



UNIVERSIDAD AUTÓNOMA DE YUCATÁN

TESIS DE MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

**Plataforma no intrusiva para el reconocimiento
de actividades al interior del hogar de los
adultos mayores**

Autor:
I.B. Cristhian Antonio Carrillo
Polanco

Directores de Tesis:
Dr. Jorge Ricardo Gómez
Montalvo
Dr. Francisco José Moo Mena

*Una tesis presentada para la obtención del grado de
Maestro en Ciencias de la Computación*

Mérida, Yucatán
Julio de 2018

Declaración de Autoría

Yo, I.B. Cristhian Antonio Carrillo Polanco, declaro que esta tesis titulada, «Plataforma no intrusiva para el reconocimiento de actividades al interior del hogar de los adultos mayores» y el trabajo aquí presentado son de mi autoría. Y confirmo que:

- Este trabajo fue realizado durante el período de maestría que cursé en esta Universidad.
- Cualquier parte de esta tesis que haya sido previamente presentada para la obtención de un título en esta Universidad u otra institución, fue mencionado claramente.
- Cualquier consulta realizada de la publicación de terceros, fue atribuido claramente.
- Cualquier cita mencionada en este trabajo cuenta con su referencia bibliográfica. Con la excepción de dichas citas, esta tesis es completamente mi propio trabajo.

«La educación es el arma más poderosa que puedes usar para cambiar al mundo»

Nelson Mandela

Resumen

En esta tesis se presenta el desarrollo de una plataforma para el reconocimiento de actividades al interior del hogar de los adultos mayores, en específico de aquellos que puedan realizar sus labores cotidianas de manera independiente. Las tres pautas que guiaron el desarrollo de la plataforma fueron: 1) establecer los parámetros que permitan caracterizar las actividades; 2) determinar la implementación del reconocimiento (inferencia o identificación) de actividades, es decir, los componentes involucrados y sus respectivas tareas; 3) minimizar la intrusión en la vida de los adultos mayores.

Como parte de las contribuciones de la tesis, se diseñó una arquitectura general para el reconocimiento de actividades. Dicha arquitectura está compuesta por un grupo de sensores, un dispositivo de monitoreo, un servidor Web, y un cliente Web. Los sensores son los encargados de proporcionar la información de la ubicación de los adultos mayores al interior de su hogar; el dispositivo de monitoreo es el encargado de recolectar la información de los sensores, generar más datos que permitan solventar los parámetros necesarios que caracterizan a las actividades, enviar dicha información al servidor, recibir la actividad inferida y realizar una retroalimentación al servidor con base en el criterio del usuario; el servidor se encarga de recibir la información necesaria para realizar la inferencia, la cual se realiza mediante un sub-módulo denominado motor de inferencias, además se encarga de devolver la actividad inferida al dispositivo cliente, recepcionar la retroalimentación del usuario y almacenarla en la base de datos, así como proporcionar la información solicitada por el cliente Web; por último, el cliente Web se encarga de solicitar la información almacenada en la base de datos del servidor, y con base en ella generar gráficas que permitan al cuidador del adulto mayor visualizar de manera amena algunas estadísticas de las actividades realizadas por éste.

Con base en la arquitectura general, se desarrolló OPAIEH, una plataforma basada en ontologías (base de conocimiento) y un algoritmo de consultas (motor de inferencias) para la identificación de actividades al interior del hogar de los adultos mayores. Además se desarrolló de manera paralela una red neuronal para el reconocimiento de actividades al interior del hogar, denominada IAR-NN, con la finalidad de comparar la precisión en la inferencia de actividades realizada por dicha red neuronal con la del motor de inferencias de OPAIEH.

Para cuantificar la precisión del motor de inferencias de OPAIEH y IAR-NN, se solicitó a un usuario que se monitoreara durante 4 semanas; con base en los datos de las actividades registradas se realizaron pruebas para determinar las precisiones en la inferencia de actividades obtenidas por OPAIEH y IAR-NN, las cuales fueron de 96.6 % y 82.43 % respectivamente.

Si bien la precisión en la inferencia de actividades obtenida por OPAIEH es superior a la de IAR-NN, es importante recalcar una característica sobresaliente de las redes neuronales, su generalización. Esta capacidad de generalización se obtiene gracias a los patrones que las redes neuronales aprenden durante su fase de entrenamiento, en IAR-NN esto se traduce en inferencia pura, es decir, generar conocimiento nuevo del ya existente. Esta cualidad de IAR-NN es más sólida que el principio inferencial de OPAIEH, dado que en dicha plataforma la actividad

obtenida es el resultado de realizar consultas sobre una base de conocimiento de actividades registradas por el usuario.

Un punto a considerar en el futuro es realizar las pruebas con una mayor cantidad de datos, ya que en este trabajo de maestría no se tuvo el tiempo suficiente para adquirir un gran volumen de información, es decir, adquirir información durante varios meses, así como de diversos usuarios finales. El poseer una mayor cantidad de datos proporcionará una mayor confianza respecto al comportamiento inferencial tanto de OPAIEH como de IAR-NN.

Agradecimientos

A Dios por guiar y bendecir mi camino.

A mi madre Susana por estar conmigo en todo momento y darme las herramientas que me han permitido llegar tan lejos, sin su apoyo este logro no habría podido ser posible.

A mi novia Azalea por escucharme, aconsejarme y apoyarme, así como ser mi fuente de motivación constante.

A mi hermano Eduardo y mi cuñada Cintya por darme su apoyo cuando más lo he requerido.

A mis asesores, el Dr. Jorge Gómez y el Dr. Francisco Moo, por sus enseñanzas y por su dedicación puesta durante el desarrollo de este proyecto de tesis.

A mis maestros por brindarme valiosos conocimientos.

Al coordinador de la maestría, el Dr. Fernando Curi, por estar ahí para orientarme durante todo el posgrado.

Al Conacyt por darme la oportunidad de estudiar un posgrado.

Índice general

Declaración de Autoría	III
Resumen	VII
Agradecimientos	IX
1. Introducción	1
1.1. Preliminares	1
1.1.1. Plataformas desarrolladas para el reconocimiento de actividades en los entornos internos	2
1.2. Contexto y problemática	4
1.2.1. Definición de actividades	4
1.2.2. Identificación de actividades	4
1.2.3. Minimizar intrusión	4
1.3. Objetivos	5
1.3.1. Objetivo general	5
1.3.2. Objetivos específicos	5
1.4. Contribuciones	5
1.5. Estructura de la tesis	6
2. Arquitectura general de la plataforma	9
2.1. Estructura de la arquitectura	9
2.2. Funcionamiento de los módulos de la arquitectura	9
2.2.1. Sensores	9
2.2.2. Dispositivo cliente o dispositivo de monitoreo	10
2.2.3. Servidor Web	11
2.2.4. Cliente Web	12
2.3. Consideraciones para minimizar la intrusión	12
3. Plataforma basada en ontologías (OPAIEH)	13
3.1. Marco teórico	13
3.1.1. Lenguaje natural y lenguaje formal	13
3.1.2. Representación del conocimiento y lógicas de descripción	15
3.1.3. Ontologías	16
OWL	16

3.1.4.	Trabajos relacionados	17
	Discusión sobre el estado del arte	18
3.2.	Definición de actividad	18
3.3.	Modelo ontológico empleado en OPAIEH	19
3.3.1.	Clases de IAR-ONTO	19
3.3.2.	Individuos de IAR-ONTO	20
3.3.3.	Propiedades de IAR-ONTO	21
3.4.	Diseño de OPAIEH	22
3.4.1.	Sensores	22
3.4.2.	OPAIEHClient	24
	Registro de usuario	25
	Registro de habitaciones y actividades	25
	Monitoreo	25
3.4.3.	OPAIEHServer	28
	Aplicación Web	29
	Base de datos	31
3.4.4.	OPAIEHClient Web	32
	Número de entradas por habitación	33
	Duración promedio de actividades	34
	Trayectoria del usuario entre las habitaciones de su hogar por día	35
	Grafo dirigido de la relación entre las actividades y las habitaciones por día	35
	Línea de tiempo de las actividades	35
3.5.	Consultas de ontología	36
3.5.1.	Caso 1	39
3.5.2.	Caso 2	40
3.5.3.	Caso 3	40
	Primera condición	40
	Segunda condición	41
3.5.4.	Caso 4	41
3.6.	Escenario de prueba, resultados y análisis	43
3.6.1.	Escenario de prueba	43
3.6.2.	Resultados y análisis	44
	Actividad no inferida No. 5	45
	Actividad no inferida No. 43	47
	Actividad no inferida No. 73	47
3.7.	Conclusiones	47
4.	Reconocimiento de actividades basado en redes neuronales	49
4.1.	Marco teórico	49
4.1.1.	Introducción, breve historia y aplicaciones de las redes neuronales	49
4.1.2.	Características de las redes neuronales	51
	Arquitecturas de las redes neuronales	52
	Funciones de activación	53

Algoritmos de aprendizaje	53
4.1.3. Algoritmo backpropagation	55
4.1.4. Entrenamiento y prueba	56
4.1.5. Trabajos relacionados	57
Discusión sobre el estado del arte	58
4.2. Diseño y desarrollo de IAR-NN	58
4.2.1. Diseño de IAR-NN	59
Número de neuronas de entrada	59
Número de capas ocultas	60
Número de neuronas de salida	61
Estructuras de IAR-NN	62
4.2.2. Desarrollo de IAR-NN	63
IAR-NN Training	63
IAR-NN Test	63
4.3. Pruebas experimentales, resultados y análisis	64
4.3.1. Prueba experimental	64
4.3.2. Resultados y análisis	66
4.4. Conclusiones	68
5. Conclusiones	71
A. Diagrama de clases de OPAIEHClient	75
B. Diagrama de clases de OPAIEHServer	77
C. Algoritmo backpropagation	81
Bibliografía	83

Índice de figuras

2.1. Arquitectura general de la plataforma para el reconocimiento de actividades al interior del hogar.	10
2.2. Aplicación de la arquitectura general de la plataforma en el hogar.	11
3.1. Relación de inclusión entre las áreas: inteligencia artificial (IA), representación del conocimiento (KR) y lógicas de descripción (DL).	15
3.2. Diagrama de clases de IAR-ONTO.	20
3.3. Clase DaysofTheWeek y sus individuos creados en tiempo de diseño.	21
3.4. Visualización de los atributos del objeto <i>Sleep</i> de la clase <i>Activity</i>	21
3.5. Propiedades de la ontología IAR-ONTO. De lado izquierdo se muestran las propiedades de objeto y de lado derecho las propiedades de tipo de dato.	22
3.6. Arquitectura de OPAIEH.	23
3.7. Funcionamiento de la señal broadcast de los sensores Beacons y configuración de los IDs al interior del hogar.	24
3.8. Registro de habitaciones. En (a) se muestra el registro de la habitación <i>Bedroom</i> , y en (b) se muestra la interfaz después de registrar varias habitaciones.	26
3.9. Registro de actividades. En (a) se muestra el registro de la actividad <i>Sleep</i> , y en (b) se muestra la interfaz después de registrar varias actividades.	27
3.10. Pantalla de monitoreo de OPAIEHClient antes de enviar los datos al servidor. . .	28
3.11. Pantalla de monitoreo de OPAIEHClient al recibir la inferencia de OPAIEHServer y esta resulta ser correcta.	29
3.12. Pantalla de monitoreo de OPAIEHClient en caso que la actividad inferida no sea correcta, pero la actividad realizada por el usuario se encuentra registrada en la base de datos de OPAIEHClient.	30
3.13. Pantalla de monitoreo de OPAIEHClient en caso que la actividad inferida no sea correcta y tampoco se encuentre registrada en la base de datos.	31
3.14. OPAIEHClientWeb. En (a) se muestra la interfaz de login, y en (b) se muestra la interfaz de usuario.	33
3.15. Gráfica de número de entradas por habitación.	34
3.16. Gráfica de duración promedio de actividades.	34
3.17. Gráfica de la trayectoria del usuario entre las habitaciones de su hogar por día. .	35
3.18. Grafo dirigido de la relación entre las actividades y las habitaciones por día. . .	36
3.19. Línea de tiempo de las actividades.	36
3.20. Consulta con base en un intervalo de hora de inicio (Query StartTime o QST). . .	37

3.21. Consulta con base en un intervalo de duración (Query Duration o QD).	37
3.22. Diagrama de flujo del algoritmo de consultas en IAR-ONTO.	39
3.23. Consulta para conocer el StartTime del resultado de Query Duration (Query Find StartTime o QFST).	41
3.24. Consultas para conocer la actividad con hora de inicio más próxima a la hora de inicio consultada (Query Nearest StarTime o QNST). Las consultas (a) y (b) tienen la misma estructura, con la diferencia que (a) se emplea cuando la hora de inicio consultada se retrocede (Backward), y (b) se emplea para cuando la hora de inicio se adelanta (Forward).	42
3.25. Gráfica de las actividades confirmadas por categoría.	44
3.26. Gráfica de la inferencia por actividades registradas.	45
3.27. Duración promedio de la actividad Take a shower.	46
4.1. Estructura de una neurona artificial.	51
4.2. Redes neuronales unicapa y multicapa.	52
4.3. Redes neuronales parcialmente conectadas y totalmente conectadas.	53
4.4. Redes neuronales feedforward y feedback.	54
4.5. Función identidad.	54
4.6. Tres diferentes tipos de funciones de activación (escalón, sigmoidea, lineal) en formato unipolar y bipolar.	55
4.7. Arquitectura de IAR-NN.	61
4.8. Conversión de las actividades y sus correspondientes parámetros al formato requerido de los vectores de entrada.	62
4.9. Obtención de las características de los vectores de entrada de la tabla de confirmación de actividades de la base de datos de OPAIEHServer.	64
4.10. Esquema de la validación cruzada.	65
4.11. Diagrama de flujo de la prueba experimental.	66
4.12. Gráfica del desempeño de diferentes estructuras de IAR-NN.	68
4.13. Gráfica de la comparación del promedio de precisión de inferencia en los entrenamientos y el desempeño obtenido de la estructura 12-4-6 para diferentes épocas de entrenamiento.	69
A.1. Diagrama de clases de OPAIEHClient.	76
B.1. Diagrama de clases de OPAIEHServer.	79

Índice de tablas

3.1. Ejemplo de registros en la tabla <i>users</i>	32
3.2. Ejemplo de registro de confirmaciones de actividades de <i>userX</i>	32
3.3. Casos posibles de resultados de la realización de QST y QD.	38
3.4. Ejemplos de actividades registradas en una réplica de IAR-ONTO.	38
3.5. Actividades registradas por el usuario en su réplica correspondiente de IAR-ONTO.	43
3.6. Registro de actividades no inferidas.	45
4.1. Binarización de las habitaciones.	59
4.2. Binarización de los días de la semana.	60
4.3. Precisiones obtenidas del entrenamiento realizado con el conjunto de entrenamiento 1 para diferentes estructuras.	67
4.4. Desempeños obtenidos para diferentes estructuras entrenadas durante diferentes números de épocas.	67
4.5. Relación entre las actividades inferidas erróneamente y las esperadas resultantes de la pruebas realizadas sobre la estructura 12-4-6 entrenada durante 2000 épocas de entrenamiento.	69

Dedicado a mi mamá, Susana.

Capítulo 1

Introducción

1.1. Preliminares

Diversos estudios realizados por el Instituto Nacional de Estadística Geografía e Informática (INEGI) [1, 2], así como reportes de la Organización Mundial de la Salud (OMS) [3], muestran que el sector de la población que es mayor de 60 años está en franco crecimiento. Ciertamente, la esperanza de vida de las personas ha aumentado en los últimos años, lo que ha permitido que el porcentaje de los adultos mayores de 60 años crezca. Sólo por mencionar algunos datos, la OMS menciona que, entre 2015 y 2050, el sector de la población mundial con más de 60 años de edad pasará de 900 millones hasta 2000 millones, lo que representa un aumento del 12 % al 22 %. En México, el Censo de Población y Vivienda 2010 [2], contabilizó 10.1 millones de adultos mayores, lo que representa 9.0 % de la población total. A nivel colectivo, el envejecimiento de la población implica nuevas problemáticas y nuevos retos para cualquier sociedad, por ejemplo: ofrecer servicios de salud enfocados a los adultos mayores, contar con sistemas de transporte adaptados a la movilidad del adulto mayor, ofertar espacios laborales en los que el adulto mayor pueda desempeñarse con su experiencia, más centros de recreación e inclusive servicios de tecnologías de información orientados al adulto mayor [4].

Para solventar los problemas de la edad, mejorar la calidad de vida y desarrollar entornos que brinden ayuda, soporte y asistencia a las personas de la tercera edad durante la realización de sus actividades cotidianas, surgen los campos de la Gerontotecnología y de la Vida Asistida por el Entorno (AAL, por sus siglas en inglés). El término gerontotecnología nace en Europa en la década de 1990 como la composición de dos palabras: gerontología, el estudio del envejecimiento, y tecnología. Se define entonces como el estudio de la tecnología y el envejecimiento para la mejora del funcionamiento diario de los adultos mayores [5]. El área de la AAL también tuvo sus inicios en la década de 1990, pero no fue sino hasta la mitad de la década de 2000 cuando su importancia se hizo notar [6]. La AAL es un campo multidisciplinario emergente que utiliza en gran medida las tecnologías de la información y la comunicación (ICT por sus siglas en inglés) en los sistemas personales de salud y telesalud para contrarrestar los efectos de la creciente población de adultos mayores [7]. Los sistemas AAL proporcionan un ecosistema de sensores médicos, computadoras, redes inalámbricas y aplicaciones para el monitoreo de la

salud. El principal objetivo de las soluciones AAL es prolongar el tiempo durante el cual los adultos mayores puedan vivir independientemente en su ambiente preferido usando tecnologías ICT para preservar su salud personal. La asistencia del adulto mayor en el hogar mediante sensores inteligentes es un servicio que en la actualidad está adquiriendo mayor importancia, ya que permite incrementar la autonomía e independencia de los adultos mayores mientras minimiza los riesgos de que vivan solos [8]. El reconocimiento y monitoreo de las actividades humanas son piezas claves al momento de proporcionar servicios de atención médica y asistencia a las personas de la tercera edad que viven solas [9].

Desde el punto de vista del ambiente, en [10] clasifican a los entornos AAL en internos y externos, según la ubicación en la cual los adultos mayores realizan sus actividades. Los entornos internos son aquellos que se encuentran al interior de un inmueble, por ejemplo, el hogar o la oficina. Por otra parte, los entornos externos son los que se encuentran en la comunidad del usuario, por ejemplo, un parque, un centro comercial o incluso durante el trayecto de un punto a otro.

Una diferencia que se observa en los trabajos de investigación [10, 11, 12, 13, 14, 15, 16, 17, 18, 19] relacionados con el reconocimiento de actividades en los entornos internos y externos, es el empleo de diferentes herramientas y tecnologías para realizar la identificación de las actividades. Generalmente en el exterior se emplea el sensor GPS incluido en los celulares inteligentes (smartphones) [10, 11, 12]. Por el contrario, en un entorno interno sería complicado emplear la señal GPS para el reconocimiento de actividades, dado que no cuenta con la precisión suficiente para determinar la posición de una persona al interior de un hogar [20]. Es por esta razón que en los entornos internos se han empleado videocámaras [13], diferentes tipos de sensores [14, 15, 16], así como señales (WiFi y de radio) [17, 18, 19].

1.1.1. Plataformas desarrolladas para el reconocimiento de actividades en los entornos internos

Los sistemas AAL para el reconocimiento de actividades en los entornos internos se caracterizan por el uso de diferentes herramientas para conseguir dicho objetivo.

En [13] se propone un sistema basado en video que permite reconocer actividades mediante la extracción de las características temporales y espaciales de los datos recogidos por una videocámara. Si bien los sistemas basados en video son muy útiles para monitorear en tiempo real y brindan información muy completa y precisa, presentan un inconveniente, exponen en gran medida la privacidad del usuario. La plataforma desarrollada en este trabajo de tesis permite informar al cuidador del adulto mayor acerca de las actividades que este realiza tratando de minimizar la intrusión en su vida diaria.

Otro tipo de sensores empleados en el reconocimiento de actividades son los sensores magnéticos. En [14] se describe una plataforma basada en el uso de un imán por parte del usuario y la colocación de magnetómetros en la vivienda, los cuales sensan los cambios en el campo magnético que se producen cuando el usuario se aproxima a los sensores. Si bien para el usuario

portar un imán en forma de brazalete puede ser muy práctico, la desventaja de esta plataforma se basa en que hay un compromiso entre la fuerza del imán a utilizar, si es muy fuerte puede poner en riesgo al usuario (puede atraer objetos metálicos que pongan en riesgo su integridad física), y si es muy débil no podrá ser detectado de manera adecuada por los magnetómetros.

En [15] se describe otra plataforma que usa sensores magnéticos pero con un enfoque diferente, se basa en el hecho de que en un hogar hay diversos aparatos eléctricos, por lo tanto equipando al usuario con un guante que tenga embebidos sensores Hall y empleando árboles de decisión como motor para realizar el análisis de la información, es posible reconocer las actividades que realiza el usuario mediante el sensado de los campos magnéticos que se generan al momento de entrar en contacto con los diferentes dispositivos eléctricos y electrónicos que se encuentran en el hogar.

El trabajo realizado en [16] describe un sistema de adquisición de datos que usa un lector y etiquetas (tags en inglés) de identificación por radiofrecuencia (RFID por sus siglas en inglés), así como un motor probabilístico y redes bayesianas, que a partir de los datos recolectados por el lector RFID (al entrar en contacto con los objetos que están “etiquetados”) permiten inferir las posibles actividades que se realizan. Los trabajos realizados en [15, 16] presentan la desventaja de tener que equipar a los usuarios dispositivos electrónicos voluminosos, si bien en la plataforma presentada en este proyecto de tesis se tiene que instalar una aplicación en el smartphone del usuario, y este debe cargar dicho dispositivo conforme se desplace en su hogar, se tiene la justificación que los smartphones son dispositivos que ya se encuentran presentes en la vida cotidiana de las personas.

Además de utilizar sensores, en los entornos internos también se han empleado diferentes señales [17, 18, 19] para el reconocimiento de actividades, como la señal WiFi y la de radiofrecuencia. En [17, 18] se describe una plataforma que emplea la señal WiFi, en específico el indicador de fuerza de la señal recibida (RSSI por sus siglas en inglés), y un algoritmo que combina clasificadores k-NN y árboles de clasificación con la finalidad de reconocer las actividades que realiza un usuario en una habitación. Esta plataforma se basa en el principio de que la señal WiFi es bloqueada o reflejada de diferentes maneras de acuerdo a la actividad que realice el usuario, y por lo tanto para cada una de las actividades se retorna una marca única de RSSI en el receptor, la cual es analizada mediante las técnicas de clasificación anteriormente mencionadas, lo que permite reconocer aquella actividad que se realiza en un momento en específico.

En [19] se describe una plataforma que emplea el RSSI de la señal de radiofrecuencia para el reconocimiento de actividades. Su principio es similar al de la señal WiFi, las señales de radio son sensibles a los cambios en el ambiente, el cuerpo humano la puede absorber, dispersar o reflejar. La variación de la calidad de la señal es perjudicial en el área de comunicaciones, pero este efecto secundario es útil en el reconocimiento de actividades. La desventaja de las plataformas basadas en las señales WiFi y de radio es que hasta el momento solo pueden reconocer tres actividades (caminar, pararse o sentarse) al interior de la habitación donde se coloquen los emisores y receptores de dichas señales. Por otra parte, la plataforma presentada en este proyecto de tesis, es capaz de reconocer todas las actividades que el usuario registre; el área de monitoreo abarcada consistirá en las habitaciones en donde se coloquen los sensores.

El reconocimiento de actividades juega un papel importante en la asistencia al interior del hogar de los adultos mayores, dado que el reconocer las actividades de manera precisa es importante para la determinación del riesgo que conlleva realizarlas, lo que se traduce en la generación de conocimiento que puede ser empleado con la finalidad de preservar la integridad física de los adultos mayores.

1.2. Contexto y problemática

En esta sección se presenta el contexto y la problemática que este trabajo de tesis pretende atacar.

1.2.1. Definición de actividades

La tarea primordial de la plataforma desarrollada es reconocer (inferir o identificar) qué actividad se encontraba realizando el adulto mayor. Para conseguir esto, primero que nada es necesario definir o caracterizar las actividades que realizan los adultos mayores al interior de su hogar. Por lo tanto, se requiere dar solución a la interrogante ¿Qué parámetros serán considerados para definir qué es una actividad en el entorno de la plataforma?

Es importante mencionar que para efectos de este trabajo los términos reconocer, inferir o identificar serán empleados de manera indistinta.

1.2.2. Identificación de actividades

Habiendo establecido la definición de las actividades, el siguiente desafío conlleva en implementar un sistema que sea capaz de monitorear al adulto mayor, es decir, extraer la información de las actividades que este realiza y enviárselas a un módulo que sea capaz de identificar la actividad que el adulto mayor realizaba con base en la información que le fue entregada.

Para lograr la identificación de actividades, es necesario determinar ¿Quién será el encargado de obtener la información sobre las actividades que realiza el adulto mayor?, ¿Qué deberá contener esa información?, ¿Quién será el encargado de analizar esa información?.

1.2.3. Minimizar intrusión

Un factor importante para el éxito en el uso de las herramientas y tecnologías para el monitoreo digital de la vida diaria de las personas de la tercera edad, es mantener en el usuario la sensación de vida independiente, de no sentirse observadas ni monitoreadas. Un sistema digital que informa o notifica absolutamente todos los movimientos de una persona a terceros (familiares o médicos), termina por desmotivar a su usuario, dado que su vida privada queda

notoriamente expuesta. Por lo tanto, el monitoreo no intrusivo es un requerimiento fundamental en el desarrollo de plataformas enfocadas al cuidado de los adultos mayores [21].

Ahora bien, la cuestión es ¿Hasta dónde debe intervenir un sistema de monitoreo inteligente en la vida de los adultos mayores mientras estos realizan sus actividades en el hogar? Por ejemplo, la falta de intervención puede ocasionar que el sistema no reconozca adecuadamente las actividades que este realiza al interior de su hogar. Por otra parte, la excesiva intervención (intrusión) desmotiva el uso del sistema mismo.

1.3. Objetivos

Con base en la problemática presentada se fijan el objetivo general y los objetivos específicos.

