UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

"REDES NEURONALES DE RETROPROPAGACIÓN COMO TÉCNICA DE INFERENCIA DE LA UBICACIÓN DE USUARIOS MÓVILES DE WLANS DENTRO DE EDIFICIOS"

TESIS
PARA OBTENER EL TÍTULO DE
INGENIERO EN COMPUTACIÓN

PRESENTA
EDGAR ALBERTO MARTÍNEZ CRUZ

DIRECTORES DE TESIS
M. C. RAÚL CRUZ BARBOSA
DR. JESÚS FAVELA VARA

HUAJUAPAN DE LEÓN, OAXACA,
ABRIL DE 2004.

Redes Neuronales de Retropropagación como Técnica de Inferencia de la Ubicación de Usuarios Móviles de WLANs dentro de Edificios

Edgar Alberto Martínez Cruz

Abril 2004

Contenido

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

El desarrollo de la tecnología en comunicaciones inalámbricas se ha incrementado en los últimos años. Nuevas tecnologías inalámbricas innovan el campo de las comunicaciones haciendo cada vez más fácil la transmisión de datos de un lugar a otro. Si a esto le aunamos el auge que están teniendo las computadoras portátiles o *handhelds*, que permiten a los usuarios acceder a recursos de cómputo independientemente de su localización, tenemos una tecnología que permite el acceso a información y servicios de cómputo en escenarios que requieren la movilidad del usuario.

El incremento en la movilidad crea situaciones donde el contexto del usuario (como la ubicación espacial de éste, la gente y los objetos alrededor de él), es dinámico. Con una amplia variedad de posibles situaciones del usuario, se requiere adaptar los servicios que se le puedan proporcionar, en pos de dar un buen soporte a la interacción humano-computadora.

Para proporcionar un servicio adecuado, muchos de los sistemas emergentes del cómputo móvil necesitan conocer la ubicación física de las personas u objetos. El conocimiento de la ubicación del usuario tiene múltiples aplicaciones en el ambiente civil, comercial, militar y de emergencias. Desde ayudar a un turista en su paso por un pueblo hasta avisar a un restaurante que hay gente cercana buscando comida. Algunas preguntas a las que las aplicaciones podrían responder conociendo la ubicación del usuario serían : ¿Cuál es la impresora más cercana a mí en este edificio?, ¿Cómo debería nuestro equipo de rescate moverse rápidamente a través del edificio para localizar a las víctimas del siniestro?, ¿Podré mostrar automáticamente mis resultados en el pizarrón electrónico que tengo enfrente de mí?

1.1 Planteamiento del Problema

La ubicación es difícil de detectar porque requiere el uso de sensores y dispositivos de cómputo especializados, y con frecuencia la información cambia con el tiempo: por ejemplo, un guía turístico móvil que actualiza su pantalla conforme el usuario cambia su posición, debe hacer uso de múltiples sensores para cubrir el área en la que el usuario navega.

La efectividad de las técnicas utilizadas para el rastreo y posicionamiento variará dependiendo de la aplicación, que les demandará mayor o menor resolución espacial o temporal.

Dada la importancia del tema, se han propuesto varios sistemas y métodos para la determinación de la ubicación de las personas. Sistemas para ambientes exteriores como el Sistema de Posicionamiento Global (GPS) [DjG+01] permiten determinar la posición de computadoras móviles con precisión pero son inefectivos dentro de edificios. En técnicas de localización en interiores, existe infraestructura especializada, ya sea basada en infrarrojo [WaR+92]-[WaR+96][AbG+97] o en ultrasonido [HaA+01][PrN+00], sin embargo, su costo es alto por requerir de equipo especializado. Existen también, dentro de las técnicas para estimar localización en interiores, sistemas basados en video [DaT+98], que hacen uso de técnicas de reconocimiento de patrones, pero que requieren mucho tiempo de entrenamiento para lograr resultados aceptables.

En sistemas de localización basados en Radio Frecuencia (RF), existen sistemas que utilizan la infraestructura existente de una red LAN inalámbrica y la intensidad de la señal presente en la red - que todo dispositivo móvil dentro de la red inalámbrica recibe - para estimar la posición del móvil. Estos sistemas hacen uso de técnicas como Redes Bayesianas [CaP+01], reconocimiento de patrones [SmA+01] o distancias Euclidianas [BaP+00].

El método que se propone en este trabajo es basado en RF y utiliza como técnica de estimación de la ubicación a redes neuronales de retropropagación, las cuales reciben la intensidad de la señal del dispositivo móvil al punto de acceso como entrada y transforman esta información a coordenadas físicas.

1.2 Objetivo General

Diseñar la arquitectura de la red neuronal y evaluar el desempeño de las redes neuronales de retropropagación aplicadas como técnica de inferencia

de la ubicación de usuarios móviles de redes LAN inalámbricas dentro de edificios.

1.3 Objetivos Específicos

- Definir un método y la plataforma que permita medir las intensidades de las señales de los diversos puntos de acceso de una WLAN hacia un dispositivo portátil, en diversos puntos dentro de un edificio.
- Entrenar una red neuronal de retropropagación con los datos recolectados, probando con diversas arquitecturas, así como con diversos algoritmos de entrenamiento.
- Evaluar resultados de desempeño de la red neuronal.
- Comparar los resultados obtenidos con otras técnicas de localización.
- Construir una aplicación de software, que haga uso de la información de la ubicación, que le proporciona la red neuronal en una fase operacional dentro del edificio.

La organización de la tesis está dividida en cinco capítulos. En el capítulo uno se tiene la introducción, el planteamiento del problema y los objetivos. En el capítulo dos se describen las técnicas existentes para determinar la ubicación de un usuario móvil, se describe el concepto de cómputo consciente del contexto, y se presentan ejemplos. En el capítulo tres se presenta el concepto de red neuronal artificial, sus características y las arquitecturas de red más usadas. El desarrollo del proyecto es descrito en el capítulo cuatro. En el capítulo cinco se plasman los resultados obtenidos. Enseguida, las conclusiones y las perspectivas. Finalmente, se presentan los apéndices.

Capítulo 2

Computación Consciente del Contexto

En este capítulo se introduce el concepto del cómputo consciente del contexto y su influencia directa en la creación de los diversos sistemas de localización de usuarios, que también aquí se describen.

2.1 Cómputo Consciente del Contexto

2.1.1 Introducción

Dos tecnologías permiten a los usuarios moverse con poder de cómputo y recursos de red a la mano: computadoras portátiles y comunicaciones inalámbricas. El usuario ha incrementado su libertad de movilidad, lo cual le crea situaciones donde el contexto en el que se encuentre (tal como su ubicación, la gente o los objetos a su alrededor) sea más dinámico.

Los asistentes personales digitales (PDAs) y el cómputo ubicuo (también llamado cómputo pervasivo, en el cual se permite el uso de cómputo haciendo a muchas computadoras disponibles para el usuario en el ambiente físico, pero haciéndolas invisibles para él [WeM91]) le han dado al usuario la noción de que puede acceder a servicios de información en cualquier momento y en cualquier lugar.

Investigadores de cómputo móvil han tratado de esconder la movilidad y hacer que las desconexiones frecuentes sean transparentes para el usuario final. Ejemplos de esto incluye el sistema de archivos Coda [SaM+93] y en

trabajo de móvil IP [PeCh+96]. Sin embargo, además de adaptar sistemas y aplicaciones en donde la movilidad se oculta, se debería explorar y proveer de infraestructura que soporte nuevas aplicaciones conscientes de la movilidad. Tales aplicaciones serían más efectivas y adaptativas a las necesidades de información de los usuarios sin consumir mucha de su atención si pudieran tomar ventaja de las características del ambiente dinámico, tales como la ubicación del usuario, gente cercana, hora del día e incluso intensidad de luz y niveles de ruido [ChG+00].

2.1.2 Qué es el Contexto

Muchos investigadores han tratado de definir que es el contexto en términos del cómputo móvil. Schilit divide el contexto en tres categorías [ShB+94]:

- **Contexto de Cómputo**, tales como la conectividad de la red, los costos de comunicación y el ancho de banda, además los recursos cercanos como las impresoras, pantallas y estaciones de trabajo.
- **Contexto del usuario**, tales como el perfil del usuario, su ubicación espacial, la gente que está cerca de él e incluso su situación social actual.
- **Contexto físico**, como la intensidad de la luz, niveles de ruido, condiciones de tráfico y temperatura.

El tiempo es también un contexto natural e importante para muchas aplicaciones y en [ChG+00] proponen agregar una cuarta categoría llamada contexto temporal, que incluye la hora del día, semana, mes y estación del año. Además, cuando el usuario y su paso por el contexto físico son grabados a través del tiempo, se obtiene un contexto histórico o historial, el cual puede ser también útil para ciertas aplicaciones.

En [DeA+99] definen al contexto como "Cualquier información que puede ser usada para caracterizar la situación de una entidad, donde una entidad puede ser una persona, lugar u objeto que es considerado relevante en la interacción entre un usuario y una aplicación, incluyendo al usuario y aplicación mismos".

Combinando diversos aspectos del contexto se puede generar un mejor entendimiento de la situación actual. Los contextos "primarios" incluyen a la ubicación, identidad, actividad y tiempo, que corresponden al *donde, quien, que y cuando* se está realizando una actividad. Estos 4 contextos actúan

como índices para otras fuentes de información contextual [DeA+99]. Por ejemplo, conociendo la ubicación actual del usuario, el día y la hora, junto con el calendario del usuario se puede tener una buena idea de su situación social actual, si está presente en una reunión, sentado en clase, esperando en el aeropuerto, etc.

Un aspecto importante del contexto son aquellas características del ambiente circundante que, por un lado, *determinan* el comportamiento de las aplicaciones móviles y aquellas que son *relevantes* a las aplicaciones pero no críticas [ChG+00]. Para las aplicaciones no será necesario adaptarse a las características del segundo tipo excepto para mostrarlas a los usuarios interesados.

2.1.3 Cómputo Consciente del Contexto

Dey define a la computación consciente del contexto como "el uso del contexto para proveer información relevante y/o servicios al usuario, en donde sea que éste se encuentre". Tres importantes comportamientos conscientes del contexto que una aplicación puede exhibir son [DeA+99]:

- **La presentación de información y servicios a un usuario**. Un ejemplo es la información de dónde se encuentra el banco más cercano, otro ejemplo podría ser una interfaz de usuario que cambia sus opciones dependiendo de la hora, día o ubicación. Esta categoría se refiere a aplicaciones que presentan información contextual al usuario o usan el contexto para proponer opciones apropiadas.

- **La ejecución automática de un servicio**. Un ejemplo es cuando una persona entra en una habitación y su correo electrónico le es mostrado en la terminal mas cercana, otro ejemplo podría ser el cambio en el volumen del teléfono de acuerdo al nivel del ruido en el ambiente. Esta categoría se refiere a aplicaciones que activan una instrucción, o reconfiguran el sistema en beneficio del usuario y de acuerdo al cambio del contexto.

- **Etiquetar el contexto de la información para permitir su futura recuperación**. Esta categoría se refiere a aplicaciones que etiquetan los datos adquiridos con información relevante del contexto. Como por ejemplo dejar notas virtuales en una ubicación para que posteriormente sean leídas por alguien más.

2.2 Sistemas de Localización

Para que una aplicación consciente del contexto pueda hacer uso de la información de la ubicación del usuario, debe haber un sistema de localización que lo soporte. Hasta ahora la mayoría de los que crean las aplicaciones concientes del contexto crean también sus propios sistemas de localización. En esta sección se presentan diversos sistemas de localización y las técnicas de las que hacen uso para tal fin que han sido reportadas en la literatura.

2.2.1 Sistemas y Técnicas de Localización en Ambientes de Exteriores

GPS

GPS (Sistema de Posicionamiento Global) es un sistema de navegación mundial basado en satélites. El sistema consta de 24 satélites, igualmente espaciados en seis órbitas planas y a una altura de 20,200 metros sobre la tierra, que transmiten dos señales portadoras de código, una para uso civil y otra para uso militar y del gobierno [DjG+01]. Los satélites transmiten mensajes de navegación, donde un receptor GPS los utiliza para determinar su posición. Los receptores GPS procesan la señal para determinar la posición en 3D – latitud, longitud y altitud – con una precisión de 10 metros o menos. Sin embargo, los receptores GPS necesitan una línea de recepción de los satélites despejada y recibir la señal de al menos tres o cuatro satélites. Además, la señal GPS no trabaja adecuadamente dentro de edificios porque la intensidad de la señal es muy baja para penetrar un edificio, lo cual excluye a GPS como alternativa tecnológica para determinar la posición de un receptor dentro de los edificios [ChG+00].

Identificación de Células (Cell-ID)

El método de identificación de células es una técnica básica. El método toma en cuenta el hecho que redes móviles pueden dar un aproximado de la posición de un móvil conociendo la célula en la que se encuentre en ese momento. Esta técnica está en uso y es soportada por una amplia variedad de dispositivos móviles. Sin embargo, la precisión del método depende del tamaño de la célula, y ésta generalmente es de varios kilómetros, al menos en el caso de telefonía celular [GiG+02].

Dirección del ángulo de llegada (AOA)

En esta técnica, la idea básica es mover en el espacio una antena direccional hasta que la dirección de la máxima intensidad de la señal o fase coherente sea detectada [SaS94]. En los sistemas satelitales móviles la movilidad en la dirección de la antena es difícil de implementar, la cual es necesaria para lograr una precisión adecuada en aplicaciones de cómputo conscientes del contexto.

Retardo de tiempo

Como las ondas electromagnéticas viajan a una velocidad constante en el espacio, la distancia entre dos puntos puede ser inferida midiendo el retardo en el tiempo de las ondas de radio transmitidas entre ellos. Este método es muy útil y llevado a cabo por los sistemas de satélites. Existen dos tipos de métodos de retardos de tiempo: Medición Absoluta del Tiempo de Llegada (TOA) y Medición del Tiempo Diferencial de Llegada (TDOA).

En la medición absoluta del tiempo de llegada, la posición es estimada del tiempo absoluto que la onda tardó en viajar entre el transmisor y el receptor, o viceversa. Esto implica que el receptor conoce exactamente el tiempo de transmisión.

En la medición del tiempo diferencial de llegada se elimina el problema de que tenga que haber una sincronización entre el transmisor y el receptor utilizando muchos transmisores sincronizados a un tiempo común base y midiendo la diferencia del tiempo de llegada en el receptor [ZeV+03].

2.2.2 Sistemas de Localización para Ambientes dentro de Edificios

Sistemas basados en Infrarrojo

El sistema de localización *Active Badge* fué desarrollado en los laboratorios Olivetti [WaR+92] y consiste en un sistema de proximidad que utiliza tecnología basada en el infrarrojo (IR). Cada persona que el sistema puede localizar trae consigo un dispositivo metálico de tecnología de infrarrojo o *badge*. El badge emite un único identificador cada diez segundos o cuando se requiere. Un servidor central es el que reúne los datos de los sensores infrarrojos colocados en distintas ubicaciones de un edificio.

La ubicación de un badge es simbólica, representa, por ejemplo, la sala en que el badge está ubicado. La información simbólica puede ser utilizada en el desarrollo de arquitecturas de software que requieran el uso de esta información, como en [HaA+01]. Sistemas similares basados en IR son el sistema *ParcTab* de Xerox [WaR+92] y el proyecto *Cyberguide* [AbG+97]. En sistemas basados en infrarrojo se tiene el problema de interferencia en lugares con luz fluorescente o luz solar, además de que el rango de alcance de estos sistemas es de unos pocos metros.

Sistemas basados en Ultrasonido

Active Bat. Investigadores de AT&T desarrollaron el sistema de localización Active Bat, el cual utiliza dispositivos transmisores en forma similares a los de Active Badge, sin embargo, estas unidades llamadas *Bats* emiten pulsos ultrasónicos [HaA+99]. Una estación base transmite periódicamente un mensaje de radio que contiene un identificador, que hace que el correspondiente Bat emita un pulso corto de ultrasonido. Al mismo tiempo que la estación base emite el identificador de radio frecuencia hacia el Bat, también se manda una señal de reset hacia los sensores ubicados en la habitación. Cada sensor receptor en el sistema mide el tiempo entre la señal de reset y la llegada del pulso ultrasónico del Bat y calcula la distancia hacia él. El controlador local entonces manda las mediciones de los sensores a un controlador central. Las distancias del Bat a tres o más receptores pueden ser utilizadas para encontrar la posición en 3D usando el proceso de multilateración. El sistema puede localizar dispositivos o Bats con una precisión de 9 cm. de la ubicación real en un 95% de los casos. Sin embargo, la infraestructura de paneles, que realizan el cálculo de la señal recibida en *Active Bat*, requiere una alta escalabilidad, desarrollo y costo.

Cricket. El sistema de localización Cricket utiliza emisores ultrasónicos en la infraestructura y receptores dentro del objeto a ser localizado [PrN+00]. En Cricket el objeto móvil es el que realiza su propia triangulación para determinar su posición.

Como el sistema Active Bat, Cricket utiliza una señal de control de radio frecuencia y paquetes de datos ultrasónicos para calcular el tiempo entre la emisión y la recepción. Pero este sistema no requiere que los múltiples sensores estén controlados centralmente porque cada receptor móvil realiza el cálculo del tiempo y su posición. Cricket puede delinear una región de

1.2 x 1.2 metros cuadrados. Las ventajas de Cricket son la privacidad y una escalabilidad descentralizada. Sin embargo, el costo del equipo especializado es alto.

Sistemas basados en Electromagnetismo

MotionStar. El uso de ondas electromagnéticas para posicionamiento es un método clásico [RaF+79]. Sistemas como *MotionStar* generan pulsos magnéticos en una antena transmisora y el sistema computa la posición y la orientación de las antenas receptoras midiendo la respuesta en los tres ejes ortogonales del pulso transmitido.

Estos sistemas magnéticos tienen la ventaja de una alta precisión, en el orden de menos de 1mm de resolución espacial. Sin embargo tienen un alto costo de implementación y la necesidad de que el objeto a ser localizado lleve consigo una unidad de control. Además, los sensores deben estar de uno a tres metros del transmisor y la precisión se degrada con la presencia de objetos metálicos en el ambiente.

Sistemas basados en Visión

El uso de tecnología de visión computacional para saber la ubicación de una persona u objeto ha sido también tratada por muchos grupos de investigadores. En [DaT+98] se utilizan diversas cámaras en tiempo real, posicionadas en diferentes puntos del edificio. Aunque se utilizan cámaras de alta precisión, los sistemas de visión requieren importantes recursos de cómputo para analizar las imágenes capturadas. El tipo de análisis que se hace a las imágenes incluye principalmente estimación de profundidad, segmentación del color y clasificación de patrones de intensidad.

Los sistemas basados en visión computacional, tienen el problema de que el rango de visión es limitado al número de cámaras en el sistema y al alcance que éstas tengan. Además trabajan sólo con un reducido número de personas dentro de una sala.

Sistemas basados en Contacto Físico

SmartFloor. Un tipo más de sistema de ubicación de personas es el sistema de localización *SmartFloor*, basado en contacto físico de la persona con el sistema [OrR+00]. En el sistema, sensores de presión capturan las pisadas de las personas y el sistema utiliza los datos para el rastreo de la persona

por el lugar, así como para el reconocimiento de la persona, por medio de un análisis de la forma del pie. Aunque este sistema no requiere que la gente cargue algún dispositivo consigo para su localización, tiene la desventaja de una pobre escalabilidad y un alto costo porque en todo el lugar donde se instale, el piso se tiene que alterar para instalar los sensores de presión.

Sistemas basados en RF

PinPoint. Un sistema basado en radio frecuencia (RF) es el sistema *PinPoint*. PinPoint utiliza dispositivos receptores que leen las señales de RF, los cuales deben cargar consigo el equipo o las personas que desean ser localizadas [WeJ+98].

Las antenas en el sistema emiten radio-señales de 2.4 GHz hacia los dispositivos receptores, y las radio-señales o tags son reflejadas por éstos con una señal de respuesta a 5.8 GHz junto con un código de identificación. Las señales son recibidas por antenas y mandan los resultados a células controladoras, que continuamente realizan una triangulación de las señales reflejadas por los dispositivos móviles para saber su ubicación. El número de antenas receptoras determina la precisión de ubicación, sin embargo, el mejor resultado es de tres metros de error en promedio de la ubicación real. Una desventaja es que el costo del sistema es alto, además de ser especializado para rastreo y ubicación de personas y equipo móvil.

