B) & \sigma_{2,3}(A,B) & \sigma_{3,3}(A,B) \\ \sigma_{1,2}(A,B) & \sigma_{2,2}(A,B) & \sigma_{3,2}(A,B) \\ \sigma_{1,1}(A,B) & \sigma_{2,1}(A,B) & \sigma_{3,1}(A,B) \end{bmatrix} \tag{3.2}$$

3.3.1 Component Relations

The *extended state information* is designed to allow inference of relation $R(a_i, b_j)$ without CORE9 computation where possible; such inference is possible if a_i and b_j are not in the same core. Lemma 3.1 proves that if a_i and b_j are in the same core then it would only be in $core_{22}$. It follows from Lemma 3.1 that in order to determine whether two components a_i and b_j are in the same core, it is sufficient to look at the state of $core_{22}$ i.e. σ_{22}. It is to be noted here that default relations can be inferred for all components that are not in σ_{22}; a detailed discussion of such inference is given in Section 3.3.1.1.

Lemma 3.1. *Given two extended objects and the* extended state information*, the state of the central core, i.e.* σ_{22}*, determines whether two components* a_i *and* b_j *are in the same core.*

Proof. Let A and B be two extended objects with m and n components respectively. For any two extended objects the *extended state information* consists of the states of nine cores formed using boundaries of MBR(A) and MBR(B); we refer to this as the *core grid*. To prove that, in order to determine whether any two components a_i and b_j are in the same core it is sufficient to examine the state of the central core, σ_{22}. We show that *if any component (say* b_j*) is not in* σ_{22} *then*

it is not possible for that the component to be in the same core as a component of the other extended object (say a_i).

For the proof we consider the possible cases how the nine cores may be formed by MBR(A) and MBR(B). There are three possible cases how the two opposite boundaries parallel to the X-axis may be formed; similarly there are three cases for the opposite boundaries parallel to the Y-axis. Herein, we give the proof only with respect to boundaries parallel to the X-axis as similar arguments can be given for boundaries parallel to the Y-axis. The three possible cases are:

(a) MBR(A) and MBR(B) coincide and define at least one of the two opposite boundaries of the core grid.

(b) Both the opposite boundaries of the core grid are defined by either MBR(A) or MBR(B) (but not both).

(c) One of the opposite boundaries of the core grid is defined by MBR(A) and the other MBR(B).

The three cases are illustrated in Figure 3-2. Let us consider components a_i and b_j such that $b_j \notin \sigma_{22}$ and $b_j \in \sigma_{p3}$ where $p \in \{1, 2, 3\}$ i.e., b_j is in either of the upper cores. For a_i and b_j to be in the same core, $a_i \in \sigma_{p3}$.

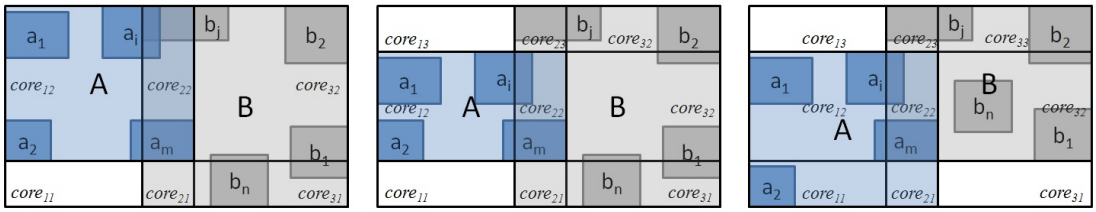


Figure 3-2: The three possible cases for core boundaries parallel to the X-axis

From Figure 3-2, it is clear that for the first case the area covered by $core_{13}, core_{23}$ and $core_{33}$ is zero. Therefore it is impossible for a_i or b_j to be in any of these cores. For the second and third cases, there can be two further possibilities for a_i:

1. a_i defines a boundary of the MBR(A).

2. a_i does not define any boundary of the MBR(A).

Assuming that a_i defines the upper boundary of MBR(A), a_i would also define the upper boundary of the middle cores, $core_{12}, core_{22}$ and $core_{32}$, i.e. $a_i \in \sigma_{q2}$ where $q \in \{1, 2, 3\}$ (see Figure 3-2). It would appear that if only a_i were to move

slightly upwards it would be in σ_{q3} and if $p = q$, then $a_i, b_j \in \sigma_{q3}$. But by moving a_i upwards, the upper boundary of $core_{q2}$ also moves upward; therefore $a_i \notin \sigma_{q3}$.

On the other hand if a_i does not define the upper boundary of MBR(A), then some other component, a_k, would define the upper boundary of MBR(A); this in turn would mean that a_k is above a_i. So if $a_i \in \sigma_{q3} \Rightarrow a_k \in \sigma_{q3}$. However, it has already been shown that if a_k defines the upper boundary MBR(A), then $a_k \notin \sigma_{q3}$. Therefore, $a_i \notin \sigma_{q3}$ and a_i and b_j are not in the same core.

Similar arguments can be given for each of the outer cores of the core grid, that if $b_j \notin \sigma_{22}$ then a_i and b_j are never in the same core. □

Given Lemma 3.1 and the fact that default relations can be inferred for all components that are not in σ_{22}, a recursive algorithm is formulated that opportunistically computes the topological, directional and distance relations between a pair of extended objects. For any two extended objects A and B (with m and n components respectively), in the average case, only some components of either object are in σ_{22}, say p components of A and q components of B. At the topmost level of the recursion, the relations corresponding to $m-p+n-q$ components not in σ_{22} are inferred by default. In the next level of recursion, new extended object A' is created using the p components and B' is created using the q components. At this level too, some relations are obtained by default and for the remaining we go down to the next level of recursion. The recursive call is continued until one reaches a level where both objects have one component each or all components of both objects are in the central core. If it is the latter case, all of the remaining relations are computed by pairwise CORE9 computation. The details of the recursion are given in Algorithm 1.