1.3.1. Objetivo general

Desarrollar una plataforma inteligente no intrusiva que permita reconocer las actividades que realizan los adultos mayores al interior de su hogar.

1.3.2. Objetivos específicos

1. Definir las variables que caracterizan las actividades que realizan los adultos mayores, y que permitan identificarlas.
2. Diseñar un modelo que permita inferir las actividades que realizan los adultos mayores al interior de su hogar.
3. Implementar el modelo de inferencia en un servidor web.
4. Desarrollar una aplicación móvil que permita monitorear al adulto mayor y envíe información al servidor Web.
5. Diseñar un cliente Web que permita visualizar la información de las actividades del usuario.
6. Evaluar el funcionamiento de la plataforma desarrollada.

1.4. Contribuciones

En este proyecto de tesis se desarrolló una plataforma para realizar el reconocimiento de actividades al interior del hogar de los adultos mayores, particularmente de los que viven de

manera independiente, es decir que gozan de buena salud física y mental, y pueden realizar sus actividades por sí solos y sin cuidados especiales al interior de su hogar.

Es importante mencionar que la plataforma propuesta en este proyecto de tesis fue una prueba de concepto (proof of concept, en inglés), por las siguientes razones:

- Las pruebas experimentales para evaluar el funcionamiento de la plataforma no fueron realizadas en adultos mayores, dado que era necesario que el usuario empleara un smartphone, y a la mayoría de los adultos mayores se les dificulta el empleo de dichos dispositivos [22].
- Además, el cargar consigo mismo un smartphone a cada una de las habitaciones del hogar suele ser muy incómodo, por lo que el empleo del smartphone en la plataforma no fue pensado como dispositivo final, lo ideal sería emplear un dispositivo más portátil como un reloj inteligente (smartwatch).

En este trabajo de tesis se propuso una arquitectura genérica para el desarrollo de una plataforma para el reconocimiento de actividades, dicha arquitectura consta de diversos componentes, entre los que destacan: sensores, dispositivo de monitoreo, una aplicación Web en el cual se encuentre embebido un algoritmo de inferencia de actividades, una base de datos en la cual se almacenen los registros de las actividades realizadas por el usuario, un cliente Web que permita representar la información de la base de datos de manera inteligible.

Además se desarrollaron dos modelos basados en la arquitectura propuesta. El primer modelo desarrollado fue basado en ontologías, en torno a este se desarrollaron cada uno de los componentes de la arquitectura propuesta. El segundo modelo desarrollado fue basado en redes neuronales, para este únicamente se desarrolló el motor de inferencia de actividades, dado que la finalidad de la arquitectura es que cada uno de los componentes funcionen como piezas de un rompecabezas y puedan ser reemplazados.

La finalidad de este proyecto de tesis es que la plataforma para el reconocimiento de las actividades que realizan los adultos mayores al interior del hogar, sea aplicada en sistemas de asistencia para adultos mayores, los cuales permitan monitorearlos de manera no intrusiva.

1.5. Estructura de la tesis

Esta tesis se estructura de la siguiente forma:

En el Capítulo 2 se presenta la arquitectura general de la plataforma para el reconocimiento de actividades al interior del hogar, además se plantean las características implementadas para minimizar la intrusión en la vida de los usuarios.

En el Capítulo 3 se describe el diseño del modelo ontológico de la plataforma para el reconocimiento de actividades al interior del hogar, la arquitectura particular de este modelo, las pruebas experimentales realizadas y los resultados obtenidos.

En el Capítulo 4 se describe el diseño del modelo basado en redes neuronales de la plataforma para el reconocimiento de actividades al interior del hogar, las pruebas experimentales realizadas y los resultados obtenidos.

Finalmente, en el Capítulo 5 se dan las conclusiones del proyecto, haciendo énfasis en la comparación de los dos modelos propuestos, así como la propuesta de trabajo futuro.

Capítulo 2

Arquitectura general de la plataforma

En este capítulo se presenta la propuesta de arquitectura general para la plataforma de reconocimiento de actividades al interior del hogar de los adultos mayores.

2.1. Estructura de la arquitectura

La arquitectura general de la plataforma (Fig. 2.1) está compuesta por cuatro módulos principales: sensores, dispositivo de monitoreo (o dispositivo cliente), un servidor Web y un cliente Web. En conjunto, dichos módulos deben conseguir dos objetivos principales:

1. Realizar el reconocimiento de las actividades que realiza el adulto mayor al interior de su hogar.
2. Permitir al cuidador del adulto mayor la visualización del análisis de las actividades que este último realiza al interior de su hogar.

2.2. Funcionamiento de los módulos de la arquitectura

A continuación se describirán con más detalle cada uno de los módulos de la arquitectura. En Fig. 2.2 se muestra el esquema de funcionamiento o aplicación de la plataforma al interior del hogar, cabe mencionar que tanto el servidor Web como el cliente Web serán módulos que se encontrarán fuera del hogar.

2.2.1. Sensores

La función de los sensores en la arquitectura será proporcionar información acerca de la ubicación del adulto mayor al interior del hogar, es importante recordar que en el capítulo anterior se mencionó que, en los entornos internos, la señal GPS no puede ser empleada para

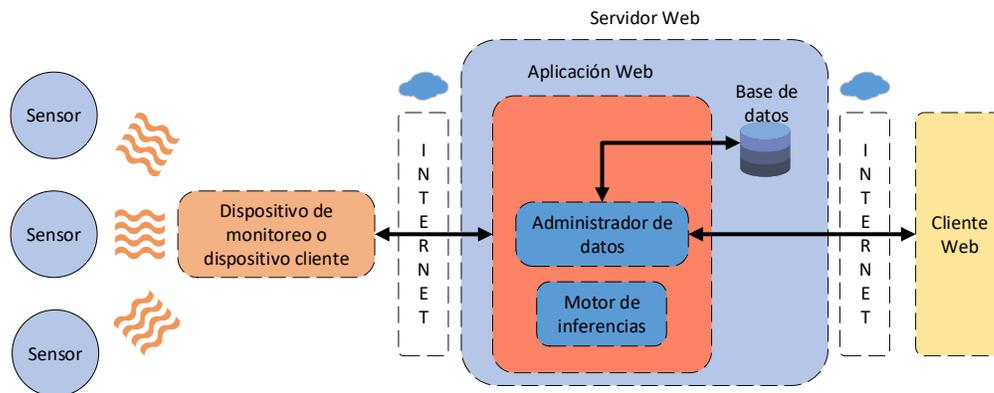


FIGURA 2.1: Arquitectura general de la plataforma para el reconocimiento de actividades al interior del hogar.

conocer la ubicación de una persona al interior de un inmueble. Por cada habitación en la que se desee monitorear las actividades que realiza el adulto mayor deberá colocarse un sensor.

2.2.2. Dispositivo cliente o dispositivo de monitoreo

Antes de iniciar el reconocimiento de actividades es necesario tener una fase de registro; en dicha fase, el dispositivo cliente deberá permitir al usuario: ingresar sus datos personales, instanciar las actividades que este considera que realiza al interior de su hogar e instanciar las habitaciones que serán monitoreadas. Los datos ingresados en el dispositivo cliente serán enviados al servidor Web, el cual se encargará de almacenarlos.

Además de formar parte del registro, el dispositivo determinará la ubicación del adulto mayor al interior de su hogar con base en la información que le será proporcionada por los sensores. Sin embargo, es importante mencionar que la ubicación del adulto mayor no será suficiente para realizar el reconocimiento de actividades. Es por esta razón que se requiere que el dispositivo cliente, además de recibir la información de los sensores, sea capaz de generar más información (parámetros) que permita la inferencia de las actividades.

Por último, el dispositivo cliente deberá enviar la información (parámetros) de la actividad que realizaba el usuario al servidor Web, con la finalidad de que éste determine la posible actividad que realizaba el usuario. Se emplea el término posible ya que no se tiene el 100% de certeza de que la actividad inferida sea correcta. Es por esta razón que se sugiere que exista una retroalimentación por parte del usuario, es decir, que el servidor Web le indique al dispositivo cliente cuál fue la actividad inferida, con la finalidad de que el usuario indique si dicha inferencia fue acertada o no.

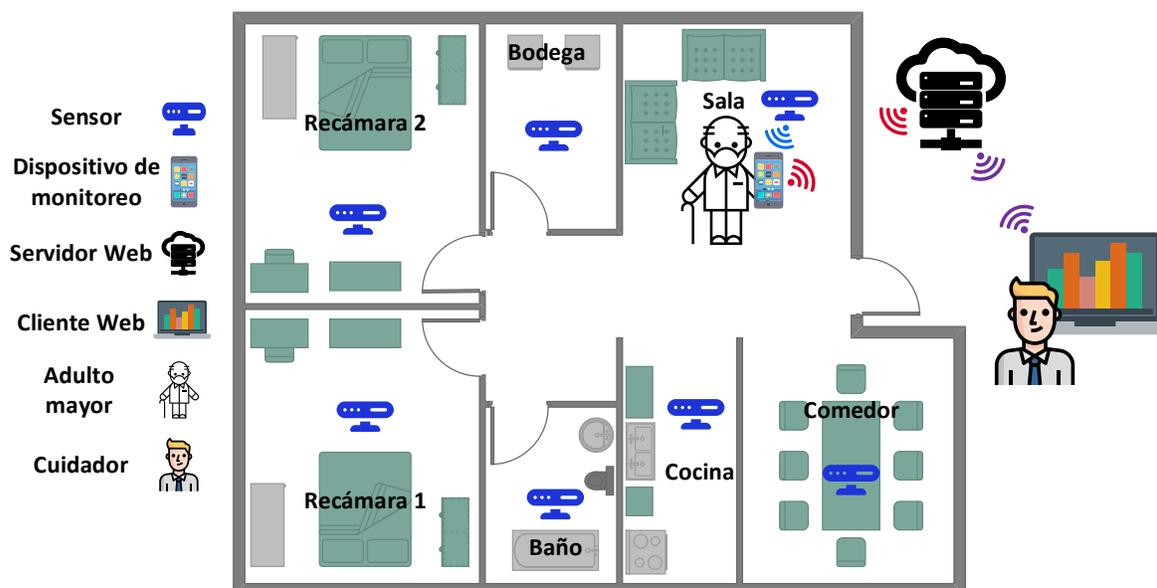


FIGURA 2.2: Aplicación de la arquitectura general de la plataforma en el hogar.

2.2.3. Servidor Web

El Servidor Web será el módulo encargado de: 1) recibir la información proveniente del dispositivo cliente (registro, parámetros de la actividad realizada por el usuario, retroalimentación o confirmación de la actividad inferida); 2) realizar la inferencia de las actividades; 3) dar a conocer la actividad inferida al dispositivo cliente; 4) recibir la confirmación del usuario acerca de la inferencia realizada; 5) proporcionar al Cliente Web las confirmaciones realizadas por el usuario; 6) almacenar en la base de datos los registros de los usuarios y las confirmaciones realizadas por estos. Para realizar estas tareas el servidor Web estará formado por dos sub-módulos: una aplicación Web y una base de datos.

1. *Aplicación Web*. Este módulo estará comprendido por dos sub-módulos: administrador de datos y motor de inferencias.
 - a) *Administrador de datos*. Comprenderá la parte del registro del lado del servidor, es decir, estará a cargo del registro del usuario, de las actividades que este indique que realiza al interior de su hogar, y de las habitaciones. Se encargará de recibir las confirmaciones de las actividades realizadas por el usuario. Además proporcionará al cliente Web la información que éste le solicite, con la finalidad de dársela a conocer al cuidador del adulto mayor. Este módulo deberá almacenar la información que reciba en la base de datos.

- b) *Motor de inferencias*. Será el encargado de recibir los parámetros de la actividad que realizaba el usuario enviados por parte del dispositivo cliente, con dicha información este módulo deberá inferir la posible actividad. Habiendo inferido la actividad, el motor de inferencias deberá enviársela al dispositivo cliente.
2. *Base de datos*. Será el espacio de almacenamiento de los datos de los usuarios. Deberá ser estructurada de tal manera que exista una tabla con los datos personales de cada uno de los usuarios; además cada usuario deberá tener su propio espacio en la base de datos que le permita almacenar las actividades que el considera que realiza al interior de su hogar y las habitaciones en las que desea ser monitoreado, así como las confirmaciones de todas las actividades que realice al interior de su hogar.

2.2.4. Cliente Web

Este componente será el encargado de presentar al cuidador del adulto mayor un análisis de las actividades que este realiza al interior de su hogar, para obtener dicha información el Cliente Web requerirá realizar solicitudes al Administrador de Datos de la Aplicación Web contenida en el Servidor. Con base en esa información, se pretende que el cuidador pueda monitorear y vigilar al adulto mayor.

2.3. Consideraciones para minimizar la intrusión

Como se mencionó en el capítulo anterior, una de las problemáticas a resolver es minimizar la intrusión en la vida del adulto mayor, es decir, evitar que sienta que su vida privada es invadida. La intrusión a la vida del adulto mayor será considerada en tres niveles de la arquitectura: a nivel del dispositivo cliente, a nivel de las actividades, y a nivel del Cliente Web.

1. *A nivel del dispositivo cliente*. La intrusión se minimizará procurando recopilar la menor cantidad de parámetros posibles necesarios para realizar la inferencia de actividades (realizada por el Servidor Web), es decir, utilizar los parámetros mínimos necesarios que permitan realizar el reconocimiento de actividades.
2. *A nivel de las actividades*. De todas las actividades que puede realizar el adulto mayor al interior de su hogar, las necesidades fisiológicas son las más personales e íntimas, y exponer a terceras personas dicha información es sumamente intrusivo. Es por esta razón que se debe diseñar la plataforma de tal manera que se evite este tipo de intrusión en la vida del adulto mayor.
3. *A nivel del Cliente Web*. La información proporcionada por este módulo al cuidador del adulto mayor deberá ser presentada de tal manera que se evite exponer en gran medida la vida del adulto mayor.

Capítulo 3

Plataforma basada en ontologías (OPAIEH)

En este capítulo se presenta el diseño y desarrollo de OPAIEH (Ontology Platform for Activity Identificación for Elderly at Home), la plataforma basada en ontologías para la identificación (o reconocimiento) de actividades al interior del hogar de los adultos mayores. De manera precisa, OPAIEH emplea la ontología IAR-ONTO (Indoor Activity Recognition Ontology) para realizar la inferencia de actividades.

3.1. Marco teórico

En esta sección se explican los conceptos básicos de lenguaje natural, lenguaje formal, representación de conocimiento, lógicas de descripción, ontologías y OWL, esenciales para la comprensión del desarrollo de OPAIEH, en la cual se emplea un enfoque ontológico para representar el concepto de actividad y con base en este realizar la inferencia de actividades. Además, se presentan los trabajos relacionados con un enfoque ontológico.

3.1.1. Lenguaje natural y lenguaje formal

El *lenguaje natural* es el medio que utilizamos de manera cotidiana para establecer nuestra comunicación con las demás personas [23]. Si dos personas hablan el mismo idioma, podrán entablar una comunicación e intercambiar información. Como humanos, entender una oración o un enunciado que escuchamos o leemos es tan sencillo que casi nunca pensamos conscientemente sobre lo que significa comprender una oración y cómo logramos esta tarea [24]. Se tomará como ejemplo la siguiente oración:

Justo después del gol inicial de Puskás, Hungría amplió su ventaja. Pero los alemanes pronto alcanzaron el nivel, y un disparo de largo alcance de Rahn finalmente determinó la sorprendente victoria de Alemania.

De la oración se pueden realizar varias inferencias: 1) la oración habla de un resumen de un partido de fútbol soccer; 2) al mencionar a Hungría y a Alemania el texto no se refiere a los países, si no a sus respectivas selecciones de fútbol soccer; 3) en el partido se marcaron 5 goles (2 de Hungría y 3 de Alemania); 4) los futbolistas Puskás y Rahn juegan para Hungría y Alemania respectivamente.

La tarea de comprender una oración en un lenguaje natural es posible en parte gracias a la extracción de la información que se encuentra implícita. Los humanos podemos entender dicha información gracias al conocimiento previo que hemos adquirido a lo largo de nuestra vida. Sin embargo una tarea que parece relativamente fácil para los humanos, no lo es para las computadoras, dado que carecen de esa base de conocimiento y por lo tanto el realizar esas inferencias es difícil para ellas [25].

Otro problema que presentan los lenguajes naturales es la ambigüedad, se dice que algo es ambiguo cuando puede ser entendido en dos o más sentidos [26]. Por ejemplo si se tienen las siguientes oraciones:

1. Juan utilizó el gato para cambiar el neumático de su automóvil.
2. El gato de Juan se comió toda su comida.

En ambas oraciones aparece la palabra gato, sin embargo, en la primera se emplea para referirse a una herramienta mecánica y en la segunda a un ser vivo.

Por otra parte, el *lenguaje formal*, es aquel que el hombre ha desarrollado para expresar las situaciones que se dan en específico en cada área del conocimiento científico. Los lenguajes formales pueden ser utilizados para modelar una teoría física, matemática, ingeniería, crear lenguajes de programación, etc., con la ventaja de que en éstos toda ambigüedad es eliminada [23]. Por ejemplo el siguiente código tiene exactamente un solo significado según lo establecido en el lenguaje C.

```
int main(){
    int num1;
    int num2;
    int sum;
    sum=num1+num2;
    printf("La suma es: %d,sum);
}
```

Si bien, los lenguajes formales permiten eliminar la ambigüedad, aún queda la necesidad de que las computadoras tengan una base de conocimiento que les permita realizar inferencias sobre ese conocimiento explícito. Por esto suena razonable suponer que si se desea un sistema computacional inteligente, es necesario encontrar la manera que conozca cosas, y esto implica encontrar una manera de representar las cosas que deseamos que sepa para que puedan codificarse dentro de dicho sistema [27].

3.1.2. Representación del conocimiento y lógicas de descripción

La representación del conocimiento (KR por sus siglas en inglés) es el área de la inteligencia artificial (AI por sus siglas en inglés) que se preocupa por cómo un agente usa lo que sabe para decidir qué hacer [28], es decir, estudia cómo poner el conocimiento en un formato en el cual las computadoras puedan razonar sobre ese conocimiento [29]. La representación del conocimiento se enfoca en el diseño de formalismos que sean adecuados tanto epistemológicamente como computacionalmente para la representación del conocimiento de un dominio en particular. Una de las principales líneas de investigación se ha concentrado en el principio de que el conocimiento debería representarse mediante la caracterización de clases de objetos y las relaciones entre ellos. La organización de las clases utilizadas para describir un dominio de interés se basa en una estructura jerárquica, que no solo proporciona una representación efectiva y compacta de la información, sino que también permite que el razonamiento se realice de manera efectiva [30].

Las lógicas de descripción (DLs por sus siglas en inglés) son una familia de lenguajes para la representación del conocimiento, que son usadas para representar el conocimiento de un dominio de aplicación de una forma estructurada y formalmente bien entendida [31]. Las DLs forman parte de las diferentes aproximaciones empleadas en el área de KR (que a su vez pertenece al área de AI, ver Fig. 3.1).

Las DLs proporcionan medios para expresar conocimiento sobre conceptos que componen una terminología (TBox), así como conocimiento sobre hechos concretos (objetos que instancian los conceptos) que forman una descripción del mundo (ABox). Dado que las DLs proporcionan una sintaxis y una semántica formal, facilitan la formulación de algoritmos de razonamiento sólidos y completos [32]. Las lógicas de descripción proporcionan las bases formales para las ontologías [33].

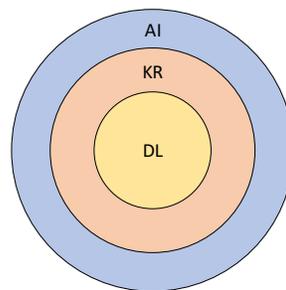


FIGURA 3.1: Relación de inclusión entre las áreas: inteligencia artificial (IA), representación del conocimiento (KR) y lógicas de descripción (DL).

3.1.3. Ontologías

Originalmente, la ontología es una antigua rama de la filosofía que tiene como objetivo establecer la verdad sobre la realidad, mediante la búsqueda de la respuesta a la pregunta ¿Qué existe?. La ontología, en su significado original, se encarga de la descripción objetiva de la realidad y cualquier dominio de objetos [34], mediante la definición de clases y estructuras de los objetos, propiedades, eventos, procesos y relaciones en cada área de la realidad [35]. Sin embargo, este término se ha expandido a otras áreas de la ciencia y ha adquirido diferentes interpretaciones en cada una de ellas.

En el área de cómputo, una ontología es definida como “una formal y explícita especificación de una conceptualización compartida” [36, 37, 38]. En términos generales, las ontologías proporcionan un vocabulario de términos y relaciones mediante los cuales se puede representar un dominio [39], con la intención que este permita el intercambio de conocimiento, re-uso de la información y razonamiento sobre dicho conocimiento [40].

La palabra ontología puede hacer referencia a diferentes objetos en el área de cómputo según el contexto [41], por ejemplo una ontología puede ser: un diccionario en el área de la recuperación de información; o un modelo representado en OWL (Web Ontology Language) en el área de vinculación de datos; o un esquema XML (eXtensible Markup Language) en el contexto de base de datos; etc.

En la actualidad, se cuenta con varios lenguajes ontológicos como Formato de Intercambio de Conocimiento (KIF por sus siglas en inglés), Ontolingua, Cyc y Lenguaje de Ontologías Web (OWL por sus siglas en inglés) [34]. En este trabajo de tesis, se desarrollo un modelo ontológico basado en OWL.

OWL

El World Wide Web Consortium (W3C) recomienda el uso de OWL como lenguaje de ontologías, porque posee una gran cantidad de vocabulario para describir propiedades y clases [42]. OWL se basa en la capacidad que tiene XML para definir esquemas de etiquetado personalizado y en el enfoque flexible de RDF (Resource Description Framework) para la representación de datos [34].

OWL propone 3 versiones de lenguajes [42]:

- *OWL Lite*: Es el sub-lenguaje OWL sintácticamente más simple. Está destinado a ser utilizado en situaciones donde solo se necesita una jerarquía simple de clases y restricciones simples.
- *OWL DL*: Es mucho más expresivo que OWL-Lite y se basa en las lógicas de descripción, y por esta razón OWL-DL permite realizar razonamiento automático. Además, es posible calcular automáticamente la jerarquía de clasificación y comprobar si hay incoherencias en la ontología.

- *OWL Full*: Es el sub-lenguaje OWL más expresivo. Está destinado a ser utilizado en situaciones donde la expresividad muy alta es más importante que poder garantizar la decidibilidad o integridad computacional del lenguaje. No es posible realizar razonamiento en OWL-Full [43].

El lenguaje OWL, entonces, es usado para formalizar un dominio mediante la definición de clases, propiedades de esas clases, e individuos, los cuales son instancias de esas clases. Además, permite razonar sobre esa base de conocimiento.

3.1.4. Trabajos relacionados

En los últimos años ha crecido el interés en el uso de técnicas basadas en ontologías para el reconocimiento automático de datos contextuales complejos, como las actividades humanas. En el área del cómputo ubicuo, el lenguaje OWL DL se ha estado usando para construir ontologías para la representación de actividades, y con base en datos contextuales, realizar el reconocimiento [44]. Esto ha sido posible gracias a las propiedades de las ontologías: simplicidad, flexibilidad, extensibilidad, interoperabilidad, expresividad y producción de código automático [45].

En [46] se describe una arquitectura para el reconocimiento de actividades basada en el empleo de sensores, videocámaras y una estructura ontológica. La ontología diseñada, se encarga de describir las relaciones e interacciones entre las actividades, el usuario, los objetos, y los datos proporcionados por los sensores y las videocámaras, con la finalidad de inferir la actividad que el usuario se encuentra realizando. La videocámara es empleada con la finalidad de complementar la información proveniente de los sensores, así como también, para solventar problemas referentes a información incierta o incompleta que estos puedan proporcionar. Otro aspecto importante a destacar es que en este proyecto se considera una actividad como la secuencia de varias sub-actividades; por el momento, la única actividad en la cual se enfoca el reconocimiento de esta arquitectura es “preparar una taza de té”, cuya realización involucra diversas sub-actividades, entre las cuales destacan hervir la tetera, obtener una taza, obtener una bolsa de té, etc.

En [47] se presenta la arquitectura OBAR (Ontology Based Activity Recognition). En esta arquitectura se desarrolló una infraestructura consciente del contexto compuesta de diversos sensores (infrarrojos, magnéticos, presión, etc.) y un sistema de reconocimiento ontológico capaz de reconocer actividades tomando como parámetros la localización del usuario al interior de su hogar, la hora de realización de actividad, el objeto empleado en la actividad, y en algunos casos la postura corporal utilizada al realizar la actividad (detectada por los sensores ultrasónicos), este último parámetro es empleado para reducir la ambigüedad en ciertas circunstancias. Además en la ontología diseñada, se propusieron los conceptos de actividades basadas en ubicación (aquellas que se realizan en una única habitación) y las actividades basadas en objetos (actividades que pueden realizarse en diferentes lugares con el mismo objeto).

En [48] se presenta el método ARoM (ADL Recognition Method) para el reconocimiento de las ADLs (actividades de la vida diaria, por sus siglas en inglés) realizadas por una persona al interior de una casa inteligente (Smart Home en inglés) equipada de sensores. Para el reconocimiento de las ADLs, ARoM emplea una base ontológica para la definición semántica de la ubicación, el dispositivo y el entorno, así como un sistema de razonamiento capaz de determinar la actividad que ejecuta una persona y con base en esa información ejecutar la acción de asistencia más idónea.

En [45] se presenta una clasificación de los diferentes trabajos de investigación relacionados con el reconocimiento de actividades basados en ontologías, con la finalidad de compararlos y evaluarlos. En dicho trabajo se clasifican los métodos ontológicos para el reconocimiento de actividades en dos grandes categorías: 1) modelado contextual inteligente simple, en esta categoría se encuentran OWL1 y OWL2; 2) modelado semántico inteligente complejo, en esta categoría se encuentran modelos basados en lógica (SWRL, SPARQL, etc.), modelos híbridos basados en ontologías-probabilidad-estadística como COSAR [49], modelos ontológicos con meta-datos, modelos basados en ontologías difusas.