2.2.3 Técnicas de localización RF para Ambientes dentro de Edificios y que Utilizan la Infraestructura de una LAN Inalámbrica

Este tipo de técnicas de localización son particularmente interesantes porque no requieren de infraestructura física especial y son alternativas viables para espacios cerrados. Además, con el crecimiento de las redes Wi-Fi podrían funcionar en muchos lados. La técnica propuesta en esta tesis corresponde a este tipo.

En el proyecto *Daedalus* se utiliza la infraestructura de una LAN inalámbrica para determinar la posición de un usuario móvil [HoT+97]. Los puntos de acceso transmiten señales junto con sus coordenadas físicas y el dispositivo móvil calcula su posición, que será la misma que la del punto de acceso del cual está recibiendo la señal. Aquí, sin embargo, la precisión estará limitada por el rango de alcance de la célula que forme el punto de acceso.

Existe otro conjunto de métodos, que utilizan la intensidad de la señal de los puntos de acceso de una WLAN, como son los métodos de los proyectos de Microsoft Research, CMU, UCLA y de la Universidad de Trento, para estimar la ubicación de un usuario móvil dentro de un edificio. La idea fundamental que comparten estos métodos es que en una red inalámbrica, el nivel de energía o intensidad de la señal de un paquete o *beacon*, que emiten los puntos de acceso hacia los dispositivos móviles, es relativa a la ubicación del dispositivo móvil receptor dentro del edificio [BaP+00]. Esto provee una alternativa para estimar la ubicación de un usuario móvil.

La intensidad de la señal se utiliza en la construcción, en una fase de registro, de un *Mapa* o base de datos de las mediciones de las intensidades conforme el usuario móvil se desplaza por todo el edificio y en diversos puntos geográficos registra la intensidad de la señal de los diversos puntos de acceso. En la base de datos se registran tuplas como $(x, y, z, ss_1, \ldots, ss_i, \ldots, ss_n)$, donde x, y, z son coordenadas físicas del punto del edificio donde se realizó la medición y ss_i es la intensidad de la señal recibida en el dispositivo móvil y que fue emitida por el i-ésimo punto de acceso. En el sistema RADAR, desarrollado por Microsoft Research [BaP+00], se utiliza como alternativa para crear la base de datos, un modelo matemático de la propagación de la señal RF dentro del edificio, en este modelo se toma en cuenta en la fórmula el número de paredes y demás obstrucciones en un edificio de la señal que se propaga en el espacio entre el emisor y el receptor.

Posterior a la fase de registro, en una fase de aplicación, el dispositivo móvil para poder estimar su posición registra las intensidades en ese momento de los puntos de acceso que estén en su rango de alcance y busca dentro de la base de datos o Mapa de Intensidades para encontrar la tupla $(x, y, z, ss_1, \ldots, ss_i, \ldots, ss_n)$ más cercana a la tupla que se acaba de crear con la medición.

La técnica de búsqueda que RADAR realiza en el Mapa de Intensidades es llamada *Nearest Neighbor in Signal Space* (NSSS). El algoritmo NSSS calcula la distancia Euclidiana (en el espacio de la señal) entre cada tupla en el Mapa de Intensidades y la tupla medida. El algoritmo escoge entonces aquella tupla que minimiza la distancia en espacio de la señal y toma sus coordenadas físicas y las declara como la estimación de la posición del usuario móvil. Una variante del algoritmo NSSS consiste en que cuando hay más de una tupla que son las más cercanas a la tupla medida, realiza un promedio de las coordenadas físicas del conjunto de tuplas para estimar la ubicación del móvil.

En el sistema desarrollado por [CaP+01], en UCLA, se registra la relación señal a ruido (SNR) para construir el Mapa, en vez de la intensidad de la señal. En este sistema se aplican modelos de Redes Bayesianas sobre el Mapa de Intensidades para estimar la ubicación del usuario móvil.

En la Universidad Carnegie Mellon (CMU), se desarrollaron dos algoritmos, el CMU-PM (Emparejamiento de Patrones) y el CMU-TMI (Triangulación, Mapeo e Interpolación) [SmA+01]. CMU-PM implementa un algoritmo de emparejamiento de patrones, similar al NSSS de RADAR. CMU-TMI necesita conocer las posiciones de los puntos de acceso en el área con el fin de trasladar las intensidades de las señales a distancias y realizar una triangulación con las distancias obtenidas de al menos tres puntos de acceso. Después de realizar la triangulación, el área de espacio de posibles posiciones es promediada para obtener un estimado de la posición en el espacio. El Mapa de Intensidades es usado entonces para mapear la posición en el espacio a una posición física. CMU-TMI utiliza también interpolación para utilizar el mínimo de datos en el Mapa de Intensidades.

En la Universidad de Trento, Italia, se utilizan redes neuronales para inferir la ubicación del dispositivo móvil [BaR+02]. Aquí no se utiliza un algoritmo de búsqueda o de emparejamiento sobre el Mapa, como en RADAR, CMU y UCLA, sino que el Mapa es utilizado para entrenar a la red neuronal, y que, mediante el aprendizaje por ejemplos, traslade las intensidades de las señales de los puntos de acceso a una posición física. El método de entrenamiento de la red neuronal utilizado es el de la Secante en un Paso [BaR92].

2.3 Aplicaciones Conscientes del Contexto

En esta sección, se presentan algunas aplicaciones reportadas en la literatura sobre consciencia del contexto y cómo la información contextual fue manejada en ellas.

Call Forwarding

Creado por el grupo Olivetti y basado en el sistema Active Badge, la ubicación del usuario es presentada al recepcionista, quien dirige la llamada telefónica hacia el teléfono más cercano al usuario [WaR+92].

Active Map

Desarrollado por el grupo XEROX Parc, utiliza la ubicación del usuario como información contextual, que es recolectada por el sistema PARC Tab. La ubicación de la gente es mostrada en un mapa, el cual es actualizado cada pocos segundos, permitiendo a la gente ser fácilmente localizada o conocer en donde se está efectuando una reunión en la que uno podría estar interesado [WaR+96].

Comunicaciones Móviles en Hospitales

En este proyecto, la comunicación entre el personal del hospital, como son las enfermeras, los doctores y otros asistentes, se apoya en un sistema que reconoce el contexto en el cual los trabajadores están realizando sus tareas. El sistema, por medio de clientes instalados en las computadoras portátiles o de bolsillo que el personal lleva consigo, permite el intercambio de mensajes instantáneos entre ellos, además del acceso a servicios, tales como obtener el historial médico del paciente de la cama que el doctor está atendiendo en ese momento o dejar una recomendación para la siguiente enfermera que venga a ver al paciente. Además, se puede visualizar en un mapa del dispositivo portátil la ubicación de todo el personal a través del hospital. El proyecto está siendo desarrollado en el Centro de Investigación Científica y de Educación Superior de Ensenada [MuM+03].

Asistente de Compras

El dispositivo puede guiar a los compradores a través de una tienda, proveerles detalles de los artículos, ayudarles a localizarlos, hacer un análisis comparativo de precios y más. Hay un aspecto de privacidad relativo a este proyecto, donde la tienda mantiene los perfiles de los compradores. Como consecuencia, los compradores son divididos en dos clases: clientes regulares, quienes compran anónimamente sin un perfil y compradores frecuentes, quienes se registran en el sistema, el cual mantiene sus perfiles de compra. Este proyecto fue desarrollado por el grupo de laboratorios AT&T [AsA+94].

Guía Turístico

Desarrollado por el grupo de Futuros Ambientes Computacionales, del Instituto de Tecnología de Georgia, utiliza la ubicación del turista para proveerle

de servicios cercanos a él, por ejemplo, el turista puede encontrar direcciones, obtener información y dejar comentarios en el mapa interactivo. El viaje diario es automáticamente compilado usando la historia de a donde ha viajado un turista y es usado por el sistema para hacer sugerencias de lugares de interés para visitar. La información de la ubicación es recolectada en ambientes externos por el sistema GPS y en ambientes dentro de edificios por un sistema de posicionamiento por infrarrojo [AbG+97].

Asistente de Conferencias

Desarrollado también por el grupo de Futuros Ambientes Computacionales del Instituto de Tecnología de Georgia. El asistente de conferencias utiliza una amplia variedad de información de contexto para ayudar a los participantes de las conferencias, entre los que está la ubicación, el tiempo local y un programa de participaciones de la conferencia. El asistente examina el programa de la conferencia, los temas de las presentaciones, la ubicación del usuario y sus intereses de investigación, con el fin de sugerirle a que presentaciones asistir. Cuando el usuario entra a una sala donde se efectúa la conferencia, el Asistente de Conferencias despliega automáticamente en el dispositivo inalámbrico el nombre del presentador, el título de la presentación y otra información interesante para el usuario. El Asistente graba automáticamente las diapositivas de la presentación, así como los comentarios y preguntas para usarlas posteriormente [DeAK+99].

Trabajo de Campo

El sistema se ocupa en proveer un conjunto de herramientas para la asistencia del trabajador de campo en sus observaciones y actividades de recolección de datos. El sistema utiliza la ubicación como información contextual para ayudarlo, por ejemplo, en recolectar la información acerca del ambiente en el que se encuentre. Para facilitar aún más la recolección de datos de manera automática, aparte de utilizar información contextual como la ubicación y el tiempo, la información también es etiquetada con la ubicación en el mapa para un análisis posterior. El sistema fue desarrollado por la Universidad de Kent, en Canterbury [PaJ+98].

Asistente de Oficina

Desarrollado por el Laboratorio MIT Media, el asistente es un agente que interactúa con los visitantes en la puerta de la oficina y administra el programa de trabajo del propietario de la oficina. El Asistente es activado cuando un visitante se aproxima, el cual es detectado por dos alfombrillas ubicadas en ambos lados de la puerta de la oficina, y éste adaptará su comportamiento conforme a la información contextual, como la identidad del visitante y el estado de la agenda del propietario de la oficina [YaH+00].

Capítulo 3

Redes Neuronales Artificiales

Dentro de un edificio, la relación entre la distancia y la intensidad de la señal recibida de los distintos puntos de acceso en una WLAN se vuelve una función compleja debido a la geometría del edificio y a la infraestructura que contiene, considerando además que a la señal la afectan también el clima, obstrucciones como personas y objetos y la presencia de otros objetos que emitan señales de RF. Es muy difícil obtener un modelo matemático que describa la relación distancia-intensidad de señal porque tiene que tomar en cuenta en su fórmula a todos los objetos que interfieren y que hacen que la intensidad de la señal disminuya; además, el modelo matemático tiene que calcularse nuevamente para cada edificio. El uso de redes neuronales se vuelve entonces una opción importante a considerar debido a la relación compleja de distancia-intensidad y a la falta de información completa que describa totalmente el ambiente [BaR+02].

En este capítulo se describen los conceptos básicos de las redes neuronales artificiales. Se explican también las diversas estrategias de aceleramiento del algoritmo de retropropagación y que fueron aplicadas al entrenamiento de la red neuronal en el desarrollo de este trabajo, con el fin de mejorar su rendimiento

3.1 Red Neuronal Biológica

Una red neuronal biológica tiene tres tipos de componentes que son de particular interés para las redes neuronales artificiales: dendritas, el cuerpo o soma y el axón. Las dendritas son las que reciben señales de otras neuronas,

formando una *sinapsis*. El cuerpo de la neurona o soma contiene el núcleo, el cual se encarga de todas las actividades metabólicas de la neurona y recibe la información de otras neuronas vecinas a través de las conexiones sinápticas. La transmisión de una señal de una célula a otra por medio de la sinápsis es un proceso químico: en él se liberan substancias transmisoras en el lado del emisor de la unión y su efecto es elevar o disminuir el potencial eléctrico dentro del cuerpo de la célula receptora. Si el potencial alcanza el umbral se envía un pulso o potencial de acción por el axón y se dice, entonces, que la célula se disparó. Este pulso alcanza a otras neuronas a través de la distribución de los axones. En la figura 3.1 se muestran los componentes básicos de una neurona biológica.

Una característica importante que las redes neuronales biológicas comparten con las redes artificiales es la tolerancia a fallos. Por ejemplo, los humanos somos capaces de reconocer muchas señales de entrada que sean un poco diferentes de alguna señal que hayamos visto antes. La tolerancia a daños en el sistema neurológico es otra característica, en donde a pesar de la pérdida continua de neuronas, el sistema sigue aprendiendo [FaL94].

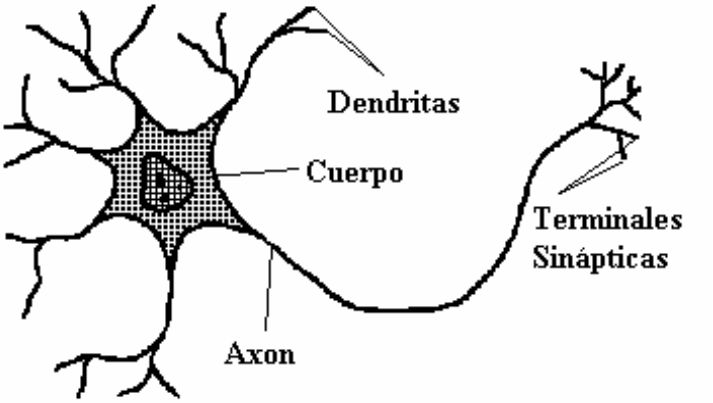

Figura 3.1: Neurona Biológica

3.2 Red Neuronal Artificial

En forma general, una red neuronal artificial es una máquina que modela la forma en la cual el cerebro realiza una tarea en particular o función de interés [HaS94]. Para realizar su objetivo, una red neuronal artificial emplea las interconexiones masivas entre unidades simples de cómputo o neuronas.

Una red neuronal puede definirse de la siguiente manera [HaS94]:

> *Una red neuronal es un procesador distribuido masivamente paralelo que tiene una propensión natural para almacenar conocimiento experimental y hacerlo disponible para su uso. La red neuronal asemeja al cerebro en dos aspectos:*

1. *El conocimiento es adquirido por la red a través de un proceso de aprendizaje*
2. *Las fuerzas de conexión entre neuronas conocidas como pesos sinápticos son utilizados para almacenar el conocimiento.*

3.3 El Modelo de una Neurona Artificial

La neurona es la unidad de procesamiento básica de una red neuronal. En ella se pueden identificar algunos elementos básicos, los cuales se mencionan a continuación [HaS94]:

- *Un conjunto de ligas sinápticas*, donde cada una de ellas tiene un peso que la caracteriza. Una señal de entrada x_j conectada en la sinapsis j de la neurona k es multiplicada por el peso sináptico w_{kj}.
- *Un sumador o combinador lineal*, para sumar o agregar las señales de entrada, que previamente fueron multiplicadas por los respectivos pesos sinápticos w_{kj} de la neurona.
- *Una función de activación* $\varphi()$, que limita o normaliza la amplitud de la señal de salida de la neurona.
- *Un umbral* θ_k, para disminuir la entrada de la función de activación. Cuando se tiene el objetivo de aumentar la entrada, se emplea un *bias* o sesgo, en vez de un umbral.

En la figura 3.2 se puede observar el modelo de una neurona. Matemáticamente, se puede describir una neurona k de la siguiente manera [HaS94]:

$$u_k = \sum_{j=1}^{p} w_{kj} x_j, \tag{3.1}$$

además de la siguiente ecuación:

$$y_k = \varphi(u_k - \theta_k), \tag{3.2}$$

donde la ecuación 3.1 es el sumador o combinador lineal y la ecuación 3.2 corresponde a la salida y_k, que representa el estado de activación de la neurona. Si se define v_k como

$$v_k = u_k - \theta_k, \tag{3.3}$$

se puede introducir el umbral θ_k en la ecuación 3.1 de la siguiente manera:

$$v_k = \sum_{j=0}^{p} w_{kj} x_j, \tag{3.4}$$

donde la entrada cero se define como

$$x_0 = -1, \tag{3.5}$$

y cuyo peso sináptico es

$$w_{k0} = \theta_k. \tag{3.6}$$

3.4 Funciones de activación

Como se mencionó anteriormente, la operación básica de una neurona artificial incluye el sumar las señales de entrada previamente multiplicadas por los pesos sinápticos y posteriormente la aplicación de una función de activación a esta suma, la cual va a normalizar la salida de la neurona. A continuación se presentan las funciones de activación más comunes [HaS94].

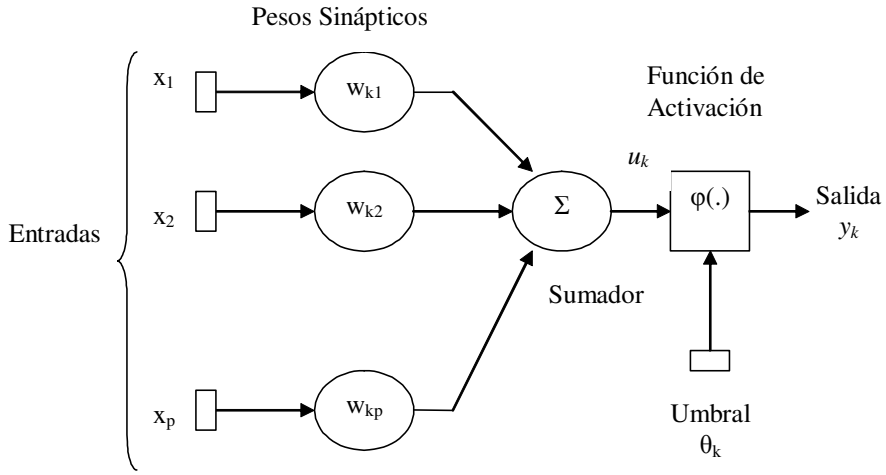

Figura 3.2: Modelo de una neurona artificial

3.4.1 Función Escalón

La función escalón se describe mediante la siguiente ecuación:

$$\varphi(v) = \begin{cases} 1, & si\ v \geq 0 \\ 0, & si\ v < 0 \end{cases} . \tag{3.7}$$

Igualmente, la salida de la neurona k, que emplea esta función es:

$$y_k = \begin{cases} 1, & si\ v_k \geq 0 \\ 0, & si\ v_k < 0 \end{cases} , \tag{3.8}$$

donde v_k es el nivel de actividad interna de la neurona, que es:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j - \theta_k. \tag{3.9}$$

En la Fig. 3.3 a) se puede observar la función escalón.

3.4.2 Función Identidad

La función identidad se define como

$$\varphi(v) = v \qquad Para\ toda\ v. \tag{3.10}$$

Igualmente la salida de la neurona k será:

$$y_k = v_k \qquad Para\ toda\ v_k. \tag{3.11}$$

Esta función se utiliza cuando se quiere conservar el nivel de actividad interna de la neurona en la salida. En la Fig. 3.3 b) se observa la función identidad.

3.4.3 Función Sigmoide

La función sigmoide es una función de activación muy utilizada en las redes de retropropagación porque tiene la característica de ser continua, diferenciable y monotónicamente no decreciente. Es además computacionalmente eficiente, ya que su derivada es fácil de computar.

Un ejemplo de función sigmoide es la función *logística*, que se define como:

$$\varphi(v) = \frac{1}{1 + e^{-av}}, \tag{3.12}$$

donde a es el parámetro de inclinación de la función sigmoidal. El rango de salida de la función es de 0 a +1. En la Fig. 3.3 c) se observa esta función.

Otro tipo de función sigmoidal utilizada en la construcción de redes neuronales es la función *tangente hiperbólica*, cuya salidas están en el rango de -1 a +1 y está definida como:

$$\varphi(v) = \tanh\,(\frac{v}{2}) = \frac{1 - e^{-v}}{1 + e^{-v}}. \tag{3.13}$$

3.5 Arquitecturas de Red

La manera en que las neuronas de una red neuronal están estructuradas está fuertemente ligada con el algoritmo de aprendizaje utilizado para entrenar la red. A continuación se presentan las arquitecturas de red más comunes.