Given Lemma 3.1, it can be shown that there are four distinct possible configurations. The following lemma proves that Extended CORE9 is able to compute qualitative spatial relations for all possible configurations.

Lemma 3.2. *Given any two extended objects, all topological, directional and distance relations between them can be computed using Extended CORE9.*

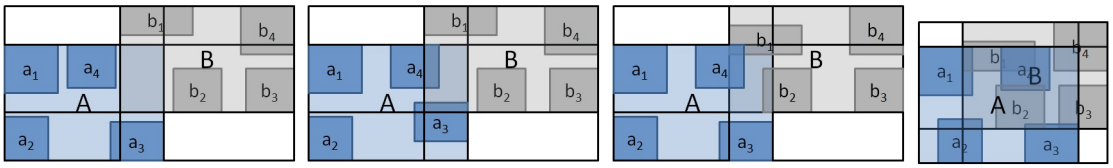
Proof. For a given pair of extended objects, only one of four possible cases may occur, as shown in Figure 3-3. The four cases are as follows:

Case 1. No components in σ_{22}; shown in Figure 3-3a.

Case 2. Components of only one object in σ_{22}; shown in Figure 3-3b.

Case 3. Some components of both objects are in σ_{22}; shown in Figure 3-3c.

Case 4. All components of both objects are in σ_{22}; shown in Figure 3-3d.



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

Figure 3-3: The four possible cases of configurations between a pair of extended objects

The recursive algorithm proposed to compute relations between a pair of extended objects (Algorithm 1) computes all the component relations opportunistically. The four possible cases mentioned above are handled by the algorithm as follows:

- In cases 1 and 2, the topological, direction and distance relations are obtained by default (see Section 3.3.1.1).
- In case 3, some of the component relations are obtained by default, the remaining relations are obtained recursively. From the recursive call, once again we have four cases. The recursion terminates when all relations are either computed by the cases 1, 2 or 4.
- In case 4, none of the component relations are obtained by default. As such, all computations are obtained using pairwise CORE9.

□

For example, to compute topological relations, the four cases are handled as follows: **Case 1.** Since there are no components in σ_{22}, all components are by default topologically *disjoint*. **Case 2.** Since components of only one object are in σ_{22}, again all are by default topologically *disjoint*. **Case 3.** Since there are some components of A and some components of B in σ_{22}, relations for all components not in σ_{22} are by default *disjoint*. For all remaining components, new relations are computed recursively for new extended objects composed of components in σ_{22}. **Case 4.** Since all components of A and B are in σ_{22}, no relations can be inferred by default. As such, all relations are computed using pairwise CORE9 computations.

For components not in σ_{22} topological, directional and distance relations are computed by default as discussed in Section 3.3.1.1. For components that are in σ_{22} relations are computed recursively as discussed in Section 3.3.1.2.

3.3.1.1 For components not in σ_{22}

The topological component relations are expressed as RCC5 relations and directional relations expressed as cardinal directions. Relations are computed using Algorithm 1, that takes advantage of the fact that when two components are completely in different cores, the topological relation between them can be immediately inferred to be *Disjoint* or the DR relation of RCC5. For example, the topological relation between components a_1 and b_1 in Figure 3-1a is DR. This is because the nine cores of the *core grid* are disjoint by geometry. Consequently, any two components that are in two disjoint cores are also disjoint.

The directional relation between two components a_i and b_j that are completely in different cores can be inferred by analyzing the indices of the respective cores. This is possible because the indices of cores refer to their geometric positioning. For example, $core_{11}$ is at the bottom-left corner and $core_{33}$ is at top-right corner of the core grid[1]. Therefore, from the definition of CDC relations, one can infer that if a_i is in $core_{11}$ and b_j is in $core_{33}$ then a_i is *southwest* of b_j or the SW relation of CDC. Table 3.1 gives the complete set of inferences required to compute the default directional relation. In the table, the contents of cell $[p, q]$ is the direction relation that holds between components a_i and b_j if the condition in row p and column q holds. Here, x, y, z, w are the indices of the cores of a_i and b_j, such that a_i is in $core_{xy}$ and b_j is in $core_{zw}$.

	$x < z$	$x = z$	$x > z$
$y > w$	NW	N	NE
$y = w$	W	B	E
$y < w$	SW	S	SE

Table 3.1: Inference of Cardinal Directional Relation for components a_i and b_j

The distance relations for components are expressed as approximate estimates of the qualitative distance relations defined in [72]. We use a subset of the qualitative distance relations: *connected* (C), *strictly close* (SCl), *strictly near* (SN), and *away* (A). Using only the *extended state information* we are able to provide a rough estimate of the distance relations, with the exception of the *connected* relation. We call these rough estimates of distance relations as *roughlyClose*, *roughlyNear*, and *roughlyAway*. and are defined based whether two components belong to geometrically adjacent cores, non-adjacent cores or same cores. Table 3.2 gives a detailed description of the conditions for inferring distance relations.

[1]assuming that the core grid is always in the first quadrant of the reference frame

roughlyAway	a_i and b_j are in completely different cores and none of the cores containing a_i is adjacent to any of the cores containing b_j
roughlyNear	a_i and b_j are in completely different cores and at least one of the cores containing a_i is adjacent to one of the cores containing b_j
roughlyClose	When a_i and b_j are in the same core, the algorithm is called recursively as described in Algorithm 1. When the recursion terminates, if the topological relation between them is found to be DR, then the distance relation is *roughlyClose*.
Connected	If the topological relation between a_i and b_j them is not DR.