Discusión sobre el estado del arte

Los trabajos realizados en [46, 47] proponen ontologías que permiten representar de manera semántica todos aquellos parámetros que intervienen en la realización de una actividad, como pueden ser ubicación, objetos, sensores involucrados, así como parámetros muy específicos como postura corporal o la información proveniente de videocámaras [47]. La ontología desarrollada en este proyecto de tesis fue diseñada de tal forma que en ella intervengan únicamente 4 parámetros (habitación, día, hora y duración), los cuales permiten representar de manera semántica el concepto de actividad presentado en este trabajo de tesis, cuyo diseño se basa en el supuesto que las personas tienden a realizar sus actividades al interior del hogar de manera rutinaria [50], especialmente los adultos mayores [51]. El involucrar un menor número de parámetros en el reconocimiento de actividades conlleva a diseñar una ontología que modele de manera general las actividades que realizan las personas al interior de su hogar, así como también permite minimizar la intrusión en la vida de las personas, dado que se expone una menor cantidad de aspectos de la vida diaria de éstas.

3.2. Definición de actividad

Para realizar el reconocimiento de actividades lo más importante es tener en claro qué será considerado como una actividad. A continuación se presentan los parámetros que se emplearán para la caracterización de las actividades (tomando como referencia los implementados en [10]):

- *Room (Habitación)*. Es el lugar del hogar donde se realiza la actividad (por ejemplo el baño, la cocina, el comedor, etc.).

- *Day (Día)*. Es el día (o los días) de la semana en que se realiza la actividad.
- *StartTime (Hora de inicio)*. Es la hora en la cual el usuario empezó a realizar la actividad.
- *Duration (Duración)*. Es el tiempo que le toma al usuario realizar la actividad. La actividad inicia cuando el usuario entra en una habitación, y termina cuando éste se sale de dicha habitación.

Por lo tanto una actividad se define como la acción que realiza un adulto mayor en una habitación, en uno o varios días de la semana, con una hora de inicio, y con una duración. Por ejemplo un adulto mayor realiza la actividad *desayunar* en el comedor, de lunes a sábado, a las 7 am, y le toma realizar dicha actividad 20 minutos.

3.3. Modelo ontológico empleado en OPAIEH

En este trabajo se propone IAR-ONTO, una ontología OWL que es empleada por OPAIEH como una base de conocimiento para el reconocimiento de actividades al interior del hogar. IAR-ONTO fue diseñada para satisfacer la definición de actividad planteada en la sección anterior, dado que representa de manera semántica el concepto de actividad.

Para el diseño de IAR-ONTO se empleó Protégé [52], una plataforma de código libre gratuita que permite la creación, visualización y manipulación de ontologías en varios formatos de representación [48], incluido OWL.

Es importante mencionar que IAR-ONTO es la ontología base, la cual puede ser replicada cuantas veces sea necesario (con el nombre base de *UserX*, donde *X* es el número de réplica), con la finalidad de que cada una de esas réplicas (que se encontrarán en el servidor Web) sean asignadas a un usuario en específico, es decir, a cada usuario le corresponderá una réplica de IAR-ONTO.

Las ontologías OWL están compuestas por clases, propiedades de esas clases, así como instancias de dichas clases. A continuación se explicarán cada uno de estos componentes.

3.3.1. Clases de IAR-ONTO

Las clases OWL se interpretan como conjuntos que contienen individuos del mismo tipo, estas se describen utilizando descripciones formales que establecen con precisión los requisitos que se requieren para ser miembro de cada una de esas clases [43]. IAR-ONTO consta de 3 clases: *DaysofTheWeek*, *Room* y *Activity*. En Fig. 3.2 se muestra el diagrama de clases de IAR-ONTO.

La clase *DaysofTheWeek* representa los días de la semana.

La clase *Room* representa una habitación. Esta clase posee los atributos de *hasSensorID* y *hasSensorName* cuyos tipos de datos son *int* y *String* respectivamente. El atributo *hasSensorID*

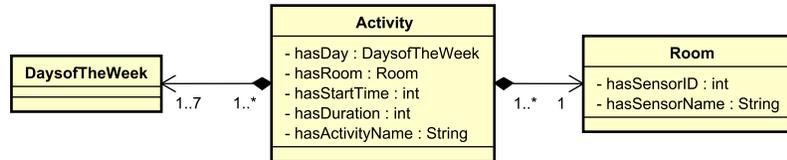


FIGURA 3.2: Diagrama de clases de IAR-ONTO.

permite ingresar el ID del sensor, mientras que el atributo *hasSensorName* permite ingresar el nombre del sensor (el cual se nombra de acuerdo a la habitación donde se encuentre ubicado).

La clase *Activity* representa una actividad. Esta clase posee 5 atributos, de los cuales 4 de estos (*hasDay*, *hasRoom*, *hasStartTime*, *hasDuration*) corresponden a los parámetros que forman parte de la definición de actividad presentada en la sección anterior. El atributo *hasDay*, de tipo *DaysofTheWeek*, indica el día (o los días) de la semana en los cuales se lleva a cabo la actividad; el atributo *hasRoom*, de tipo *Room*, indica la habitación en la cual se lleva a cabo la actividad; *hasStartTime*, de tipo *int*, es la hora, en minutos, en que la actividad inicia (las horas en formato de 24h son modificadas por el sistema para tratarlas en minutos, tomando en cuenta que un día tiene 1440 minutos, las 15:00 serán consideradas por el sistema como 900); *hasDuration*, de tipo *int*, indica la duración de la actividad en minutos; *hasActivityName*, de tipo *String*, es el nombre de la actividad.

3.3.2. Individuos de IAR-ONTO

En OWL los individuos representan objetos en el dominio de interés. En OPAIEH se consideran dos tipos de individuos, los creados en tiempo de diseño y los creados en tiempo de ejecución. Los individuos creados en tiempo de diseño son aquellos que fueron instanciados (desde la interfaz de Protégé) justo después de la creación de la IAR-ONTO, su instanciación es posible debido a que cualquier actividad que sea creada por cualquier usuario requerirá de éstos. IAR-ONTO solo cuenta con 7 individuos creados en tiempo de diseño pertenecientes a la clase *DaysofTheWeek* (Fig 3.3), los cuales corresponden a los 7 días de la semana (Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Los individuos creados en tiempo de ejecución (correspondientes a las clases *Room* y *Activity*) son creados por cada usuario y los valores de sus atributos son asignados de manera explícita por ellos mismos a través de OPAIEHClient (la aplicación móvil que será instalada en el smartphone de cada usuario). La creación de las instancias en tiempo de ejecución debe ser

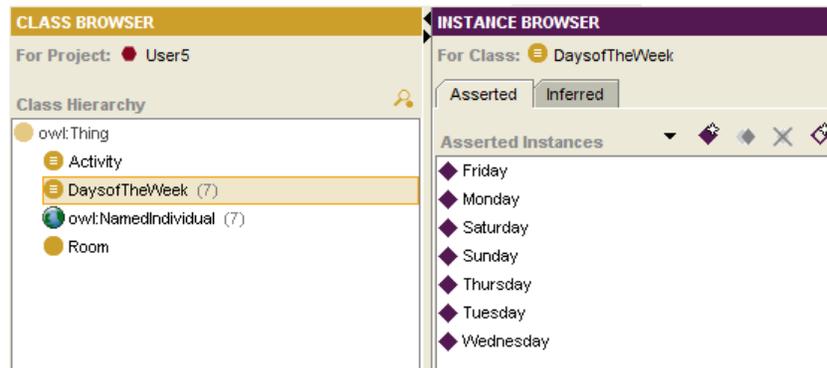


FIGURA 3.3: Clase DaysofTheWeek y sus individuos creados en tiempo de diseño.

realizada en el siguiente orden, primero el usuario debe instanciar las habitaciones de su hogar, para después realizar las instancias de las actividades, esto se debe a que uno de los atributos de la clase *Activity* es *hasRoom* de tipo *Room*, en otras palabras, no se pueden crear instancias de la clase *Activity* si no existen instancias de la clase *Room*. En Fig. 3.4 se muestra la visualización de los atributos del objeto instanciado en tiempo de ejecución *Sleep* perteneciente a la clase *Activity*.

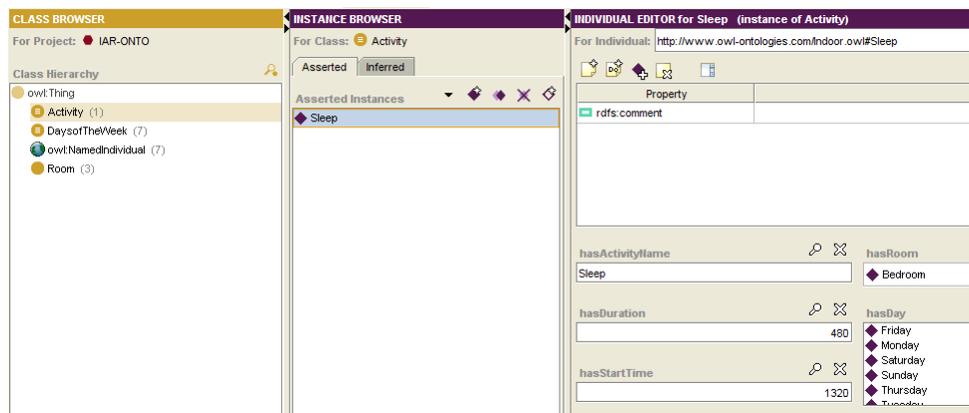


FIGURA 3.4: Visualización de los atributos del objeto *Sleep* de la clase *Activity*.

3.3.3. Propiedades de IAR-ONTO

Las propiedades en OWL representan relaciones entre dos individuos. Existen dos tipos de propiedades principales, las propiedades de objeto y las propiedades de tipo de dato.

Las propiedades de objeto vinculan un individuo con otro. Por ejemplo, *Sleep*, instancia de la clase *Activity*, se relaciona con el individuo *Bedroom*, instancia de la clase *Room*, mediante la propiedad *hasRoom*; esto significa que la actividad *Sleep* se realiza en el cuarto *Bedroom*.

Por otra parte, las propiedades de tipo de dato vinculan a un individuo con un XML Schema Datatype [43]. Por ejemplo, la propiedad *hasDuration*, vincula el individuo *Bedroom*, instancia de la clase *Activity*, con un XML Schema Datatype de tipo *int*, el cual representa los minutos que conlleva realizar dicha actividad. En Fig. 3.5 se muestran las propiedades requeridas en IAR-ONTO.

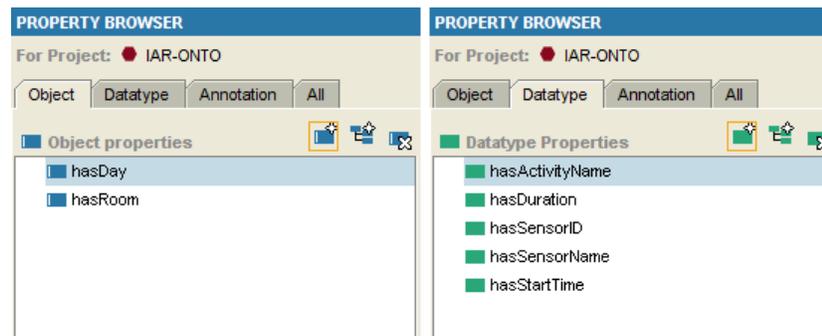


FIGURA 3.5: Propiedades de la ontología IAR-ONTO. De lado izquierdo se muestran las propiedades de objeto y de lado derecho las propiedades de tipo de dato.

3.4. Diseño de OPAIEH

La plataforma OPAIEH está compuesta por cuatro componentes: sensores, OPAIEHClient, OPAIEHServer y OPAIEHClient Web, los cuales en conjunto permiten el reconocimiento de las actividades al interior del hogar, así como la visualización de las estadísticas de éstas. En Fig. 3.6 se muestra el diagrama esquemático de la arquitectura de OPAIEH. En las siguientes secciones se explicarán con más detalle cada uno de los componentes de OPAIEH.

3.4.1. Sensores

Los sensores empleados en OPAIEH son sensores Beacon, los cuales son pequeños dispositivos inalámbricos que emiten una señal de radio de corto alcance, con tecnología Bluetooth 4.0 (llamado Bluetooth Low Energy o BLE) [53]. Estos dispositivos emiten una señal broadcast que contiene el ID del dispositivo (un número entero positivo) y el indicador de fuerza de la señal recibida (RSSI por sus siglas en inglés), entre otros parámetros.

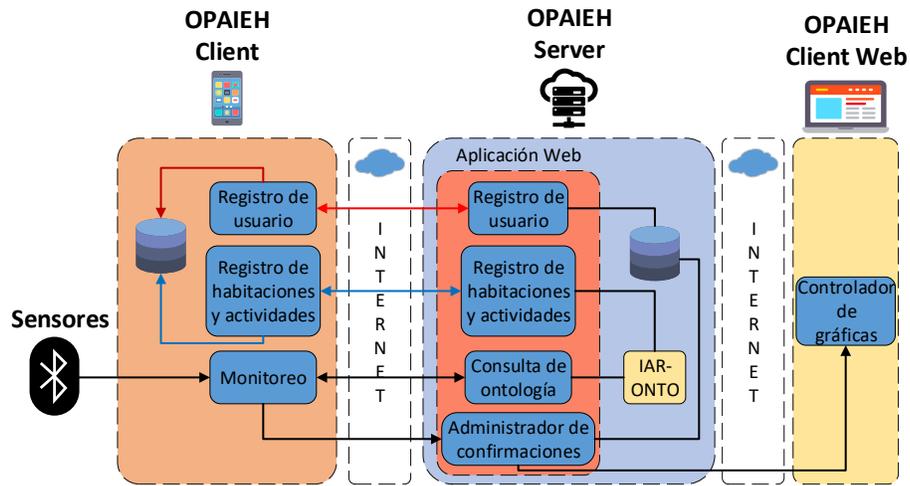


FIGURA 3.6: Arquitectura de OPAIEH.

La señal tipo broadcast de los sensores Beacons colocados al interior de un inmueble brinda la posibilidad de que un receptor, en el caso de OPAIEH un smartphone, reciba de manera simultánea las señales provenientes de todos los sensores que se encuentren en el área de alcance. En Fig. 3.7 se muestra un diagrama ilustrativo del funcionamiento de la señal broadcast de los Beacons.

El ID de los sensores Beacon es un parámetro muy importante en OPAIEH, ya que permite vincular (mediante OPAIEHClient) cada una de las habitaciones con el sensor al cual se coloque en su interior. Por medio de las aplicaciones móviles oficiales de los distribuidores es posible modificar el ID de acuerdo a la necesidad del problema a resolver. En Fig. 3.7 se muestra un ejemplo de la configuración de los IDs al interior del hogar. La cantidad de Beacons necesarios para un hogar dependerá de cuántas habitaciones se desean monitorear.

La señal RSSI es la medida de la potencia de la señal de radio recibida. Esta señal permite determinar que tan cerca (mayor intensidad de la señal) o lejos (menor intensidad de la señal) se encuentran los sensores del receptor, lo cual para OPAIEH se traduce en determinar dónde se encuentra el usuario al interior de su hogar. Tomando como ejemplo el diagrama ilustrativo en Fig. 3.7, se puede intuir que la señal del sensor con ID igual a 6 será la que llegará con más intensidad al smartphone, dado que el sensor del que proviene dicha señal es el más cercano al smartphone, por lo tanto, con esa información se puede determinar que el usuario se encuentra en su sala.

Otra característica que se puede configurar de los sensores, es el intervalo de emisión de la señal, es decir, cada cuanto tiempo la señal es repetida por los sensores. Para su empleo en OPAIEH, los sensores fueron configurados con un intervalo de emisión de 100ms.

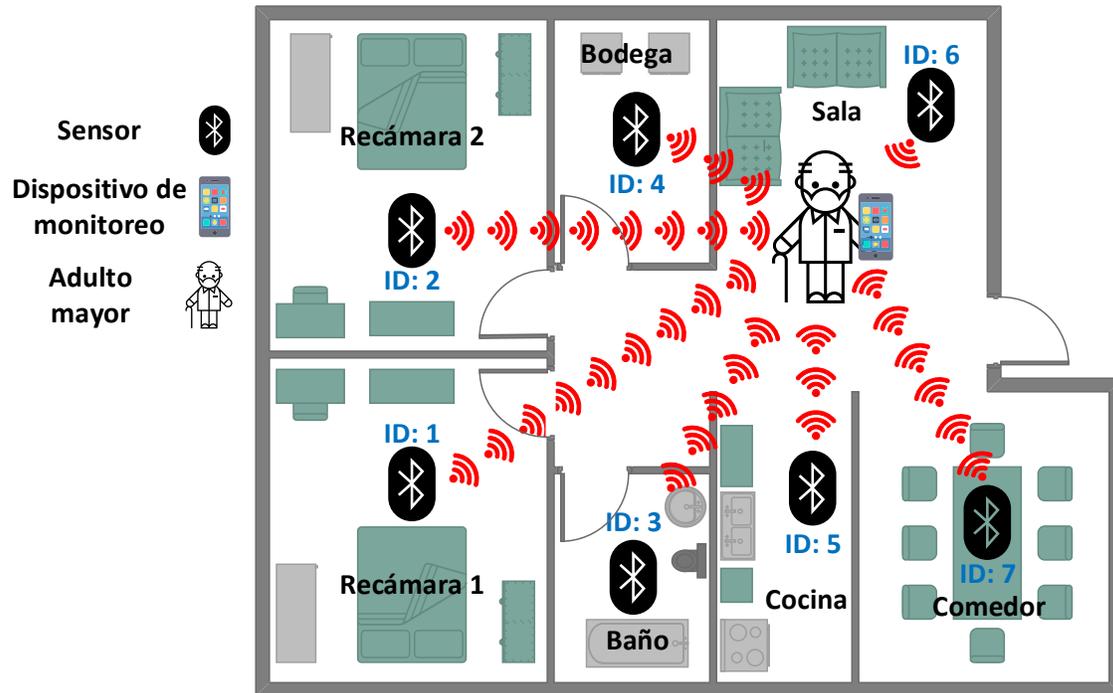


FIGURA 3.7: Funcionamiento de la señal broadcast de los sensores Beacons y configuración de los IDs al interior del hogar.

3.4.2. OPAIEHClient

OPAIEHClient es una aplicación móvil Android (diseñada para la versión 6 Marshmallow), cuya función en la plataforma consiste en permitir al usuario hacer su registro en el servidor (OPAIEHServer), monitorea las actividades que el usuario realiza en su hogar con la finalidad de enviar al servidor los datos de éstas para realizar su inferencia. La inferencia de actividades en OPAIEH tiene un enfoque supervisado, por lo que OPAIEHClient fue diseñado para consultar al usuario si las inferencias realizadas por OPAIEHServer son correctas o no. OPAIEHClient está compuesta de tres módulos principales: *Registro de usuario*, *Registro de habitaciones y actividades*, y *Monitoreo*, que en conjunto permiten realizar las tareas anteriormente mencionadas. A continuación se explicará con más detalle cada uno de los módulos de OPAIEHClient. En el apéndice A se muestra el diagrama de clases de OPAIEHClient.

Registro de usuario

Este módulo permite al usuario ingresar sus datos personales, los cuales son: su nombre de usuario (username), su contraseña y su edad. Estos datos son enviados al servidor (OPAIEHServer) para ser almacenados en su base de datos, el cual devuelve la confirmación de almacenamiento exitoso o fallido, esto con la intención de sincronizar el almacenamiento de las bases de datos del cliente y la del servidor, es decir, al confirmarse el almacenamiento exitoso del lado del servidor, el dispositivo cliente procede a almacenar dichos datos en su base de datos. Las flechas rojas en Fig. 3.6 ilustran la sincronía de almacenamiento.

Registro de habitaciones y actividades

Mediante este módulo el usuario puede registrar las diferentes habitaciones de su hogar, así como las actividades que realiza en estas.

- *Registro de habitaciones.* Este apartado permite vincular los sensores instalados con las habitaciones. La vinculación consiste en asignar el nombre de la habitación donde se encuentra el sensor con el ID de éste (ver Fig. 3.8). De esta manera, cuando la aplicación reciba las señales de los sensores, y por ende los parámetros que en ella viajan (ID del sensor, RSSI, etc.), ésta podrá conocer cuál es el sensor que está enviando la señal más fuerte (es el que se encuentra más cerca del usuario con el smartphone), y con base en esto determinar en cuál habitación se encuentra el usuario.
- *Registro de actividades.* Este apartado permite al usuario registrar las actividades que él considera que realiza al interior de su hogar. Para registrar una actividad, el usuario debe asignarle, un nombre, una habitación, establecer sus horas de inicio y de fin en formato de 24 horas (la duración de las actividades se calcula en base a estos dos parámetros), y los días de la semana en los cuales realiza dicha actividad. En Fig. 3.9 se muestra la interfaz de registro de actividades de OPAIEHClient.

Existe una sincronización en el almacenamiento de las habitaciones y actividades entre OPAIEHClient y OPAIEHServer, del lado del cliente se guarda en su base de datos, y del lado del servidor se guarda en la réplica de IAR-ONTO asignada al usuario. Las flechas azules en Fig. 3.6 ilustran la sincronía de almacenamiento.

Monitoreo

Este módulo se encarga de tres funciones: 1) recibir la señal emitida por los sensores Bluetooth y determinar en cuál habitación se encuentra el usuario; 2) enviar los datos al servidor necesarios para que este infiera la actividad que el usuario realizaba; 3) realizar la confirmación de la actividad inferida (por medio de la intervención del usuario).

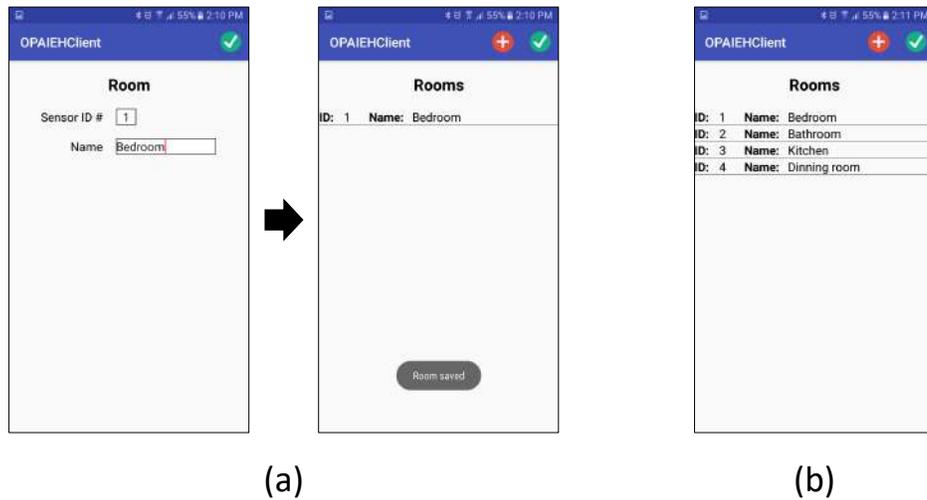


FIGURA 3.8: Registro de habitaciones. En (a) se muestra el registro de la habitación *Bedroom*, y en (b) se muestra la interfaz después de registrar varias habitaciones.

- *Recepción y análisis de la señales.* Este componente se encarga de recibir la señal Bluetooth proveniente de los sensores con la finalidad de determinar en cuál habitación se encuentra el usuario, es decir, cuándo entra y sale de una habitación. Debido a que los sensores se encuentran relativamente cerca unos de otros al interior de un hogar, en ocasiones la señal Bluetooth de estos es volátil, es por esta razón que se decidió tomar 10 muestras del RSSI de cada una de las señales enviadas por los sensores, y con base a ellas, realizar un promedio para determinar en cuál habitación se encuentra el usuario.
- *Envío de los datos de la actividad.* En OPAIEH, la inferencia de actividades se realiza cada vez que el usuario se desplaza de una habitación a otra (en cambio de habitación), dado que es cuando el módulo de monitoreo se encarga de enviar los datos de la actividad que se encontraba realizando el usuario a OPAIEHServer, siempre y cuando la duración de la actividad sea mayor a 1 minuto. En Fig. 3.10 se muestra la interfaz del módulo de monitoreo de OPAIEHClient antes de enviar los datos a OPAIEHServer. Los datos enviados por este módulo a OPAIEHServer son:
 - *Day.* Es el día de la semana en el cual se inició la actividad.
 - *Room.* Es la habitación en la que el usuario realizaba la actividad.
 - *Hour.* Es la hora en la que se inició la actividad. Si bien el usuario ve en la pantalla (ver Fig. 3.10) la hora en formato de hora y minutos (HH:MM), OPAIEHClient envía la hora en términos de minutos, de tal manera que la hora 14:18 será enviada como 858.

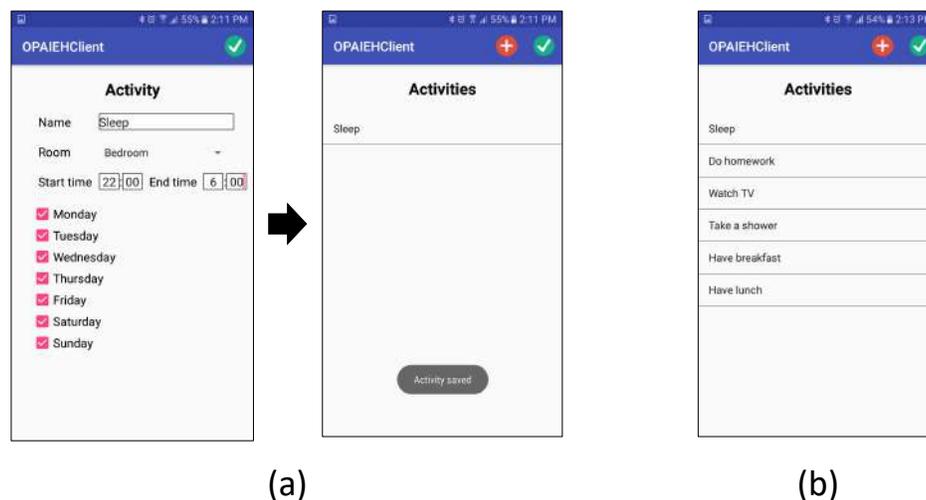
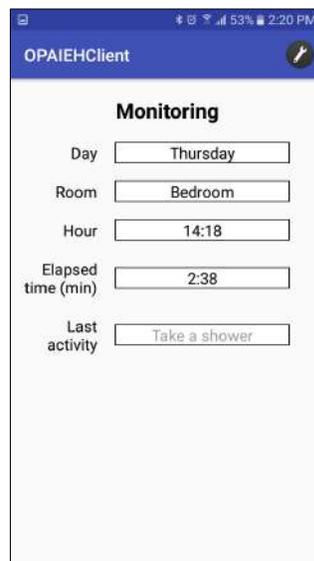


FIGURA 3.9: Registro de actividades. En (a) se muestra el registro de la actividad *Sleep*, y en (b) se muestra la interfaz después de registrar varias actividades.