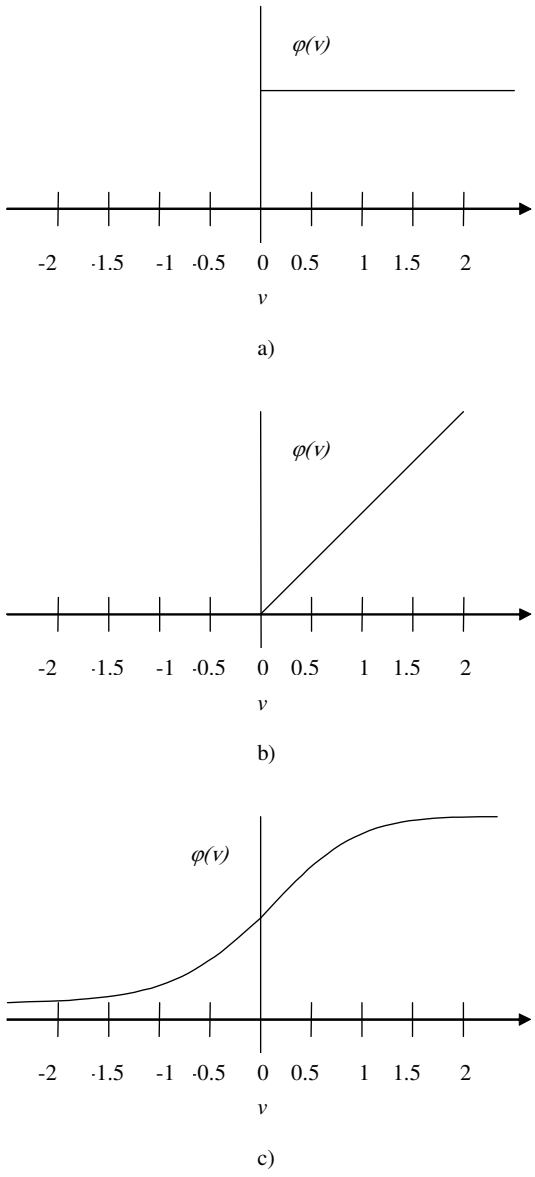

Figura 3.3: a) Función escalón. b) Función identidad. c) Función logística sigmoidal

3.5.1 Redes de Alimentación hacia Delante de Capa Sencilla

Comúnmente, en una red neuronal las neuronas están organizadas en capas. La arquitectura más sencilla de una red es la que tiene una capa de *nodos de entrada* y una sola capa de *neuronas de salida*. Se dice que es una red de alimentación hacia delante porque la computación se lleva a cabo estrictamente de la capa de entrada hacia la capa de neuronas de salida y de una sola capa porque en la capa de entrada no se realiza computación alguna, como se observa en la Fig. 3.4.

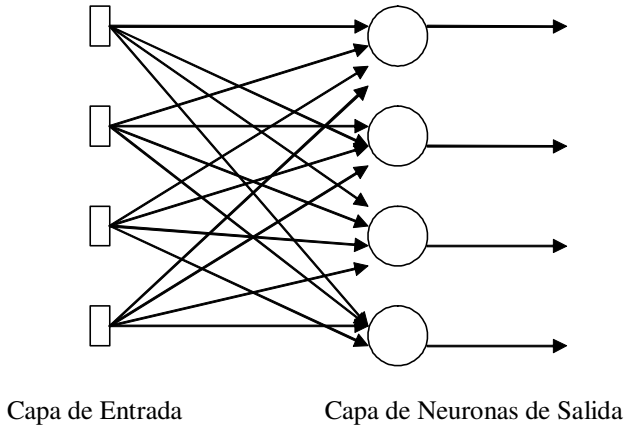

Figura 3.4: Red de alimentación hacia delante de capa sencilla

3.5.2 Redes de Alimentación hacia Delante Multicapa

En esta arquitectura, se tienen una o más capas de neuronas *ocultas*, es decir, todas aquellas capas de la red excepto la de entrada y la de salida. Con una o más capas ocultas, la red es capaz de extraer estadísticas de orden superior, la red adquiere una perspectiva global por su conjunto extra de conexiones sinápticas y la dimensión extra de interacciones entre neuronas [ChP+92]. En la Fig. 3.5 se observa una red multicapa con una capa oculta.

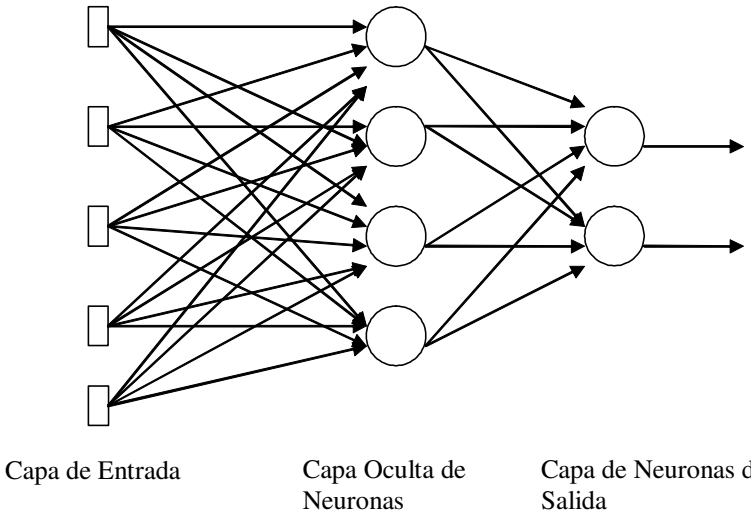

Figura 3.5: Red multicapa de alimentación hacia delante 5 - 4 – 2: 5 entradas, 4 neuronas intermedias y 2 neuronas de salida

3.5.3 Redes Recurrentes

Una red neuronal recurrente se puede distinguir de las arquitecturas de alimentación hacia delante porque tiene al menos un ciclo de *retroalimentación*. Como se observa en la Fig. 3.6, las salidas de las neuronas de una capa están conectadas a sus mismas entradas, o a las de las demás neuronas de la misma capa. Para lograr la retroalimentación, las redes recurrentes hacen uso de unidades de retardo.

3.6 El Proceso de Aprendizaje

Las redes neuronales tienen la propiedad de *aprender* de su medio y de modificar su comportamiento según éste. El aprendizaje en una red neuronal es un proceso iterativo, en el cual los pesos sinápticos son ajustados, así como los umbrales. Una definición de aprendizaje es la siguiente:

> *Aprendizaje es el proceso por el cual los parámetros libres de una red neuronal son adaptados a través de un proceso continuo de*

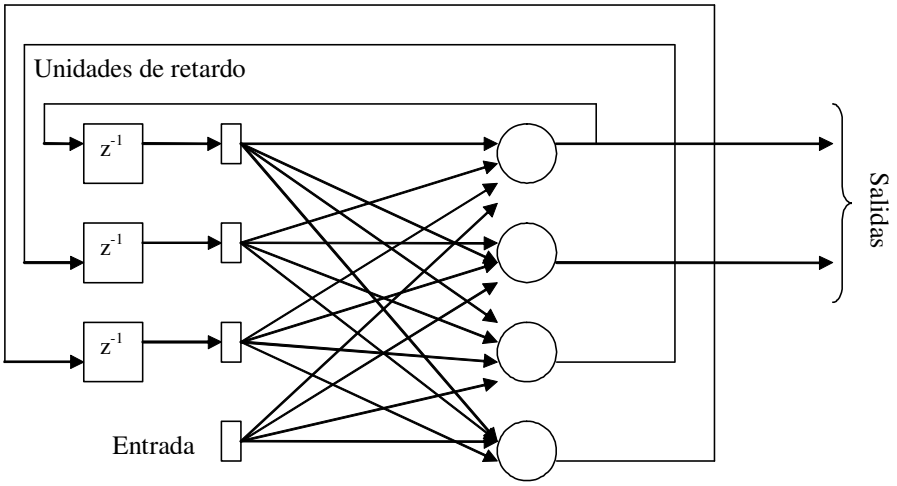

Figura 3.6: Red recurrente con neuronas ocultas

> *estimulación por el ambiente en el cual la red está inmersa. El tipo de aprendizaje está determinado por la manera en la cual se hacen los cambios a los parámetros [HaS94].*

Como se mencionó en la definición anterior, se somete a la red neuronal a interactuar con el medio en el cual se encuentra. El ambiente le proporciona a la red las entradas o estímulos y ésta va a responder con cambios en su estructura interna, específicamente, en sus pesos sinápticos. A cada entrada, la red neuronal va a responder de diferente manera debido a las modificaciones en su estructura, que la red sufre en cada estímulo.

3.6.1 Aprendizaje Supervisado

El aprendizaje supervisado o activo requiere de un *maestro externo*, el cual tiene el conocimiento del ambiente en el sentido de que conoce un conjunto de ejemplos de la forma entrada – salida, es decir, dada una entrada, conoce la salida que le corresponde. Para cada vector de entrada aplicado a la red neuronal, el maestro externo es capaz de proveerle a la red una respuesta deseada, que se comparará con la que se obtenga de la red neuronal, y el resultado de esta comparación repercutirá en un ajuste de los pesos sinápticos de la red. El proceso de comparación y ajuste será un proceso iterativo hasta

llegar a un estado de la red que se considere óptimo según el criterio de paro. Cuando el proceso de aprendizaje termina, entonces se deja a la red neuronal interactuar por ella sola con el ambiente.

El tipo de aprendizaje supervisado utilizado en las redes neuronales de retropropagación es el *aprendizaje por corrección de error*, en el cual, como medida de rendimiento de la red, se toma el valor instantáneo de la suma de errores cuadráticos, definido como:

$$\varepsilon(n) = \frac{1}{2}\sum_{k} e_k^2(n), \tag{3.14}$$

donde $\varepsilon(n)$ es el valor instantáneo de la suma, de la neurona k en la iteración n. En la ecuación 3.14, $e_k(n)$ es el error y está definido como:

$$e_k(n) = d_k(n) - y_k(n), \tag{3.15}$$

donde $d_k(n)$ es la respuesta deseada y $y_k(n)$ es la respuesta obtenida en la red. La red entonces es optimizada minimizando $\varepsilon(n)$ con respecto a los pesos sinápticos.

La función de error cuadrático promedio puede visualizarse como una superficie multidimensional del error, con los parámetros libres como coordenadas [HaS94]. Cualquier operación de la red bajo la supervisión del maestro es representado como un punto en la superficie de error y el objetivo es que el punto de operación tiene que moverse sucesivamente hacia un punto mínimo de error, el cual puede ser un mínimo local o global. En un sistema supervisado con aprendizaje por corrección de error, el sistema utiliza el *gradiente negativo*, el cual es un vector en la superficie de error que apunta en la dirección en la cual el error desciende más rápido.

El algoritmo least-mean-square (LMS) [WiB+60] y el algoritmo de retropropagación [WeP74], el cual es una generalización del primero, son ejemplos de algoritmos de aprendizaje supervisado. En la Fig. 3.7 se muestra un diagrama del aprendizaje supervisado.

Aprendizaje en Línea y Fuera de Línea

En el aprendizaje en línea o aprendizaje por patrón, se actualizan los pesos sinápticos de la red neuronal después de que un patrón o vector de entrenamiento (un vector que contiene la entrada y la salida deseada al estímulo) es introducido a la red. En el aprendizaje en línea se calcula el error - conforme a la ecuación 3.15 - después de cada patrón de aprendizaje, lo que

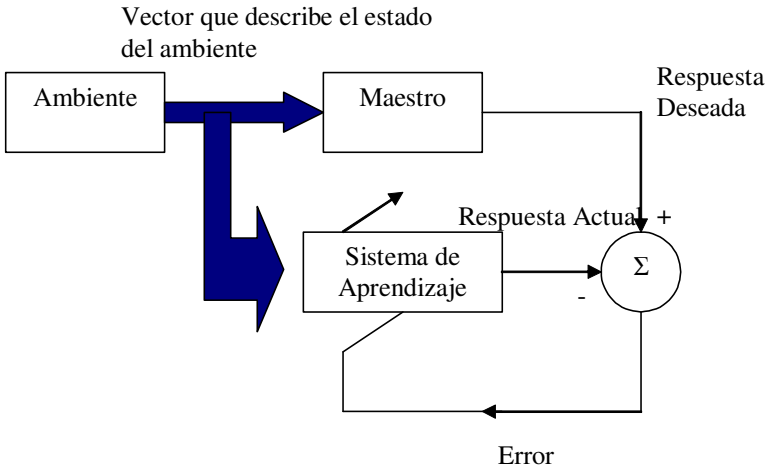

Figura 3.7: Diagrama a bloques del aprendizaje supervisado

ocasiona que, en el caso del algoritmo de retropropagación, los pesos sinápticos se actualicen muchas veces durante una *época* (el paso del conjunto completo de patrones de entrenamiento sobre la red neuronal).

En el aprendizaje fuera de línea se actualizan los pesos sinápticos después del conjunto completo de datos de entrenamiento, calculándose el error sobre el conjunto entero de datos. El aprendizaje fuera de línea es conocido también como aprendizaje por lotes.

3.6.2 Aprendizaje No Supervisado

En el aprendizaje no supervisado o auto-organizado, no hay un maestro externo que vigile el proceso de aprendizaje. Es decir, no hay ejemplos específicos de la función a aprenderse por la red neuronal. Una vez que la red ha aprendido las *regularidades* o características de los datos de entrada, ésta desarrolla la habilidad de formar representaciones internas para desarrollar características de la entrada y entonces crear nuevas clases automáticamente [BeS91]. El sistema de aprendizaje en una red no supervisada requiere de una regla de aprendizaje competitiva, la cual consiste en que las neuronas de la red compiten unas con otras por la "oportunidad" de responder a los

vectores de entrada. En la Fig. 3.8 se muestra el diagrama a bloques del aprendizaje no supervisado.

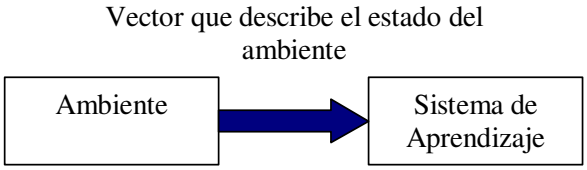

Figura 3.8: Diagrama a bloques del aprendizaje no supervisado

3.7 Perceptrón Simple

El perceptrón es la forma más simple en que una red neuronal puede clasificar todos aquellos patrones que tengan la característica de ser linealmente separables [HaS94], por ejemplo, patrones que caen en lados opuestos de un hiperplano. El perceptrón simple consiste de una sola neurona con pesos sinápticos y un umbral. El algoritmo usado para ajustar los pesos sinápticos del perceptrón fue desarrollado por Rosenblatt (1962). El perceptrón está limitado a clasificar entre sólo dos clases de patrones por estar conformado sólo por una neurona. En la Fig. 3.9 se observa el perceptron simple.

El modelo McCulloch-Pitts de una neurona es la base en la operación del perceptrón de Rosenblatt, el cual consiste de un combinador lineal seguido de una función de activación. El nodo del combinador lineal realiza una suma de las entradas multiplicadas por sus pesos sinápticos, el cual también toma en cuenta un umbral externo. El resultado del combinador lineal es aplicado a la función de activación, la cual produce una salida igual a +1 si su entrada es positiva y -1 si es negativa.

En el modelo del perceptrón, las señales de entrada a la red son denotadas por $x_1, x_2, \ldots x_p$. Los pesos sinápticos se denotan por $w_1, w_2, \ldots, w_p$. P es el número de entradas de la red. El umbral externo es θ. El combinador lineal entonces se puede definir como:

$$v = \sum_{i=1}^{p} w_i x_i - \theta. \tag{3.16}$$

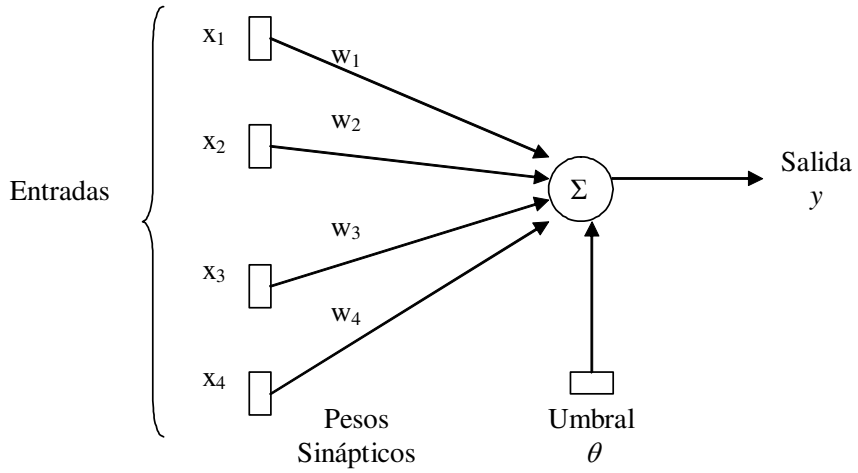

Figura 3.9: Perceptrón simple

Las entradas $x_1, x_2, \ldots x_p$ serán entonces clasificadas en dos clases, esta clasificación se hace respecto a la salida y de la función de activación, si es $+1$ pertenecerá a una clase, y pertenecerá a la segunda clase si es -1.

3.8 Perceptrón Multicapa

El perceptrón multicapa, con alimentación hacia delante, lo constituyen primeramente, *la capa de entrada*, que son unidades receptoras. Están, además, como se puede observar en la Fig. 3.10, una o más *capas ocultas*, que son las que realizan la computación interna de la red, y la *capa de salida*, que también realiza cómputo en la red. Cuando una señal entra en la red se propaga hacia delante a través de éstas capas.

Los perceptrones multicapa han logrado resolver una gran cantidad de problemas con el algoritmo de entrenamiento conocido como el algoritmo de retropropagación, el cual está basado en la regla de aprendizaje por corrección de error, la cual es una generalización a su vez del algoritmo least-mean-square (LMS).

Un perceptrón multicapa tiene tres características principales [HaS94]:

1. El modelo de cada neurona en la red tiene una no-linealidad en su salida. Esta no-linealidad es suave, es decir, es diferenciable en cualquier punto, distinto al perceptrón de Rosenblatt cuya salida era de sólo dos valores. Para satisfacer el requerimiento de no-linealidad se utiliza la función logística sigmoidal[1].

2. La red neuronal contiene una o más capas de neuronas ocultas, éstas no constituyen ni la capa de entrada ni la de salida. Las capas ocultas permiten a la red aprender tareas complejas extrayendo las características más importantes de los patrones de entrada.

3. La red tiene un alto grado de *conectividad*, que está determinado por sus conexiones sinápticas.

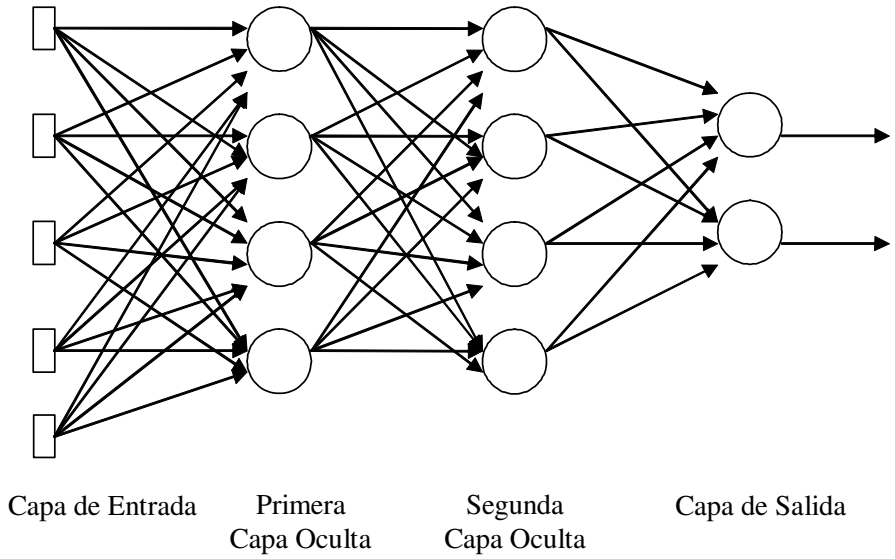

Figura 3.10: Perceptrón multicapa de dos capas ocultas

[1]Descrita en la sección 3.4.3

3.9 Algoritmo de Retropropagación

La idea de la retropropagación del error en redes neuronales fue utilizada por primera vez por Werbos en 1974 en su tesis doctoral [WeP74]. Después, Rumelhart, Hinton y Williams en 1986 redescubrieron el concepto y lo publicaron en el libro *Parallel Distributed Processing* [RuD+86]. Paralelamente un algoritmo similar era desarrollado por Parker y también por Lecun, en 1985.

El proceso de retropropagación consiste en dos recorridos a través de las diferentes capas de la red neuronal: el recorrido hacia delante y hacia atrás. En el recorrido hacia delante, el vector de entrada o patrón de entrada es presentado a los nodos de la capa de entrada y el efecto se propaga a todas las demás neuronas de la red, capa por capa. Este efecto llega hasta las neuronas de salida, donde, a la salida de éstas, constituye la respuesta de la red al patrón de entrada. En el recorrido hacia atrás, los pesos sinápticos son ajustados conforme la regla de corrección de error. El recorrido hacia atrás comienza cuando se calcula el error, el cual lo constituye la sustracción de la respuesta de la red a la respuesta deseada. Una vez calculado el error, éste es propagado hacia atrás, en contra de la dirección de las conexiones sinápticas. De aquí el término de retropropagación. Entonces, los pesos sinápticos de la red son ajustados, capa por capa, con la finalidad de que el error sea cada vez menor por cada patrón de entrada que es presentado a la red. El efecto de la retropropagación puede ser observado en la Fig. 3.11 [HaS94].