Table 3.2: Inference of Distance Relation for components a_i and b_j

It is worth noting that CORE9 uses relative sizes of cores to measure the *relative closeness* of two rectangular objects. Depending on whether the size of a core is increasing or decreasing CORE9 infers whether the objects are moving closer or further away [13]. In contrast, we use an approach that takes into account whether two components are in adjacent cores or not and at which level of the recursion, to determine a qualitative distance relation.

3.3.1.2 For components in σ_{22}

For components that are not in σ_{22} all component relations can be computed by default. However, for components that are in σ_{22}, the relations can be computed using the recursive algorithm as detailed in Algorithm 1.

The whole relations between extended objects A and B can be inferred by looking at the component relations collectively. The computation of whole relations from component relation is discussed in the following section.

3.3.2 Whole-Relations

Whole relations between the extended objects are derived from the component relations computed using the recursive algorithm. Given the set of topological component relations for extended objects A and B, the topological *whole* relation between A and B is obtained as the most general relation from the subsumption lattice of RCC [70] (see Section 2.4.2 of Chapter 2). Similarly, the directional *whole* relation between A and B is the most general directional relation. This is computed by assuming A and B to be extended objects with single components, MBR(A) and MBR(B) respectively. Extended CORE9 computation degenerates to CORE9 computation for single-component objects; the directional *whole* rela-

Algorithm 1: *boolean* $Rel(\sigma(A,B))$, Algorithm to find $R(a_i,b_j) \forall i \in \{1...m\}, \forall j \in \{1..n\}$

Input: $\sigma(A,B)$
Output: *boolean*
begin
  if $\forall core_{xy}(A,B),\ \sigma_{xy}(A,B) \cap \{a_i,b_j\} = \phi$ **then**
    $R(a_i,b_j) = \Psi$; $\Psi \in \Re$, $\Re$ is a set of spatial relations
  else if $\forall core_{xy}(A,B),\ \exists \sigma_{xy}(A,B) - \{a_i,b_j\} = \phi$ **then**
    Compute $R(a_i,b_j)$ using CORE9 SI
  else if $\forall core_{xy}(A,B), \exists \sigma_{xy}(A,B) - \{a_i,b_j\} = \{a_{i_1},...a_{i_k},b_{j_1},...b_{j_k}\} \neq \phi$ **then**
    $A' = MBR(a_{i_1},a_{i_2},...a_{i_k})$
    $B' = MBR(b_{j_1},b_{j_2},...b_{j_k})$
    $newr \leftarrow Rel(\sigma(A',B'))$; Recursively find relations using ESI
    if $newr = FALSE$ **then**
      forall $i \in \{i_1,i_2,...i_k\}$ & $j \in \{j_1,j_2,...j_k\}$ **do**
        Compute $R(a_i,b_j)$ using CORE9 SI
  else
    return $FALSE$; no new relations are computed
  return $TRUE$; at least one new relation is computed

tion is thus computed using CORE9 for the MBRs of A and B.

On the other hand, the qualitative distance *whole* relation is the component relation describing the closest components. Table 3.3 is used to infer the distance whole relation. In the table, the contents of a cell [p,q] indicate whether the q^{th} relation is in the set of distance component relations for the p^{th} whole relation to be inferred. The contents of the cell [p,q] can be either of the three value: *true*, $false$ and $\times$. If the contents of cell [p,q] is *true*, then it indicates *there is at least one* component relations of the type corresponding to the q^{th} column. Similarly, $false$ and $\times$ indicate *there are no* and *there may or may not be* a component relation of the type corresponding to the q^{th} column, respectively.

Component Relation	*Connected*	*roughlyClose*	*roughlyNear*	*roughlyAway*
Connected	*true*	$\times$	$\times$	$\times$
roughlyClose	$false$	*true*	$\times$	$\times$
roughlyNear	$false$	$false$	*true*	$\times$
roughlyAway	$false$	$false$	$false$	*true*

Table 3.3: Computing whole relation for the distance relations

From our discussions thus far, we are in a position to compute component relations and whole relations for a given pair of extended objects. The set of

relations computed using Extended CORE9 for a pair of extended objects include the set of component relations and the whole relation. In the following subsection, the complete process of extracting spatial relations between a pair of extended objects using Extended CORE9 is illustrated with the help of an example.

3.3.3 An Illustrative Example

Consider the extended objects A and B ($A = a_1 \cup a_2 \cup a_3$ and $B = b_1 \cup b_2 \cup b_3$) as shown in Figure 3-4.

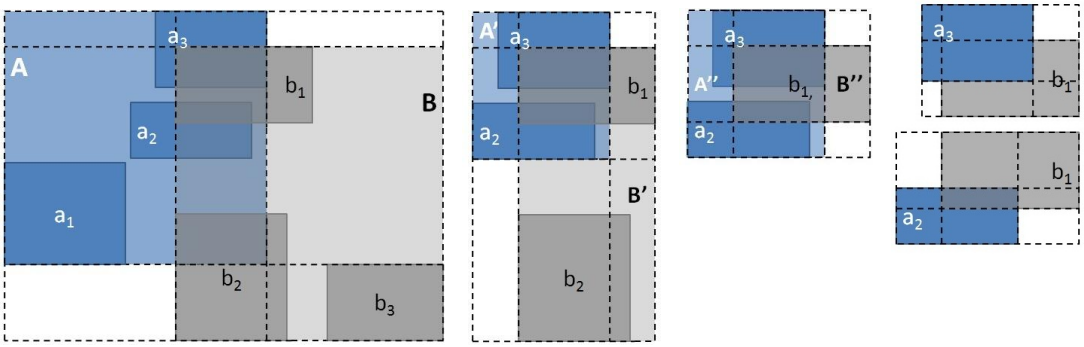


Figure 3-4: [Illustrative example: The objects in the first three levels of recursion and the base case.