- *Elapsed time (Duration)*. Es la duración de la actividad, es decir, el tiempo total transcurrido desde que el usuario entró a la habitación hasta que salió de ella.
- *Confirmación de la actividad*. Enviados los datos a OPAIEHServer, este los analiza y devuelve a OPAIEHClient el nombre de la posible actividad que el usuario realizaba. Con base en esto, el módulo de monitoreo se encarga de preguntarle al usuario si dicha actividad es la que se encontraba realizando o no, si el usuario responde *YES* el mensaje de confirmación es enviado al servidor (ver Fig. 3.11), si responde *NO*, entonces se le presenta un listado de actividades que se encuentran registradas en la base de datos de OPAIEHClient, y que se realizan en la misma habitación en la que el usuario se encontraba realizando la actividad, así el usuario tiene la posibilidad de seleccionar alguna de esas actividades y enviar el mensaje de confirmación a OPAIEHServer presionando el botón *OK* (ver Fig. 3.12), o en caso que no sea ninguna de esas seleccionar la opción *Another activity* e ingresar el nombre de la actividad que realizaba en el cuadro de texto que se le presenta, y presionar *OK* para enviar el mensaje de confirmación (ver Fig. 3.13). El mensaje de confirmación está compuesto por:
 - *Date*. Es la fecha en formato *aaaa:MM:dd HH:mm:ss* en la cual inicio la actividad.
 - *Room*. Es la habitación donde el usuario realizaba la actividad.
 - *Duration*. Es la duración de la actividad, es decir, el tiempo total transcurrido desde que el usuario entró a la habitación hasta que salió de ella.

- *Inferred_Activity*. Es un dato de tipo booleano, el cual tendrá valor de 1 si el usuario indica que la actividad inferida es la que se encontraba realizando, en caso contrario valdrá 0.
- *List_Activity*. Es un dato de tipo booleano, el cual tendrá el valor de 1 cuando el usuario indique que la actividad que realizaba no fue la inferida, pero esta se encontraba entre las opciones que se le mostraron, en caso contrario valdrá 0.
- *User_Activity*. Es el nombre de la actividad que el usuario indicó que se encontraba realizando. Si la actividad fue inferida correctamente, *User_Activity* toma el valor de la actividad inferida (ver Fig. 3.11). Si la actividad no fue inferida correctamente, pero fue seleccionada de la lista que se le proporcionó al usuario, *User_Activity* toma el valor de la opción seleccionada (ver Fig. 3.12). Si la actividad no se encontraba en el listado que se le proporcionó al usuario, por lo que este tuvo que ingresar el nombre de la actividad que realizaba, *User_Activity* será igual al valor ingresado (ver Fig. 3.13).



The screenshot shows a mobile application interface titled "OPAIEHClient" with a "Monitoring" section. It contains five input fields: "Day" with the value "Thursday", "Room" with "Bedroom", "Hour" with "14:18", "Elapsed time (min)" with "2:38", and "Last activity" with "Take a shower". The top status bar indicates a battery level of 53% and the time 2:20 PM.

FIGURA 3.10: Pantalla de monitoreo de OPAIEHClient antes de enviar los datos al servidor.

3.4.3. OPAIEHServer

OPAIEHServer es el servidor de la plataforma y está compuesto de tres módulos principales: *Aplicación Web*, *Base de datos* y *IAR-ONTO* (descrita en la sección 3.3). En el apéndice B se muestra el diagrama de clases de OPAIEHServer. A continuación se explicarán con más detalle los dos primeros módulos.

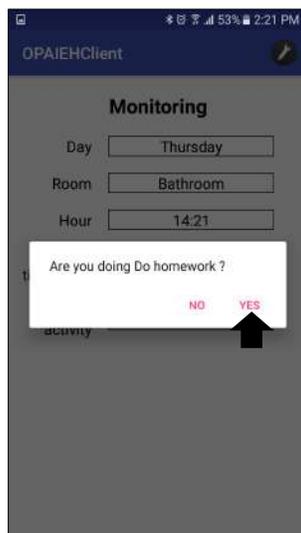


FIGURA 3.11: Pantalla de monitoreo de OPAIEHClient al recibir la inferencia de OPAIEHServer y esta resulta ser correcta.

Aplicación Web

Este módulo contiene los componentes que permiten registrar al usuario, las habitaciones y las actividades, así como realizar las consultas en la réplica de IAR-ONTO asignada a cada uno de los usuarios y administrar las confirmaciones de las actividades que ellos realizan al interior de su hogar.

- *Registro de usuario.* Este componente registra los datos personales del usuario que le son enviados por OPAIEHClient, en la base de datos de OPAIEHServer. Si el registro es exitoso, al usuario se le asigna un ID desde la base de datos.
- *Registro de habitaciones y actividades.* Gestiona el almacenamiento de las habitaciones y actividades en la réplica de IAR-ONTO que le corresponde a cada usuario, e informa al dispositivo cliente del éxito o fracaso de dicha operación. Además, este componente se encarga de almacenar *actividades predefinidas*, que son aquellas actividades que realizan la mayoría de los usuarios en el hogar en múltiples ocasiones del día (son actividades en las que la hora de inicio no es un parámetro que permita identificarlas), por lo que el servidor las registra por defecto en la base de conocimiento con un nombre, en una habitación, con hora de inicio 0, con una duración y todos los días de la semana (de lunes a domingo). En OPAIEH se tomaron en cuenta dos actividades predefinidas, *lavar platos (Washing dishes)*, con duración de 6 min) y *actividades del baño (Toilet activities)*, con duración de 5 min), esta última se refiere a las actividades sanitarias del usuario: necesidades fisiológicas y lavarse los dientes. Estas dos actividades son almacenadas en la réplica de IAR-ONTO

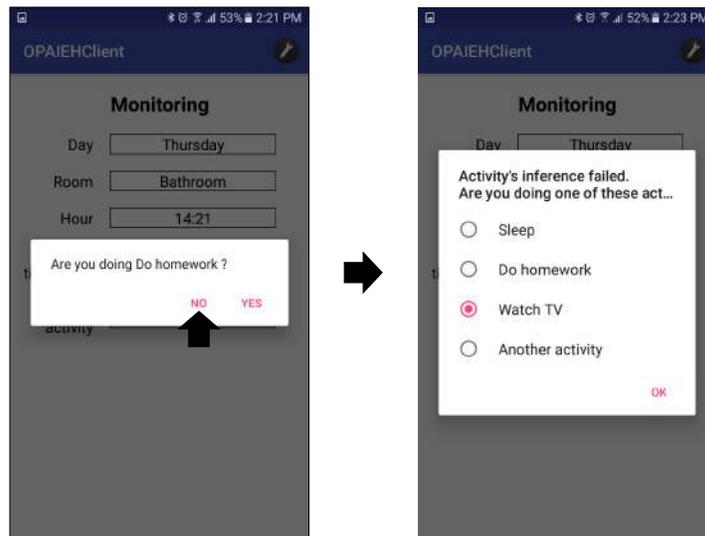


FIGURA 3.12: Pantalla de monitoreo de OPAIEHClient en caso que la actividad inferida no sea correcta, pero la actividad realizada por el usuario se encuentra registrada en la base de datos de OPAIEHClient.

que le corresponde al usuario, cuando éste registra las habitaciones *Kitchen* y *Bathroom* respectivamente. Es importante mencionar que las actividades predefinidas únicamente se almacenan en la réplica de IAR-ONTO del usuario, dado que OPAIEHServer no le informa a OPAIEHClient de la creación de las actividades predefinidas, y por esta razón no son registradas en la base de datos de OPAIEHClient.

- *Consulta de ontología.* Este componente se encarga de recibir los datos que le son enviados por OPAIEHClient desde el módulo de *Monitoreo*, y hacer consultas a la réplica de IAR-ONTO que le corresponde a dicho usuario, con la finalidad de determinar cuál es la posible actividad que este realizaba, y dársela a conocer al módulo de *Monitoreo* de OPAIEHClient, con la finalidad que el usuario realice la confirmación. En la siguiente sección (3.5) se explica con más detalle el procedimiento para realizar consultas en las réplicas de IAR-ONTO.
- *Administrador de confirmaciones.* Este componente se encarga de registrar las confirmaciones de las actividades realizadas por el usuario (desde OPAIEHClient) en la base de datos, en específico, en las tablas de confirmación de actividades (a cada usuario le corresponde una tabla o registro). Otra función del *Administrador de confirmaciones* es realizar consultas en la tablas de confirmación de actividades de la base de datos para extraer la información necesaria que le es solicitada por OPAIEHClient Web.

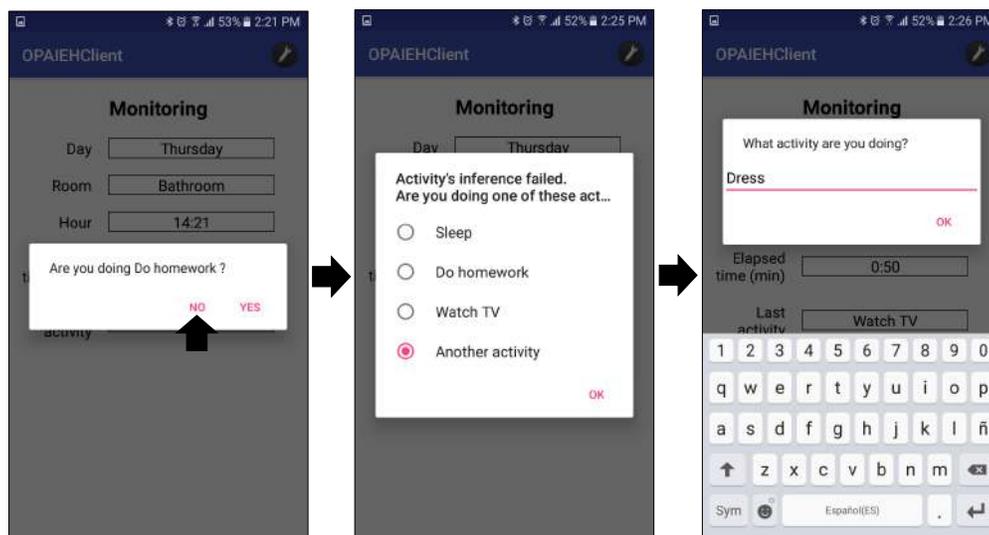


FIGURA 3.13: Pantalla de monitoreo de OPAIEHClient en caso que la actividad inferida no sea correcta y tampoco se encuentre registrada en la base de datos.

Base de datos

La base de datos de OPAIEHServer está compuesta por dos tipos de tablas: 1) tabla de usuarios; 2) tabla de confirmaciones de actividades.

La tabla de usuarios (*users*) solo cuenta con una instancia en la base de datos; esta tabla esta destinada al almacenamiento de los datos personales de los usuarios, de su contraseña para acceder a OPAIEHClient y a OPAIEHClient Web, así como la asignación del ID que le corresponderá al usuario en la plataforma. La tabla *users* está compuesta por 4 columnas, *ID*, *Username*, *Password*, y *Age*. La columna *ID* es la llave primaria de la tabla, la cual consiste en un número entero que se auto-incrementa (comenzando desde el 1) cada vez que se registra a un nuevo usuario, este ID se le asigna a cada usuario, con la intención de que OPAIEHServer pueda leer y escribir tanto en la réplica de IAR-ONTO que le corresponde al usuario, como en la tabla de confirmaciones de actividades que le fue asignada. La columna *Username* de tipo *Unique* almacena el nombre con el que el usuario se registró en la aplicación. La columna *Password* almacena la contraseña del usuario. La columna *Age* almacena la edad del usuario. En la Tabla 3.1 se muestra un ejemplo de registros almacenados en la tabla *users* de la base de datos.

La tabla de confirmaciones de actividades se requiere instanciar según el número de usuarios, es decir, a cada usuario le corresponderá una instancia de dicha tabla. Las instancias se nombrarán con el formato *userX*, de tal manera que la *X* sea reemplazada por el ID que le fue asignado a cada usuario. La tabla de confirmaciones de actividades está compuesta por 7 columnas, *No*, *Date*, *Room*, *Duration*, *Inferred activity*, *List activity*, y *User activity*. La columna *No*

TABLA 3.1: Ejemplo de registros en la tabla *users*.

ID	Username	Password	Age
1	José	jose1234	65
2	Juan	juanABCD	62
3	Ana	ana12AB	66

TABLA 3.2: Ejemplo de registro de confirmaciones de actividades de *userX*.

No	Date	Room	Duration	Inferred activity	List activity	User activity
1	2018-01-23 09:15:09	Bathroom	7	1	0	Toilet activities
2	2018-01-23 09:28:42	Dinning room	7	1	0	Have breakfast
3	2018-01-23 09:36:07	Kitchen	3	1	0	Washing dishes
⋮	⋮	⋮	⋮	⋮	⋮	⋮
43	2018-01-30 15:00:26	Bathroom	3	0	0	Tooth brush
⋮	⋮	⋮	⋮	⋮	⋮	⋮
73	2018-02-05 15:49:35	Bathroom	9	0	1	Take a shower

es la llave primaria de la tabla, la cual consiste en un número que se auto-incrementa (comenzando desde el 1) cada vez que se registra una nueva confirmación de actividad. La columna *Date* almacena la fecha y hora de inicio la actividad confirmada con el formato aaaa-MM-dd HH:mm:ss. Las columnas *Room* y *Duration* almacenan la habitación y la duración de la actividad confirmada respectivamente. Las columna *Inferred activity*, de tipo booleano, almacena un 1 si el usuario indicó que la actividad fue inferida correctamente, o un 0 en caso contrario. La columna *List activity*, de tipo booleano, almacena un 1 si el usuario seleccionó alguna actividad de la lista que se le presentó, o un 0 en caso contrario. La columna *User activity* almacena el nombre de la actividad que el usuario indicó que se encontraba realizando. En la Tabla 3.2 se muestra un ejemplo de confirmaciones de actividades almacenadas en la tabla *userX* de la base de datos.

3.4.4. OPAIEHClient Web

OPAIEHClient Web es una página Web que permite al cuidador del adulto mayor visualizar de forma gráfica las estadísticas de actividades (confirmaciones) que éste realiza, tomando en consideración minimizar la intrusión en la vida del adulto mayor, lo cual se consigue generando gráficas que expongan la menor cantidad de información acerca de la vida de los usuarios.

OPAIEHClient Web se encarga de solicitar información al *Administrador de confirmación*, la cual se encuentra contenida en las tablas de confirmaciones de los usuarios en la base de datos, con base en esta información, el componente *Controlador de gráficas* de este módulo se encarga de generar gráficas (así como actualizarlas) que permitan visualizar e interpretar mejor dicha información.

Para acceder a OPAIEHClient Web el cuidador del adulto mayor debe conocer el nombre de usuario y la contraseña del adulto mayor. En Fig. 3.14 se muestra la interfaz de login, así como parte de la interfaz que visualiza el cuidador del adulto mayor al acceder a la cuenta.

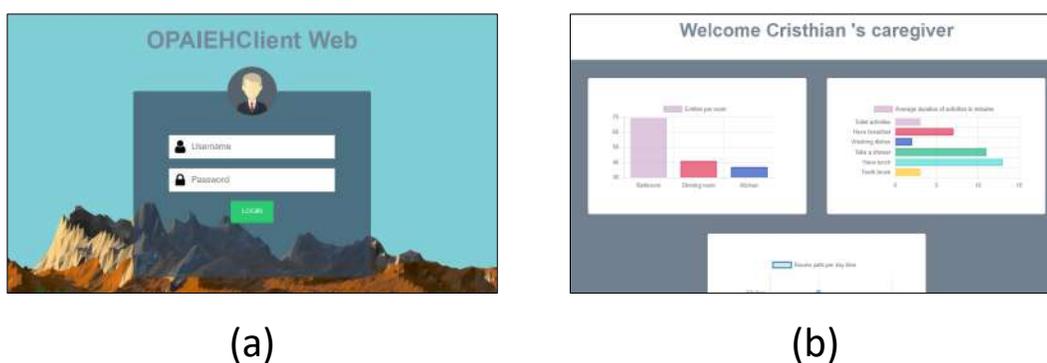


FIGURA 3.14: OPAIEHClientWeb. En (a) se muestra la interfaz de login, y en (b) se muestra la interfaz de usuario.

OPAIEHClient Web genera 5 tipos de gráficas diferentes, las cuales fueron creadas con las herramientas proporcionadas por las librerías ChartJS [54], D3 [55] y SIMILE Timeline [56]. A continuación se explican con mayor detalle cada una de las gráficas.

Número de entradas por habitación

Esta gráfica de tipo barras verticales creada con la librería Chart JS [54], representa el número de veces que el usuario accede a cada una de las habitaciones donde se encuentran instalados los sensores, en el eje horizontal se encuentran las habitaciones y en el eje vertical la frecuencia. Esta gráfica es de tipo global, dado que los datos graficados abarcan desde el inicio del monitoreo (desde que el usuario empezó a usar la plataforma OPAIEH) hasta el momento en que se accedió a la cuenta de OPAIEHClient Web. En Fig. 3.15 se muestra un ejemplo de este tipo de gráfica.



FIGURA 3.15: Gráfica de número de entradas por habitación.

Duración promedio de actividades

Esta gráfica de tipo barras horizontales creada con la librería Charts JS [54], representa la duración promedio en minutos de las actividades (confirmaciones) que realiza el usuario, en el eje horizontal se encuentra la duración promedio en minutos y en el eje vertical las actividades. Esta gráfica es de tipo global, dado que los datos graficados abarcan desde el inicio del monitoreo hasta el momento en que se accedió a la cuenta. En Fig. 3.16 se muestra un ejemplo de este tipo de gráfica.

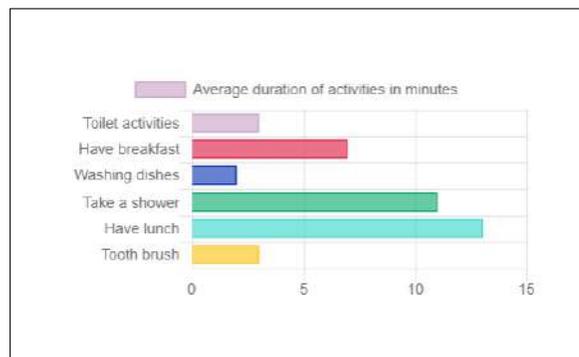


FIGURA 3.16: Gráfica de duración promedio de actividades.

Trayectoria del usuario entre las habitaciones de su hogar por día

Esta gráfica de puntos creada con la librería Chart JS [54], representa la trayectoria de las habitaciones que el usuario ha recorrido en el transcurso del día, en el eje horizontal se encuentra la hora y en el eje vertical la habitación. Esta gráfica permite al cuidador del adulto mayor seleccionar la trayectoria realizada por día. En Fig. 3.17 se muestra un ejemplo de este tipo de gráfica.



FIGURA 3.17: Gráfica de la trayectoria del usuario entre las habitaciones de su hogar por día.

Grafo dirigido de la relación entre las actividades y las habitaciones por día

Esta grafo dirigido creado con la librería D3 [55], representa la relación que existe entre la secuencia de las actividades realizadas en un día (Next), así como las habitaciones en donde estas se realizan (Done in). Las abreviaciones A_1, A_2, \dots, A_n indican la secuencia de realización de actividades, donde A_1 es la primera, seguida de A_2 , y así sucesivamente; de igual manera las abreviaciones R_1, R_2, \dots, R_n indican el orden en el cual las habitaciones fueron visitadas, de tal manera que la habitación R_1 fue la primera en visitarse en ese día, la siguiente fue R_2 , y así sucesivamente. Esta gráfica permite al cuidador del adulto mayor generar el grafo de un día en específico. En Fig. 3.18 se muestra un ejemplo de este tipo de gráfica.

Línea de tiempo de las actividades

Esta línea del tiempo creada con la librería SIMILE Timeline [56], representa todas las actividades que el usuario ha realizado desde que se inicio el monitoreo hasta el momento en que se accedió a la cuenta. La línea del tiempo permite visualizar todas las actividades mediante el desplazamiento horizontal por año, mes y hora. La visualización de las actividades en la línea

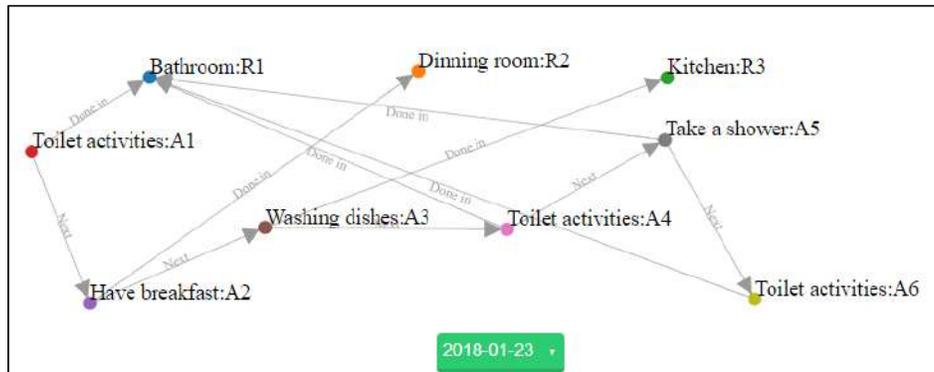


FIGURA 3.18: Grafo dirigido de la relación entre las actividades y las habitaciones por día.

del tiempo se muestra con un punto azul fuerte con el nombre de la actividad a lado, así como una barra horizontal de color azul claro cuya longitud es proporcional a la duración de la actividad. En Fig. 3.19 se muestra un ejemplo de este tipo de gráfica.

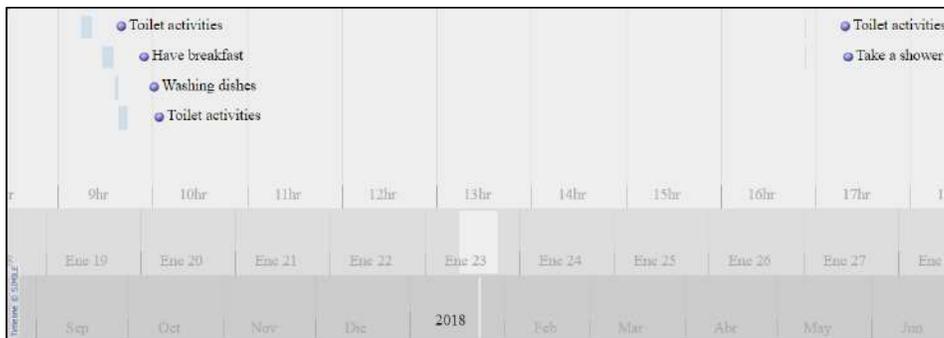


FIGURA 3.19: Línea de tiempo de las actividades.

3.5. Consultas de ontología

En esta sección se explica la manera en que se realizan las consultas en las réplicas IAR-ONTO correspondientes a los usuarios, las cuales contienen la información de las actividades

que realizan los usuarios al interior de su hogar. Las consultas se realizan en el lenguaje SPARQL [57].

Es importante recordar, que las consultas las realiza el módulo *Consulta de ontología* de *OPAIEHServer* cada que que el usuario se desplaza de una habitación a otra, dado que es cuando *OPAIEHClient* le envía los datos de la actividad que el usuario se encontraba realizando, los cuales son *Day*, *Room*, *StartTime*, y *Elapsed time*. Con base en esos parámetros el módulo de *Consulta de ontología* realiza dos consultas (o *queries* en inglés) diferentes: 1) consulta en términos de un intervalo de hora de inicio (*Query StartTime* o *QST*), el intervalo es de ± 10 con base a la hora de inicio recibida; 2) consulta en términos de un intervalo de duración (*Query Duration* o *QD*), los intervalos de duración son de ± 4 , ± 5 y ± 6 , para las actividades del baño, para las actividades de la cocina, y para el resto de las actividades respectivamente. En las Figuras 3.20 y 3.21 se muestran los formatos de *Query StartTime* y *Query Duration* respectivamente. Con base en los resultados que se obtengan con esas consultas se pueden presentar cuatro casos: 1) Que ambas consultas tengan resultado no nulo; 2) *Query StartTime* tenga resultado no nulo y *Query Duration* tenga resultado nulo; 3) *Query StartTime* tenga resultado nulo y *Query Duration* tenga resultado no nulo; 4) Que ambas consultas tengan resultado nulo. En la Tabla 3.3 se muestra una tabla de verdad con los 4 casos.

```
“SELECT ?X WHERE { ?X a:hasDay a:day . ?X
a:hasRoom a:room . ?X a:hasStartTime ?Y .
FILTER (?Y >= minRange && ?Y <= maxRange) }”
```

FIGURA 3.20: Consulta con base en un intervalo de hora de inicio (*Query StartTime* o *QST*).

```
“SELECT ?X WHERE { ?X a:hasDay a:day . ?X
a:hasRoom a:room . ?X a:hasDuration ?Y .
FILTER (?Y >= minRange && ?Y <= maxRange) }”
```

FIGURA 3.21: Consulta con base en un intervalo de duración (*Query Duration* o *QD*).

Con base en los datos de la Tabla 3.4, se ejemplificarán los 4 casos anteriormente mencionados. En Fig. 3.22 se muestra el diagrama de flujo de la realización de consultas en IAR-ONTO.

TABLA 3.3: Casos posibles de resultados de la realización de QST y QD.

Caso	QST	QD
1	1	1
2	1	0
3	0	1
4	0	0

TABLA 3.4: Ejemplos de actividades registradas en una réplica de IAR-ONTO.