El algoritmo de retropropagación se puede describir en cinco fases, que se presentan a continuación [HaS94]:

1. **Inicialización**. El conjunto de pesos sinápticos es inicializado a valores razonables. Comúnmente son valores aleatorios pequeños entre 0 y 1 ó entre -1 y 1.

2. **Presentación de los ejemplos de entrenamiento**. A la red se le presenta una época o conjunto de ejemplos de entrenamiento. Por cada ejemplo del conjunto de entrenamiento se va a realizar la computación hacia delante y hacia atrás.

3. **Computación hacia delante**. El vector de entrenamiento $[x(n), d(n)]$ es presentado a la red, $x(n)$ es el vector de entrada aplicado a los nodos de entrada y $d(n)$ es el vector de respuesta que se desea se produzca en la salida de la red. Se calcula entonces la señal de entrada capa por

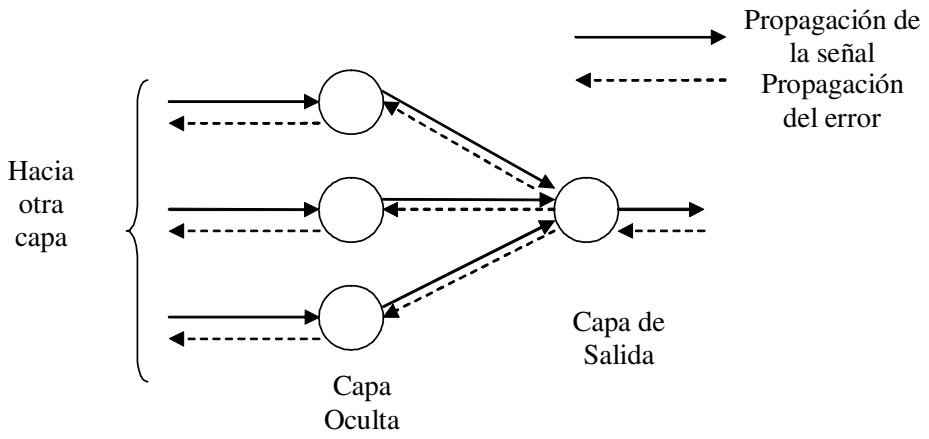

Figura 3.11: Propagación de la señal de entrada y la señal de error en el algoritmo de retropropagación

capa hacia delante, pasando por las funciones de activación para cada neurona. El nivel de actividad interna $v_j^{(l)}(n)$ para la neurona j de la capa l en la n-ésima iteración es:

$$v_j^{(l)}(n) = \sum_{i=0}^{p} w_{ji}^{(l)}(n) y_i^{(l-1)}(n), \tag{3.17}$$

$y_i^{(l-1)}(n)$ es la señal de salida de la neurona i en la capa anterior $l-1$, en la iteración n. $w_{ji}^{(l)}(n)$ es el peso sináptico que conecta a la neurona j en la capa l con la neurona i en la capa $l-1$. Para $i=0$, se tiene que $y_0^{(l-1)} = -1$ y que $w_{j0}^{(l)}(n) = \theta_j^{(l)}(n)$, donde $\theta_j^{(l)}(n)$ es el umbral aplicado a la neurona j en la capa l. La función de activación de la neurona j en la capa l es:

$$y_j^{(l)}(n) = \frac{1}{1 + \exp(-v_j^{(l)}(n))}, \tag{3.18}$$

y si la neurona j está la primera capa, entonces

$$y_j^{(0)}(n) = x_j(n), \tag{3.19}$$

donde $x_j(n)$ es el j-ésimo elemento del vector de entrada $x(n)$. Si la neurona j está en la capa de salida, entonces

$$y_j^{(l)}(n) = o_j(n), \tag{3.20}$$

aquí $o_j(n)$ es el j-ésimo elemento del vector de salida . Cuando la señal llega a la capa de salida, se calcula el error, que es

$$e_j(n) = d_j(n) - o_j(n), \tag{3.21}$$

en el cual $d_j(n)$ es el j-ésimo elemento del vector de respuestas deseadas $d(n)$. La suma instantánea de los errores cuadráticos es entonces:

$$\varepsilon(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n), \tag{3.22}$$

C es el conjunto que incluye a todas las neuronas en la capa de salida de la red.

4. **Computación hacia atrás**. El cálculo de los gradientes de la red se realiza en esta fase y se calcula capa por capa, el gradiente para una neurona se calcula como:

$$\delta_j^{(L)}(n) = e_j^{(L)} o_j(n)[1 - o_j(n)], \tag{3.23}$$

si la neurona j está en la capa de salida L. Para la neurona j en la capa oculta l se tiene:

$$\delta_j^{(l)}(n) = y_j^{(l)}(n)[1 - y_j^{(l)}(n)] \sum_{k=1}^{P} \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n), \tag{3.24}$$

donde P es el número de neuronas en la capa $l + 1$. El ajuste de los pesos sinápticos de la red en la capa l se hace de acuerdo con la regla delta:

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha[w_{ji}^{(l)}(n) - w_{ji}^{(l)}(n-1)] + \eta \delta_j^{(l)}(n) y_i^{(l-1)}(n), \tag{3.25}$$

aquí η es el coeficiente de aprendizaje y α es la constante de momentum, que pueden tomar valores en el intervalo (0, 1). La constante de momentum es la que controla el nuevo incremento en w_{ji} en relación al pasado incremento en w_{ji}.

5. **Iteraciones**. Las iteraciones se realizan presentando nuevas épocas de ejemplos de entrenamiento hasta que los parámetros libres (pesos sinápticos) se estabilizan en un punto donde el error cuadrático promedio ε_{av} calculado sobre el conjunto entero de entrenamiento alcanza un valor pequeño aceptable. ε_{av} está definido como:

$$\varepsilon_{av} = \frac{1}{N} \sum_{n=1}^{N} \varepsilon(n), \tag{3.26}$$

$\varepsilon(n)$ es la suma instantánea de errores cuadráticos - definida en la ecuación 3.22 - y N es el número de ejemplos del conjunto de entrenamiento. En la Fig. 3.12 se muestra una ilustración de las variables involucradas en el algoritmo de retropropagación. Se hace notar en esta figura que

$$\varphi(n) = y(n), \tag{3.27}$$

en donde *y(n)* es igual a la función de activación sigmoidal.

En este trabajo se decidió utilizar la arquitectura del perceptrón multicapa, con el algoritmo de retropropagación como método de entrenamiento, para mapear las intensidades de la señal RF a coordenadas en 2D, por sus características de aprendizaje de tareas complejas y la no-linealidad en su salida. El capítulo cuatro presenta la implementación de la red de retropropagación de acuerdo al problema de la ubicación de usuarios móviles.

3.10 Aceleración de Convergencia del Algoritmo de Retropropagación

En su forma estándar, el algoritmo de retropropagación es un algoritmo de máximo descenso, que busca por un mínimo en una superficie de error E, usando pasos fijos pequeños, especificados por el coeficiente de aprendizaje η:

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}, \tag{3.28}$$

donde w_{ij} pertenece al conjunto de pesos sinápticos de la red neuronal.

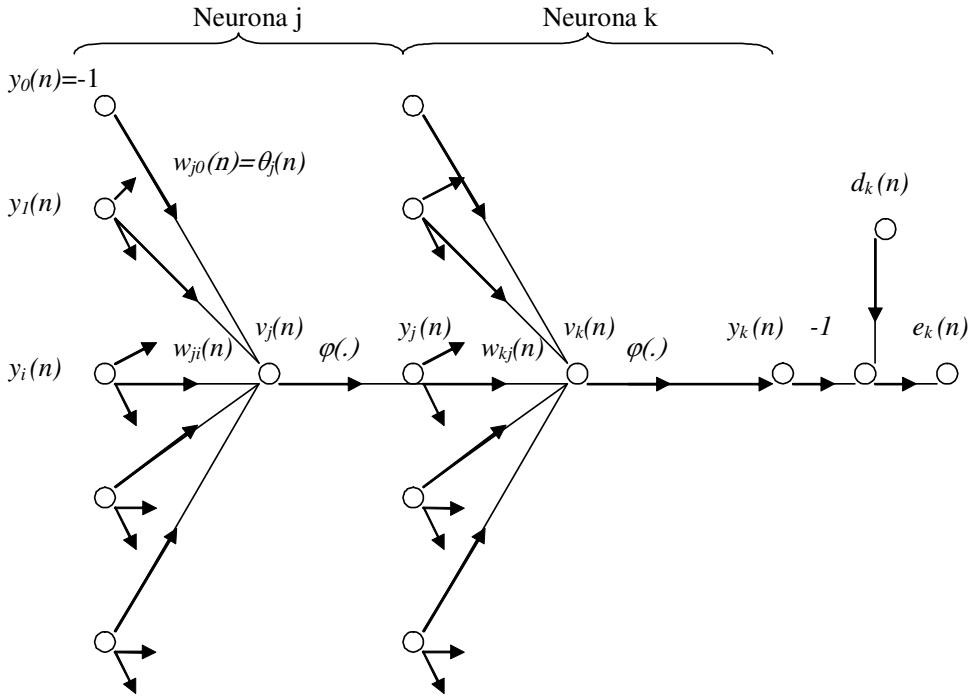

Figura 3.12: Gráfica de flujo de la señal de la neurona k conectada a la neurona oculta j

Sin embargo, en esta forma estándar, el algoritmo de retropropagación tiene varios inconvenientes [SoB+92]:

1. **Baja Velocidad de Convergencia**. Cuando la superficie de error es no lineal y sólo la información del gradiente local está disponible, el coeficiente de aprendizaje η debe mantenerse bastante pequeño para asegurar una convergencia estable. Esto, por supuesto, aumenta el tiempo del proceso de aprendizaje considerablemente.

2. **Mínimos Locales**. La superficie de error sobre la cual está el gradiente descendente podría tener mínimos locales [HaD90], los cuales siempre atraparán a un algoritmo descendente estricto, como es el caso del algoritmo de retropropagación estándar.

3. **Oscilaciones**. Cuando la no-circularidad del contorno de la superficie de error se vuelve extrema, su forma es como la de una "hondonada" bastante estrecha, inclinándose suavemente hacia el mínimo real y fuertemente en una dirección casi ortogonal. Bajo tales circunstancias, un algoritmo de máximo descenso con un coeficiente de aprendizaje fijo puede entrar en una situación oscilatoria, saltando de un lado a otro de la hondonada en vez de ir progresivamente hacia el mínimo.

4. **El Tamaño de Paso**. En la ecuación 3.28, el tamaño de paso (Δw_{ij}) es directamente proporcional a la magnitud del gradiente ($\frac{\partial E}{\partial w_{ij}}$). Entonces, para un coeficiente de aprendizaje fijo, el paso es largo en las partes donde la pendiente de la superficie es grande y es corto donde la pendiente es pequeña. Estrictamente, esta política es apropiada sólo en superficies cuadráticas. En superficies con altiplanicies y declives que cambian rápida y arbitrariamente, el coeficiente de aprendizaje fijo puede llevar a un bajo rendimiento en el aprendizaje de la red.

En esta sección se detallan ocho algoritmos que de distintas maneras aceleran la convergencia del algoritmo de retropropagación. El primer algoritmo que se presenta es el algoritmo de retropropagación elástico, que no toma en cuenta la magnitud del gradiente, sino el signo de éste; en seguida se presenta a los algoritmos del gradiente conjugado, quienes hacen uso de un vector dirección para evitar las oscilaciones en la búsqueda del mínimo en la superficie de error; por último, se detallan los algoritmos Quasi-Newton, quienes hacen uso de la información de la derivada de segundo orden de la

función de evaluación de costo (el error cuadrático promedio) para mejorar el rendimiento del error de entrenamiento.

3.10.1 Algoritmo de Retropropagación Elástico

Las redes multicapa utilizan frecuentemente funciones de activación sigmoidales, que lo que hacen es normalizar una entrada con un rango infinito de valores a una salida con un rango pequeño. Las funciones sigmoidales tienen la característica de que su pendiente debe acercarse a cero cuando la entrada es muy grande, lo que significa un problema cuando el gradiente tiene una magnitud pequeña, porque causará cambios muy pequeños en los parámetros libres de la red, aún cuando éstos están aún lejos de sus valores óptimos.

El algoritmo de retropropagación elástico no toma en cuenta la magnitud del gradiente, sólo el signo de éste, el cual le determinará la dirección de la actualización del peso sináptico. El cambio en el peso sináptico estará determinado por valores diferentes al gradiente, éstos serán dos valores, uno de incremento y otro de decremento: el primero actualizará los pesos sinápticos cuando el signo del gradiente es el mismo para dos iteraciones sucesivas y el segundo cuando el signo del gradiente cambia. Si el gradiente es cero, entonces el peso sináptico continúa siendo el mismo [RiM+93].

3.10.2 Método del Gradiente Conjugado

El método del gradiente conjugado utiliza el vector dirección, que es una combinación lineal de vectores dirección previos y el vector gradiente negativo actual, a fin de reducir el comportamiento oscilatorio y reforzar el ajuste de los pesos sinápticos en concordancia con el camino de vectores dirección anteriores.

Sea $p(n)$ el vector dirección en la iteración n del algoritmo. Entonces el vector de pesos sinápticos es actualizado conforme a la regla delta:

$$w(n+1) = w(n) + \eta(n)p(n), \tag{3.29}$$

donde $\eta(n)$ es el coeficiente de aprendizaje en la iteración n. El vector inicial de la dirección es igual al negativo del vector gradiente en el punto $n = 0$, se tiene entonces que

$$p(0) = -g(0), \tag{3.30}$$

los sucesivos vectores dirección son entonces calculados como una combinación lineal entre el actual vector gradiente y el vector dirección anterior:

$$p(n+1) = -g(n+1) + \beta(n)p(n), \tag{3.31}$$

donde $\beta(n)$ es un parámetro que varía con el tiempo. Las variaciones al algoritmo del gradiente conjugado se hacen precisamente en la manera en que $\beta(n)$ es calculado a partir de $g(n)$ y $g(n+1)$: la fórmula *Fletcher-Reeves* calcula la dirección $\beta(n)$ de la siguiente manera [FlR+64]:

$$\beta(n) = \frac{g^T(n+1)g(n+1)}{g^T(n)g(n)}. \tag{3.32}$$

Por su parte, la fórmula *Polak - Ribière* calcula de una manera alternativa $\beta(n)$:

$$\beta(n) = \frac{g^T(n+1)[g(n+1) - g(n)]}{g^T(n)g(n)}. \tag{3.33}$$

En [KrA+89] y en [JoE+91] se demuestra que el aprendizaje por retropropagación basado en el método del gradiente conjugado requiere menos épocas para converger que el algoritmo estándar, aunque es computacionalmente más complejo.

La Búsqueda lineal

El cálculo del coeficiente de aprendizaje $\eta(n)$ en la fórmula 3.29 realiza una búsqueda lineal, cuyo propósito es encontrar el valor de η para el cual la función de costo (el error cuadrático promedio) $\varepsilon_{av}(w(n)+\eta p(n))$ es mínima. En esta búsqueda, se evalúan varios valores de $w(n)$ y $p(n)$.

$\eta(n)$ está definido entonces como:

$$\eta(n) = \arg\min \left\{ \varepsilon_{av}(w(n) + \eta(n)p(n)) \right\}. \tag{3.34}$$

Los algoritmos típicos de búsqueda lineal prueban una serie de valores candidatos para $\eta(n)$, deteniendo la búsqueda cuando algún valor para $\eta(n)$ cumple con algunas condiciones. La búsqueda lineal se hace en dos etapas: Una etapa de agrupamiento, que encuentra un intervalo $[a, b]$ conteniendo los valores aceptables para $\eta(n)$, y una etapa de bisección o interpolación, que lo que hace es computar un valor para $\eta(n)$ dentro del intervalo [NoJ+99].

Los algoritmos de búsqueda lineal pueden ser clasificados de acuerdo al tipo de información que utilizan. Están aquellos que utilizan sólo la información de los valores de la función y los que utilizan la información del gradiente para determinar si un buen coeficiente de aprendizaje $\eta(n)$ ha sido encontrado [NoJ+99].

La Búsqueda Lineal Charalambous. El método de búsqueda lineal Charalambous fue diseñado para usarse en combinación con un algoritmo del gradiente conjugado para el entrenamiento de redes neuronales. Es un algoritmo híbrido, ya que utiliza tanto interpolacion cúbica como bisección. Este método toma en cuenta en su algoritmo la Condición Fuerte de Wolfe para asegurarse de que se encuentre un coeficiente de aprendizaje adecuado, además, implementa una estrategia que toma en cuenta el último cambio en la función de costo, el gradiente de la función y el vector dirección para calcular el primer paso tentativo inicial. Detalles del algoritmo se pueden encontrar en [Cha92].

Reinicio Powell-Beale

En los algoritmos del gradiente conjugado anteriormente explicados, la dirección de búsqueda es periódicamente reiniciada al negativo del gradiente (ecuación 3.30). El momento estándar de hacer este reinicio es cuando el número de iteraciones es igual al número de parámetros en la red (los pesos sinápticos y bias). Sin embargo, en la fórmula Powell-Beale [PoM77] el reinicio se efectúa si existe una ortogonalidad muy pequeña entre el gradiente actual y el previo. Dicha ortogonalidad se prueba con la siguiente desigualdad:

$$\left|g(n-1)^T g(n)\right| \geq 0.2\left|g(n)\right|^2, \tag{3.35}$$

si esta condición es satisfecha, la dirección de búsqueda es reiniciada al negativo del gradiente.

Algoritmo del Gradiente Conjugado Escalado

El algoritmo del gradiente conjugado escalado es una variación del algoritmo estándar del gradiente conjugado, en el que el coeficiente de aprendizaje es encontrado a través de una búsqueda lineal que involucra el cálculo del error y la primera derivada. En el algoritmo del gradiente conjugado escalado el

coeficiente de aprendizaje es estimado por un mecanismo de escalado, sin tener que hacer la búsqueda lineal [MoMF93]. El coeficiente de aprendizaje está dado por:

$$\eta(n) = \frac{-p(n)^T g(n)}{p(n)^T s(n) + \lambda(n) \left|p(n)\right|^2}, \tag{3.36}$$

donde $p(n)$ es el vector dirección, definido en la ecuación 3.29, y $s(n)$ es

$$s(n) = g^2(n)p(n). \tag{3.37}$$

Sin embargo $s(n)$ puede ser aproximado como una diferencia de primeras derivadas en la siguiente ecuación:

$$s(n) = \frac{\varepsilon'(w(n) + \sigma(n)p(n)) - \varepsilon'(w(n))}{\sigma(n)} \qquad , 0 < \sigma(n) < 1 \tag{3.38}$$

donde $\varepsilon(w(n))$ es la función de suma instantánea de los errores cuadráticos, definida en la ecuación 3.22.

En la ecuación 3.36, $\lambda(n)$ es el parámetro de escalado cuya función es similar al parámetro de escalado de los métodos Levenberg-Marquardt, $\lambda(n)$ en cada iteración es disminuido o incrementado de acuerdo a que tan buena es la aproximación de segundo orden del error real. La fórmula del ajuste al vector de pesos sinápticos está dada entonces como:

$$\Delta w(n) = \eta(n)p(n). \tag{3.39}$$

3.10.3 El Método de Newton

El método de Newton [HaS94] utiliza la ecuación:

$$w(n+1) = w(n) - H^{-1}g(n), \tag{3.40}$$

para calcular la actualización de w. Donde g es el vector gradiente que se define como:

$$g = \frac{\partial \varepsilon_{av}(w)}{\partial w}, \tag{3.41}$$

y H es la matriz Hessiana:

$$H = \frac{\partial^2 \varepsilon_{av}(w)}{\partial w^2}, \tag{3.42}$$

ecuación, que como se observa, realiza la segunda derivada del error promedio de los pesos sinápticos y bias. La aplicación del método de Newton para el entrenamiento de las redes neuronales requiere una cantidad grande de cómputo debido al cálculo de la matriz Hessiana y su inversa.