In the highest level of recursion, i.e. level 0 in Figure 3-5, ESI of A and B will be:

$$\sigma(A,B) = \begin{bmatrix} \{a_3\} & \{a_3\} & \phi \\ \{a_1,a_2,a_3\} & \{a_2,a_3,b_1,b_2\} & \{b_1,b_2\} \\ \phi & \{b_2\} & \{b_2,b_3\} \end{bmatrix}$$

From this ESI, we can infer $R(a_1,b_1)$, $R(a_1,b_2)$, $R(a_1,b_3)$, $R(a_2,b_3)$, $R(a_3,b_3)$ are DC. Rest of the relations are recursively obtained from new objects $A' = a_2 \cup a_3$ and $B' = b_1 \cup b_2$ (where $a_2,a_3,b_1,b_2 \in core_{22}(A,B)$) as shown in Figure 3-4; this happens at level 1 in Figure 3-5. The ESI of A' and B' will be:

$$\sigma(A',B') = \begin{bmatrix} \{a_3\} & \{a_3\} & \phi \\ \{a_2,a_3\} & \{a_2,a_3,b_1\} & \{b_1\} \\ \phi & \{b_2\} & \{b_2\} \end{bmatrix}$$

From this ESI, we further infer that $R(a_2,b_2), R(a_3,b_2)$ are DC. For the rest of the relations we recursively compute $A'' = a_2 \cup a_3$ and $B'' = b_2$ (where $a_2,a_3,b_2 \in$

$core_{22}(A', B')$) as shown in Figure 3-4; this is level 2 in Figure 3-5.

$$\sigma(A'', B'') = \begin{bmatrix} \{a_3\} & \{a_3\} & \phi \\ \{a_2, a_3\} & \{a_2, a_3, b_1\} & \{b_1\} \\ \{a_2\} & \{a_2\} & \phi \end{bmatrix}$$

At this stage, no new information is obtained using the ESI; hence CORE9 state information is used to infer $R(a_3, b_1)$ and $R(a_2, b_1)$ as PO. This is the base case and level 3 in Figure 3-5.

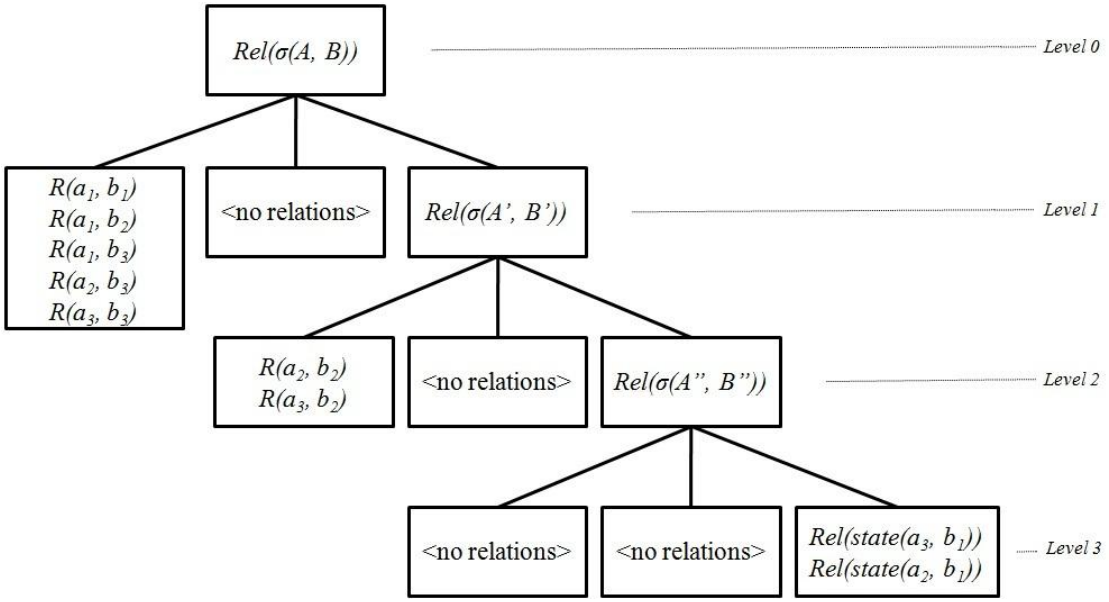


Figure 3-5: Illustrative example: The tree of recursive calls by Algorithm 1 on A and B of Figure 3-4

3.3.4 Theoretical Analysis

Given any video, a spatial representation of the interactions can be obtained using Extended CORE9 by abstracting all entities as extended objects. This follows directly from Observation 1 and Lemma 3.2. All entities in a video frame can be abstracted as *extended objects*, and thereafter Extended CORE9 can be used to compute topological, direction, and distance relations between any pair of extended objects. Extended CORE9 uses geometric reasoning to minimize the number of computations where possible. For any two extended objects A and B (with m and n components respectively) those components that are not in σ_{22} are inferred by default. If say, only p components of A and q components of B are in σ_{22} then component relations of $m - p + n - q$ are inferred by default. Therefore, even though the total number of component relations between A and B is mn,

the number of *core grid* and *extended state information* computations required to obtain the mn relations is much less than that. The following theorem proves that the average case complexity of Extended CORE9 is $O(n\ log\ n)$.