Name	Day	Room	StartTime (min)	Duration (min)
Have breakfast	Monday,Tuesday,Wednesday,Thursday Friday,Saturday,Sunday	Dinning room	480	10
Have lunch	Monday,Tuesday,Wednesday,Thursday Friday,Saturday	Dinning room	840	20
Take a shower	Monday,Tuesday,Wednesday,Thursday Friday,Saturday,Sunday	Bathroom	900	12
Toilet activities	Monday,Tuesday,Wednesday,Thursday Friday,Saturday,Sunday	Bathroom	0	5

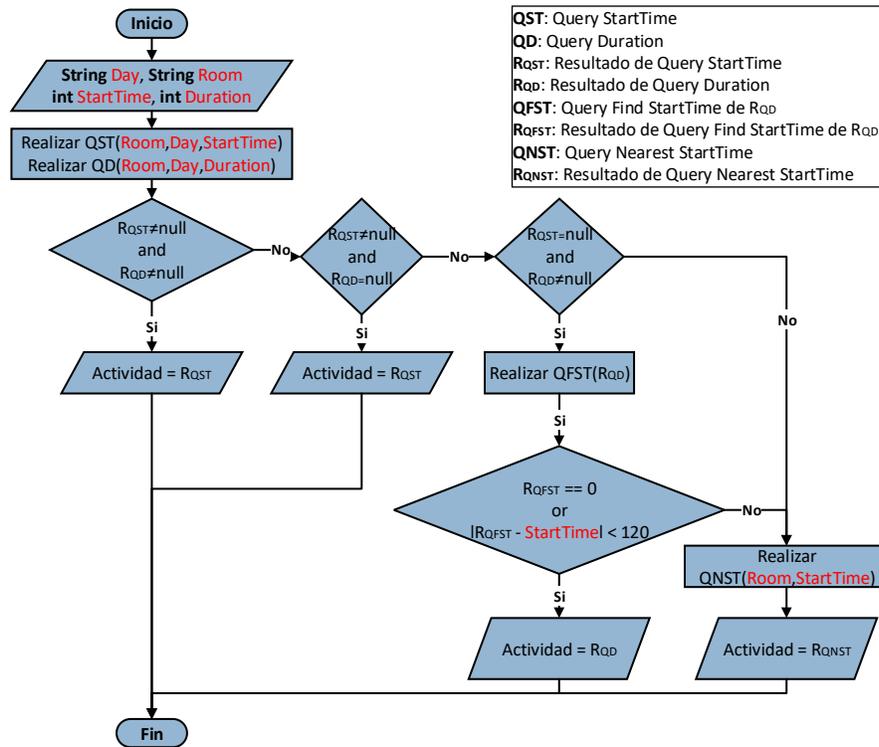


FIGURA 3.22: Diagrama de flujo del algoritmo de consultas en IAR-ONTO.

3.5.1. Caso 1

Si el módulo de Consulta de ontología de OPAIEHServer recibe los datos *Monday, Dinning room, 475 y 14*, este efectúa dos consultas, una con base en un intervalo de hora de inicio (Query StartTime) y otra con base en un intervalo de duración (Query Duration). La Query StartTime (en Fig. 3.20 se muestra el formato genérico en SPARQL de Query Starttime) en lenguaje natural es “Seleccionar la actividad que se realiza en el día Monday, en la habitación Dinning room, y con un intervalo de hora de inicio de 465 a 485” (475 ± 10), en la Tabla 3.4 se puede observar que la actividad *Have breakfast* es una actividad que se realiza en el día Monday, en la habitación Dinning room y su hora de inicio de 480 se encuentra incluida dentro del intervalo de hora de inicio, por lo tanto el resultado de la Query StartTime es la actividad *Have breakfast*. La Query Duration en lenguaje natural (en Fig. 3.21 se muestra el formato genérico en SPARQL de Query Duration) es “Seleccionar la actividad que se realiza en el día Monday, en la habitación Dinning room, y con un intervalo de duración de 8 a 20” (14 ± 6), en la Tabla 3.4 se puede observar que la actividad *Have breakfast* es una actividad que se realiza en el día Monday, en la habitación Dinning room y su duración de 10 se encuentra incluida dentro del intervalo de duración, por lo tanto el

resultado de la Query Duration es la actividad *Have breakfast*. Dado que ambas consultas tienen un resultado no nulo, la actividad inferida estará dada por el resultado de Query StartTime.

3.5.2. Caso 2

Con los datos *Tuesday, Bathroom, 908, 20*, el módulo de Consulta de ontología realiza Query StartTime y Query Duration, las cuales son, “*Seleccionar la actividad que se realiza en el día Tuesday, en la habitación Bathroom, y con un intervalo de hora de inicio de 898 a 918*” (908 ± 10), y “*Seleccionar la actividad que se realiza en el día Tuesday, en la habitación Bathroom, y con un intervalo de duración de 16 a 24*” (20 ± 4), respectivamente. Con base en los datos de la Tabla 3.4, se puede determinar que el resultado de Query StartTime es *Take a shower*, y el resultado de Query Duration es nulo, ya que en la actividad *Take a shower* se satisfacen dos de sus parámetros (día y habitación), sin embargo, la duración de dicha actividad (12) no se encuentra incluida en el intervalo de duración establecido en la Query Duration. Como únicamente Query StartTime tuvo resultado no nulo, la actividad inferida estará dada por el resultado de dicha consulta.

3.5.3. Caso 3

Si al realizar las Query StartTime y Query Duration resulta que la primera tiene resultado nulo pero la segunda no, la actividad inferida estará dado por Query Duration si se cumplen cualquiera de estas dos condiciones: 1) el resultado de Query Duration sea una actividad predefinida; 2) la actividad resultante de Query Duration tenga una hora de inicio con una diferencia menor o igual a 120 con respecto a la hora de inicio consultada. Si el resultado de Query Duration no cumple con ninguna de estas dos condiciones, se aplicará el método empleado en el Caso 4 (ver apartado 3.5.4).

Primera condición

Las Query StartTime y Query Duration con los datos *Saturday, Bathroom, 1200 y 8*, son “*Seleccionar la actividad que se realiza en el día Saturday, en la habitación Bathroom, y con un intervalo de hora de inicio de 1190 a 1210*” y “*Seleccionar la actividad que se realiza en el día Saturday, en la habitación Bathroom y con un intervalo de duración de 4 a 12*”, respectivamente. Con base en los datos de la Tabla 3.4, se puede determinar que el resultado de Query StartTime es nulo, puesto que no existe una actividad que se realice en el día Saturday, en la habitación Bathroom y con una hora de inicio dentro del intervalo de 1190 a 1210; por otra parte el resultado de Query Duration sería *Toilet activities*, dado que es una actividad que se realiza en el día Saturday, en la habitación Bathroom y su duración (5) se encuentra incluida dentro del intervalo de duración de la Query Duration. Como el resultado de Query StartTime es nulo y el de Query Duration no es nulo (*Toilet activities*), el siguiente paso es determinar la hora de inicio de la actividad resultante de Query Duration. La manera de determinar la hora de inicio de la actividad resultante de Query

Duration, es mediante la consulta Query Find StartTime o QFST por sus siglas en inglés, en Fig. 3.23 se muestra el formato genérico en SPARQL de este tipo de consulta. La QFST para la actividad Toilet activities en lenguaje natural es “Seleccionar la hora de inicio de la actividad con nombre Toilet activities”, el resultado de dicha consulta es 0 (ver Tabla 3.4). Como el resultado de Query Duration es una actividad predefinida, Toilet activities, la actividad inferida estará dada por el resultado de dicha consulta.

Segunda condición

Las Querys StartTime y Query Duration con los datos *Friday, Dinning room, 920* y *25*, son “Seleccionar la actividad que se realiza en el día Friday, en la habitación Dinning room, y con un intervalo de hora de inicio de 910 a 930” y “Seleccionar la actividad que se realiza en el día Friday, en la habitación Dinning room, y con un intervalo de duración de 19 a 31”, respectivamente. Con base en los datos de la Tabla 3.4, se puede determinar que el resultado de Query StartTime es nulo, puesto que no existe una actividad que se realice en el día Friday, en la habitación Dinning room y con una hora de inicio dentro del intervalo de 910 a 930; por otra parte el resultado de Query Duration sería *Have lunch*, dado que es una actividad que se realiza en el día Friday, en la habitación Dinning room y su duración (20) se encuentra incluida dentro del intervalo de duración de la Query Duration. Como el resultado de Query StartTime es nulo y el de Query Duration no es nulo (*Have lunch*), es necesario determinar la hora de inicio de la actividad resultante de Query Duration. La QFST en lenguaje natural para determinar la hora de inicio de la actividad *Have lunch* es “Seleccionar la hora de inicio de la actividad con nombre *Have lunch*”, el resultado de dicha consulta es 840 (ver Tabla 3.4). Como la hora de inicio de la actividad resultante de Query Duration (*Have lunch* con hora de inicio 840) tiene una diferencia de 80 con respecto a la hora de inicio consultada (920), la actividad inferida estará dada por el resultado de Query Duration.

```
“SELECT ?StartTime WHERE { ?activity
a:hasStartTime ?StartTime . ?activity
a:hasActivityName “ResultQueryDuration” }”
```

FIGURA 3.23: Consulta para conocer el StartTime del resultado de Query Duration (Query Find StartTime o QFST).

3.5.4. Caso 4

Con los datos *Wednesday, Dinning room, 920* y *27*, las consultas Query StartTime y Query Duration son “Seleccionar la actividad que se realiza en el día Wednesday, en la habitación Dinning room, y con un intervalo de hora de inicio de 910 a 930” y “Seleccionar la actividad que se realiza en el

día Wednesday, en la habitación Dinning room y con un intervalo de duración de 21 a 33” respectivamente. Con base en los datos de la Tabla 3.4, se puede determinar que tanto el resultado de Query StartTime como el de Query Duration (la duración de Have lunch, 20, no se encuentra incluida en el intervalo de duración) son nulos. Debido a que los resultado de ambas consultas son nulos, se prosigue a realizar el método Query Nearest StartTime o QNST por sus siglas en inglés, el cual contiene las consultas que llevan el mismo nombre. El método consiste en realizar consultas con base en la habitación y la hora de inicio inicialmente consultadas (Dinning room y 920), sin tomar en cuenta el día de la realización de la actividad. Tomando como referencia la hora de inicio, en la iteración i se realizan dos consultas, una que se emplea cuando la hora de inicio se retrocede en una unidad y otra que se emplea cuando la hora de inicio se adelanta en una unidad, con la finalidad de determinar la actividad con la hora de inicio más cercana a la hora de inicio consultada. En Fig. 3.24 se muestra el formato genérico en SPARQL de este tipo de consultas. Se realizarán tantas iteraciones hasta que alguna de las consultas (Forward o Backward) obtenga un resultado no nulo. Con los datos Dinning room y 920, tras 80 iteraciones se consigue un resultado no nulo en la QNST Backward, el cual es la actividad *Have lunch*, y para QNST Forward cuya hora de inicio es 1000, su resultado es nulo. Con base en los resultados obtenidos de las consultas QNST, la actividad inferida estará dada por QNST Backward.

```
“SELECT ?X WHERE { ?X a:hasRoom a:room . ?X
a:hasStartTime StartTimeBackward }”
```

(a)

```
“SELECT ?X WHERE { ?X a:hasRoom a:room . ?X
a:hasStartTime StartTimeForward }”
```

(b)

FIGURA 3.24: Consultas para conocer la actividad con hora de inicio más próxima a la hora de inicio consultada (Query Nearest StarTime o QNST). Las consultas (a) y (b) tienen la misma estructura, con la diferencia que (a) se emplea cuando la hora de inicio consultada se retrocede (Backward), y (b) se emplea para cuando la hora de inicio se adelanta (Forward).

TABLA 3.5: Actividades registradas por el usuario en su réplica correspondiente de IAR-ONTO.

Name	Day	Room	StartTime (min)	Duration (min)
Have breakfast	Monday,Tuesday,Wednesday,Thursday Friday,Saturday,Sunday	Dinning room	480	10
Have lunch	Monday,Tuesday,Wednesday,Thursday Friday,Saturday	Dinning room	840	20
Take a shower	Monday,Tuesday,Wednesday,Thursday Friday,Saturday,Sunday	Bathroom	900	12
Toilet activities	Monday,Tuesday,Wednesday,Thursday Friday,Saturday,Sunday	Bathroom	0	5
Washing dishes	Monday,Tuesday,Wednesday,Thursday Friday,Saturday,Sunday	Kitchen	0	6

3.6. Escenario de prueba, resultados y análisis

3.6.1. Escenario de prueba

En esta sección se explica el escenario de prueba, así como los resultados obtenidos en la prueba experimental realizada; la finalidad de dicha prueba consiste en determinar la precisión con la que OPAIEH realiza la inferencia de las actividades, la cual está dada por la ecuación 3.1.

$$\text{Precisión} = \frac{\text{Actividades inferidas correctamente}}{\text{Total de actividades}} \times 100 \quad (3.1)$$

Debido a que el diseño de OPAIEHClient (la aplicación móvil) está basado en retroalimentación, su uso por parte de adulto mayores sería complicado; es por esta razón que con la finalidad de probar el funcionamiento y determinar la precisión de OPAIEH, el usuario final de la prueba experimental no fue un adulto mayor, si no una persona de mediana edad capaz de manejar un smartphone sin dificultad alguna.

Para llevar a cabo la prueba experimental se requirieron algunos preparativos preliminares, los cuales consistieron en la instalación de los sensores Beacons en 3 de las habitaciones que el usuario estipuló (Baño, Cocina y Comedor); la instalación de OPAIEHClient en su smartphone, el cual es un Galaxy J5 con versión de Android 6.0.1, 1.5 GB de RAM y una versión de Bluetooth 4.1; el registro del usuario en la plataforma, así como el registro de las habitaciones y las actividades, en la Tabla 3.5 se muestra la relación de las actividades y las habitaciones que el usuario registró.

La prueba experimental consistió en que el usuario debía utilizar OPAIEHClient (monitoreo y confirmación de actividades) durante 4 semanas (28 días) cada vez que fuera a realizar algunas de las actividades que registró desde dicha aplicación móvil (Have breakfast, Have lunch, Take a shower), así como las actividades involucradas en las actividades predefinidas (Toilet activities y Washing dishes), previamente se le explicó al usuario en que consistían dichas actividades predefinidas.

3.6.2. Resultados y análisis

Para exponer los resultados es necesario entender bien la estructura de las tablas de confirmaciones de actividades (ver sección 3.4.3). Tomando en cuenta la estructura de la Tabla 3.2, en la presentación y análisis de los resultados se brindará principal atención en las columnas *Inferred activity* y *List activity*, con base en las cuales las actividades confirmadas se pueden agrupar en tres categorías: 1) actividad inferida correctamente, 2) actividad no inferida correctamente pero se encontraba registrada en la base de datos de OPAIEHClient; 3) actividad no inferida correctamente y no se encontraba en la base de datos de OPAIEHClient.

El total de actividades confirmadas al cabo de los 28 días que duró la prueba experimental fue de 147, en Fig. 3.25 se muestra la cantidad de actividades confirmadas por cada categoría.



FIGURA 3.25: Gráfica de las actividades confirmadas por categoría.

Con base en los datos proporcionados en Fig. 3.25, y aplicando la fórmula de la ecuación 3.1, se puede calcular la precisión de OPAIEHClient en la inferencia de actividades, la cual es de 96.6 %.

En Fig. 3.26 se muestra un desglose de las inferencias realizadas por actividad registrada en la réplica de IAR-ONTO del usuario; se puede observar que de las 5 actividades registradas, 4 de ellas (Have breakfast, Have lunch, Toilet activities y Washing dishes) obtuvieron un 100 %

TABLA 3.6: Registro de actividades no inferidas.

No	Date	Room	Duration	Inferred activity	List activity	User activity
5	2018-01-23 16:53:23	Bathroom	1	0	1	Take a shower
43	2018-01-30 15:00:26	Bathroom	3	0	0	Tooth brush
73	2018-02-05 15:49:35	Bathroom	9	0	1	Take a shower
87	2018-02-07 13:50:33	Bathroom	9	0	1	Take a shower
136	2018-02-18 11:40:39	Bathroom	1	0	1	Take a shower

de inferencia acertada, por otra parte, Take a shower en 4 ocasiones no fue inferida correctamente, esto quiere decir que la plataforma OPAIEH infirió otra actividad que no era esa, y el usuario retroalimentó a la plataforma corroborando que en realidad se encontraba realizando la actividad Take a shower.

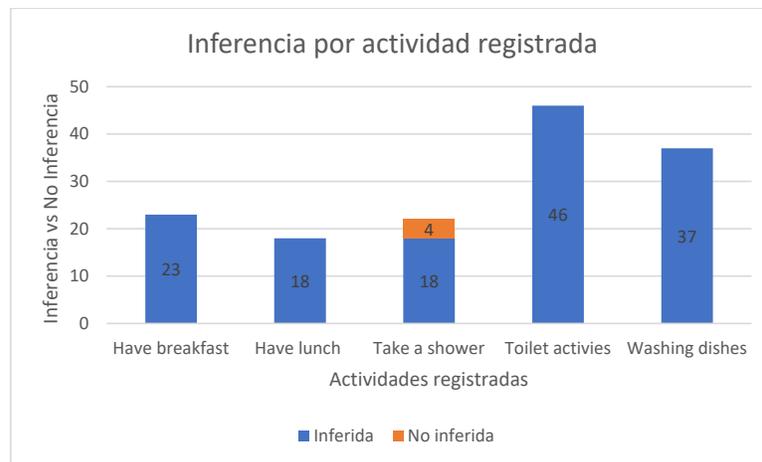


FIGURA 3.26: Gráfica de la inferencia por actividades registradas.

Debido a que las actividades no inferidas (4 registradas y 1 no registrada) fueron pocas, a continuación se procede a hacer el análisis del por qué OPAIEH no las infirió correctamente. En la Tabla 3.6 se muestra el registro de las 5 confirmaciones de las actividades no inferidas.

Actividad no inferida No. 5

Tomando en cuenta la fecha y hora de inicio de esta actividad (2018-01-23 16:53:23), se puede determinar que esta se realizó en un día Tuesday y con una hora de inicio de 1013 ($16 \times 60 + 53$). Por lo tanto los datos que recibió el módulo *Consulta de ontología* fueron Tuesday, Bathroom,

1013 y 1. Las Query StartTime y Query Duration son “Seleccionar la actividad que se realiza en el día Tuesday, en la habitación Bathroom y con un intervalo de hora de inicio de 1003 a 1023” y “Seleccionar la actividad que se realiza en el día Tuesday, en la habitación Bathroom y con un intervalo de duración de 1 a 5” respectivamente. Con base en las consultas y en los datos de la Tabla 3.5, se puede determinar que el resultado de QST es nulo y el de QD es Toilet activities. La actividad inferida por OPAIEHServer fue Toilet activities, sin embargo el usuario retroalimentó a la plataforma indicando que la actividad que en realidad se encontraba realizando era Take a shower (la cual le fue presentada por OPAIEHClient como otras opciones). Ahora bien, en Fig. 3.27 se muestra la gráfica de duración promedio de actividades obtenida de OPAIEHClient Web, en la cual se muestra que la duración promedio de la actividad Take a shower es de 11 minutos, entonces ¿Existe la posibilidad que el usuario se haya bañado en 1 minuto? La respuesta es no, entonces, ¿Por qué se presenta esta situación? Esto ocurre debido a que la señal Bluetooth es volátil, el método implementado en la sección 3.4.2 solo logró minimizar más no erradicar dicho inconveniente. Lo que sucede es que a pesar que el usuario permanece en una habitación, y por ende se supone que la señal de mayor intensidad que reciba OPAIEHClient será de aquel sensor que se encuentra en dicha habitación, en ocasiones el módulo de monitoreo de OPAIEHClient recibe con mayor intensidad la señal proveniente de otro sensor ubicado en otra habitación, por lo que OPAIEHClient interpreta como un cambio de habitación, y envía los datos de la actividad que realizaba el usuario (Day, Room, Start Time y Duration) a OPAIEHServer.

De igual forma, la actividad No. 136 no fue inferida por la misma razón que la actividad No. 5.

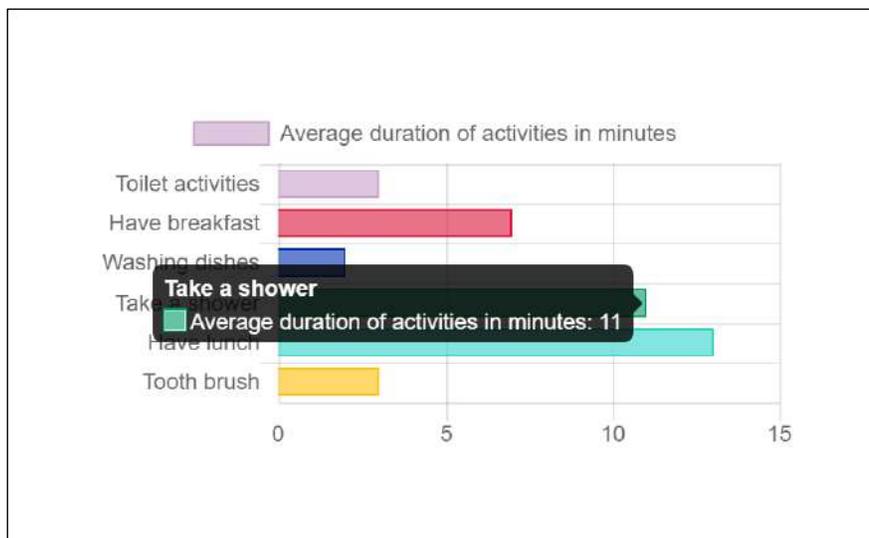


FIGURA 3.27: Duración promedio de la actividad Take a shower.

Actividad no inferida No. 43

Las consultas realizadas por el módulo de Consulta de ontología para la actividad No. 43 fueron con los datos Tuesday, Bathroom, 900 y 3. Los resultados de las Query StartTime y Query Duration fueron respectivamente Take a shower y Toilet activities, por lo que la actividad inferida por OPAIEHServer fue Take a shower. Con los datos de la confirmación de la actividad No. 43 se puede determinar que el usuario retroalimentó que la actividad que se encontraba no era Take a shower, además como la única actividad registrada en OPAIEHClient que se realiza en la habitación Bathroom es Take a shower (recordar que las actividades predefinidas únicamente se guardan en la ontología), ninguna otra actividad le fue presentada al usuario por OPAIEHClient, por lo que el usuario retroalimentó que se encontraba realizando la actividad *Tooth brush*, actividad que si bien forma parte de la definición de Toilet activities, como se comentó anteriormente está no es mostrada dentro de las opciones presentadas al usuario, esto se realizó con la finalidad de determinar que actividades suele hacer el usuario en las habitaciones donde fueron registradas las actividades predefinidas.

Actividad no inferida No. 73

Las consultas realizadas por el módulo de Consulta de ontología para la actividad No. 73 fueron con los datos Monday, Bathroom, 949 y 9. Los resultados de las Query StartTime y Query Duration fueron nulo y Toilet activities respectivamente, por lo que la actividad inferida por OPAIEHServer fue Toilet activities. Con los datos de la confirmación de la actividad No. 73 se puede determinar que el usuario retroalimentó a la plataforma que la inferencia fue errónea, e indicó que la actividad Take a shower era la que en realidad se encontraba realizando. La inferencia no acertada de la actividad Take a shower se debe a que el usuario no realizó la actividad en un intervalo de hora de inicio de 900 ± 10 minutos, por lo que la QST da un resultado nulo, además como la QD obtiene como resultado la actividad predefinida Toilet activities, esta será la actividad inferida.

De igual forma, la actividad No. 87 no fue inferida por la misma razón que la actividad No. 73.

3.7. Conclusiones

En este apartado de la tesis se desarrolló OPAIEH, una plataforma basada en ontologías para el reconocimiento de las actividades que realizan los adultos mayores al interior del hogar. El uso de las ontologías OWL permitió crear de manera fácil y sencilla (gracias a la semántica que estas proveen) una base de conocimiento (IAR-ONTO) basada en la definición de actividad propuesta en este proyecto de tesis, la cual permite almacenar la información acerca de las actividades que realizan los usuarios al interior de su hogar. IAR-ONTO es la pieza clave del

reconocimiento de actividades, dado que las inferencias se realizan en torno a esta, mediante la realización de consultas en formato SPARQL.

Los resultados obtenidos por OPAIEH fueron satisfactorios, dado que de las 147 confirmaciones realizadas por parte del usuario, 142 fueron inferidas correctamente y solo 5 no, con base en esto se puede decir que la precisión de OPAIEH es de un 96.6 %. Además es importante recalcar que 2 de las 5 actividades no inferidas, se debieron a problemas técnicos que existen con la señal Bluetooth, la cual en ocasiones resulta ser volátil.

Pese a que OPAIEH no pudo probarse con adultos mayores, debido a su diseño e implementación, esta sirve como una exitosa prueba de concepto para determinar la eficiencia del uso de las ontologías para el reconocimiento de actividades.

Capítulo 4

Reconocimiento de actividades basado en redes neuronales

En este capítulo se presenta el diseño y desarrollo de la red neuronal IAR-NN (Indoor Activity Recognition Neural Network) para el reconocimiento de actividades al interior del hogar de los adultos mayores, dicha red neuronal fue diseñada con una arquitectura feedforward fully connected, y se empleó el algoritmo backpropagation para su entrenamiento.

4.1. Marco teórico

En esta sección se presenta de manera breve la historia de las redes neuronales, se detallan de manera general las arquitecturas (o topologías) más comunes empleadas para el diseño de las redes neuronales, se expone la clasificación de los algoritmos empleados para el aprendizaje de las redes neuronales, se explica el algoritmo backpropagation, así como el método de validación cruzada de n capas (k-fold cross-validation en inglés). Además, se presentan los trabajos relacionados con el reconocimiento de actividades mediante redes neuronales.

4.1.1. Introducción, breve historia y aplicaciones de las redes neuronales

Una red neuronal artificial (ANN por sus siglas en inglés) es en términos simples un modelo computacional biológicamente inspirado, que consiste en elementos de procesamiento (llamados neuronas), y conexiones entre ellos con coeficientes (pesos) vinculados a las conexiones [58]. En [59] describen a las redes neuronales artificiales como invenciones matemáticas inspiradas en los sistemas biológicos, el propósito de una red neuronal es mapear una entrada en una salida deseada, tal como lo hace una función matemática. En [60] definen a las redes neuronales como un sistema de procesamiento de información que tienen características de rendimiento en común con las redes neuronales biológicas. El hecho de que existan muchas definiciones acerca de lo que son las redes neuronales se debe a que existen diferentes paradigmas, cuyo diferencia radica en cómo entrenar las redes neuronales y dónde usarlas.