3.10.4 Algoritmos Quasi- Newton

Los algoritmos Quasi-Newton están basados en el método de Newton, sin embargo, éstos no requieren del cálculo de la segunda derivada de ε_{av}, ya que ellos actualizan un aproximado de la matriz Hessiana, o su inversa, en cada iteración del algoritmo (que usualmente es $H_0 = 1$) [DeJ+83]. El objetivo de los algoritmos Quasi-Newton es encontrar un $H^u(n)$ que actualice a la matriz Hessiana:

$$H(n+1) = H(n) + H^u(n). \tag{3.43}$$

Existen diversos métodos para calcular la aproximación de la matriz Hessiana, que se presentan en las subsecciones siguientes.

Actualización Broyden, Fletcher, Goldfarb y Shanno

La fórmula Broyden, Fletcher, Goldfarb y Shanno (BFGS) es uno de los algoritmos Quasi-Newton más utilizados. Ésta encuentra la actualización de la matriz Hessiana en base a la siguiente fórmula [DeJ+83]:

$$B(n+1) = B(n) + R - S, \tag{3.44}$$

donde R es

$$(\frac{1 + q(n)^T B(n) q(n)}{q(n)^T p(n)}) \frac{p(n) p(n)^T}{p(n)^T q(n)},$$

y S es

$$\frac{p(n) q(n)^T B(n) + B(n) q(n) p(n)^T}{q(n)^T p(n)}.$$

En esta fórmula, $B(n+1) = H^{-1}(n+1)$, $q(n) = g(n+1) - g(n)$, y $p(n) = w(n+1) - w(n)$.

Algoritmo de la Secante en un Paso

Mientras que el algoritmo BFGS almacena la aproximación de la matriz Hessiana (o su inversa), el método de la Secante en un Paso (OSS) requiere sólo de los gradientes [BaR92]. La nueva dirección de búsqueda $p(n+1)$ se obtiene por:

$$p(n+1) = -g(n+1) + A(n)s(n) + B(n)y(n), \tag{3.45}$$

donde $s(n)$ es el último paso, es decir, s es $\Delta w(n)$, que en el algoritmo de Newton es igual a $-H^{-1}g(n)$, $g(n+1)$ es el gradiente actual y $y(n)$ es la diferencia de gradientes: $y(n) = g(n+1) - g(n)$.

Los escalares $A(n)$ y $B(n)$ son definidos como:

$$A(n) = -(1 + \frac{y(n)^T y(n)}{s(n)^T y(n)})\frac{s(n)^T g(n+1)}{s(n)^T y(n)} + \frac{y(n)^T g(n+1)}{s(n)^T y(n)};$$

$$B(n) = \frac{s(n)^T g(n+1)}{s(n)^T y(n)}.$$

Búsqueda lineal de *backtracking*. Un tipo de búsqueda lineal que los algoritmos Quasi-Newton OSS y BFGS utilizan es el del tipo *backtracking* [DeJ+83]. En este tipo de búsqueda, en el primer paso se utiliza el valor de la función de costo en el punto actual y un valor de $\eta = 1$. También se utiliza el valor de la derivada de la función para obtener una aproximación cuadrática del rendimiento de la función a lo largo de la dirección de búsqueda. El mínimo de la aproximación cuadrática se vuelve un punto óptimo tentativo y se evalúa el rendimiento de la función de costo en ese punto. Si el rendimiento no se reduce lo suficiente se realiza una interpolación cúbica y el mínimo de la interpolación se vuelve el nuevo punto óptimo tentativo. Este proceso se continúa hasta que se obtiene una reducción suficiente en el rendimiento de la función de costo ε_{av}.

Algoritmo Levenberg - Marquardt

Cuando la función de error tiene la forma de suma de cuadrados, que es el caso con el error cuadrático promedio (ecuación 3.26), la matriz Hessiana puede ser aproximada como:

$$H = J^T J, \tag{3.46}$$

y el gradiente puede ser expresado como:

$$g = J^T e, \tag{3.47}$$

donde J es la matriz Jacobiana, que contiene las primeras derivadas de los errores de la red con respecto a los pesos y bias, e es el vector de errores de la red. La matriz Jacobiana puede ser computada con la técnica de retropropagación y además es mucho menos compleja que la matriz Hessiana [HaM+94].

Entonces, el algoritmo Levenberg-Marquardt utiliza la aproximación de la matriz Hessiana para actualizar los pesos sinápticos:

$$w(n+1) = w(n) - [J^T J + \mu I]^{-1} J^T e, \tag{3.48}$$

donde μ es un escalar que se decrementa después de cada paso exitoso y se incrementa cuando un paso tentativo podría incrementar la función de error. I es la matriz identidad.

Capítulo 4

Desarrollo del Proyecto

En el presente capítulo se describe la manera en que los datos fueron obtenidos y dispuestos para el entrenamiento de la red neuronal. Se especifica además el hardware y el software utilizado durante el desarrollo del proyecto. Por último, se presenta la aplicación consciente del contexto que se construyó y que hace uso de la información de la ubicación, proporcionada por la red neuronal.

4.1 Especificaciones de Hardware y Software

A continuación se mencionan los requerimientos de hardware y software en el desarrollo del proyecto.

4.1.1 Recolección de Muestras

El dispositivo móvil inalámbrico que se utilizó para la recolección de muestras de intensidad de la señal provenientes de los distintos puntos de acceso fue una computadora portátil Intel Pentium II a 331 MHz con 192 Mb de memoria RAM. La tarjeta inalámbrica utilizada en la computadora portátil fue una modelo *Orinoco Silver*, con una velocidad de transmisión de datos de 11Mbps y compatible con el estándar IEEE 802.11b. La plataforma de cómputo utilizada fue el sistema operativo *Windows XP* v2002. Para la recolección de las muestras de intensidad de la señal se utilizó el software *Client Manager* de Orinoco, versión 2.9, el cual tiene comunicación directa con la tarjeta inalámbrica. Este software permitió obtener información especializada como

lo es la intensidad de la señal de los distintos puntos de acceso que recibe la tarjeta inalámbrica en todo momento en el que se encuentra dentro de la red LAN inalámbrica, así como el nivel de ruido y la intensidad de la relación señal a ruido (SNR). La razón para utilizar el software Client Manager es que es uno de los pocos que corre bajo el sistema operativo Windows, además de que tiene compatibilidad con la tarjeta inalámbrica Orinoco Silver, que fue la que se utilizó durante el proyecto.

4.1.2 Instalación de la Red Inalámbrica

La red inalámbrica que funciona dentro del edificio del Instituto de Electrónica y Computación de la Universidad Tecnológica de la Mixteca (cuya infraestructura se detalla en la sección siguiente) está constituida por cinco puntos de acceso. Tres de ellos son modelo *AP-200* de Orinoco (señalados con cuadros y con los números 2, 4 y 5 en la Fig. 4.1) y dos puntos de acceso son modelo *Apple Airport*, uno tipo *Dual Ethernet* y otro del tipo *Extreme* (señalados con los números 1 y 3 en la Fig. 4.1).

Los cinco puntos de acceso funcionan bajo el protocolo estándar IEEE 802.11b. La red inalámbrica opera en la banda de 2.4 GHz, la cual es de licencia libre ISM (Industrial, Científica y Médica) y que tiene un rango de alcance de 160 m, 50 m y 25 m para espacios abiertos, semi-abiertos y cerrados, respectivamente. La banda de 2.4 GHz está dividida en 13 canales de los cuales 5 son utilizados en esta red inalámbrica: los canales 1, 3, 7, 9 y 11, con el fin de minimizar la interferencia entre cada uno de los puntos de acceso.

4.1.3 Simulación de la Red Neuronal

El hardware utilizado para el entrenamiento y prueba de la red neuronal es una computadora Pentium IV a 2.4 GHz y con 256 Mb de memoria RAM. El software utilizado es el *Toolbox* de Redes Neuronales de *Matlab v.6.5*. Las razones para utilizar dicho software es que el Toolbox incluye:

- Rutinas predefinidas de los diversos algoritmos de entrenamiento de la red neuronal.
- Un buen manejo de los vectores de entrada, al poder presentarlos en forma matricial ante la red neuronal.

- Herramientas de graficación que permiten, entre otras cosas, la visualización gráfica del proceso de aprendizaje de la red durante su entrenamiento.
- Un conjunto de rutinas que permiten crear la arquitectura y topología de red neuronal más adecuada al problema.

4.1.4 Construcción de la Aplicación Consciente del Contexto

La aplicación que muestra sobre un mapa la posición de la persona móvil dentro del Instituto de Electrónica y Computación de la UTM, fue construida en una computadora Pentium IV de 2.4 GHz y 256 MB de memoria RAM. El software utilizado fue *Microsoft Visual Basic v.6.0*.

4.2 Lugar de Prueba

En el desarrollo del proyecto se utilizó el segundo piso del Instituto de Electrónica y Computación, de la Universidad Tecnológica de la Mixteca, como el lugar donde se estimaría la posición del usuario móvil. Las dimensiones del piso son de 40 x 20 metros, el cual incluye a 28 cuartos. En la Fig. 4.1 se muestra el plano del segundo piso del Instituto[1].

Dentro del edificio se encuentra funcionando una red inalámbrica de área local (WLAN) basada en el estándar 802.11b. Los cinco puntos de acceso que conforman la WLAN están señalados y numerados en cuadros dentro de la Fig. 4.1.

4.3 Registro de la Intensidad de la Señal

En la Fig. 4.2 se muestran sobre el plano del segundo piso del Instituto de Electrónica y Computación las marcas de los lugares donde se registró la intensidad de la señal proveniente de los puntos de acceso hacia la tarjeta inalámbrica.

[1]Plano proporcionado por el arquitecto José Lázaro, encargado de la construcción del edificio. El arquitecto pertenece al Instituto de Diseño.

En cada punto seleccionado del edificio se tomaron muestras con la posición de la computadora portátil hacia cada uno de los cuatro puntos cardinales. En cada dirección u orientación se registraron 7 muestras y se calculó el promedio de éstas. En general, se tuvieron entonces 4 mediciones-promedio por cada punto en el edificio, una por cada orientación. Se hizo el registro de las mediciones en 154 puntos sobre la planta del edificio, obteniéndose entonces un total de 616 mediciones-promedio para todo el lugar.

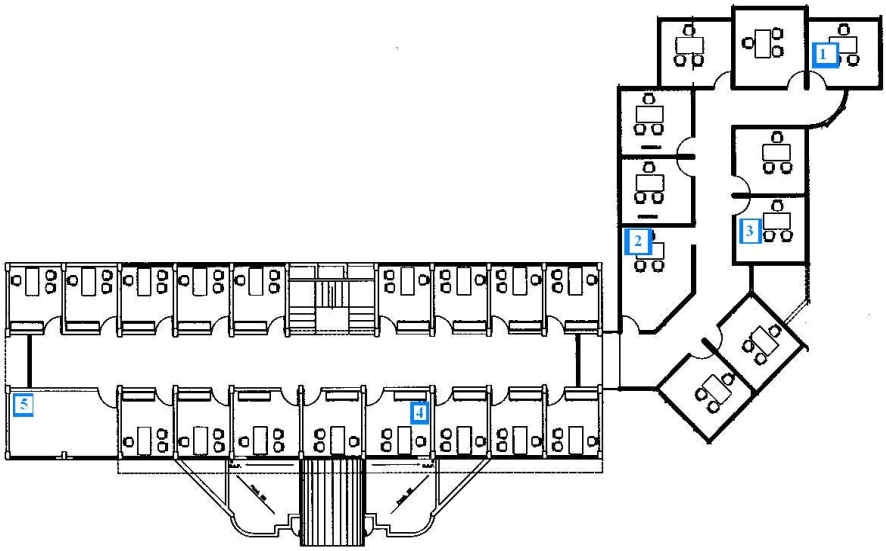

Figura 4.1: Plano del segundo piso del Instituto de Electrónica y Computación de la UTM

Para posicionar adecuadamente el lugar en donde se hicieron las mediciones se tomó como referencia un plano cartesiano, cuyo origen está en la esquina inferior izquierda de la planta, como se muestra en la Fig. 4.3[2].

El formato de cada muestra-promedio que se registró es del tipo *(IS1, IS2, IS3, IS4, IS5, x, y)*, donde *ISn* es la intensidad de la señal del punto de acceso n en el punto geográfico con coordenadas *(x, y)*, en metros [3]. Las

[2] El plano se encuentra también en el archivo de AutoCAD *planoIEC.dwg*.

[3] En el archivo de Excel, *Mediciones.xls*, se encuentra el registro entero de las mediciones recolectadas de la intensidad de la señal así como las mediciones de nivel de ruido y la relación señal a ruido.

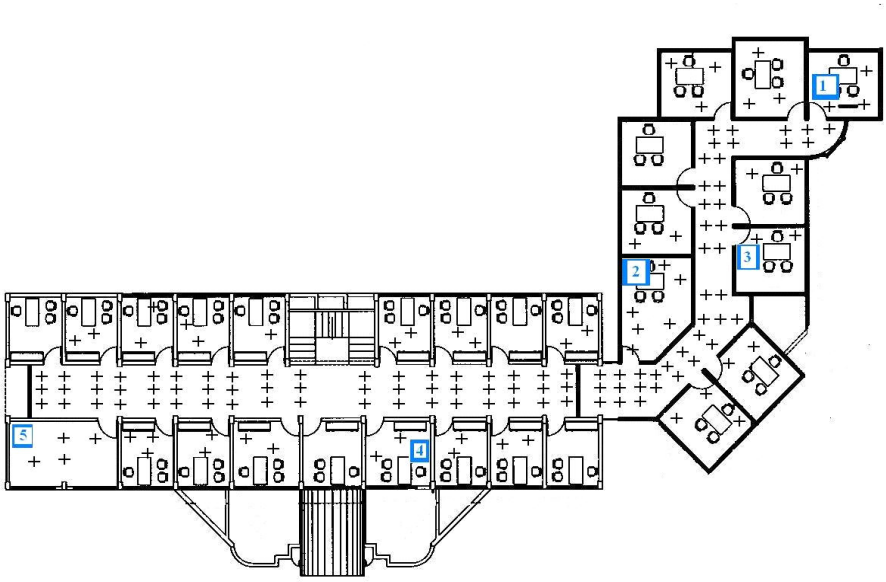

Figura 4.2: Plano del segundo piso del Instituto de Electrónica y Computación de la UTM. En él se marcan con una cruz los distintos puntos geográficos en los que se hizo el registro de la intensidad de la señal

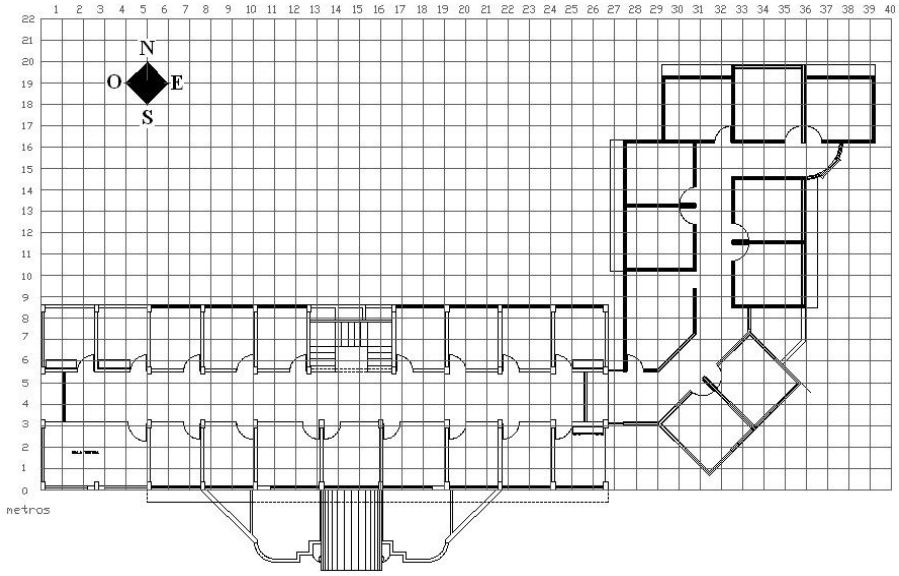

Figura 4.3: Plano del Instituto de Electrónica y Computación de la UTM con Sistema de Coordenadas Cartesianas

unidades de la intensidad de la señal son en dBm, el cual es un valor de potencia en decibeles en relación a un milivatio. También se registraron los niveles de ruido y de relación señal a ruido: *(R1, R2, R3, R4, R5, x, y)* y *(SNR1, SNR2, SNR3, SNR4, SNR5, x, y)* respectivamente, con el fin de manejarlos también como patrones de entrada a la red neuronal.

4.4 Implementación de la Red Neuronal con el Algoritmo de Retropropagación

Los vectores de entrada que le fueron presentados a la red neuronal son de tres tipos, de acuerdo al tipo de dato al que pertenecen: intensidad de la señal, relación señal a ruido y la combinación intensidad de la señal con relación señal a ruido.

1. **Intensidad de la señal**. En el caso en que los vectores de entrada se refieren a la intensidad de la señal, la capa de entrada de la red neuronal consiste de 5 neuronas, en donde cada nodo de entrada recibe

el valor de la intensidad de la señal proveniente de uno de los cinco puntos de acceso. Entonces, el vector de entrada está conformado de los siguientes valores, por ejemplo:

$$[-52.286, -83.714, -80.571, -82, -102].$$

El valor específico de -102 dBm era introducido a algún vector de entrada cuando no se recibía señal de un punto de acceso en el lugar en el que se efectuaba la medición, ya que las intensidades más pequeñas que se alcanzaban a registrar eran del rango de -90 a -100 dBm.

Las unidades de la intensidad de la señal recibida están en una relación de decibel-miliwatt, por lo que el signo menos está presente en todos los valores debido a que la potencia de la señal es regularmente menor a 1 miliwatt.

2. **Relación señal a ruido**. El vector del nivel de ruido está conformado por valores que están también en dBm, por ejemplo:

$$[-93.714, -91, -91.429, -96.143, -102].$$

De la misma manera que en el registro de la intensidad de la señal, el valor de -102 dBm era introducido a aquél vector de entrada correspondiente al lugar en el que no se registraba señal alguna por parte de algún punto de acceso.

De la intensidad de la señal y el nivel de ruido, se deriva la relación señal a ruido como la resta aritmética de la intensidad de la señal y el nivel de ruido. Siguiendo los ejemplos anteriores, el vector resultante de la relación señal a ruido es:

$$[41.429, 7.286, 10.857, 14.143, 0].$$

Al utilizar la relación señal a ruido como vector de entrada, la topología de la red neuronal consiste de 5 nodos en su capa de entrada, al igual que cuando se utiliza la intensidad de la señal como vector de entrada.

3. **Intensidad de la señal y Relación señal a ruido**. En este caso, la capa de entrada consiste de 10 nodos, los cuales reciben un vector conformado por diez elementos, cinco correspondiendo a la intensidad de la

señal y cinco a la relación señal a ruido. Siguiendo con los ejemplos de mediciones de los tipos anteriores, el vector de entrada correspondiente es:

$$[-52.286, -83.714, -80.571, -82, -102, 41.429, 7.286, 10.857, 14.143, 0].$$

La capa de salida de la red para los tres casos mencionados anteriormente está conformada por dos nodos, los cuales indican la posición del usuario móvil en coordenadas (x, y) cuyas unidades están en metros; un ejemplo es:

$$[1.7, 4.85].$$

Un esquema general de la topología de red utilizada se muestra en la Fig. 4.4. El entrenamiento de la red neuronal se hizo por lotes. Las entradas tanto como las salidas fueron normalizadas. La función de activación que se utilizó en los nodos de la capa intermedia de la red fue la logística sigmoidal, en los nodos de la capa de salida se utilizó la función identidad.

De un conjunto de 616 patrones, 432 se utilizaron para el entrenamiento de la red y 184 para la fase de prueba.

4.5 Una Aplicación Consciente del Contexto Básica

Se implantó una aplicación que muestra, por medio de un mapa, la posición de una persona que se traslada con un dispositivo móvil a través de la segunda planta del Instituto de Electrónica y Computación, en la Universidad Tecnológica de la Mixteca. La pantalla de la aplicación se muestra en la Fig. 4.5.

La aplicación se basa en dos módulos, que le proporcionan los datos de la posición del usuario: el módulo de la red neuronal previamente entrenada, y el módulo que obtiene de la tarjeta inalámbrica las intensidades de las señales de los puntos de acceso. En la Fig. 4.6 se muestra el diagrama de secuencia de la interacción entre el usuario y la aplicación.

La red neuronal que utiliza la aplicación es la que obtuvo los mejores resultados en las pruebas realizadas en el presente trabajo de tesis. En el manual de usuario del Apéndice B se detalla la forma de poner en funcionamiento la aplicación.