Theorem 3.1. *The average case complexity of the recursive algorithm of Extended CORE9 is $O(n\ log\ n)$, where n is the maximum number of components in either of the extended objects.*

Proof. Let the two extended objects be A and B such that A has m components and B has n components.
Best Case: From Lemma 3.2, the best case for the recursive algorithm when either there are no components in the central core or components of only object are in the central core. In such a case, all of the relations can be inferred by default. In other words, the best case time complexity is $O(1)$.
Worst Case: The worst case is when all the components are in the central core. In such a case, all of the relations has to be obtained using pairwise CORE9 computations. Therefore in the worst case, the time complexity is $\Theta(mn)$, where m and n are the number of components in each of the two extended objects.
Average Case: In the average case, only some components of each extended object are in the central core. At each level of the core, $m+n$ components of the extended objects are split into $p+q$ and $m-p+n-q$ components, such that p components of A and q components of B are in the central core.

Recursion is applied on the p components of A and q components of B by creating new extended objects A' and B'; this can be done in $O(p+q)$ time. Since the number of ways m, n can be split into p, q and $m-p, n-q$, is $(m-1)(n-1)$. Thus the recurrence equation can be written as,

$$T(m,n) = \frac{1}{(m-1)(n-1)} \sum_{p=1,q=1}^{m,n} T(p,q) + O(m+n) \tag{3.3}$$

To solve the above recurrence by substitution, we consider $T(m,n) = (m+n)log(m+n)$. Therefore,

$$\begin{aligned}\sum_{p=1,q=1}^{m,n} T(p,q) &= \sum_{p=1,q=1}^{m,n} (p+q)log(p+q) \\ &\leq \frac{(m+n)^2(m+n+2)}{4} log(m+n)\end{aligned}$$

Substituting in Equation 3.3 we obtain,

$$T(m,n) \approx O(m+n)log(m+n)$$
$$\approx O(n\ log\ n)\ \text{, where } n > m$$

□

For extended objects, A and B (m and n components respectively), CORE9 could use either of the two variants CORE9$_w$ and CORE9$_c$ for representation. In CORE9$_c$, the number of computations is quadratic in the total number of components of A and B, i.e. O($(m+n)^2$). Note that CORE9$_w$ requires constant number of computations, this is at the expense of information loss (as detailed in Section 3.2.1).

3.4 Experimental Evaluation

The effectiveness of the proposed representation schema relies on how effectively it can describe an activity so that better interaction models can be obtained for classification. This in turn is reflected in the classification results for the activities. We describe activities as a bag-of-words, where the words are the qualitative relations obtained using Extended CORE9. Experiments are conducted using four different classifiers with such a representation. The details of the experimental setup are discussed in the following section.

3.4.1 Experimental Setup

We extract the qualitative topological, directional and distance relations amongst the interacting entities for each activity sequence using Extended CORE9. As discussed previously in Section 3.3, the relations computed by Extended CORE9 between a pair of extended objects, comprises of the set of *component* relations as well as the set of *whole* relations. However, for the purpose of our experiments we consider the following variants -

- Relations for a pair of extended objects is the set of component relations and the whole relation; we write this as ExtCORE9$_{cw}$
- Relations for a pair of extended objects is the component relations without the whole-relation; we write this as ExtCORE9$_c$

- Relations for a pair of extended objects is only the whole-relation; we write this as ExtCORE9$_w$.

The qualitative relations obtained using Extended CORE9 are treated as a bag of words describing the activity within a video. Each video is treated as a document and the activity classes as topics or categories that the classifier is to model. A detailed description of how a video is converted to Extended CORE9 bag of words can be found in Appendix A. The aim of our experiments is to evaluate the representation schema Extended CORE9. To do so, we believe a bag-of-words based approach is sufficient. The bag words description is converted into a word vector before applying the classifier. We have conducted experiments using four classifiers namely, *K-Nearest Neighbour Classifier* (KNN), *Naive Bayes Classifier*, *Support Vector Machines* (SVM) and *Deep Learning*. It is to be noted that, research towards better classification approaches that best utilizes Extended CORE9 representation schema is not the aim of the experiments discussed in this chapter.

Further, for all the variants of Extended CORE9 discussed above, we have separate bag-of-words description for each video. Similarly we also have different bag-of-words description for CORE9$_c$ and CORE9$_w$ (see Sec 3.2.1). Therefore, we conduct similar experiments and compare the results for all variants of Extended CORE9 and CORE9.

3.4.2 Experimental Results

We have experimented on short video sequences from the UT-Interaction dataset [3], the Mind's Eye dataset and SBU Kinect Interaction dataset [117]. For our experiments on the UT-Interaction dataset, we use 50 videos for five activities - *handshaking, hugging, kicking, punching* and *pushing* - that are interactions involving more than one human bodies. For the Mind's Eye dataset we consider 110 videos for 11 activities from the dataset - *approach, carry, catch, collide, drop, follow, hold, kick, pickup, push* and *throw*. For both UT-Interaction and Mind's Eye dataset, we use keyframes of the videos [2] and manually label the humans and objects involved in each of the keyframes. The SBU Kinect Interaction dataset consists of eight activities over 282 videos. For this dataset, we use the available skeleton tracks to obtain the extended object representation.

We evaluate ExtCORE9$_w$, ExtCORE9$_c$, and ExtCORE9$_{cw}$ on the three datasets using different classifiers. We obtain topological, directional and distance fea-