La historia de las redes neuronales artificiales empezó a inicios de 1940 [61], Warren McCulloch y Walter Pitts fueron los primeros en introducir un modelo matemático de una neurona [62]; su neurona artificial realizaba operaciones lógicas con dos o más entradas y producía una salida si un valor umbral era excedido [63]. En 1949, Donal Hebb diseñó la primera ley de entrenamiento para redes neuronales [64]. A finales de 1950 se dio la primera aplicación práctica a las redes neuronales, Frank Rosenblatt inventó el perceptron [65], una red neuronal para el reconocimiento de patrones. En 1960 fue desarrollado ADALINE [66] por Bernard Widrow y Marcian Hoff, fue la primera aplicación de las redes neuronales de uso comercial, era empleada para el procesamiento de señales, se podía encontrar en casi todos los teléfonos analógicos, su algoritmo de entrenamiento era la regla de Widrow-Hoff o la regla delta. En 1969 Marvin Minsky y Seymour Papert publicaron un análisis matemático preciso [67] del perceptron donde demostraron que este no era capaz de resolver el problema XOR, de hecho demostraron que una capa única de perceptrones (single-layer perceptrons en inglés) solo podían separar categorías que eran linealmente separables. Debido a la publicación de Minsky y Seymour y al hecho que no existían computadoras digitales poderosas en las cuales hacer experimentos, muchas personas creyeron que las investigaciones sobre las redes neuronales se encontraban en un callejón sin salida. Por un década la investigación en redes neuronales fue parcialmente suspendida [68].

Dos nuevos conceptos fueron los responsables del renacimiento en la investigación de las redes neuronales. El primero fue el uso de la mecánica estática para explicar la operación de cierta clase de redes recurrentes que podían ser usadas como memoria asociativa, las redes Hopfield, en honor a su creador John Hopfield, el cual se inspiró en las leyes del magnetismo. El segundo concepto desarrollado fue el algoritmo backpropagation (una generalización de la regla delta) para el entrenamiento de redes neuronales multicapa, este descubrimiento fue realizado independientemente por muchos investigadores, pero la publicación más influyente fue la realizada por David Rumelhart y James McClelland [69]. Los problemas linealmente separables podían ser resueltos por perceptrones multicapa [61]. Estos nuevos desarrollos revitalizaron el campo de las redes neuronales. Desde la década de 1980, se han escrito miles de artículos, las redes neuronales han encontrado innumerables aplicaciones, y el campo se ha mantenido activo con nuevos trabajos teóricos y prácticos [68].

Entre las aplicaciones de las redes neuronales se encuentran:

- Aproximación de funciones.
- Asociación de patrones.
- Agrupamiento de datos, clasificación, categorización y conceptualización.
- Aprendizaje de parámetros estadísticos.
- Acumulación de conocimiento mediante entrenamiento.
- Extracción de conocimiento a través del análisis de los pesos de las conexiones entre neuronas.

- Inserción de conocimiento en una red neuronal con la finalidad de razonamiento aproximado [58].

La aplicación que se empleará en IAR-NN para el reconocimiento de actividades será la clasificación, dado que el reconocer o inferir las actividades que el adulto mayor realiza al interior de su hogar es posible de lograr mediante la clasificación de estas.

4.1.2. Características de las redes neuronales

Antes de analizar las características de una red neuronal, es importante conocer la estructura de su componente básico, la neurona artificial, en Fig. 4.1 se muestra su esquema. La neurona artificial consiste en diversas partes: entradas (x_1, x_2, \dots, x_n), pesos ($w_1, w_2, w_3, \dots, w_n, w_b$), una sumatoria (Σ), y finalmente su función de activación ($f(\cdot)$) o de transferencia [70]. La salida de la neurona estará dada por:

$$z = f\left(\sum_{i=1}^n w_i x_i + w_b\right) \quad (4.1)$$

Es importante mencionar que la ecuación 4.1 carece de una definición formal de función de transferencia f , más adelante se hablará más al respecto. Otro punto a considerar es que por simplicidad de los diagramas no se incluirá el nodo bias en los esquemas de las redes neuronales, pero si se incluirá en los cálculos.

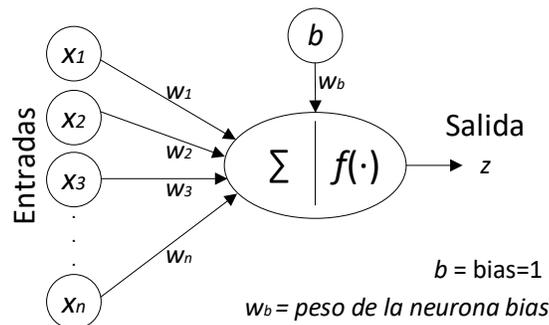


FIGURA 4.1: Estructura de una neurona artificial.

Una red neuronal se compone de tres elementos: 1) el patrón de conexiones entre las neuronas (su arquitectura o topología); 2) la función de activación o de transferencia de sus neuronas; 3) el método para determinar los pesos entre las conexiones (el algoritmo de aprendizaje) [60].

Arquitecturas de las redes neuronales

El arreglo de las neuronas en capas (layers en inglés), y los patrones de conexión dentro y entre esas capas se conoce como la arquitectura de la red neuronal. Las redes neuronales generalmente son clasificadas como unicapa o multicapa, de acuerdo al número de capas que posean. Las neuronas de entrada no se cuentan como una capa, dado que estas no realizan ningún cálculo. El número de capas en la red se puede definir de acuerdo al número de capas de conexiones con pesos que interconectan a las neuronas entre bloques de neuronas [60]. En Fig. 4.2 se muestran ejemplos de redes neuronales unicapa y multicapa respectivamente.

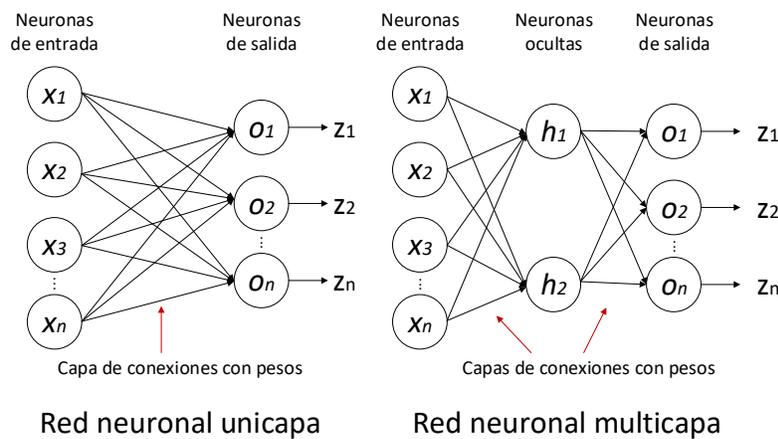


FIGURA 4.2: Redes neuronales unicapa y multicapa.

De acuerdo a la forma en que estén conectadas las neuronas entre capa y capa, las redes neuronales se pueden clasificar en parcialmente conectadas (partially connected) o totalmente conectadas (fully connected) [58]. Una red neuronal totalmente conectada es aquella en la que cada neurona i está conectada a todas las neuronas de la siguiente capa [61], a diferencia de las redes neuronales parcialmente conectadas en las cuales esta restricción no se cumple. En Fig. 4.3 se muestran ejemplos de redes neuronales parcialmente conectadas y totalmente conectadas respectivamente.

Según la dirección de las conexiones entre las neuronas, éstas se pueden clasificar en feed-forward (alimentación hacia adelante) y feedback (alimentación hacia atrás). En una red feed-forward cada neurona de una capa en específico tiene únicamente conexiones con las neuronas de la siguiente capa (las conexiones van en dirección hacia la capa de salida) [61]. Por otra parte, en una red feedback existen conexiones que van desde las neuronas de salida a las neuronas de entrada [58]. En Fig 4.4 se muestran ejemplos de redes neuronales feedforward y feedback respectivamente.

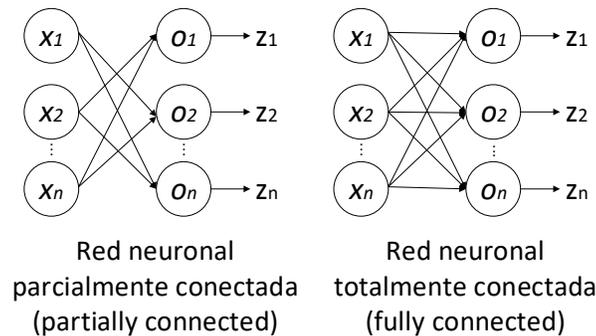


FIGURA 4.3: Redes neuronales parcialmente conectadas y totalmente conectadas.

Funciones de activación

La función básica de una neurona artificial consiste en sumar su señal de entrada ponderada y aplicar una función de activación. Para las neuronas entradas, esta función es la función de identidad (Fig. 4.5). Para las demás neuronas, el diseñador de la red neuronal tiene para escoger entre varias funciones de transferencia, como se muestra en Fig. 4.6, las cuales se pueden emplear tanto en forma unipolar (con salidas de 0 o 1) o bipolar (con salidas de -1 y 1). Sin embargo, la función de transferencia más comúnmente utilizada es la sigmoidea o logística, esto se debe a las propiedades matemáticas que posee (monotonidad, continuidad y diferenciabilidad), que son muy útiles cuando se entrena una red neuronal con gradiente descendente [59]. Por lo general, la misma función de activación se usa para todas las neuronas en cualquier capa particular de una red neuronal [60].

En el diseño de IAR-NN se empleará la función de activación sigmoidea con salida unipolar en todas las neuronas, a excepción de las neuronas de entrada cuya función de activación es la de identidad. En la ecuación 4.2 se muestra la salida de la neurona de Fig. 4.1 al aplicarle la función de activación sigmoidal unipolar.

$$z = \frac{1}{1 + e^{-(\sum_{i=1}^n w_i x_i + w_b)}} \quad (4.2)$$

Algoritmos de aprendizaje

La característica más interesante de las redes neuronales es su generalización, es decir, su capacidad de familiarizarse con los problemas mediante el entrenamiento y, después de un entrenamiento suficiente, sean capaz de resolver problemas desconocidos de la misma clase. El implementar el aprendizaje en una red neuronal se puede realizar de diversas maneras, sin

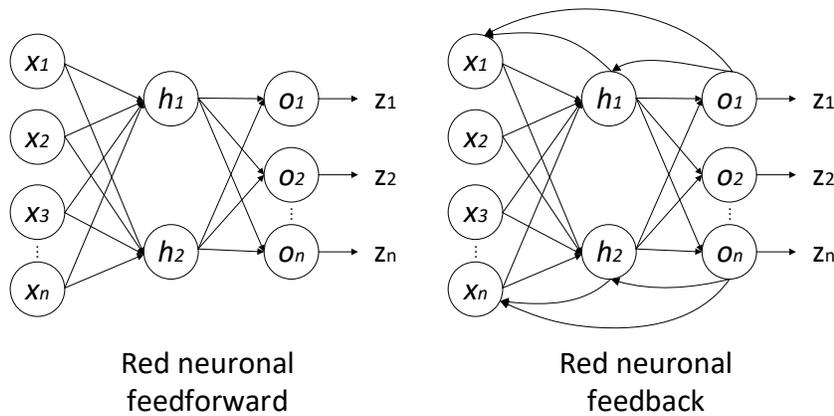


FIGURA 4.4: Redes neuronales feedforward y feedback.

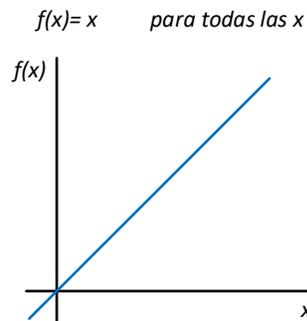


FIGURA 4.5: Función identidad.

embargo, el método más común es la modificación de los pesos de las conexiones entre las neuronas [61].

Los algoritmos de aprendizaje son aquellos que se utilizan para entrenar a las redes neuronales, es decir, son los métodos que permiten establecer los valores de los pesos de las conexiones entre las neuronas. Por conveniencia, estos algoritmos han sido clasificados en dos categorías [60]:

- *Aprendizaje supervisado*. Los ejemplos de entrenamiento consisten de vectores de entrada (x) y vectores de salida deseados o etiquetas (y). El entrenamiento se realiza hasta que la red neuronal “aprende” a asociar cada vector de entrada (x) con su correspondiente vector de salida (y) [58]. Algunas de las técnicas más populares de aprendizaje supervisado son la regla de Hebb, la regla delta y backpropagation [60].

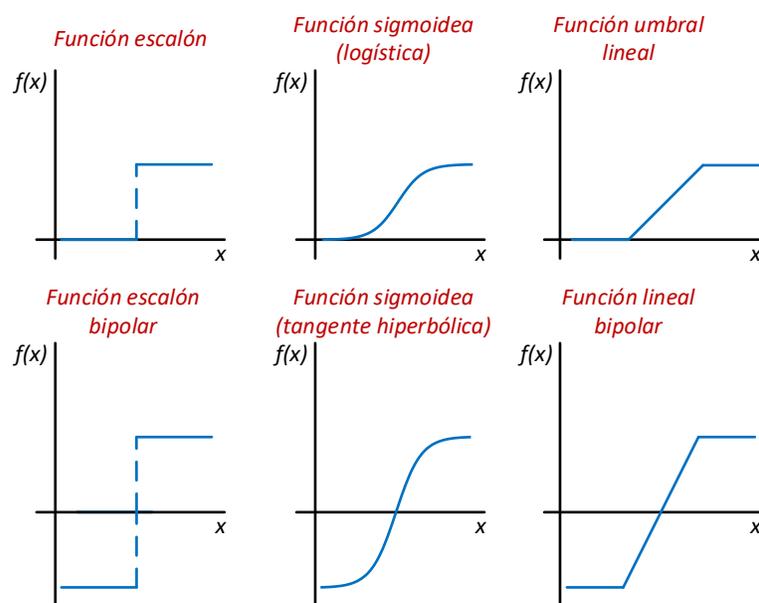


FIGURA 4.6: Tres diferentes tipos de funciones de activación (escalón, sigmoidea, lineal) en formato unipolar y bipolar.

- *Aprendizaje no supervisado.* Solo los vectores de entrada (x) son proporcionados. El entrenamiento se realiza hasta que la red neuronal identifica patrones similares y con base en ellos clasifica los vectores de entrada (x) en categorías similares [61]. Dos de las más populares técnicas de aprendizaje no supervisado son Self-Organizing Map (SOM) y Adaptive Resonance Theory (ART) [59].

IAR-NN será entrenada mediante aprendizaje supervisado, en específico con el algoritmo backpropagation.

4.1.3. Algoritmo backpropagation

Cuando se habla del método de entrenamiento backpropagation, se hace referencia a una red neuronal multicapa con conexión feedforward entrenada mediante el algoritmo backpropagation.

El entrenamiento de una red mediante el algoritmo backpropagation involucra tres etapas: 1) la etapa feedforward ; 2) el cálculo y retropropagación (backpropagation) del error asociado; 3) el ajuste o actualización de los pesos. Después del entrenamiento, durante la fase de prueba o evaluación, solo se realiza la etapa de feedforward. A continuación se describe como funciona el algoritmo backpropagation para una red neuronal con una capa de neuronas ocultas.

Durante la fase feedforward, cada neurona de entrada (x_i) recibe una señal de entrada y la trasmite a cada una de las neuronas ocultas de la siguiente capa. Cada neurona oculta (h_i) calcula su salida mediante su función de activación y la envía a cada una de las neuronas de salida. Cada neurona de salida (o_i) calcula su salida para formar la respuesta de la red para el ejemplo de entrenamiento dado.

Durante el entrenamiento, cada neurona de salida compara su salida con el valor esperado (t_k) para determinar el error asociado para ese ejemplo de entrenamiento en esa neurona de salida, basado en ese error se calcula el factor δ_k , el cual es usado para distribuir el error asociado a la neurona de salida o_i hacia las demás neuronas en la capa previa (las neuronas conectadas a o_i). De manera similar, el factor δ_h es calculada para cada neurona oculta (h_i).

Después de determinar todos los factores δ , los pesos de las conexiones de todas las capas de neuronas son actualizados. El ajuste de los pesos se realiza con base en el valor de δ y la salida de la neuronas [60]. En el apéndice C se muestra de forma detallada el algoritmo backpropagation.

4.1.4. Entrenamiento y prueba

Para realizar de manera adecuada el entrenamiento y las pruebas correspondientes, es necesario conocer algunos conceptos importantes como son: época, conjunto de entrenamiento (training set), conjunto de prueba (test set), validación cruzada de k iteraciones (k-fold cross validation). Además, es necesario definir el concepto de desempeño, al cual se le dio un significado especial en este proyecto de tesis en el ámbito de la red neuronal diseñada y del reconocimiento de actividades.

Una *época* es un ciclo mediante el cual se entrena la red neuronal con todos los ejemplos del conjunto entrenamiento. Por lo general, se requiere entrenar la red neuronal durante varias épocas [60].

Los datos recabados para el problema que se desea resolver con la red neuronal, deben ser divididos en un conjunto de entrenamiento y en un conjunto de prueba, los cuales deben ser disjuntos [63]. El *conjunto de entrenamiento* es usado para el aprendizaje, es decir, para ajustar los parámetros (pesos) del clasificador. El *conjunto de prueba* es usado para evaluar al clasificador previamente entrenado [71].

Una técnica para dividir los datos en los conjuntos de entrenamiento y de prueba, es *K-fold cross validation*, la cual consiste en dividir los datos en k subconjuntos, por lo tanto, la evaluación de la red neuronal se realizará k veces. En cada iteración, uno de los k subconjuntos será usado como conjunto de prueba y los otros $k - 1$ subconjuntos se agruparán para formar el conjunto de entrenamiento [72].

Para evaluar la red neuronal, es necesario conocer la precisión con la que esta clasifica las actividades, la cual esta dada por la fórmula:

$$\text{Precisión} = \frac{\text{Ejemplos clasificados correctamente}}{\text{Total de ejemplos del conjunto de prueba o de entrenamiento}} \times 100 \quad (4.3)$$

La evaluación de la red neuronal consiste en determinar las precisiones con la que esta clasifica los ejemplos de los conjuntos de prueba, es decir las precisiones obtenidas en las k pruebas, con base en dicha información es posible calcular el *desempeño de la red neuronal*, el cual estará dado por el promedio de las precisiones obtenidas en las k pruebas:

$$\text{Desempeño} = \frac{\sum \text{Precisiones de las pruebas}}{k} \quad (4.4)$$

4.1.5. Trabajos relacionados

Las redes neuronales se han empleado ampliamente en el reconocimiento de actividades, dado que permiten resolver problemas que son muy complejos y dinámicos (empleo de diversos sensores en diferentes ambientes, muchas posibles actividades por clasificar, etc.) para las tecnologías convencionales, mediante las cuales la solución sería muy complicada de implementar [70]. En [73] se proponen tres algoritmos (Quick Propagation, Levenberg Marquardt and Batch Back Propagation) para el entrenamiento de redes neuronales, cuya finalidad es entrenar dichas redes para el reconocimiento de actividades. En [74] se presenta OCEAN (Opportunistic Computing model for wearable Activity recognition), en el cual se implementan tres técnicas (mapeo aleatorio, agrupamiento difuso y actualización de pesos) que permiten a la red neuronal propuesta adaptarse a los cambios dinámicos en el ambiente, como el empleo de diferentes tipos de sensores, agregación de nuevas actividades al modelo creado, diferentes estilos de uso de los sensores por parte del usuario (colocarse los sensores o dispositivos en diferentes partes del cuerpo como la muñeca o portarlo en el bolsillo).

Otro tipo de redes neuronales que se han aplicado para el reconocimiento de actividades son las redes neuronales recurrentes [75], así como las redes neuronales convolucionales [76, 77]. El trabajo de investigación presentando en [75] emplea el uso de redes neuronales recurrentes basadas en neuronas de largo a corto plazo (LSTM-RNN por sus siglas en inglés) para el reconocimiento de actividades (un total de 21 actividades realizadas al interior del hogar). Los vectores de entrada proporcionados para el reconocimiento de actividades provienen de la señal triaxial de aceleración registrada por un smartphone, así como el audio y video grabado por una videocámara deportiva.

Para el reconocimiento de actividades, también se han propuesto arquitecturas de redes neuronales profundas como en [78] y [79]. En [78] se presenta una arquitectura de red neuronal profunda que contiene un conjunto de redes neuronales convolucionadas (GoogLeNet, AlexNet, SqueezeNet) conectadas a través de diferentes métodos de fusión (redes neuronales, media aritmética, máquinas de vectores de soporte o SVM por sus siglas en inglés, neuronas

de largo a corto plazo o LSTM) con la finalidad de realizar el reconocimiento de las actividades que se realizan al cocinar. Los datos de entrada de la red neuronal son proporcionados por las imágenes captadas por una videocámara.

Discusión sobre el estado del arte

Las redes neuronales desarrolladas para el reconocimiento de actividades presentan muchas diferencias en cuanto estructura y métodos de entrenamiento usados para entrenarlas, dado que para realizar el diseño de una red neuronal se tiene que tomar en cuenta el tipo y cantidad de datos que serán ingresados a esta. Haciendo hincapié en su estructura, se encuentran aquellas redes neuronales muy simples como en [73] que poseen solo tres capas de neuronas (una de entrada, una oculta y una de salida), pero incluso existen aquellas complejas, como la presentada en [78] donde emplean redes neuronales profundas; además se han desarrollado redes neuronales adaptables que permiten cambiar su estructura de manera dinámica para solventar los cambios provenientes de los datos del entorno, como la descrita en [74]. La red neuronal propuesta en este trabajo de tesis (IAR-NN) fue diseñada de tal forma que se pueda modificar su estructura tanto a lo largo como a lo ancho, es decir modificar el número de capas de neuronas ocultas así como el número de neuronas ocultas por cada capa, con la finalidad de buscar aquella estructura que proporcione los mejores resultados en el reconocimiento de actividades.

Otro aspecto muy importante a considerar respecto a la estructura de las redes neuronales es el número de neuronas de entrada, las cuales dependen de las características de los vectores de entrada, por ejemplo el vector de entrada red neuronal descrita en [75] está conformado por un total de 56 características las cuales son obtenidas de los datos provenientes de la señal de aceleración de un smartphone y de la señal de audio de la grabación de una videocámara. En el diseño de IAR-NN se tomó en cuenta el desarrollar una red neuronal con estructura simple, por lo que se trató que el número de características del vector de entrada sea el menor posible, dado que un mayor número de características involucra mayor tiempo computacional [80].

Tomando en cuenta el método de entrenamiento empleado, se encuentran aquellas redes neuronales que combinan diferentes métodos para realizar el reconocimiento de actividades, como en el caso de [78], sin embargo, el emplear un mayor número de algoritmos implica aumentar el tiempo computacional para el reconocimiento de actividades, es por esta razón que IAR-NN se entrenó con un solo algoritmo, backpropagation, el cual resulta sencillo pero eficiente.

4.2. Diseño y desarrollo de IAR-NN

En esta sección se presenta el diseño y desarrollo de la red neuronal IAR-NN para el reconocimiento de actividades al interior del hogar. Es importante mencionar que IAR-NN es el equivalente del módulo *Motor de inferencias* de la arquitectura general.

TABLA 4.1: Binarización de las habitaciones.

Habitación	Valor numérico	Valor binario
Habitación 1	1	100
Habitación 2	2	010
Habitación 3	3	001

4.2.1. Diseño de IAR-NN

IAR-NN fue diseñada con una arquitectura feedforward fully connected, y fue entrenada mediante el algoritmo backpropagation (ver apéndice C), la función de transferencia de todas las neuronas es sigmoidea bipolar, a excepción de las neuronas de la capa de entrada cuya función es la de identidad. Para el diseño de IAR-NN es necesario considerar el número de neuronas de entrada, el número de capas de neuronas ocultas (así como el número de neuronas por capa oculta), y el número de neuronas de salida. A continuación se explican los detalles de cada una de las consideraciones en el diseño de IAR-NN.

Número de neuronas de entrada

El número de neuronas de entrada está dado por el número de características de los vectores de entrada. Para determinar el número de características, se tomó como referencia la definición de actividad propuesta en el desarrollo de OPAIEH (ver sección 3.2). Por lo tanto cada vector de entrada se formó con base a los 4 parámetros empleados para definir una actividad, los cuales son *Habitación*, *Día de la semana*, *Hora de inicio*, y *Duración*, además cada vector de entrada debe contar con la etiqueta correspondiente de cada vector, es decir, el valor objetivo. A continuación se explican como se manejaron cada uno de los parámetros para formar las características:

- *Habitación*. Es el lugar del hogar donde se realiza la actividad. En OPAIEH las habitaciones eran tratadas como tipo de dato String, pero en IAR-NN no pueden ser tratadas de esta forma, dado que este tipo de dato no es útil para ser empleado en las redes neuronales. Es por esta razón que se optó por realizar la binarización de las habitaciones, lo cual se consigue asignándoles a las habitaciones un valor numérico que se usan como auxiliares en la binarización, en la Tabla 4.1 se muestra un ejemplo. Dicho proceso de binarización permite obtener las características correspondientes al parámetro *Habitación*. Es importante observar el patrón de binarización, dado el número de características provenientes de este parámetro dependerá del número de habitaciones, dado que cada dígito le corresponderá a cada una de las habitaciones, el cual se marcará con un 1.
- *Día de la semana*. Es el día en que se realizó la actividad. De igual forma que las habitaciones, los días de la semana fueron convertidos a números binarios. En la Tabla 4.2 se

TABLA 4.2: Binarización de los días de la semana.

Día de la semana	Valor numérico	Valor binario
Lunes	1	1000000
Martes	2	0100000
Miércoles	3	0010000
Jueves	4	0001000
Viernes	5	0000100
Sábado	6	0000010
Domingo	7	0000001

muestra la binarización de los días de la semana. De este parámetro se obtienen 7 características, las cuales permanecerán fijas, dado que solo existen 7 días de la semana.