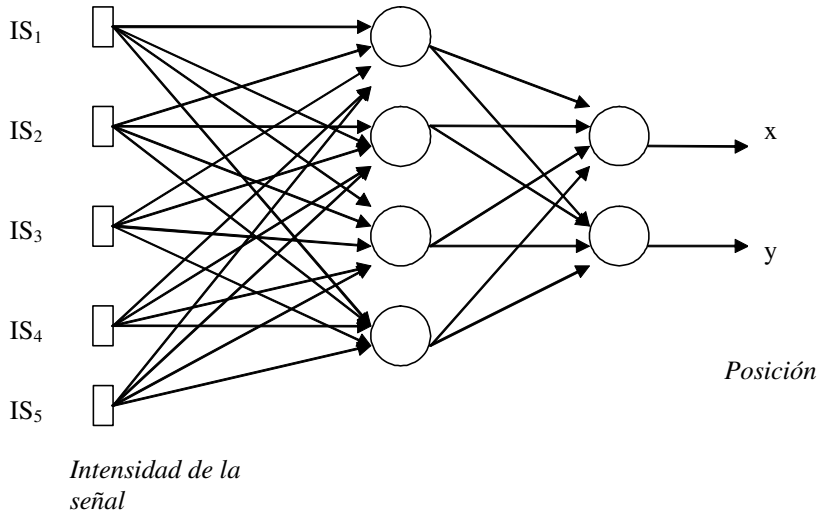

Figura 4.4: Topología de red con cinco nodos en la capa de entrada, cuatro en capa intermedia y dos en la capa de salida

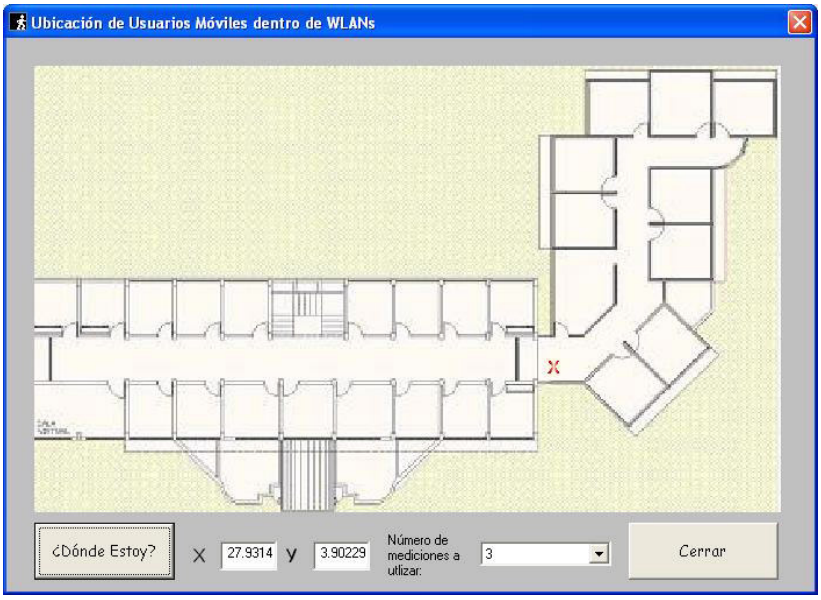

Figura 4.5: Ventana de la aplicación

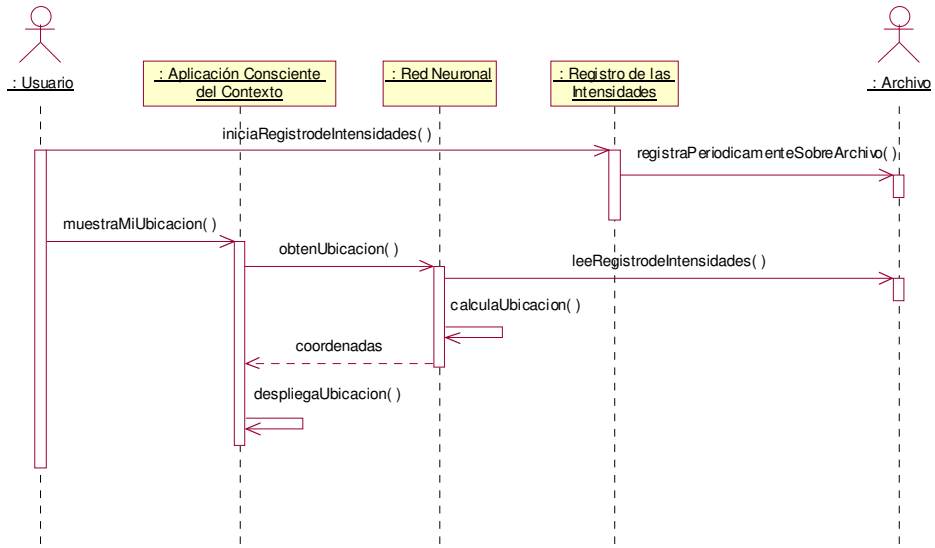

Figura 4.6: Diagrama de secuencia de la interacción usuario-aplicación

Capítulo 5

Resultados

En este capítulo se presentan los resultados obtenidos al utilizar, en una red neuronal y con los datos asociados al problema, los distintos algoritmos de entrenamiento de primer y segundo orden detallados en el capítulo tres. Se presentan además los resultados de la comparación de la técnica de redes neuronales con la técnica k-nearest.

Algoritmos de entrenamiento utilizados

Para el entrenamiento de la red neuronal se utilizaron ocho algoritmos de entrenamiento distintos, con el propósito de compararlos y observar cuáles de ellos tenían un mejor desempeño. Dichos algoritmos son:

1. Retropropagación Elástico (TrainRp).
2. Gradiente Conjugado con Actualización Fletcher-Reeves (TrainCgf).
3. Gradiente Conjugado con Actualización Polak-Ribiére (TrainCgp).
4. Gradiente Conjugado con Reinicio Powell-Beale (TrainCgb).
5. Gradiente Conjugado Escalado (TrainScg).
6. Quasi-Newton con Actualización Broyden, Fletcher, Goldfarb y Shanno (TrainBfg).
7. Quasi-Newton Secante de un Paso (TrainOss).
8. Quasi-Newton Levenberg – Marquardt (TrainLm).

En la lista anterior se proponen los nombres cortos de los algoritmos entre paréntesis para referencias posteriores.

Topologías de red utilizadas

Para el entrenamiento y prueba, se utilizó una topología de red de una sola capa intermedia (en el perceptrón multicapa, una sola capa intermedia con un gran número de unidades es suficiente para obtener una buena aproximación [BiC95] [RiB96]) y se varió el número de neuronas en esta capa en 4, 6, 8 y 16 neuronas (el método de Regularización Bayesiana [MaD92] [FoF+97] aplicado al problema mostró que utilizar un número mayor a 16 neuronas en la capa intermedia sería innecesario), con el fin de observar el comportamiento del aprendizaje de la red. En principio se espera que a mayor número de neuronas en la capa intermedia la aproximación sea mejor pero con un mayor costo en el tiempo de aprendizaje.

Tamaño del conjunto de vectores de entrada

El conjunto de vectores de entrada para el entrenamiento es de 432 muestras y de 184 para la fase de prueba.

Función de evaluación de error

Los resultados de la red neuronal sobre la estimación de la ubicación se presentan en base a la función distancia promedio de error:

$$Distancia\ Promedio\ de\ Error = \frac{1}{P}\sum_{p=1}^{P}\sqrt{(tx_p - ox_p(w))^2 + (ty_p - oy_p(w))^2},$$

donde $ox_p(w)$ y $oy_p(w)$ son las coordenadas (x, y) obtenidas por la red neuronal y tx_p y ty_p son los valores correctos de las coordenadas deseadas. P es el número de vectores de entrenamiento o de prueba.

Función de Distribución Acumulativa

La función de distribución acumulativa (CDF por sus siglas en inglés) es otra función utilizada para comparar los resultados. La CDF es una función de densidad de probabilidad para la variable distancia de error. La función de distribución acumulativa se define, para una variable X, como:

$$F(x) = P(X \leq x) = \int_{-\infty}^{\infty} f(t)dt,$$

donde la variable x, en nuestro caso, es la distancia de error. $F(x)$ es la probabilidad de obtener una distancia de error menor o igual a x.

Pruebas

Para cada uno de los ocho algoritmos se entrenó la red neuronal hasta que alcanzara alguno de los siguientes criterios de paro:

1. La norma del gradiente sea inferior a un número pequeño ($2E - 4$ en este caso).
2. El tamaño del paso η sea 0, es decir, ya no se hagan modificaciones sobre los pesos sinápticos.
3. El error sobre el conjunto de datos de prueba comience, en un determinado momento, a incrementarse. Esto, aunque el error sobre el conjunto de entrenamiento continúe en decremento.

Cada instante de tiempo (un segundo), se detuvo a la red neuronal en su entrenamiento y se evaluó la distancia de error sobre el conjunto de entrenamiento y de prueba. Cabe mencionar que en los resultados mostrados en esta sección se utilizó sólo la intensidad de la señal como el conjunto de datos de entrenamiento, en secciones posteriores se muestran los resultados al introducir la relación señal a ruido al conjunto. En la Tabla 5.1 se muestra el resultado de la ejecución de los distintos algoritmos de entrenamiento sobre el conjunto de datos de entrenamiento. Se utilizó un mismo conjunto inicial de pesos sinápticos para todos los tipos de entrenamiento. Se resalta en negritas, para cada algoritmo de entrenamiento, la menor distancia de error.

La Tabla 5.2 muestra los resultados sobre el conjunto de datos de prueba, que es el conjunto sobre el que nos interesa conocer el comportamiento de la red, ya que son los datos que la red no "conoce".

En las figuras 5.1-5.4 se muestra el comportamiento de las redes neuronales entrenadas con los ocho distintos algoritmos de entrenamiento. En

Tabla 5.1: Resultados de las redes neuronales sobre el conjunto de entrenamiento

Algoritmo de Entrenamiento	Distancia de Error (m). 4 Neuronas en Capa Intermedia	Distancia de Error (m). 6 Neuronas en Capa Intermedia	Distancia de Error (m). 8 Neuronas en Capa Intermedia	Distancia de Error (m). 16 Neuronas en Capa Intermedia
TrainLm	2.2197	2.0292	1.9605	**1.6421**
TrainScg	2.2905	2.1303	2.0513	**1.9035**
TrainOss	2.6518	2.1952	**2.0208**	2.0378
TrainRp	2.2448	2.2372	2.0686	**1.8426**
TrainCgp	2.6915	2.1823	2.0611	**2.0197**
TrainCgf	2.2633	2.1468	2.0394	**1.854**
TrainCgb	2.2433	2.1923	**1.9946**	2.0031
TrainBfg	2.2478	2.0978	2.0534	**2.0254**

Tabla 5.2: Resultados de las redes neuronales sobre el conjunto de prueba

Algoritmo de Entrenamiento	Distancia de Error (m). 4 Neuronas en Capa Intermedia	Distancia de Error (m). 6 Neuronas en Capa Intermedia	Distancia de Error (m). 8 Neuronas en Capa Intermedia	Distancia de Error (m). 16 Neuronas en Capa Intermedia
TrainLm	2.4912	2.3639	**2.1726**	2.4303
TrainScg	2.4517	2.2592	2.2492	**2.22**
TrainOss	2.8271	2.2947	2.2716	**2.2573**
TrainRp	2.5596	2.4067	2.3783	**2.1843**
TrainCgp	2.8194	2.2827	2.3145	**2.2733**
TrainCgf	2.4242	2.2489	2.257	**2.2242**
TrainCgb	2.4089	2.2837	2.2979	**2.2666**
TrainBfg	2.5127	2.272	**2.2393**	2.2704

las gráficas se muestra el tiempo de entrenamiento contra la distancia de error, en ellas se empieza a graficar desde el tiempo de un segundo de entrenamiento. Las figuras 5.1 y 5.2 corresponden al comportamiento de las redes con el conjunto de datos de entrenamiento y las figuras 5.3 y 5.4 con el conjunto de datos de prueba. En las figuras se presentan sólo las arquitecturas de menor distancia de error por cada algoritmo de entrenamiento, obtenidas en la Tabla 5.2.

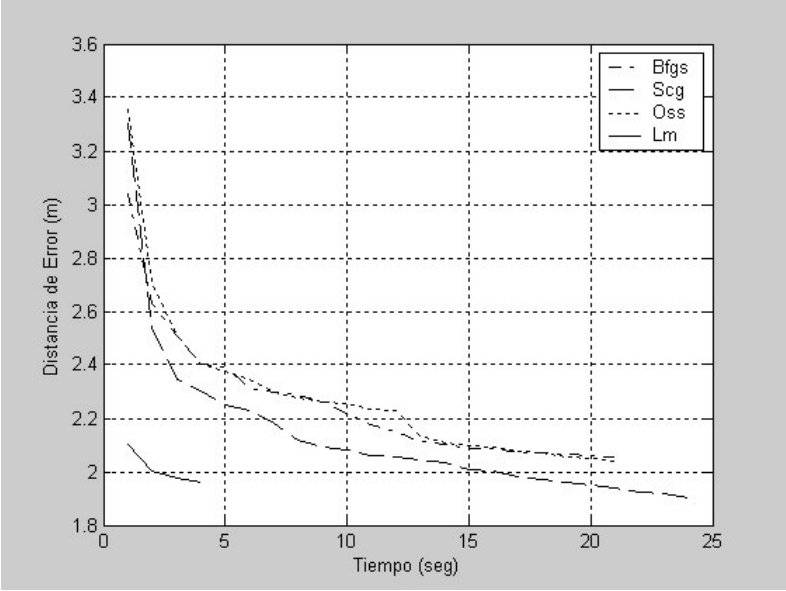

Figura 5.1: Algoritmos Quasi-Newton sobre el conjunto de datos de entrenamiento

En la Tabla 5.3 se muestran los resultados de los ocho algoritmos de entrenamiento con la topología de red que arrojó la menor distancia promedio de error en la Tabla 5.2. En la tabla se utiliza la función de distribución acumulativa en distancias de dos, tres y cuatro metros como instrumento de comparación; se muestra, además, el tiempo que se llevó en converger hacia la distancia promedio de error.

De la Tabla 5.3 se observa que la menor distancia promedio de error corresponde al algoritmo de entrenamiento Quasi-Newton Levenberg – Marquardt (TrainLm), con 2.1726 metros; con un porcentaje del 55% de las estimaciones de la red neuronal que caen dentro de los 2 metros de error de

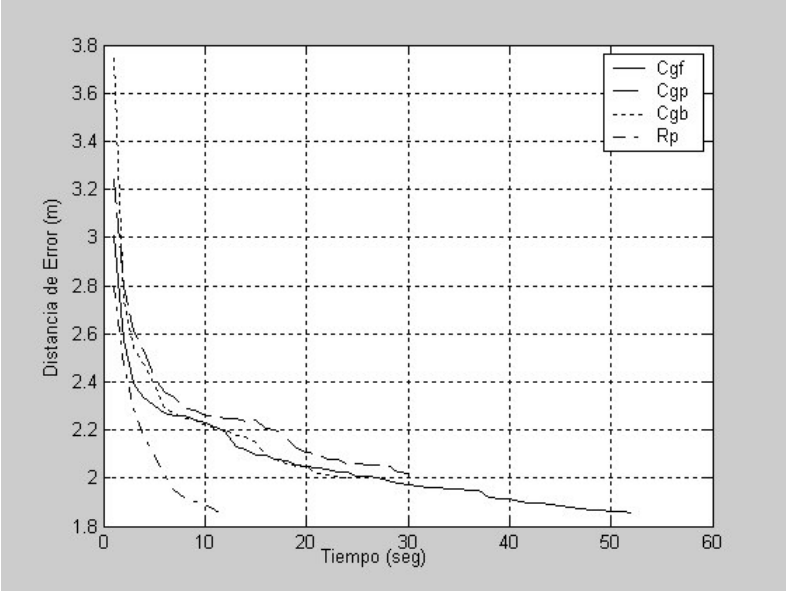

Figura 5.2: Algoritmos CG y RP sobre el conjunto de datos de entrenamiento

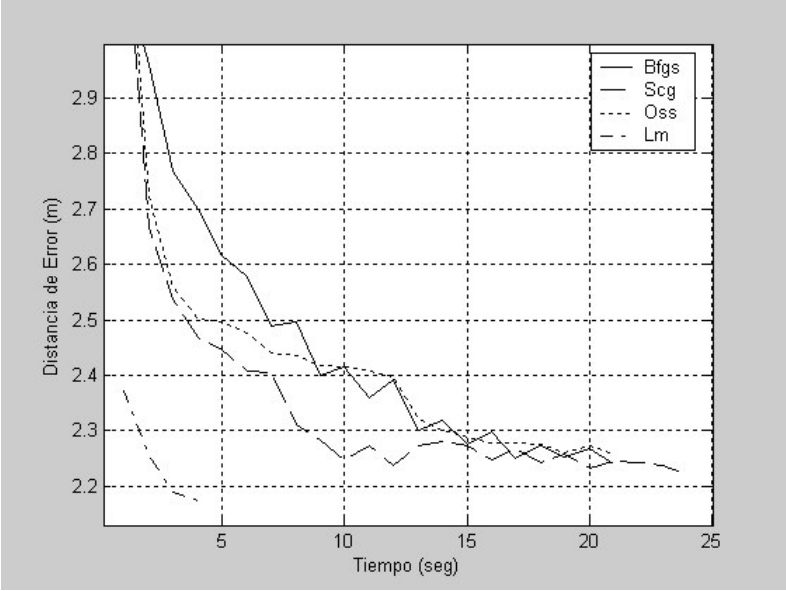

Figura 5.3: Algoritmos Quasi-Newton sobre el conjunto de datos de prueba

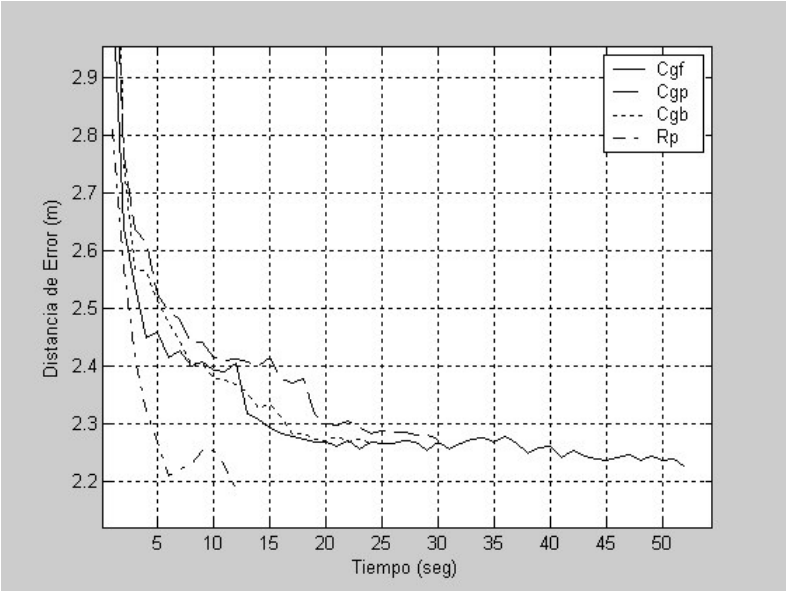

Figura 5.4: Algoritmos CG y RP sobre el conjunto de datos de prueba

Tabla 5.3: Resultados de las redes neuronales sobre el conjunto de datos de prueba

Algoritmo de Entrenamiento - Topología de Red	Distancia Promedio de Error (m.)	CDF en 2 metros (%)	CDF en 3 metros (%)	CDF en 4 metros (%)	Tiempo de Entrenamiento (seg.)
TrainLm 5-8-2	**2.1726**	**0.55**	**0.77**	**0.89**	**4**
TrainScg 5-16-2	2.22	0.52	0.77	0.91	24
TrainOss 5-16-2	2.2573	0.50	0.77	0.90	21
TrainRp 5-16-2	2.1843	0.59	0.76	0.91	12
TrainCgp 5-16-2	2.2733	0.50	0.75	0.90	30
TrainCgf 5-16-2	2.2242	0.51	0.78	0.90	52
TrainCgb 5-16-2	2.2666	0.48	0.75	0.91	24
TrainBfg 5-8-2	2.2393	0.51	0.79	0.89	21

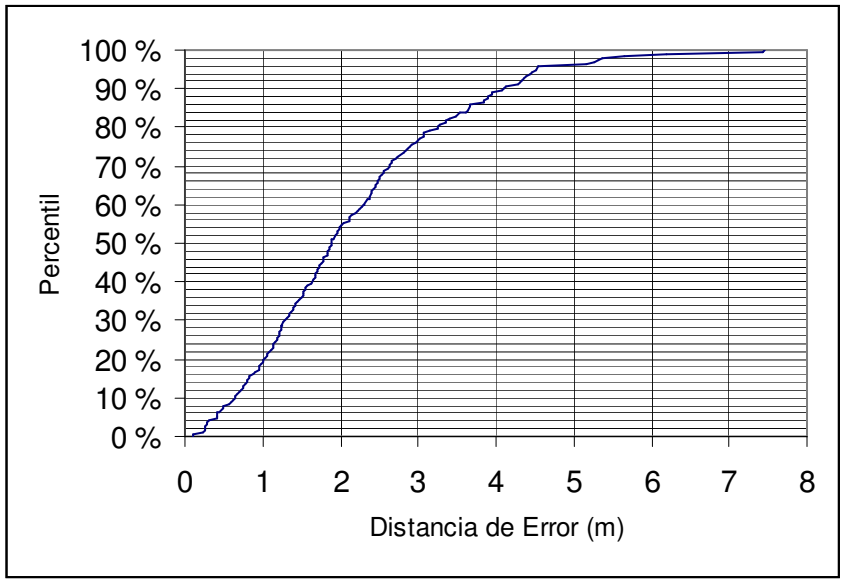

Figura 5.5: CDF para el resultado TrainLm con 8 Neuronas Intermedias

la ubicación real, un 77% dentro de los 3 metros y un 89% dentro de los 4 metros de error. En la Figura 5.5 se muestra la gráfica de la función de distribución acumulativa para este algoritmo.