[2]We use I-frames obtained using the tool *ffmpeg* as keyframes, www.ffmpeg.org

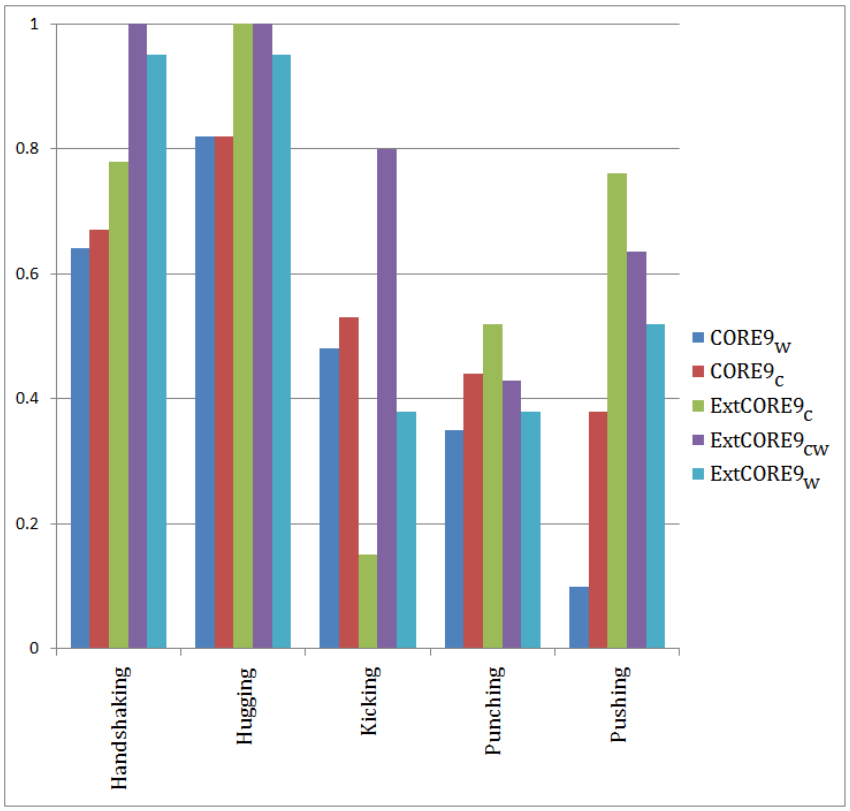


Figure 3-6: Comparison of F1-scores on UT Interaction Dataset

tures using (a) CORE9$_c$ (b) CORE9$_w$ (c) ExtCORE9$_c$ (d) ExtCORE9$_{cw}$ and (e) ExtCORE9$_w$. The experiments described above are performed on relations obtained using each of the five representation schemes. The precision, recall and f1-score for each activity class as well as the overall classification accuracies are computed. A comparison of f1-scores for all the variants of Extended CORE9 and CORE9 is shown in Figures 3-6, 3-7, and 3-8,. Tables 3.4, 3.5 and 3.6 give a comparison of the precision, recall, and f1-score of ExtCORE9$_{cw}$ with the four classifiers mentioned above. A comparison of the classification accuracies for CORE9 and Extended CORE9 is given in Table 3.7. A comparison of the classification accuracies for our approach with those in literature is given in Table 3.8.

Activity	KNN			SVM			Naive Bayes			Deep Learning		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Handshaking (10)	0.9	0.9	0.9	1	0.9	0.95	1	1	1	1	1	1
Hugging (10)	1	1	1	1	1	1	1	1	1	1	1	1
Kicking (10)	0.3	0.4	0.36	0.47	0.7	0.6	0.8	0.4	0.53	0.5	0.5	0.5
Punching (10)	0.39	0.5	0.44	0.5	0.2	0.3	0.43	0.6	0.5	0.33	0.5	0.4
Pushing (10)	0.6	0.3	0.4	0.74	0.9	0.82	0.64	0.7	0.67	1	0.5	0.67