- *Hora de inicio*. Es la hora en la que se inició la actividad. En OPAIEH, al realizar las consultas este dato es manejado en términos de minutos; en IAR-NN se utilizó en términos de minutos, pero de manera normalizada. La normalización de este tipo de dato se muestra en la ecuación 4.5, es importante recordar que un día cuenta con 1440 minutos. De este parámetro se obtiene 1 característica.

$$\text{Hora de inicio normalizada} = \frac{\text{Hora de inicio en minutos}}{1440} \quad (4.5)$$

- *Duración*. Es la duración de la actividad. De igual manera que la *Hora de inicio*, este tipo de dato se normalizará con base a 1440, como se muestra en la ecuación 4.6. De este parámetro solo se obtiene 1 característica.

$$\text{Duración normalizada} = \frac{\text{Duración en minutos}}{1440} \quad (4.6)$$

En Fig. 4.7 se pueden observar las 12 neuronas que forman parte de la capa de entrada de IAR-NN, las cuales corresponden a las características de los parámetros: *Habitación*, en la imagen se muestra como ejemplo únicamente 3 características dado que el ejemplo está dado para 3 habitaciones, pero es importante recordar que se tendrá una característica de este parámetro por cada habitación; *Día*, cuenta con 7 características; *Hora de inicio*, cuenta con 1 característica; y *Duración*, cuenta con 1 característica.

Número de capas ocultas

IAR-NN fue diseñada para ser una red neuronal flexible, es decir, que permita su modificación a lo largo (número de capas ocultas de neuronas) y a lo ancho (número de neuronas

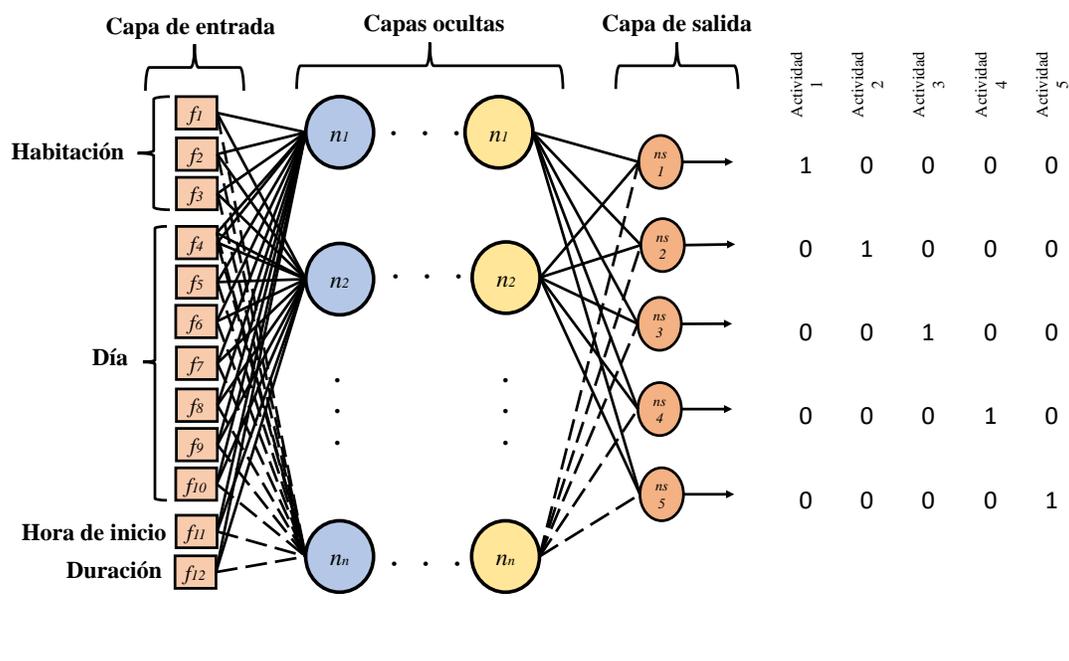


FIGURA 4.7: Arquitectura de IAR-NN.

ocultas por capa); esto permite que se puedan probar diferentes estructuras, con la finalidad que se busque aquella con la cual se obtengan mejores resultados, es decir, una mayor precisión en la inferencia de actividades.

En Fig. 4.7 se ejemplifica la flexibilidad en el número de capas ocultas de la arquitectura de IAR-NN.

Número de neuronas de salida

En IAR-NN se requieren tantas neuronas de salida como actividades registradas, es decir, por cada actividad se requiere una neurona de salida. Este enfoque permite que las salidas de las neuronas de la capa de salida generen en conjunto una respuesta binaria para cada actividad inferida, de tal manera que si existen 3 actividades, *Actividad 1*, *Actividad 2*, y *Actividad 3*, se espera que el arreglo de las salidas de las neuronas de la capa de salida sean 100, 010, 001 respectivamente, es decir, para la primera actividad solo se activa la primera neurona de salida y las otras dos no, para la segunda actividad solo se activa la segunda neurona de salida y las otras dos no, para la tercera actividad solo se activa la tercera neurona de salida y las otras dos no.

¿Qué significa que una neurona de salida se active? Cuando la salida de una neurona de salida es igual o mayor a 0.5, dicha neurona se considera activa, y su salida se convierte en un

1; en caso que la salida de la neurona de salida sea menor a 0.5, su salida se convierte en un 0. En Fig. 4.7 se muestra el funcionamiento de las neuronas de salida.

Con la finalidad de comparar las salidas binarias (actividad inferida) de las neuronas con el valor objetivo (actividad realizada) de un ejemplo de entrenamiento en la fase backpropagation, se requiere que las etiquetas, es decir las actividades, de los vectores de entrada se encuentren en formato tipo int, con la finalidad de facilitar la tarea de binarización de dicha etiqueta de manera automática por el programa desarrollado para ejecutar IAR-NN. En Fig. 4.8 se muestra con 3 ejemplos el proceso de conversión de las actividades y sus correspondientes parámetros, al formato requerido de los vectores de entrada.

Habitación	Día	Hora de inicio (min)	Duración (min)	Actividad
Habitación 1	Lunes	900	20	Actividad 1
Habitación 2	Miércoles	600	5	Actividad 2
Habitación 3	Domingo	1200	30	Actividad 3

① Actividades y sus parámetros correspondientes.

Habitación (valor numérico)	Día (valor numérico)	Hora de inicio normalizada	Duración normalizada	Actividad (valor numérico)
1	1	0.6250	0.0138	1
2	3	0.4166	0.0034	2
3	7	0.8333	0.0208	3

② Convertir a etiquetas numéricas los parámetros *Habitación*, *Día*, así como también las *Actividades*.
Normalizar los parámetros *Hora de inicio* y *Duración*.

Habitación (binario)		Día (binario)								Hora de inicio normalizada	Duración normalizada	Actividad (valor numérico)
f ₁	f ₂	f ₃	f ₄	f ₅	f ₆	f ₇	f ₈	f ₉	f ₁₀	f ₁₁	f ₁₂	Etiqueta
1	0	0	1	0	0	0	0	0	0	0.6250	0.0138	1
0	1	0	0	0	1	0	0	0	0	0.4166	0.0034	2
0	0	1	0	0	0	0	0	0	1	0.8333	0.0208	3

③ Binarización de los parámetros *Habitación* y *Día*.
Formato final de los vectores de entrada (12 características, 1 etiqueta)

FIGURA 4.8: Conversión de las actividades y sus correspondientes parámetros al formato requerido de los vectores de entrada.

Estructuras de IAR-NN

Como se mencionó anteriormente, IAR-NN fue diseñada con la finalidad de poder modificar su estructura, lo cual se consigue variando el número de capas de neuronas ocultas y el número de neuronas ocultas por capa, y de esta manera buscar aquella que proporcione los mejores resultados. A continuación se explica con dos ejemplos la nomenclatura empleada en este proyecto de tesis para denotar a las estructuras.

Cuando se hace referencia a una estructura 12-3-6, quiere decir que IAR-NN está conformada: por 12 neuronas de entrada; una capa oculta compuesta por 3 neuronas; y una capa de salida de 6 neuronas. Cuando se hace referencia a una estructura 12-3-2-6, esto quiere decir que IAR-NN está conformada por: 12 neuronas de entrada; seguida por dos capas ocultas, la primera con 3 neuronas y la segunda con 2; y una capa de salida de 6 neuronas.

4.2.2. Desarrollo de IAR-NN

Para el desarrollo de IAR-NN se utilizó el software MATLAB [81], con el cual se crearon dos programas, IAR-NN Training y IAR-NN Test. A continuación se describirán cada uno de los programas.

IAR-NN Training

Este programa permite entrenar a la red neuronal, incluye las fases feedforward, la retro-propagación del error (backpropagation), y la actualización de los pesos. Su entrada consiste en un archivo txt, el cual debe contener los vectores de entrada y las etiquetas de los ejemplos del conjunto de entrenamiento. Su salida consiste en un archivo txt el cual contiene los valores de los pesos de la red entrenada, así como un escalar, el cual representa la precisión de la inferencia de actividades obtenida al terminar el entrenamiento (ver ecuación 4.3).

El programa IAR-NN Training fue diseñado para que permita variar el número de capas de neuronas ocultas, el número de neuronas por cada capa oculta, y el número de épocas de entrenamiento. Las precisiones obtenidas en la inferencia de actividades en los entrenamientos únicamente sirven para determinar si con los parámetros anteriormente mencionados se obtienen resultados adecuados, es decir precisión de inferencia mayor o igual a $X\%$ establecido, no para evaluar el desempeño de IAR-NN. Si se obtienen resultados satisfactorios en los entrenamientos, el diseñador puede proceder a evaluar la red neuronal mediante las pruebas (IAR-NN Test).

IAR-NN Test

Este programa permite evaluar la precisión del conjunto de prueba, por lo que únicamente incluye la fase feedforward. Su entrada consiste en dos archivos txt, el primero debe contener los vectores de entrada y las etiquetas de los ejemplos del conjunto de prueba, el segundo debe contener los valores de los pesos de la red previamente entrenada. Su salida consiste en un escalar, el cual representa la precisión de la inferencia de actividades al terminar la prueba (ver ecuación 4.3). Es importante recordar que con el promedio de las precisiones de las pruebas se obtiene el desempeño de la red neuronal (ver ecuación 4.4).

4.3. Pruebas experimentales, resultados y análisis

En esta sección se explica la prueba experimental realizada sobre IAR-NN, así como los resultados obtenidos de esta; la finalidad de dicha prueba consiste en determinar el desempeño de IAR-NN en la inferencia de actividades. La prueba experimental se llevó a cabo en una computadora con procesador Intel i5 y 8 GB de memoria RAM.

4.3.1. Prueba experimental

En este proyecto de tesis no hubo la posibilidad de integrar IAR-NN dentro de la arquitectura diseñada para OPAIEH, es decir, reemplazar el motor de inferencias basado en *Consulta de ontologías* por *IAR-NN*. Por este motivo, la prueba experimental de IAR-NN se realizó con las 147 actividades confirmadas contenidas en la tabla de confirmación de actividades del usuario, las cuales se obtuvieron durante la prueba experimental de OPAIEH; es importante recordar que las tablas de confirmación de actividades se encuentran en la base de datos OPAIEHServer. La información contenida en dicha tabla fue exportada a un archivo CSV de Excel, para realizar un tratamiento previo a los datos desde dicho programa. En Fig. 4.9 se muestra de manera simplificada el procedimiento para obtener de la base de datos, las características necesarias de los vectores de entrada, así como sus respectivas etiquetas.

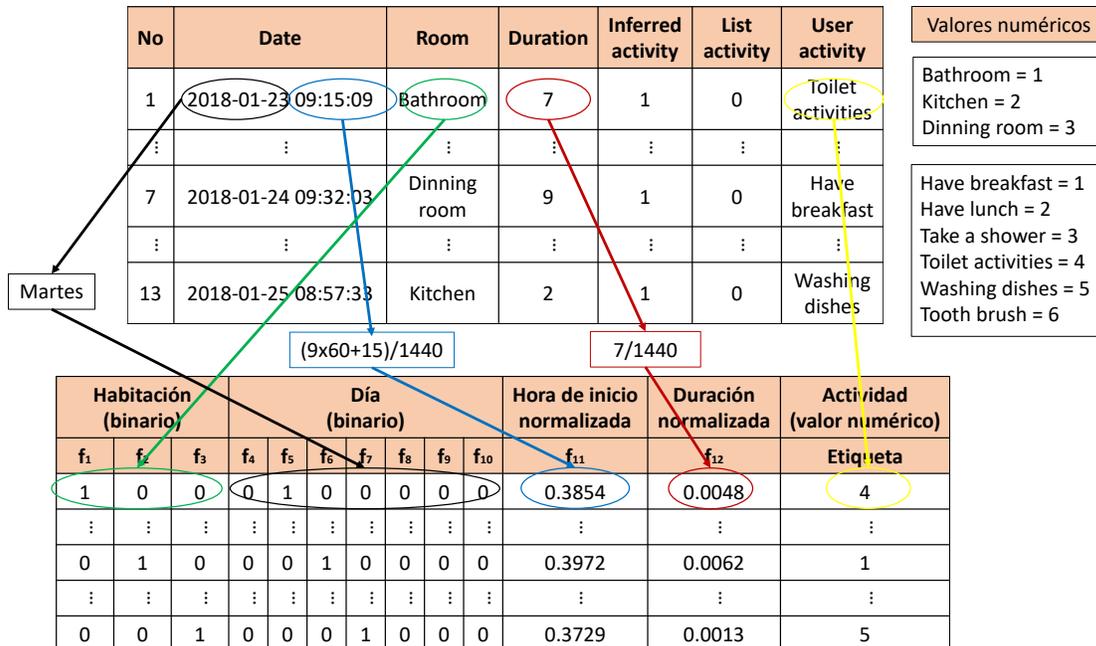


FIGURA 4.9: Obtención de las características de los vectores de entrada de la tabla de confirmación de actividades de la base de datos de OPAIEHServer.

La finalidad de la prueba experimental es probar diferentes estructuras de IAR-NN, es decir, variar el número de capas de neuronas ocultas y el número de neuronas ocultas por capa, para determinar cuál de ellas proporciona los mejores resultados, es decir, mejor precisión en la inferencia de actividades, la cual se espera que sea igual o mayor a 80 %.

Para determinar el desempeño (promedio de las precisiones de las pruebas) de cada una de las estructuras probadas de IAR-NN se realizó una validación cruzada de 4 subconjuntos, cada subconjunto se conformó por los vectores de entrada obtenidos de los registros pertenecientes a cada una de las 4 semanas. Al tener una validación cruzada con $k = 4$ implica que se tienen 4 conjuntos de entrenamiento (training set) y 4 conjuntos de prueba (test set). En Fig. 4.10 se muestra el esquema de la validación cruzada.

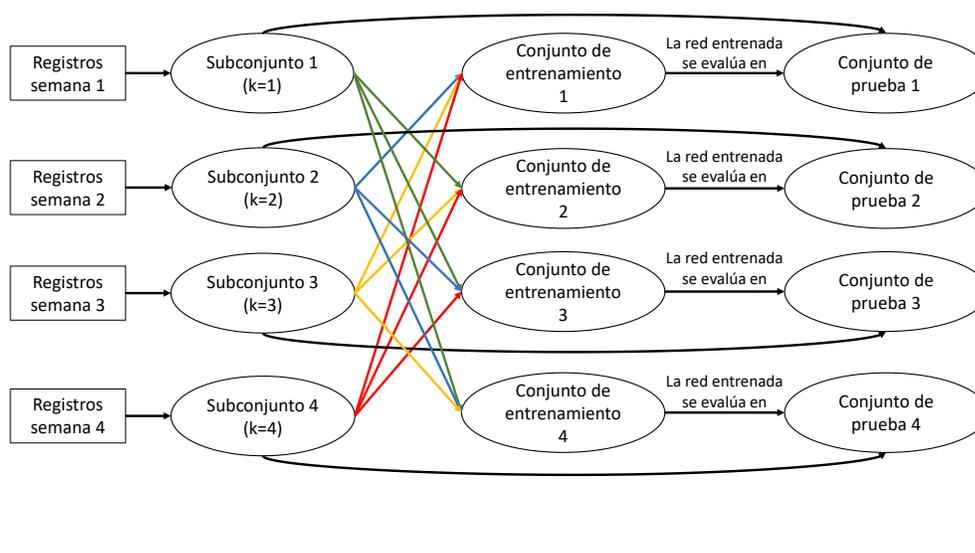


FIGURA 4.10: Esquema de la validación cruzada.

La prueba experimental consiste en seleccionar una estructura, establecer el número de épocas de entrenamiento, y con base en esto calcular la precisión de inferencia con el conjunto de entrenamiento 1, si el resultado es mayor a 80 % se procederá a calcular el desempeño de dicha estructura, esto se establece con la finalidad de no realizar los 4 entrenamientos y las 4 pruebas en aquellas estructuras que desde el primer entrenamiento muestren resultados no satisfactorios. Dicho procedimiento se realizará tantas estructuras se deseen evaluar. En Fig. 4.11 se muestra el diagrama de flujo de la prueba experimental.

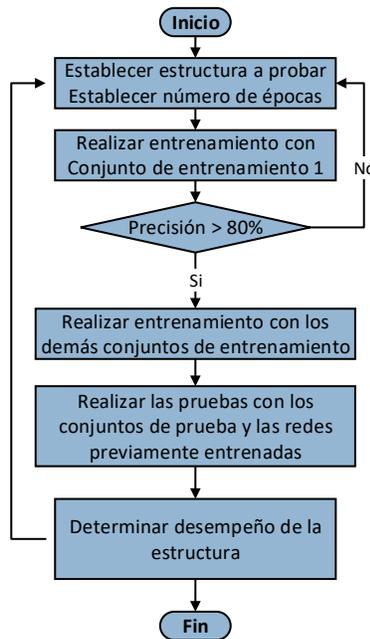


FIGURA 4.11: Diagrama de flujo de la prueba experimental.

4.3.2. Resultados y análisis

En la Tabla 4.3 se muestran las precisiones obtenidas del entrenamiento realizado con el conjunto de entrenamiento 1 para diferentes estructuras, el entrenamiento fue realizado durante 1000 épocas. De las estructuras mostradas, solo las numeradas del 3 al 7 fueron las que obtuvieron una precisión de más de 80 % al terminar el entrenamiento del conjunto de entrenamiento 1, por lo tanto solo se procedió a evaluar el desempeño de estas estructuras. Además se pudo establecer que el número ideal de capas ocultas es de 1, dado que las estructuras con más de 1 capa tienen resultados inferiores a los de 1 capa.

En Fig. 4.12 se muestra el desempeño de las estructuras 12-3-6, 12-4-6, 12-5-6, 12-7-6 y 12-9-6. De esas 5 estructuras, las primeras 3 son las que obtuvieron el mejor desempeño, es por esta razón que se realizó un análisis del comportamiento del desempeño de dichas estructuras al aumentar el número de épocas de entrenamiento, en la Tabla 4.4 se muestran los desempeños obtenidos por estructura y por número de épocas. Se esperaba que el desempeño mejorará para cada estructura según se aumentará el número de épocas, pero los datos mostrados en la Tabla 4.4 muestran un comportamiento oscilante; el mejor desempeño obtenido, 82.43 %, fue con la estructura 12-4-6 con entrenamientos de 2000 épocas.

En Fig. 4.13 se muestra una gráfica en la cual se comparan las precisiones promedio de los entrenamientos y las precisiones promedio de las pruebas (desempeño) para diferentes épocas

TABLA 4.3: Precisiones obtenidas del entrenamiento realizado con el conjunto de entrenamiento 1 para diferentes estructuras.

No	Estructura	Precisión
1	12-1-6	55.86
2	12-2-6	71.17
3	12-3-6	80.18
4	12-4-6	82.88
5	12-5-6	84.68
6	12-7-6	85.59
7	12-9-6	86.49
8	12-3-2-6	71.17
9	12-4-3-6	71.17
10	12-9-8-6	71.17
11	12-3-2-1-6	0

TABLA 4.4: Desempeños obtenidos para diferentes estructuras entrenadas durante diferentes números de épocas.

Estructura/Épocas	1000	2000	3000	4000	5000
12-3-6	79.79	78.94	79.90	79.30	76.37
12-4-6	79.24	82.43	81.11	79.65	80.33
12-5-6	80.54	81.02	81.04	81.04	79.10

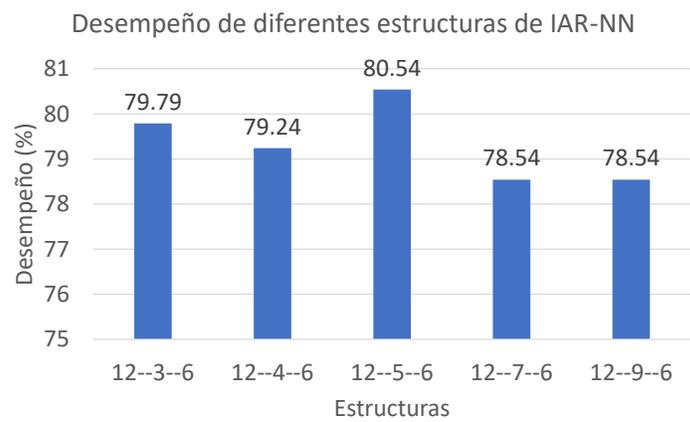


FIGURA 4.12: Gráfica del desempeño de diferentes estructuras de IAR-NN.

de entrenamiento de la estructura 12-4-6; como se puede observar en la gráfica, las precisiones promedio de los entrenamientos tienden a aumentar a medida que aumentan las épocas de entrenamiento, por otra parte el desempeño de la estructura es oscilante.

Otro aspecto importante que se tiene que analizar en torno a la inferencia de actividades, es respecto a aquellas actividades que no son inferidas correctamente. Este análisis se realiza con base en los resultados obtenidos al realizar las 4 pruebas de la validación cruzada sobre la estructura 12-4-6 entrenada durante 2000 épocas, dado que esta fue la que obtuvo el mejor desempeño. En la Tabla 4.5 se muestra la relación entre las actividades inferidas erróneamente por IAR-NN y las actividades que en su lugar se esperaba que fueran inferidas, así como la frecuencia de cada una de esas situaciones. De los resultados mostrados en la Tabla 4.5 se pueden realizar las siguientes afirmaciones: 1) las actividades en las que más se equivocó IAR-NN en la inferencia de datos fue entre Toilet activities y Take a shower; 2) algo importante que destacar es la capacidad de generalización de las redes neuronales, esto queda evidenciado al momento en que IAR-NN infirió Toilet activities en lugar de haber inferido Tooth brush, dado que la segunda actividad se encuentra englobada dentro de la definición planteada de *Toilet activities* en el capítulo anterior; 3) en una sola ocasión no infirió ninguna actividad, cuando debía inferir Take a shower.

4.4. Conclusiones

En este capítulo se propuso IAR-NN, una red neuronal para el reconocimiento de actividades al interior del hogar, la cual se diseñó con una arquitectura feedforward fully connected, y se entrenó mediante el algoritmo backpropagation. IAR-NN fue desarrollada mediante el software MATLAB y programada de manera tal que se pueda modificar dinámicamente su estructura,

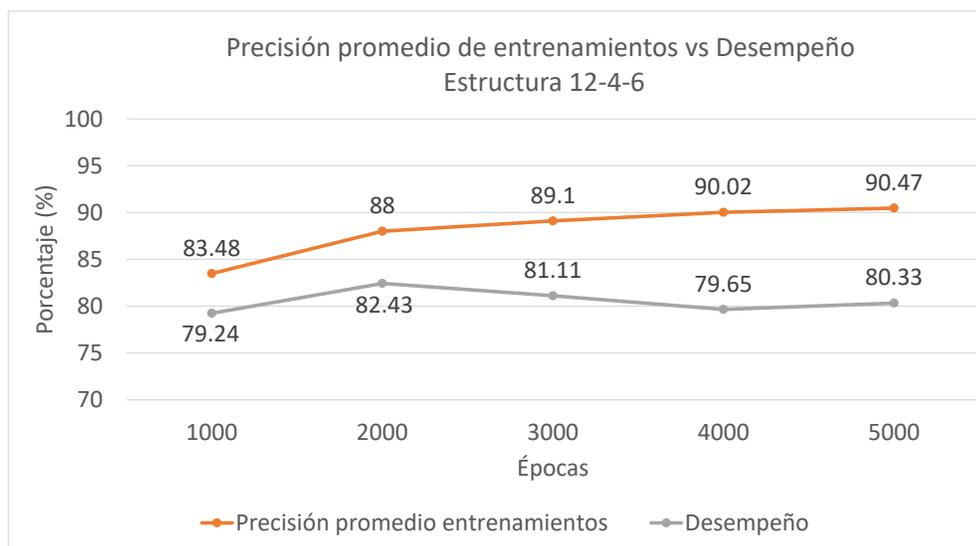


FIGURA 4.13: Gráfica de la comparación del promedio de precisión de inferencia en los entrenamientos y el desempeño obtenido de la estructura 12-4-6 para diferentes épocas de entrenamiento.

es decir, el número de capas de neuronas ocultas, así como el número de neuronas por cada una de esas capas.

La experimentación de IAR-NN fue realizada con los datos recabados de la prueba experimental de OPAIEH. La finalidad de la prueba experimental de IAR-NN fue determinar su desempeño en la inferencia de actividades, para conseguir esto se realizó una validación cruzada del conjunto de datos, con la cual se evaluaron cada una de las estructuras probadas.

El mejor desempeño obtenido de IAR-NN fue de 82.43 % en la inferencia de actividades, el cual se obtuvo con la estructura 12-4-6 y con un entrenamiento de 2000 épocas. Si bien se esperaba un mejor desempeño de IAR-NN, dado que con OPAIEH se obtuvo una precisión

TABLA 4.5: Relación entre las actividades inferidas erróneamente y las esperadas resultantes de la pruebas realizadas sobre la estructura 12-4-6 entrenada durante 2000 épocas de entrenamiento.

Actividad inferida	Actividad esperada	Frecuencia
Toilet activities	Take a shower	12
Take a shower	Toilet activities	12
Toilet activities	Tooth brush	1
Nada	Take a shower	1

de 96.6% en la inferencia de actividades, el desempeño obtenido de IAR-NN es alentador, y deja abiertas las puertas para realizar futuras pruebas, las cuales se esperan que se realicen con una mayor cantidad de registros de diferentes usuarios. Además, otro aspecto que sustenta el potencial de las redes neuronales, y el cual quedó evidenciado en los resultados, es su capacidad de generalización.