Relación Señal a Ruido como Valor de Entrada a la Red Neuronal

La relación señal a ruido (SNR) fue evaluada como vector de entrada, en vez de la intensidad de la señal, para el entrenamiento y prueba de la red neuronal. Se tomaron en cuenta a los ocho algoritmos de la Tabla 5.3. Los resultados sobre el conjunto de datos de prueba se muestran en la Tabla 5.4.

El mejor resultado en la Tabla 5.4 es el correspondiente al algoritmo TrainCgf, con una distancia promedio de error de 2.1259 metros y con un porcentaje del 55% de las estimaciones que caen dentro de los 2 metros de error de la ubicación real, un 77 % dentro de los 3 metros y un 89% en 4 metros.

Tabla 5.4: Resultados con la SNR como vector de entrada a la red neuronal

Algoritmo de Entrenamiento - Topología de Red	Distancia Promedio de Error (m.)	CDF en 2 metros (%)	CDF en 3 metros (%)	CDF en 4 metros (%)	Tiempo de Entrenamiento (seg.)
TrainLm 5-8-2	2.1953	0.53	0.80	0.90	32
TrainScg 5-16-2	2.1727	0.54	0.76	0.89	19
TrainOss 5-16-2	2.1518	0.53	0.77	0.90	32
TrainRp 5-16-2	2.135	0.52	0.79	0.90	14
TrainCgp 5-16-2	2.1533	0.55	0.75	0.89	40
TrainCgf 5-16-2	**2.1259**	**0.55**	**0.77**	**0.89**	**50**
TrainCgb 5-16-2	2.19	0.51	0.78	0.91	30
TrainBfg 5-8-2	2.314	0.53	0.75	0.90	40

Relación Señal a Ruido e Intensidad de la Señal como Valor de Entrada a la Red Neuronal

El vector de entrada de la red neuronal se amplió a diez nodos, para hacer una combinación de la intensidad de la señal con la relación señal a ruido como vector de entrada para el entrenamiento y prueba de la red neuronal. En este caso, la mejor topología de red para todos los algoritmos de entrenamiento fue la de 16 neuronas en la capa intermedia. Los resultados sobre el conjunto de datos de prueba se muestran en la Tabla 5.5.

De la Tabla 5.5 se observa que el mejor resultado corresponde al algoritmo TrainRp, con un promedio de error de 2.149 metros y un 54 por ciento de las estimaciones que caen dentro de los 2 metros de distancia de la posición real del usuario. Además, un 76 % cae dentro de los 3 metros y un 93 % dentro de los 4 metros.

Los resultados de la Tabla 5.5 mejoran a los obtenidos de la Tabla 5.3 al disminuir la distancia promedio de error en seis de los ocho algoritmos de entrenamiento. Sin embargo, al comparar los resultados con la Tabla 5.4, donde la SNR es el patrón de entrada a la red, se observa que la distancia promedio de error se incrementa en casi todos los algoritmos de entrenamiento (excepto en TrainBfg).

Tabla 5.5: Resultados con la Intensidad de la señal y la SNR como vector de entrada a la red neuronal

Algoritmo de Entrenamiento – Topología de Red	Distancia Promedio de Error (m.)	CDF en 2 m. (%)	CDF en 3 m. (%)	CDF en 4 m. (%)	Tiempo de Entrenamiento (seg.)
TrainLm 10-16-2	2.2794	0.52	0.76	0.88	3
TrainScg 10-16-2	2.2009	0.53	0.78	0.90	24
TrainOss 10-16-2	2.2004	0.52	0.78	0.90	92
TrainRp 10-16-2	**2.149**	**0.54**	**0.76**	**0.93**	**12**
TrainCgp 10-16-2	2.2199	0.53	0.76	0.89	34
TrainCgf 10-16-2	2.2261	0.51	0.79	0.89	51
TrainCgb 10-16-2	2.1973	0.52	0.77	0.89	46
TrainBfg 10-16-2	2.2346	0.51	0.75	0.90	43

Condiciones de Multi-Inicio

Con el propósito de aumentar la probabilidad de encontrar el mínimo global en la superficie de error, se reinició a la red neuronal desde múltiples puntos aleatorios sobre la superficie de error ($w_i \in [-4.8751, 4.8751]$) y se comenzó el entrenamiento a partir de esos puntos. Se probó este multi-inicio para la topología de red de 16 neuronas intermedias con el algoritmo de entrenamiento TrainCgf y con la SNR como vector de entrada, que fue la combinación con la menor distancia de error en las pruebas anteriores. El multi-inicio se realizó 260 veces y se encontró un conjunto de pesos sinápticos que arrojó una distancia promedio de error de 2.0947 metros.

Experimentación sobre los resultados obtenidos

Si tomamos en cuenta dos características del problema de la localización, que son la velocidad a la que camina una persona normalmente, que es de 1.5 m/s[1] y que se podrían hacer 4 estimaciones de la red neuronal por cada punto en el edificio en un instante menor o igual a un segundo de tiempo, se podrían mejorar los resultados obtenidos en las secciones anteriores, al desechar a aquella estimación (una como máximo) que nos arroje la red neuronal que

[1] Estimación obtenida después de realizar el experimento de medir el tiempo aproximado en que una persona se toma en trasladarse a través de un pasillo, a un ritmo normal.

sea mayor a 1.5 metros del promedio de las cuatro estimaciones hechas en un mismo punto, ya que la persona no pudo desplazarse más allá de 1.5 metros en un segundo o menos. Se realizó este experimento sobre los mejores resultados de las Tablas 5.3, 5.4 y 5.5. Los resultados de este experimento se muestran en la Tabla 5.6.

Tabla 5.6: Resultado del experimento sobre los resultados de las Tablas 5.3, 5.4 y 5.5

Algoritmo de Entrenamiento - Topología de Red	Vector de Entrada	Distancia Promedio de Error (m.)	CDF en 2 m. (%)	CDF en 3 m. (%)	CDF en 4 m. (%)
TrainLm 5-8-2	Intensidad	1.9168	0.60	0.84	0.95
TrainCgf 5-16-2	SNR	**1.8753**	**0.60**	**0.84**	**0.94**
TrainRp 10-16-2	SNR-Int.	1.8978	0.59	0.82	0.98

En la Tabla 5.6 se observa que se mejoran los tres resultados obtenidos de las Tablas 5.3, 5.4 y 5.5. El mejor resultado de las tablas anteriores, correspondiente al entrenamiento TrainCgf y con el vector SNR como patrón de entrada, disminuye a una distancia promedio de error de 1.8753 metros y con un porcentaje del 60 % de las estimaciones, realizadas sobre el conjunto de prueba, que caen dentro de los 2 metros de error de la ubicación real del usuario, además un 84 % cae dentro de los 3 metros y el 94 % dentro de los 4 metros de error.

Comparación de la técnica de Redes Neuronales con la técnica k-Nearest Base Station

La técnica de k-nearest base station [BaP+00][BaR+02] es una técnica que clasifica un patrón dado con la misma etiqueta que la del patrón de ejemplo más cercano. El algoritmo de k-nearest, aplicado al problema de la ubicación y denominado algoritmo *k-nearest en el espacio de la señal* [BaP+00], puede definirse en los siguientes puntos [BaR+02]:

1. Se utilizan, al igual que para las redes neuronales, un conjunto de datos de entrenamiento y uno de prueba.

2. En el conjunto de datos de entrenamiento se consideran las intensidades de las señales de los puntos de acceso y la posición: (IS_1^i,IS_2^i,IS_3^i,IS_4^i,-IS_5^i,x^i,y^i). En tanto que en el conjunto de prueba se consideran sólo las intensidades de la señal: $(IS_1, IS_2, IS_3, IS_4, IS_5)$.

3. Para cada tupla $(IS_1, IS_2, IS_3, IS_4, IS_5)$ del conjunto de prueba, de la que se quiera saber su posición correspondiente, se realizan los siguientes pasos:

   (a) Para cada $i \in 1...N$, donde N es el número de ejemplos del conjunto de entrenamiento, se computa la distancia Euclidiana estándar d_i entre todas las tuplas $(IS_1^i, IS_2^i, IS_3^i, IS_4^i, IS_5^i)$ del conjunto de entrenamiento y la tupla $(IS_1, IS_2, IS_3, IS_4, IS_5)$. La distancia Euclidiana estándar se define como

   $$\begin{aligned} d_i^2 &= (IS_1^i - IS_1)^2 + (IS_2^i - IS_2)^2 + (IS_3^i - IS_3)^2 \\ &\quad +(IS_4^i - IS_4)^2 + (IS_5^i - IS_5)^2. \end{aligned}$$

   (b) Se ordenan las distancias d_i de menor a mayor y se toman las tuplas con las k distancias más pequeñas (de ahí el término k-nearest).

   (c) Se computa la posición de la tupla del conjunto de prueba como el promedio de las posiciones de las k tuplas consideradas en el punto anterior:

   $$x = \frac{\sum_{i=1}^{k} x^{i_j}}{k}, \qquad y = \frac{\sum_{i=1}^{k} y^{i_j}}{k}.$$

Para comparar los resultados de las redes neuronales con la técnica k-nearest, se tomaron los mismos conjuntos de datos de entrenamiento y de prueba utilizados en las redes neuronales, específicamente el conjunto de datos de la relación señal a ruido, ya que este conjunto de datos como patrón de entrada a la red neuronal fue el que arrojó los mejores resultados. En la Figura 5.6 se muestra, aplicado al conjunto entero de datos de prueba, la distancia promedio de error contra el número de k distancias más cercanas tomadas en consideración.

De la Fig. 5.6 se observa que en el error promedio mínimo se alcanza al tomar las ocho distancias más cercanas en el espacio de la señal, y a partir

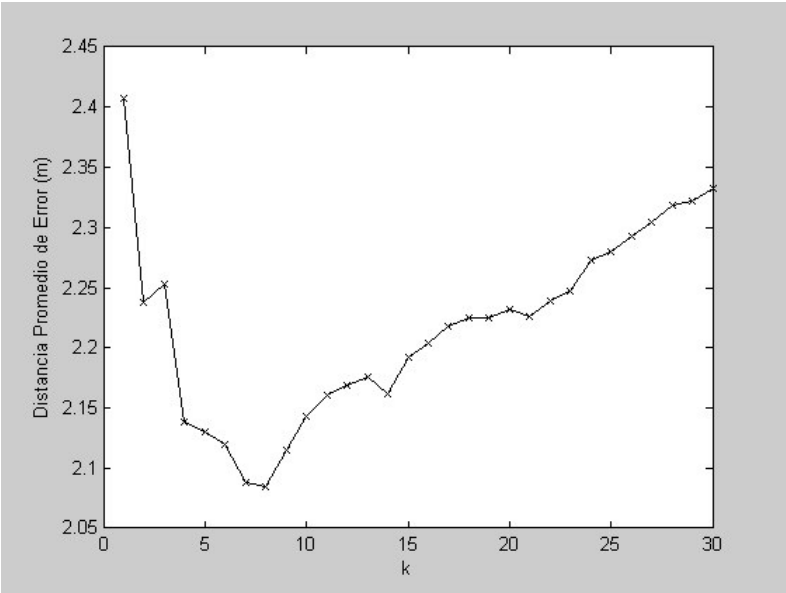

Figura 5.6: La distancia promedio de error como función del número de distancias más cercanas tomadas en consideración en el algoritmo k-nearest

de ahí, conforme k se incrementa, se incrementa la distancia de promedio de error.

En la Tabla 5.7 se muestra una comparación entre la técnica k-nearest (con una y ocho distancias más cercanas) y la técnica de redes neuronales. El resultado de redes neuronales corresponde al de la sección de condiciones de multi-inicio, que fue el mejor resultado encontrado.

Tabla 5.7: Tabla de comparación entre la técnica k-nearest y las redes neuronales

Técnica	Distancia Promedio de Error (m.)	CDF en 2 m. (%)	CDF en 3 m. (%)	CDF en 4 m. (%)
Redes Neuronales	2.0947	0.54	0.79	0.91
8-nearest neighbor	2.0841	0.56	0.81	0.92
Nearest neighbor	2.4068	0.57	0.72	0.83

En la Fig. 5.7 se muestra la gráfica CDF de k-nearest con ocho distancias más cercanas, y la gráfica CDF de redes neuronales.

Discusión

La topología de red que mostró los mejores resultados fue la de una capa intermedia, con cinco nodos en la capa de entrada, dieciséis nodos en la capa oculta y dos nodos en la capa de salida, con la relación señal a ruido como patrón de entrada tanto para el entrenamiento de la red como para el pronóstico de la ubicación.

El mejor algoritmo de entrenamiento para la topología mencionada en el párrafo anterior lo constituyó el algoritmo del Gradiente Conjugado con Actualización Fletcher-Reeves, con el resultado de una distancia promedio de error de 2.0947 metros y con un 54, 79 y 91 por ciento de probabilidad de estimar la ubicación del usuario dentro de los 2, 3 y 4 metros de la ubicación real, respectivamente.

Los algoritmos de entrenamiento Quasi-Newton Levenberg – Marquardt y de Retropropagación Elástico fueron los algoritmos de más rápida convergencia, con tiempos de entrenamiento considerablemente menores a los tiempos de los demás algoritmos (véanse Fig. 5.3 y 5.4).

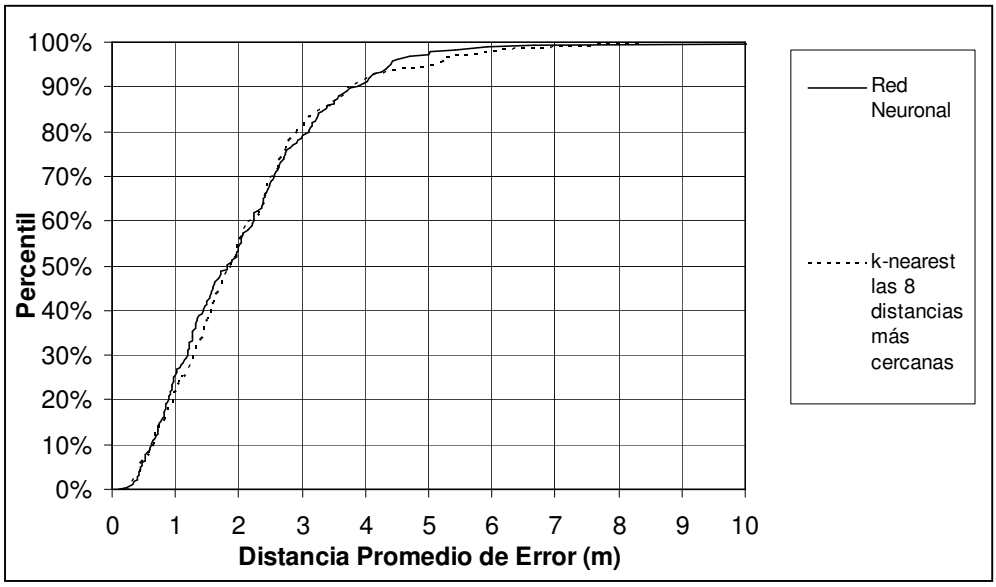

Figura 5.7: Gráfica CDF de comparación entre la técnica k-nearest y las redes neuronales

Con la relación señal a ruido como vector de entrada para el entrenamiento, las redes neuronales presentaron su mejor desempeño en minimizar la distancia promedio de error, en comparación a cuando se aplicaron los otros dos tipos de vectores de entrada: la intensidad de la señal y la combinación de la intensidad de la señal con la relación señal a ruido.

La técnica de k-nearest, al tomar sólo la distancia más cercana, es equivalente al algoritmo *Nearest Neighbor in Signal Space*, del sistema Radar [BaP+00]. Se observa en la Tabla 5.7 que las redes neuronales obtuvieron un mejor resultado que este algoritmo.

El resultado de las redes neuronales es similar, con 2.0947 metros, al mejor resultado de la técnica k-nearest, con 2.0841 metros de distancia promedio de error.

La diferencia entre los resultados obtenidos en este trabajo y en [BaR+02], al haberse aplicado en ambos trabajos el mismo algoritmo k-nearest sobre sus respectivos conjuntos de datos de entrenamiento y de prueba (obteniéndose distancias de error de 2.08 y 1.81 m. respectivamente) se debe a las metodologías que se usaron para la recolección de los datos y a la variabili-

dad del ambiente de las señales dentro de la infraestructura de ambos lugares.

El mayor esfuerzo para generar una red neuronal como herramienta para la estimación de la ubicación, radica en el registro de las intensidades de las señales de los puntos de acceso, ya que se hace de manera manual en un número grande de lugares del edificio dentro del cual se quiera inferir la ubicación de los usuarios móviles. El entrenamiento de la red neuronal, como se mostró, toma un tiempo relativamente corto.

La distancia promedio de error obtenida en los resultados es adecuada en situaciones donde las aplicaciones conscientes del contexto que hagan uso de las estimaciones de la red neuronal, requieran como mínimo, conocer la ubicación del usuario en términos de instancias de lugares tales como pasillos, cuartos, salas, cubículos, entre otros.

Discusión de la Aplicación Consciente del Contexto en Funcionamiento

Se realizó un recorrido sobre la segunda planta del Instituto de Electrónica y Computación, con la aplicación consciente del contexto ejecutándose en una computadora portátil.

Se observó, al realizar el recorrido, que la precisión de las estimaciones de la red neuronal dependen directamente de la relación que existe entre el estado de movimiento de la persona y el número n de las últimas mediciones de intensidad que se toman del archivo (sobre el que se hace un registro periódico de las intensidades de las señales[2]) y que se promedían para formar un sólo vector o patrón de entrada, que es el que se presenta ante la red neuronal para obtener las coordenadas (x, y). El parámetro n es el que en la ventana de la aplicación se señala como "Número de mediciones a utilizar", y que es seleccionado por el usuario.

Cuando la persona está en movimiento constante es preferente tomar tan sólo el último registro realizado sobre el archivo, debido a que los registros anteriores a éste seguramente ya no representan el estado actual del ambiente de las señales sobre el que se encuentra la persona.

Cuando el estado de la persona es estático, para una mayor precisión es conveniente utilizar un número n de registros mayor a uno. Esto es justificable, porque los valores de las señales son fluctuantes sobre cualquier punto del edificio pero tenderán a nivelarse a un valor promedio conforme pase el

[2]Véase el manual de usuario del apéndice B

tiempo [You+02], por lo que presentar ante la red un vector de intensidades de entrada con valores lo más cercanos a este valor promedio resulta en una mejor precisión de la estimación.

Conclusiones

La motivación para realizar este trabajo era presentar un método que facilitara la ubicación de las personas con dispositivos de tecnología móvil, en pos de que las aplicaciones hicieran uso de esta información para proporcionar mejores y oportunos servicios a los usuarios. El conocimiento de la ubicación tiene una amplia variedad de aplicaciones, las cuales se irán incrementando conforme la tecnología en comunicaciones inalámbricas esté disponible para cada vez más personas.