Table 3.4: Results for Extended CORE9 on UT Interaction dataset

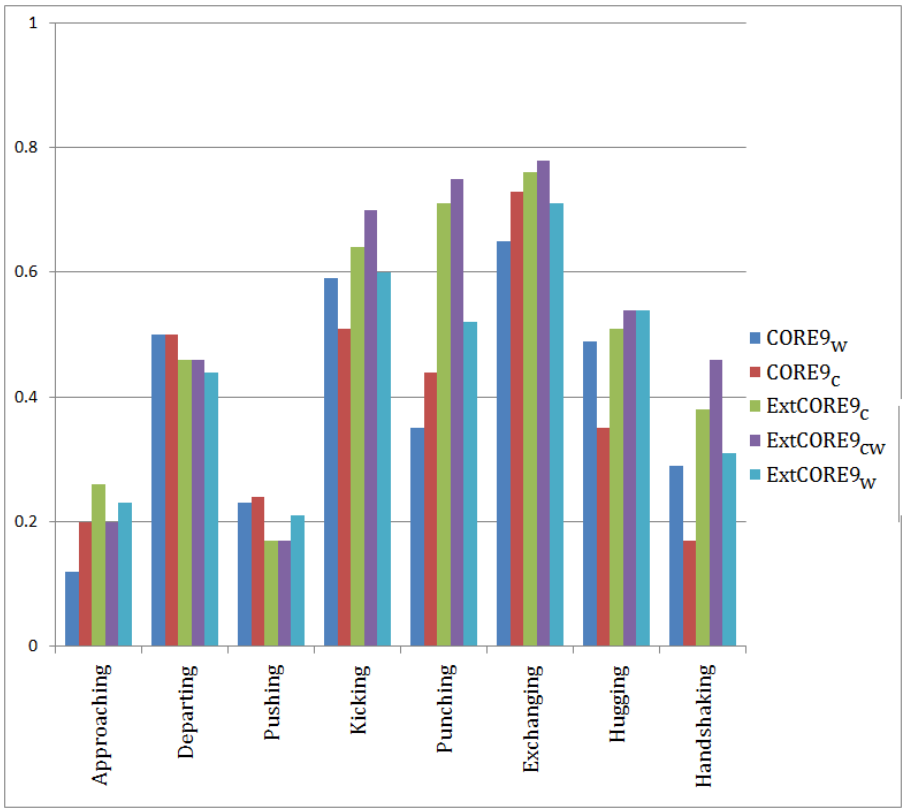


Figure 3-7: Comparison of F1-scores on SBU Kinect Interaction Dataset

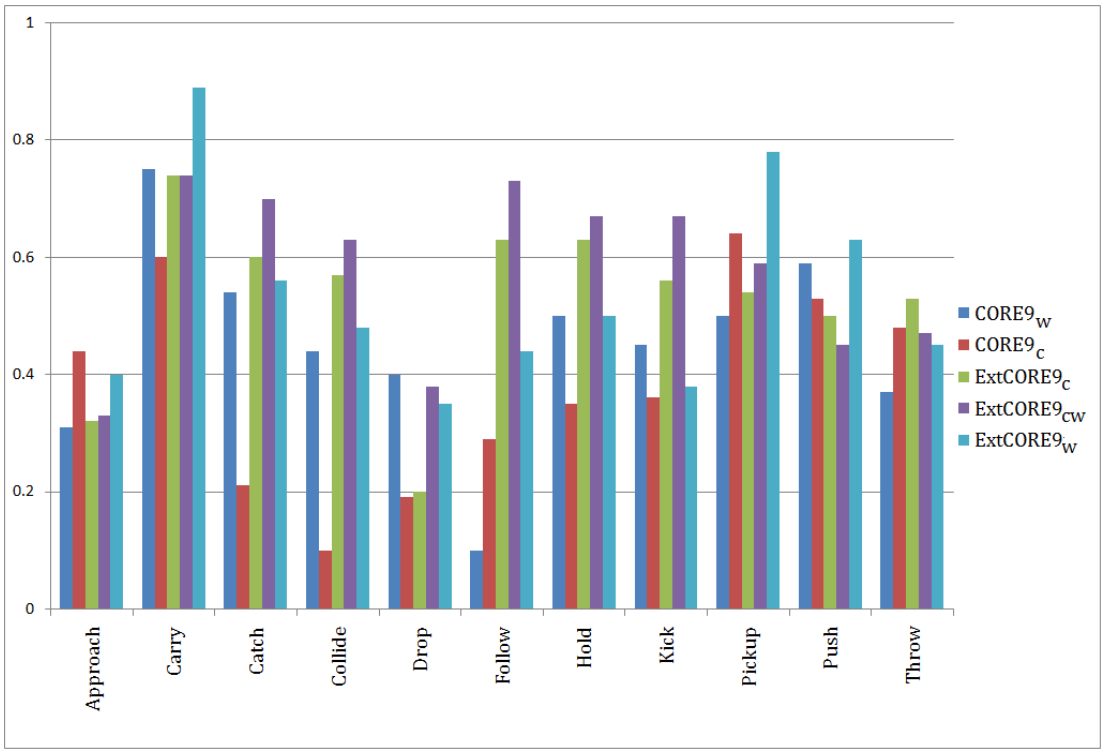


Figure 3-8: Comparison of F1-scores on Mind's Eye Dataset

Activity	KNN			SVM			Naive Bayes			Deep Learning		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Approach (10)	0.37	0.3	0.33	0.37	0.3	0.33	0.42	0.5	0.45	0.37	0.3	0.33
Carry (10)	0.78	0.7	0.74	0.87	0.7	0.78	0.71	1	0.833	0.78	0.7	0.74
Catch (10)	0.7	0.7	0.7	0.58	0.7	0.64	0.89	0.8	0.84	0.62	0.56	0.52
Collide (10)	0.67	0.6	0.63	0.57	0.8	0.67	0.33	0.4	0.36	0.33	0.4	0.36
Drop (10)	0.36	0.4	0.38	0.25	0.1	0.14	0.43	0.3	0.35	0.67	0.6	0.63
Follow (10)	0.67	0.8	0.73	0.8	0.8	0.8	0.78	0.7	0.74	0.73	0.8	0.76
Hold (10)	0.64	0.7	0.67	0.56	0.5	0.53	0.4	0.2	0.27	0.54	0.6	0.57
Kick (10)	0.57	0.8	0.67	0.67	0.6	0.63	0.71	0.5	0.59	0.42	0.5	0.45
Pickup (10)	0.71	0.5	0.59	0.4	0.6	0.48	0.47	0.7	0.56	0.58	0.7	0.64
Push (10)	0.42	0.5	0.45	0.54	0.6	0.57	0.5	0.6	0.54	0.78	0.7	0.74
Throw (10)	0.57	0.4	0.47	0.4	0.4	0.4	0.5	0.4	0.44	0.5	0.5	0.5

Table 3.5: Results for Extended CORE9 on Mind's Eye dataset

Activity	KNN			SVM			Naive Bayes			Deep Learning		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Approaching (42)	0.24	0.36	0.29	0.2	0.14	0.17	0.34	0.14	0.2	0.23	0.12	0.16
Departing (43)	0.48	0.46	0.47	0.4	0.54	0.46	0.42	0.51	0.46	0.52	0.51	0.52
Pushing (40)	0.14	0.1	0.12	0.47	0.2	0.28	0.28	0.12	0.17	0.19	0.15	0.17
Kicking (41)	0.58	0.56	0.57	0.61	0.63	0.62	0.63	0.78	0.7	0.67	0.73	0.7
Punching (18)	0.41	0.39	0.4	0.92	0.61	0.73	0.86	0.67	0.75	0.59	0.56	0.57
Exchanging(21)	0.95	0.9	0.93	0.78	0.86	0.82	0.8	0.76	0.78	0.9	0.86	0.88
Hugging(39)	0.31	0.28	0.3	0.43	0.59	0.49	0.42	0.74	0.54	0.34	0.64	0.46
Handshaking(38)	0.44	0.42	0.43	0.39	0.47	0.43	0.45	0.47	0.46	0.44	0.4	0.42