Capítulo 5

Conclusiones

El hecho que en la actualidad el sector de la población que es mayor de 60 años está en franco crecimiento, hace que surjan nuevas problemáticas y nuevos retos enfocados en la mejora de la calidad de vida de los adultos mayores. Como solución a estas situaciones, surgen áreas como la gerontotecnología y AAL, cuyo objetivo principal es prolongar el tiempo durante el cual los adultos mayores puedan vivir independientemente y de manera segura en su ambiente preferido, lo cual se consigue mediante la asistencia basada en un ambiente inteligente. Dos de las piezas claves para conseguir una adecuada asistencia a los adultos mayores son el monitoreo y reconocimiento de actividades.

En este trabajo de tesis se propuso el desarrollo de una plataforma que permita reconocer las actividades que realizan los adultos mayores al interior de su hogar, tomando siempre como prioridad el minimizar la intrusión en la vida de los usuarios. Para conseguir esto, lo primero que se realizó fue plantear una arquitectura general para el reconocimiento de actividades, en la cual se definieron los componentes mínimos necesarios para realizar dicha tarea, así como la función de cada uno de ellos dentro de la plataforma. Dicha arquitectura general, fue diseñada con la finalidad de ser un punto de referencia para la creación de plataformas basadas en diferentes enfoques. Con base en la arquitectura general, en este proyecto de tesis se abordaron dos enfoques diferentes: ontologías y redes neuronales.

El enfoque de ontologías derivó en la creación de OPAIEH, una plataforma basada en ontologías para la identificación de actividades al interior del hogar de los adultos mayores, para la cual se desarrollaron todos los módulos y sub-módulos sugeridos por la arquitectura general. Para el enfoque de redes neuronales se desarrolló IAR-NN, una red neuronal para el reconocimiento de actividades al interior del hogar, es importante mencionar que para este enfoque no se desarrolló una plataforma entera, sino que IAR-NN es únicamente el equivalente al *Motor de inferencias* de la arquitectura general. El motivo que no se haya desarrollado un entorno para IAR-NN, es que otra de las finalidades de la arquitectura general es que sus módulos funcionen como piezas que puedan ser reemplazadas, por lo tanto, se tenía la intención de sustituir el motor de inferencias de OPAIEH basado en consultas de ontología, por IAR-NN, es decir reciclar los demás módulos de OPAIEH y únicamente reemplazar el motor de inferencias con la finalidad de crear una plataforma basada en redes neuronales, sin embargo por cuestiones de tiempo no se pudo realizar esto.

La prueba experimental realizada sobre OPAIEH se realizó en un sujeto de prueba de mediana edad durante 28 días, si bien este trabajo de tesis está enfocado en los adultos mayores, todo lo desarrollado es una prueba de concepto, en parte porque el empleo de tecnología suele ser muy laborioso para los adultos mayores. La finalidad de dicha prueba fue determinar la precisión con la cual OPAIEH infiere las actividades, el resultado obtenido fue satisfactorio, dado que se obtuvo un 96.6 %.

Por otra parte la prueba experimental realizada sobre IAR-NN se realizó con base en los registros obtenidos en la prueba de OPAIEH, dado que IAR-NN no se integró dentro de una plataforma que facilitara la obtención de datos. IAR-NN se desarrolló de tal manera que su estructura pueda ser modificada, con la finalidad de determinar aquella estructura que proporcione el mejor desempeño, el cual es el término que se empleó para referirse a la precisión promedio de 4 las pruebas de la validación cruzada. La estructura que brindó el mejor desempeño fue 12-4-6, entrenada durante 2000 épocas, el cual fue de 82.43 %. Si bien el desempeño de IAR-NN fue inferior a la precisión obtenida por OPAIEH, un aspecto alentador del uso de las redes neuronales que quedó evidenciado en los resultados, es su capacidad de generalización.

Hay que recalcar que no es posible comparar a OPAIEH con IAR-NN dado que son dos cosas diferentes, el primero es una plataforma, mientras que el segundo es un componente que podría formar parte de la plataforma. Sin embargo, lo que si es posible, es compararlos a nivel del componente *Motor de inferencias* de la arquitectura general. Para realizar las inferencias en OPAIEH, es necesario tener una base de conocimiento (IAR-ONTO) sobre la cual se realicen las consultas, es decir requiere que el usuario establezca los parámetros de las actividades que realiza al interior de su hogar, por lo tanto las actividades serán inferidas según a la información proporcionada por el usuario. Por otra parte las inferencias en IAR-NN se basan en la capacidad de generalización de las redes neuronales, para lo cual es necesario contar con ejemplos que sirvan para entrenar a la red neuronal, por lo tanto las actividades serán inferidas de acuerdo a la clasificación realizada por la red neuronal. Así como tienen diferencias el motor de inferencias de OPAIEH y IAR-NN también poseen algo en común, la retroalimentación, dado que para saber si el motor de inferencias de OPAIEH infirió correctamente la actividad es necesario que el usuario realice la confirmación, en el caso de IAR-NN se necesita que las actividades se encuentren etiquetadas, dado que se trata de un aprendizaje supervisado.

Todos los objetivos planteados en este proyecto de tesis se cumplieron con el desarrollo de OPAIEH, dado que se consiguió obtener un sistema funcional que permita monitorear las actividades que realizan las personas al interior de su hogar y con base en esto realizar su reconocimiento. Por otra parte el desarrollo de IARN en esta tesis figura como punto de comparación con OPAIEH.

Los resultados obtenidos en este trabajo de tesis son satisfactorios, dado que ratifican el potencial tanto de las ontologías como de las redes neuronales, y motivan a darle continuidad a este tipo de enfoques empleados en el reconocimiento de actividades. Sin embargo, como trabajo futuro se sugiere experimentar con los siguientes enfoques: 1) realizar una plataforma para el reconocimiento de actividades híbrida, es decir, que se combinen las ontologías con las redes neuronales para aprovechar las ventajas de cada una de ellas; 2) realizar una plataforma

basada en métodos de Machine Learning basados en agrupamiento, de tal forma que se elimine la necesidad de una retroalimentación o un etiquetado de datos.

Apéndice A

Diagrama de clases de OPAIEHClient

OPAIEHClient está conformada por 25 clases y 1 interfaz, cuyas relaciones, atributos y métodos se muestran en Fig. A.1.

Las clases LoginActivity, ProfileActivity, RoomsActivity, RoomActivity, ActivitiesActivity, ActivitiActivity y MonitoringActivity heredan de la clase AppCompatActivity (en Android, una actividad representa una pantalla con interfaz de usuario), y son empleadas para logear al usuario, registrar al usuario, mostrar las habitaciones registradas, registrar una habitación, mostrar las actividades registradas, registrar una actividad, y monitorear al usuario, respectivamente.

La clase RoomListAdapter que hereda de ArrayAdapter, sirve como auxiliar en la generación de la vista personalizada de las habitaciones en la interfaz RoomsActivity (ver Fig. 3.8).

La clase SendToServer que hereda de AsyncTask permite enviar información al servidor Web, así como recibir información de este. La interfaz AsyncResponse provee el método abstracto processFinish, cuyo funcionamiento es descrito por cada una de las clases que lo implementan (LoginActivity, ProfileActivity, RoomActivity, ActivitiActivity y MonitoringActivity), la finalidad de este método es procesar la información recibida por parte del servidor.

Las clases KeySQLiteHelper, UserSQLiteHelper, RoomsSQLiteHelper y ActivitySQLiteHelper, que heredan de SQLiteHelper, permiten crear y administrar las bases de datos que OPAIEHClient requiere para su adecuado funcionamiento.

Las clases UserModel, RoomModel, ActivityModel, ActivityQueryModel y ConfirmActivityModel, permiten a OPAIEHClient crear los objetos que representan un usuario, una habitación, una actividad, una consulta de actividad, y una confirmación de actividad, respectivamente.

Las clases MyBeacons, DateandTime y MinVale proporcionan a la clase MonitoringActivity métodos auxiliares que permiten realizar un adecuado monitoreo.

Apéndice B

Diagrama de clases de OPAIEHServer

OPAIEHServer está conformada por 27 clases y 1 interfaz, cuyas relaciones, atributos y métodos se muestran en Fig. B.1.

La clase ApplicationConfig que hereda de la clase Application, brinda a OPAIEHServer los métodos necesarios para realizar la comunicación REST con OPAIEHClient.

La clase Main recibe y selecciona la acción adecuada para cada una de las peticiones que provienen de OPAIEHClient.

La clase UserDataSource contiene los métodos que permiten logear al usuario dentro de la plataforma OPAIEH, así como registrarlo en esta. La clase OntologyDataSource se encarga de registrar y administrar el almacenamiento de las habitaciones y las actividades en IAR-ONTO. La clase QueryOntology se encarga de realizar las consultas (queries) en IAR-ONTO (ver sección 3.5). La clase ActivityDataSource se encarga de registrar las confirmaciones de las actividades que realizan los usuarios en la base de datos; además contiene los métodos que extraen diferentes paquetes de información de las tablas de confirmaciones de actividades de los usuarios, con la cual OPAIEHClient Web genera las diferentes gráficas que le presenta al cuidador del adulto mayor (ver sección 3.4.4).

La clase MySQLConnection se encarga de realizar las conexiones y desconexiones en la base de datos.

Las clases LoginServlet y DBServlet, que heredan de HttpServlet, son las clases intermedias que permiten el flujo de información entre OPAIEHServer y OPAIEHClient Web. La clase LoginServlet le permite al cuidador del adulto mayor acceder a su cuenta de OPAIEHClient Web. La clase DBServlet se encarga de enviarle la información necesaria a OPAIEHClient Web, para que este pueda generar las gráficas que le presentará al cuidador del adulto mayor.

Las clases UserModel, RoomModel, ActivityModel, ActivityQueryModel, ConfirmActivityModel, permiten a OPAIEHServer crear los objetos que representan un usuario, una habitación, una actividad, una consulta de actividad, y una confirmación de actividad, respectivamente.

Las clases `EntriesPerRoomModel`, `ActivityAverageDurationModel`, `AllDatesModel`, `PathRoomModel`, `DirectedGraphStructure`, `Links`, `Nodes`, `EventStructure`, `Event`, `DGDataStructureSerialiser`, y `EventStructureSerialiser` (las dos últimas clases heredan de `JsonSerializer`) sirven como auxiliares en la manipulación de la información extraída de las tablas de confirmaciones de actividad de los usuarios, con la finalidad de que dicha información sea usada para la creación de los paquetes de información que le son enviados a `OPAIEHClient Web`.

Apéndice C

Algoritmo backpropagation

A continuación se presenta el algoritmo backpropagation para n capas de neuronas ocultas.

Paso 0. Inicializar los pesos (establecerlos con valores aleatorios pequeños).

Paso 1. Mientras la condición de paro sea falsa, realizar los pasos 2-9.

Paso 2. Para cada ejemplo de entrenamiento, realizar los pasos 3-9.

Feedforward

Paso 3. Cada neurona de entrada x_i recibe una señal de entrada y la transmite a todas las neuronas de la capa siguiente (las neuronas ocultas).

Paso 4. Cada neurona oculta o_i suma sus señales de entrada ponderadas, aplica su función de activación para calcular su señal de salida,

$$z = \frac{1}{1 + e^{-(\sum_{i=1}^n w_i x_i + w_b)}} \quad (\text{C.1})$$

y la envía a las neuronas de la siguiente capa. Este paso se realiza n veces, es decir, para todas las neuronas de las n capas de neuronas ocultas.

Paso 5. Cada neurona de salida suma sus señales de entrada ponderadas y aplica su función de activación (sigmoidea unipolar) para calcular su señal de salida.

Backpropagation

Paso 6. Cuando la señal de salida proviene de las neuronas de la capa de salida, estas reciben un patrón objetivo (t_k), la salida deseada para cada neurona de salida correspondiente al ejemplo de entrenamiento, con base en el cual cada neurona de salida calcula su término de error δ_k .

$$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k) \quad (\text{C.2})$$

Donde o_k es la salida de la neurona de salida.

Paso 7. Cada neurona oculta calcula su término de error δ_h .

$$\delta_h \leftarrow o_h(1 - o_h) \sum_{k \in K} w_{kh} \delta_k \quad (\text{C.3})$$

Donde o_h es la salida de la neurona oculta, K es el conjunto de neuronas a las que envía su señal la neurona oculta h , w_{kh} es el peso de la conexión de la neurona oculta h a la neurona k , δ_k es el término de error de la neurona k . Este paso se realiza n veces, es decir, para todas las neuronas de las n capas de neuronas ocultas.

Actualización de pesos

Paso 8. Los pesos se actualizan con base en la siguiente fórmula:

$$w_{ji}(\text{new}) \leftarrow w_{ji}(\text{old}) + \alpha \delta_j x_{ji} \quad (\text{C.4})$$

Donde w_{ji} es el peso de la conexión que va de la neurona i a la neurona j , α es la tasa de aprendizaje, δ_j es el término de error de la neurona j , x_{ji} es la información que pasa por la conexión que va de la neurona i a la neurona j (la salida de la neurona i).

Paso 9. Condición de paro.

Bibliografía

- [1] *Los Adultos Mayores en México: Perfil Sociodemográfico al Inicio del Siglo XXI*. Instituto Nacional de Estadística y Geografía, 2005.
- [2] Censo de Población y Vivienda. *Perfil sociodemográfico de adultos mayores*. Inf. téc. Instituto Nacional de Estadística y Geografía, 2010.
- [3] WHO. *World report on ageing and health*. Inf. téc. World Health Organization, 2015.
- [4] Instituto de Geriatria. «Gerontecnología». En: *Perspectivas para el desarrollo de la investigación sobre el envejecimiento y la gerontecnología en México*. Secretaría de Salud, 2010.
- [5] H. Bouma. «Gerontechnology: Making technology relevant for the elderly». En: *Gerontechnology*. IOS Press, 1992.
- [6] E. Nakagawa y col. «Relevance and perspectives of AAL in Brazil». En: *Journal of Systems and Software* 86.4 (2013).
- [7] M. Memon y col. «Ambient Assisted Living Healthcare Frameworks, Platforms, Standards, and Quality Attributes». En: *Sensors (Basel, Switzerland)* 14.3 (2014), págs. 4312-4341.
- [8] B.Athanasios y col. «The BehaviorScope Framework for Enabling Ambient Assisted Living». En: *Personal Ubiquitous Comput* 14.6 (2010).
- [9] F. Attal y col. «Physical Human Activity Recognition Using Wearable Sensors». En: *Sensors* 15.12 (2015), págs. 31314-31338.
- [10] Melchor López Mejía. «Plataforma Inteligente para la Asistencia No Intrusiva de las personas de la tercera edad que realizan actividades en ambientes externos». Tesis de maestría. Universidad Autónoma de Yucatán, 2016.
- [11] B. Furletti y col. «Inferring Human Activities from GPS Tracks». En: *Proceedings of the 2Nd ACM SIGKDD International Workshop on Urban Computing*. 5. 2013, 5:1-5:8.
- [12] M. Boukhechba y col. «Online Recognition of People's Activities from Raw GPS Data: Semantic Trajectory Data Analysis». En: *Proceedings of the 8th ACM International Conference on Pervasive Technologies Related to Assistive Environments*. 2015, 40:1-40:8.
- [13] M. Al-Wattar y col. «Activity Recognition from Video Data Using Spatial and Temporal Features». En: *2016 12th International Conference on Intelligent Environments (IE)*. 2016, págs. 250-253.
- [14] A. Popleteev. «Activity Tracking and Indoor Positioning with a Wearable Magnet». En: *Adjunct Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the ACM International Symposium on Wearable Computers*. 2015.

- [15] T. Maekawa y col. «Activity recognition with hand-worn magnetic sensors». En: *Personal and Ubiquitous Computing* 17.6 (2013), págs. 1085-1094.
- [16] M. Philipose y col. «Inferring Activities from Interactions with Objects». En: *IEEE Pervasive Computing* 3.4 (2004), págs. 50-57.
- [17] Y. Gu, L. Quan y F. Ren. «WiFi-assisted human activity recognition». En: *IEEE Asia Pacific Conference on Wireless and Mobile*. 2014.
- [18] Y. Gu, F. Ren y J. Li. «PAWS: Passive Human Activity Recognition Based on WiFi Ambient Signals». En: *IEEE Internet of Things Journal* 3.5 (2016), págs. 796-805.
- [19] X. Huang y M. Dai. «Indoor Device-Free Activity Recognition Based on Radio Signal». En: *IEEE Transactions on Vehicular Technology* 66.6 (2017), págs. 5316-5329.
- [20] H. Zou y col. «A Robust Indoor Positioning System Based on the Procrustes Analysis and Weighted Extreme Learning Machine». En: *IEEE Transactions on Wireless Communications* 15.2 (2016), págs. 1252-1266.
- [21] M. L. Mejia y col. «Intelligent Platform for Non Intrusive Assistance of Elderly People». En: *IEEE Latin America Transactions* 14.5 (2016), págs. 2433-2439.
- [22] B. Abdulrazak y col. «PhonAge: Adapted SmartPhone for Aging Population». En: *Inclusive Society: Health and Wellbeing in the Community, and Care at Home*. Springer Berlin Heidelberg, 2013, págs. 27-35.
- [23] A. Cortez, H. Vega y J. Pariona. «Procesamiento de lenguaje natural». En: *Revista de Ingeniería de Sistemas e Informática* 6.2 (2009), págs. 45-54.
- [24] P. Cimiano, C. Unger y J. McCrae. *Ontology-Bases Interpretation of Natural Language*. Morgan & Claypool Publishers, 20014.
- [25] P. Liang. «Talking to Computers in Natural Language». En: *XRDS* 21.1 (2014), págs. 18-21.
- [26] R. Panwar, Meenakshi y A. Kumar. «Natural Language Ambiguity and its Effect on Machine Learning». En: *International Journal for Scientific Research & Development* 3.4 (2015), págs. 166-170.
- [27] T. Bench-Capon. *Knowledge Representation: An Approach to Artificial Intelligence*. Academic Press Professional, Inc., 1990.
- [28] R. Brachman y H. Levesque. En: *Knowledge Representation and Reasoning*. Morgan Kaufmann, 2004.
- [29] S. J. Russell y P. Norvig. *Artificial Intelligence: A Modern Approach*. 3.^a ed. Pearson Education, 2010.
- [30] F. Baader y col., eds. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
- [31] F. Baader, I. Horrocks y U. Sattler. «Description Logics as Ontology Languages for the Semantic Web». En: *Mechanizing Mathematical Reasoning: Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday*. Ed. por D. Hutter y W. Stephan. Springer Berlin Heidelberg, 2005, págs. 228-248.

- [32] I. Rahwan y G. Simari. *Argumentation in Artificial Intelligence*. Springer Publishing Company, Incorporated, 2009.
- [33] F. Baader, I. Horrocks y U. Sattler. «Description Logics». En: *Handbook on Ontologies*. Ed. por S. Staab y R. Studer. Springer Berlin Heidelberg, 2004, págs. 3-28.
- [34] M. Viinikkala. *Ontology in Information Systems*. 2004. URL: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.201.2639> (visitado 19-02-2018).
- [35] B. Smith. *Ontology and Information Systems*. URL: http://ontology.buffalo.edu/ontology_long.pdf (visitado 19-02-2018).
- [36] T. Gruber. «A translation approach to portable ontology specifications». En: *Knowledge Acquisition* 5.2 (1993), págs. 199 -220.
- [37] W. Borst. «Construction of Engineering Ontologies for Knowledge Sharing and Reuse». Tesis doct. Institute for Telematica e Information Technology, University of Twente, 1997.
- [38] R. Studer, V. Richard y D. Fensel. «Knowledge engineering: Principles and methods». En: *Data & Knowledge Engineering* 25.1 (1998), págs. 161-197.
- [39] D. Fensel. «Ontologies». En: *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*. Springer Berlin Heidelberg, 2001, págs. 11-18.
- [40] N. Rodríguez y col. «A Survey on Ontologies for Human Behavior Recognition». En: *ACM Comput. Surv.* 46.4 (2014), 43:1-43:33.
- [41] C. Roussey y col. «An Introduction to Ontologies and Ontology Engineering». En: *Ontologies in Urban Development Projects*. Vol. 1. 2011, págs. 9-38.
- [42] W3C. *OWL Web Ontology Language Overview*. URL: <https://www.w3.org/TR/owl-features/> (visitado 23-02-2018).
- [43] M. Horridge y col. *A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0*. 2004. URL: <https://www.cse.buffalo.edu/faculty/shapiro/Courses/CSE663/Fall107/ProtegeOWLTutorial.pdf> (visitado 23-02-2018).
- [44] D. Riboni y col. «Is ontology-based activity recognition really effective?» En: *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. 2011, págs. 427-431.
- [45] S. Zolfaghari, R. Zall y M. R. Keyvanpour. «SONAr: Smart Ontology activity recognition framework to fulfill semantic web in smart homes». En: *2016 Second International Conference on Web Research (ICWR)*. 2016, págs. 139-144.
- [46] D. Mitchell, P. J. Morrow y C. D. Nugent. «A sensor and video based ontology for activity recognition in smart environments». En: *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2014, págs. 5932-5935.
- [47] K. Wongpatikaseree y col. «Activity Recognition Using Context-Aware Infrastructure Ontology in Smart Home Domain». En: *2012 Seventh International Conference on Knowledge, Information and Creativity Support Systems*. 2012, págs. 50-57.

- [48] I. Bae y H. Kim. «An Ontology-Based ADL Recognition Method for Smart Homes». En: *Communication and Networking*. Ed. por T. Kim y col. Springer Berlin Heidelberg, 2011, págs. 371-380.
- [49] D. Riboni y C. Bettini. «COSAR: Hybrid Reasoning for Context-aware Activity Recognition». En: *Personal Ubiquitous Comput.* 15.3 (2011), págs. 271-289.
- [50] N. Eagle y A. Pentland. «Reality Mining: Sensing Complex Social Systems». En: *Personal Ubiquitous Comput.* 10.4 (2006), págs. 255-268.
- [51] T. Gill, H. Allore y Z. Guo. «Restricted activity and functional decline among community-living older persons». En: *Archives of Internal Medicine* 163.11 (2003), págs. 1317-1322.
- [52] Protégé. *A free, open-source ontology editor and framework for building intelligent systems*. URL: <https://protege.stanford.edu> (visitado 27-03-2018).
- [53] Kontakt.io. *What is a beacon?* URL: <https://kontakt.io/beacon-basics/what-is-a-beacon/> (visitado 28-03-2018).
- [54] ChartJS. *Simple yet flexible JavaScript charting for designers & developers*. URL: <https://www.chartjs.org> (visitado 04-04-2018).
- [55] D3. *Data Driven Documents*. URL: <https://d3js.org> (visitado 04-04-2018).
- [56] SIMILE project. *Timeline Web Widget for Visualizing Temporal Data*. URL: <http://www.simile-widgets.org/timeline/> (visitado 04-04-2018).
- [57] W3C. *SPARQL Query Language for RDF*. URL: <https://www.w3.org/TR/rdf-sparql-query/> (visitado 04-04-2018).
- [58] S. Shanmuganathan y S. Samarasinghe. *Artificial Neural Network Modelling*. 1st. Springer, 2016.
- [59] K. Priddy y P. Keller. *Artificial Neural Networks: An Introduction*. SPIE- International Society for Optical Engineering, 2005.
- [60] L. Fausett, ed. *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Prentice-Hall, Inc., 1994.
- [61] D. Kriesel. *A Brief Introduction to Neural Networks*. 2007. URL: http://www.dkriesel.com/en/science/neural_networks.
- [62] W. McCulloch y W. Pitts. «A logical calculus of the ideas immanent in nervous activity». En: *The bulletin of mathematical biophysics* 5.4 (1943), págs. 115-133.
- [63] M. Cohen y D. Hudson. *Neural Networks and Artificial Intelligence for Biomedical Engineering*. 1st. Wiley-IEEE Press, 1999.
- [64] D. Hebb. *The organization of behavior: A neuropsychological theory*. Wiley, 1949.
- [65] F. Rosenblatt. «The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain». En: *Psychological Review* (1958), págs. 65-386.
- [66] B. Widrow y M. Hoff. «Adaptive Switching Circuits». En: *1960 IRE WESCON Convention Record* (1960), págs. 96-104.

- [67] Marvin M. Minsky y S. Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.
- [68] H. Demuth y col. *Neural Network Design*. 2nd. Martin Hagan, 2014.
- [69] D. Rumelhart y J. McClelland, eds. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. MIT Press, 1986.
- [70] F. Gaxiola, P. Melin y F. Valdez. *New Backpropagation Algorithm with Type-2 Fuzzy Weights for Neural Networks*. 1st. Springer, 2016.
- [71] B. Ripley y N. Hjort. *Pattern Recognition and Neural Networks*. 1st. Cambridge University Press, 1995.
- [72] J. Schneider. *Cross Validation*. 1997. URL: <https://www.cs.cmu.edu/~schneide/tut5/node42.html> (visitado 16-03-2018).
- [73] H. D. Mehr, H. Polat y A. Cetin. «Resident activity recognition in smart homes by using artificial neural networks». En: *2016 4th International Istanbul Smart Grid Congress and Fair (ICSG)*. 2016, págs. 1-5.
- [74] Y. Chen y col. «OCEAN: A New Opportunistic Computing Model for Wearable Activity Recognition». En: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM, 2016, págs. 33-36.
- [75] A. Tamamori y col. «An investigation of recurrent neural network for daily activity recognition using multi-modal signals». En: *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. 2017, págs. 1334-1340.
- [76] K. Adhikari, H. Bouchachia y H. Nait-Charif. «Activity recognition for indoor fall detection using convolutional neural network». En: *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*. 2017, págs. 81-84.
- [77] J. Li y col. «Convolutional neural networks (CNN) for indoor human activity recognition using Ubisense system». En: *2017 29th Chinese Control And Decision Conference (CCDC)*. 2017, págs. 2068-2072.
- [78] J. Monteiro y col. «Deep neural networks for kitchen activity recognition». En: *2017 International Joint Conference on Neural Networks (IJCNN)*. 2017, págs. 2048-2055.
- [79] W. Jiang y Z. Yin. «Human Activity Recognition Using Wearable Sensors by Deep Convolutional Neural Networks». En: *Proceedings of the 23rd ACM International Conference on Multimedia*. ACM, 2015, págs. 1307-1310.
- [80] P. Viola y M. Jones. «Rapid object detection using a boosted cascade of simple features». En: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 1. 2001, págs. I511-518.
- [81] MathWorks. *MATLAB*. URL: <https://es.mathworks.com/products/matlab.html> (visitado 20-04-2018).