Table 3.6: Results for Extended CORE9 on SBU Kinect Interaction dataset

Method	UT Interaction	Mind's Eye	SBU Kinect Interaction
CORE9 + KNN	40%	45.45%	30.14%
CORE9 + SVM	44%	38.18%	36.87%
CORE9 + Naive Bayes	42%	43.63%	41%
CORE9 + Deep Learning	48%	47.27%	40%
Extended CORE9 + KNN	62%	58.18%	40.78%
Extended CORE9 + SVM	74%	55.45%	47.16%
Extended CORE9 + Naive Bayes	74%	55.45%	49.64%
Extended CORE9 + Deep Learning	64%	57.27%	46.45%

Table 3.7: Comparison of classification accuracies on three datasets

Method	UT Interaction	Mind's Eye	SBU Kinect Interaction
Extended CORE9	74%	58.18%	49.64%
Angled CORE9 + LDA [41][a]	-	64.4%	-
BoW + SVM [118]	77%	-	-
Skeleton + Deep LSTM [58]	-	-	86.03%

Table 3.8: Comparison of classification accuracies with other approaches in literature

[a] In [41] only 5 activities are considered; in this thesis 11 activities of the dataset are considered.

3.4.3 Discussion

In our experiments we have considered a total of 24 activities from the UT Interaction dataset, the Mind's Eye dataset and the SBU Kinect Interaction dataset. For most of these activities, we obtain higher f-scores when using the features sets provided by ExtCORE9$_c$, ExtCORE9$_{cw}$, and ExtCORE9$_w$ than when using the feature set of CORE9$_w$. This is not true for only certain activities such as *Departing*, *Drop* and *Push*. Thus, we can say that the qualitative features obtained using the Extended CORE9 variations provide a better feature set in comparison to those obtained from CORE9$_w$. This goes on to strengthen the intuition that the relations amongst body-parts provide better classification information compared to the relation of the entities as a whole when it comes to recognizing human-activities in video.

A comparison of the f-scores when using CORE9$_c$, ExtCORE9$_c$ and ExtCORE9$_{cw}$ shows that for activities where several components (body-parts here) of one entity interact closely with the other entity, both ExtCORE9$_{cw}$ and ExtCORE9$_c$ perform better, compared to CORE9$_c$. Activities such as *handshaking, hugging, kicking, punching, pushing* from UT Interaction, *carry, catch, collide, drop, follow, hold, kick, throw* from Mind's Eye dataset, and *kicking, punching, exchanging, hugging, handshaking* from SBU Kinect Interaction dataset belong to this category. This is because ExtCORE9$_c$ and ExtCORE9$_{cw}$ captures the more distinctive inter-entity component relations. On the other hand, CORE9$_c$ includes several unimportant intra-entity component relations, that only adds to the confusion, bringing down the precision and recall values significantly. All of the above serves to strengthen our hypothesis that for human activity recognition, treating human bodies as extended objects and considering the relations amongst these extended objects might lead to better results.

For activities in which the relational change between the entities and their components is minimal, CORE9$_c$ gives higher f1-scores. Activities such as *approach, pickup, push* in Mind's Eye dataset, and *approaching, departing, pushing* in the SBU Kinect Interaction dataset belong to this category. This is because, in such activities, the intra-entity component relations play an important role, in distinguishing one activity from another.

We have also performed experiments using relations obtained using ExtCORE9$_w$ and have found that for activities that are human-object interactions the Extended CORE9 whole relation alone serves as a distinctive feature. Activities such as *carry, catch, collide, drop, pickup, push* of the Mind's Eye dataset belong to this

category. In the UT-Interaction dataset and SBU Kinect Interaction dataset, where all activities are human-human interactions, ExtCORE9$_w$ provides better distinctive features when a single body part of one entity interact with a single body-part of another entity. This is evident from activities *handshaking, pushing* in the UT Interaction dataset and *kicking, exchanging* activities in SBU Kinect Interaction dataset where ExtCORE9$_w$ give better f-scores.

From above analysis we can come to the conclusion that for certain activities where the body parts of one human-body interact closely with the other human or object, ExtCORE9$_c$ and ExtCORE9$_{cw}$ provides distinctive features. But for activities where the interaction between the body parts is less, a complete set of intra-entity and and inter-entity relations, as provided by CORE9$_c$ leads to better f-scores. For certain human-object interactions where the relations between the entities as a whole is a good feature, ExtCORE9$_w$ provides more distinctive features.

Table. 3.7 shows comparison of the classification accuracies for CORE9 and Extended CORE9 in combination with different classifiers. It has been observed that regardless of the classifier used, the classification accuracy obtained using an Extended CORE9 representation is much higher than compared to a CORE9 representation, for all three datasets. Further, the Naive Bayesian Classifier gives the highest classification accuracy in two out of the three datasets. The classification using the deep learning approach show that, even using a simple architecture the results are comparable to the Naive Bayesian Classifier.

3.5 Conclusion

The part-based model of the human body obtained during tracking is easily seen as an extended object; ExtCORE9$_{cw}$ provides an efficient mechanism for obtaining relations between such extended objects. By focusing on component-wise relations and whole relations of these extended objects ExtCORE9$_{cw}$ achieves better interaction models. A recursive algorithm is used to opportunistically extract the qualitative relations using as few computations as possible.

In this chapter, we have presented Extended CORE9 for extracting topological, directional and distance relations between a pair of extended objects, at a given instant of time. The classification results for ExtCORE9$_{cw}$ may be improved by incorporating temporal information of how relations between extended objects evolve over time. In the next chapter we present *temporal activity graphs* for representation of human activities that encodes the temporal evolution of the

spatial relations provided by ExtCORE9$_{cw}$. Further a *temporal activity graph kernel* is presented that allows classification of activities represented as *temporal activity graphs*.