Se cumplieron los objetivos planteados al inicio del trabajo y se detallan a continuación:

- Se diseñó y evaluó una metodología de inferencia basada en redes neuronales de retropropagación para estimar la posición de un dispositivo móvil en una red LAN inalámbrica dentro de edificios.
- Se alcanzó una distancia promedio de error de 2.09 metros en las estimaciones de la red neuronal.
- Se propuso y se probó el criterio experimental de la velocidad promedio a la que caminaría una persona dentro de un edificio, que permite descartar algunas estimaciones de la red neuronal, con lo que la distancia promedio de error disminuyó a 1.87 metros, sobre los resultados ya obtenidos.
- Los resultados obtenidos con las redes neuronales de retropropagación para la inferencia de la ubicación son comparables a los mejores resultados de la técnica k-nearest.
- Se construyó una aplicación consciente del contexto, que permitió implementar las redes neuronales como técnica de inferencia de la ubicación.

- La técnica propuesta sólo necesita una infraestructura de red inalámbrica compatible con el estándar IEEE 802.11b, en vez de requerir de una infraestructura especializada para el posicionamiento de usuarios móviles.

Perspectivas

- La recolección automática de las muestras de intensidad de la señal es una parte importante en la construcción de un módulo que proporcione a la red neuronal de esta información y se pueda obtener así la estimación de la posición del usuario en un tiempo real.

- El sistema que se requiera que funcione de manera independiente dentro de los dispositivos inalámbricos y que dé soporte de la información de la ubicación a otras aplicaciones, se conformaría de tres partes principalmente: el módulo de obtención automática de la intensidad de la señal, la red neuronal previamente entrenada y adecuada al lugar de interés, y un módulo que haga uso del mapa del lugar para trasladar la salida de la red neuronal a instancias de objetos como habitaciones, cubículos, salas, entre otros.

- Se pueden utilizar técnicas como los Modelos de Markov para predecir la ubicación, e incluir no sólo las mediciones actuales de los puntos de acceso, sino también las posiciones previas del usuario, con el fin de incrementar la precisión de la estimación.

Referencias

[AbG+97] Abowd, G. D., Atkenson, Ch. G., Hong, J., Long, S., Kooper, R. and Pinkerton, M., Cyberguide: A mobile context-aware tour guide. *Wireless Networks, v*ol. 3, no. 5, pp. 421-433, October 1997.

[AsA+94] Ashtana, A., Cravatts, M. and Krzyzanowski, An indoor wireless system for personalized shopping assistance. *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, California, IEEE Computer Society Press, pp. 69-74, December 1994.

[BaR92] Battiti, R., First and second order methods for learning: Between steepest descent and Newton's method. *Neural Computation*, vol. 4, no. 2, pp. 141-166, 1992.

[BaR+02] Battiti, R., Villani, A. and Le Nath T., Neural network model for intelligent networks: deriving the location from signal patterns. *Proceedings of The First Annual Symposium on Autonomous Intelligent Networks and Systems*, UCLA, May 8-9, 2002.

[BaP+00] Bahl, P. and Padmanabhan, V. N., Radar: An in-building RF-based user location and tracking systems. *Proceedings of IEEE INFOCOM 2000*, Tel-Aviv, Israel, IEEE Computer Society Press, March 2000.

[BeS91] Becker, S., Unsupervised learning procedures for neural networks. *International Journal of Neural Systems*, vol. 2, pp. 17-33, 1991.

[BiC95] Bishop, C.M., *Neural Networks for Pattern Recognition*, Oxford, Oxford University Press, pg. 130, 1995.

[CaP+01] Castro, P., Chiu, P., Kremenek, T. and Muntz, R. R., A probabilistic room location service for wireless networked environments. *Ubicomp*, Atlanta, GA, ACM 25, 2001.

[Cha92] Charalambous, C., Conjugate gradient algorithm for efficient training of artificial neural networks. *IEEE Proceedings*, vol. 139, no. 3, pp. 301-310, 1992.

[ChG+00] Chen, G. and Kotz, D. *A., Survey of context-Aware Mobile Computing Research*. Dartmouth Computer Science Technical Report TR2000-381, 2000.

[ChP+92] Churchland, P. S., Sejnowski, T. J., *The Computational Brain*, Cambridge, MA, MIT Press, 1992.

[DaT+98] Darrell, T., Gordon, G., Harville, M. and Woodfill, J., Integrated person tracking using stereo, color, and pattern detection. *Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, pp. 601-608, June 1998.

[DeJ+83] Dennis, J. E. and Schnabel, R. B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Englewood Cliffs, NJ, Prentice-Hall, 1983.

[DeA+99] Dey, A. K. and Abowd, G. D., *Towards a Better Understanding of context and context-awareness*. Technical Report GIT-GVU-99-22, Georgia Institute of Technology, College of Computing, June 1999.

[DeAK+99] Dey, A. K., Futakawa, M., Salber, D., and Abowd, G. D., The Conference Assistant: Combining Context-Awareness with Wearable Computing. *Proceedings of the 3rd International Symposium on Wearable Computers (ISWC'99)*, San Francisco, CA, IEEE Computer Society Press, pp. 21-28, October 1999.

[DjG+01] Djuknic, G. M. and Richton., Geolocation and Assisted GPS. *IEEE Computer*, vol. 34, no. 2, pp 123-125, 2001.

[FaL94] Fausett, L. V., *Fundamentals of neural networks: architectures, algorithms and applications*. Prentice.Hall, 1994.

[FlR+64] Fletcher, R., Reeves, C., Function minimization by conjugate gradients. *Computer J.*, vol. 7, pp. 149-154, 1964.

[FoF+97] Foresee, F. D. and Hagan, M. T., Gauss-Newton approximation to Bayesian regularization. *Proceedings of the 1997 International Joint Conference on Neural Networks*, pp. 1930-1935, 1997.

[GiG+02] Giaglis, G., Kourouthanasis P., Tsamakos, A., Towards a classification network for mobile location services. *Mobile Commerce: Technology, Theory, and Applications*, Mennecke, B.E. and Strader, T.J. (Eds.), Idea Group Publishing, 2002.

[HaM+94] Hagan, M. T. and Menhaj, M., Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989-993, 1994.

[HaD90] Hammerstrom, D., A VLSI architecture for high-performance, low-cost, on-chip learning. *International Joint Conference Neural Networks*, vol. 2, pp. 537-544, 1990.

[HaA+01] Harter, A. and Hopper, A. A., Distributed location system for the active office. *IEEE Network*, IEEE Computer Society Press, pp. 1655-1663, April 2001.

[HaA+99] Harter, A., Hopper, A., Ward, A. and Webster, P., The anatomy of a context-aware application. *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 1999)*, Seattle, WA, ACM Press, pp. 59-68, August 1999.

[HaS94] Haykin, S., *Neural Networks, a Comprehensive Foundation*, New York, NY, Macmillan, 1994.

[HoT+97] Hodes, T. D., Katz, H., Schreiber, E. S. and Rowe, L., Composable Ad Hoc Mobile Services for Universal Interaction. *Proceedings of MobiCom '97*, pp. 1-12, September 1997.

[JoE+91] Johansson , E. M., Dowla, and Goodman, D. M., Backpropagation learning for multi-layer feed-forward neural networks using the conjugate gradient method. *Int. J. Neural Systems*, vol. 2, no. 4, pp. 291-301, 1991.

[KrA+89] Kramer, A. H. and Sangiovanni-Vincentelli, A., Efficient parallel learning algorithms for neural networks. *Advances in Neural Information Processing Systems 1*, California, Morgan Kaufmann, pp. 40-48, 1989.

[MaD92] MacKay, D. J. C., Bayesian interpolation. *Neural Computation*, vol. 4, no. 3, pp. 415-447, 1992.

[MoM93] Moller, M. F., *Exact Calculation of the product of the Hessian matrix of feed-forward network error functions and a vector in O(N) time*, Technical Report PB-432, Sciencie Computer Department, Aarhus University, Denmark, 1993.

[MoMF93] Moller, M. F., A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, vol. 6, pp. 525-533, 1993.

[MuM+03] Muñoz, M., Rodriguez, M., Favela, J., González, V.M. and Martínez-Garcia, A.I., Context-aware mobile communication in hospitals. *IEEE Computer*, vol. 36, no. 8, pp. 38-46, September 2003.

[NoJ+99] Nocedal, J. and Wright, S. J., *Numerical Optimization.* Springer-Verlag New York, Inc., pp. 35-61, 1999.

[Ori03] Orinoco, *AP-200 Access Point.* URL: www.orinocowireless.com, last revision 20 Nov. 2003.

[OrR+00] Orr, R. J. and Abowd, G. D., The smart floor: A mechanism for natural user identification and tracking. *Proceedings of the 2000 Conference on human Factors in Computing Systems (CHI 2000)*, The Hague, Netherlands, ACM, April 2000.

[PaJ+98] Pascoe, J., Morse, D. and Ryan, N., Developing personal technology for the Field. *Personal Technologies*, vol. 2, no. 1, March 1998.

[PeCh+96] Perkins, Ch. E. and Johnson, D. B., Mobility support in IPV6. *Proceedings of the second annual international conference on Mobile computing and networking*, White Plains, NY, ACM Press, pp. 27-37, November 1996.

[PrN+00] Priyantha, N. B., Chakraborty A. and Balakrishnan, H., The cricket location-support system. *Proceedings of MOBICOM 2000*, Boston, MA, ACM, ACM Press, pp. 32-43, August 2000.

[PoE+69] Polak, E. and Ribiere, G., *Rev. Fr. Inform. Rech. Operation (16-R1)*, pp. 35-43, 1969.

[PoM77] Powell, M. J. D., Restart procedures for the conjugate gradient method. *Mathematical Programming*, vol. 12, pp. 241-254, 1977.

[RaF+79] Raab, F., Blood, E., Steiner, T. and Jones, H., Magnetic position and orientation tracking system. *IEEE Transactions on Aerospace and Electronic Systems*, vol. 15, no. 5, pp. 709-717, September 1979.

[RiM+93] Riedmiller, M. and Braun, H., A direct adaptive method for faster backpropagation learning: The RPROP algorithm. *Proceedings of the IEEE International Conference on Neural Networks*, 1993.

[RiB96] Ripley, B.D., *Pattern Recognition and Neural Networks*, Cambridge, Cambridge University Press., pp. 173-180, 1996.

[RoF58] Rosenblatt, F., The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, vol. 65, pp. 386-408, 1958.

[RoF62] Rosenblatt, F., Principles of Neurodynamics. *Principles of Neurodynamics*, New York, Spartan Books, 1962.

[RuD+86] Rumelhart, D. E., McClelland, J.L. and the PDP Research Group, Eds. *Parallel Distributed Processing*, MIT Press, Cambridge, MA, 1986.

[SaS94] Sakagami, S., Vehicle Position Estimates by Multi-beam Antennas in Multi-path Environments. *IEEE Transactions on Vehicular Technologies*, vol. 43, no. 4, pp. 902-908, 1994.

[SaM+93] Satyanarayanan, M., Kistler, J. J., Mummert, B., Ebling, M. R., Kumar, P. and Lu, Q., Experience with disconnected operation in a mobile computing environment. *Proceedings of USENIX*

Mobile & Location-Independent Computing Symposium, Cambridge, Massachusetts, USENIX Association, pp. 11-28, August 1993.

[ShB+94] Schilit, B., Adams, N. and Want, R., Context-aware computing applications. *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, California, IEEE Computer Society Press, pp. 85-90, December 1994, .

[SmA+01] Smailagic, A., Siewiorek, D., Anhalt, J., Kogan, D. and Wang, Y., Location Sensing and Privacy in a Context Aware Computing Environment. *Pervasive Computing*, 2001.

[SoB+92] Soucek, B. and The Iris Group, *Fast Learning and Invariant Object Recognition*. John Wiley and Sons Publishers, pp. 8-13, 1992.

[WaR+92] Want, R., Hopper, A., Falcao, V. and Gibbons, J., The active badge location system. *ACM Transactions on Information systems*, vol. 10, no. 1, pp. 91-102, January 1992.

[WaR+96] Want, R., Schilit, B. N., Adams, N., Gold, R., Petersen, K., Goldberg, D., Ellis, J. R. and Weiser M., The ParcTab Ubiquitous Computing Experiment. *Mobile Computing*, Tomasz Imielinski and Henry F. Korth eds., Kluwer Academic publisher, chapter 2, 1996.

[WeM91] Weiser, M., The computer for the 21st century. *Scientific American*, pp. 94-104, September 1991.

[WeJ+98] Werb, J. and Lanzl, C., A positioning system for finding things indoors. *IEEE Spectrum*, vol. 35, no. 9, pp. 71-78, September 1998.

[WeP74] Werbos, P. J., *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, PhD Thesis, Harvard University, 1974.

[WiB+60] Widrow, B. and Hoff, M. E., Adaptive switching circuits. *IRE WESCON*, New York, Convention Record, pp. 96-104, 1960.

[WiF03] Wi-Fi Alliance, *Wi-Fi specifications*. URL:www.wirelessethernet.org/OpenSection/index.asp, last revision in January 30 2004.

[YaH+00] Yan, H. and Selker T., Context-aware office assistant. *Proceedings of the 2000 International Conference on Intelligent User Interfaces*, New Orleans, LA, ACM Press, pp. 276-279, January 2000.

[You+02] Youssef, M., Agrawala, A., Shankar, A. V. and Noh, S. H., A probabilistic Clustering-Based Indoor Location Determination System. *Technical Report UMIACS-TR 2002-30 and CS-TR 4350*, University of Maryland, College Park, pg. 8, March 2002.

[ZeV+03] Zeimpekis, V., Giaglis, G. M. and Lekakos, G., A taxonomy of indoor and outdoor positioning techniques for mobile location services. *ACM SIGECOM Exchanges*, vol. 3, no. 4, pp. 19-27, 2003.

Apéndice A. Red LAN Inalámbrica

En una red inalámbrica, los dispositivos inalámbricos se conectan unos con otros transmitiendo y recibiendo señales en una frecuencia específica de la banda de radio. Los componentes se pueden conectar unos con otros (*peer-to-peer*) o a través de un punto de acceso (Fig. 5.8). Cuando se crea una red inalámbrica esta consistirá de dos componentes básicos: antenas o tarjetas de interfaz de red y puntos de acceso .

Figura 5.8: Punto de acceso

Las antenas o tarjetas de interfaz de red (NIC por sus siglas en inglés) son agregadas a las computadoras de escritorio, computadoras portátiles y demás dispositivos portátiles móviles para que tengan acceso a las redes inalámbricas (Fig. 5.9 (a) y (b)).

Por otra parte, los puntos de acceso actúan como "estaciones base", ellos mandarán y recibirán señales de las antenas para conectar los distintos componentes unos con otros así como hacia Internet. Todas las computadoras en una red inalámbrica pueden compartir recursos, intercambiar documentos y

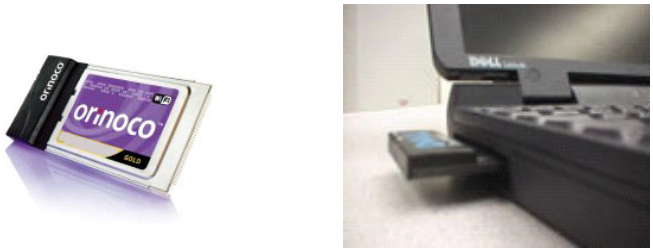

Figura 5.9: a) Tarjeta de interfaz de red. b) Posición de la tarjeta de red en una computadora portátil

usar una sola conexión de Internet. Los puntos de acceso reciben la información, la almacenan y transmiten entre la red LAN inalámbrica (WLAN) y la LAN cableada (Fig. 5.10).

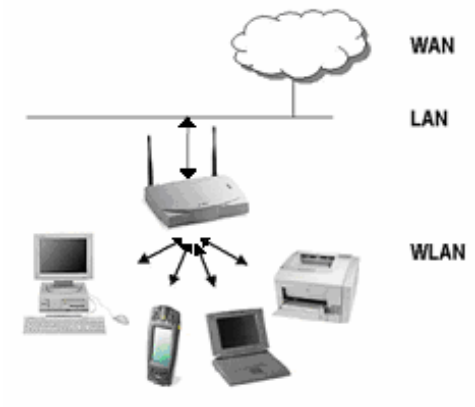

Figura 5.10: Esquema del punto de acceso como unión entre WLAN y LAN

Tarjetas PC Card

El cliente más común en una red inalámbrica es la tarjeta PC Card (Fig. 5.11), también conocida como PCMCIA (*Personal Computer Memory Card International Association*), es quien recibe la difusión de datos en la banda de radio en la que opera la red inalámbrica. Las PC Cards se pueden usar en cámaras, sistemas de audio, PDAs y otros dispositivos de cómputo móvil que tengan ranuras para PC Cards.

Figura 5.11: PC Cards

Puntos de Acceso

Incluso cuando las antenas clientes pueden ser configuradas para establecer comunicación entre ellas, una red inalámbrica opera más efectivamente cuando usa una estación base central que coordina las comunicaciones.

Los puntos de acceso incluyen ruteo NAT (Translación de dirección de red) y servicios DHCP (Protocolo de control de host dinámico). Ellos pueden crear y proveer un IP individual a todos lo clientes inalámbricos de una red y también permitir a un gateway o puerta de enlace, de proveer acceso a Internet a numerosos usuarios de una sola conexión a Internet compartida. Los puntos de acceso pueden también incluir otras aplicaciones y características como encriptación y seguridad, firewall y voz sobre el protocolo Internet (VoIP) [Ori03].

En una red inalámbrica, los puntos de acceso permiten el *roaming* (la propiedad de que un usuario móvil se mueva del área de cobertura de un punto de acceso a otra área sin perder el enlace con la red).

El Estándar 802.11a-b o Wi-Fi

Dos de los estándares más populares dentro de la industria de las redes inalámbricas son los estándares IEEE 802.11a y 802.11b. La velocidad de datos para el estándar 802.11a es de 54 Mbps y de 11 Mbps para 802.11b, trabajando en la banda de 5 GHz y 2.4 GHz respectivamente [WiF03].

Wi-Fi es el acrónimo de *Wireless Fidelity*, el cual es un término usado

Figura 5.12: Puntos de Acceso

genéricamente cuando se refiere a cualquier tipo de red 802.11, sea 802.11b, 802.11a, dual-band, u otras. El estándar 802.11b es hasta ahora el más usado, el cual tiene 11 canales de frecuencia, que van de 2400 a 2483.5 MHz; ésta es la banda de licencia libre ISM (*Industrial, Scientific and Medical*). La técnica de modulación es DSSS (*Direct Sequence Spread Spectrum*). El rango de alcance de un punto de acceso 802.11b varía dependiendo de la infraestructura del lugar. La siguiente tabla muestra el rango de alcance de la señal de un punto de acceso 802.11b en distintos ambientes.

Tabla 5.8: Rango de alcance de la señal del punto de acceso con el estándar 802.11b

Rango (metros)	**11 Mbps**	**5.5 Mbps**	**2 Mbps**	**1 Mbps**
Abierto	160 m	270 m	400 m	550 m
Semi- abierto	50 m	70 m	90 m	115 m
Cerrado	25 m	35 m	40 m	50 m

Apéndice B. Manual del Usuario

La aplicación consciente del contexto que se construyó, se basa en dos módulos más para su funcionamiento: el módulo de la red neuronal y el de recolección de muestras de intensidad de las señales de los puntos de acceso.

El módulo que contiene a la red neuronal es el archivo de Matlab *ubicacionNN.m,* el cual es llamado directamente por la aplicación. El software de *Matlab* con el *Toolbox de Redes Neuronales* debe estar instalado en la computadora portátil. La aplicación consciente del contexto abre una ventana de comandos de Matlab automáticamente al ser ejecutada y la mantiene abierta durante toda su ejecución.

El programa de registro de las intensidades que se utiliza es el *Client Manager*, versión 2.9, de Orinoco.

Precondiciones:

Antes de poner en funcionamiento la aplicación se deben haber cubierto tres aspectos:

1. Tener la tarjeta inalámbrica *Silver Orinoco* en la ranura de la computadora portátil.

2. El archivo *ubicacionNN.m* debe estar localizado en la carpeta *Work*, que se localiza dentro de la carpeta del programa Matlab. Esto, debido a que la ruta de la carpeta Work ya se encuentra en el path de Matlab, en otro caso, sólo hay que agregar al path la ruta en donde se ubique el archivo ubicacionNN.m

3. Se debe instalar y poner en funcionamiento el programa Client Manager.

Para correr el programa Client Manager dé doble clic en el ícono del programa, en la barra de inicio rápido. Al ejecutarse, el programa muestra la pantalla de la Fig. 5.13.

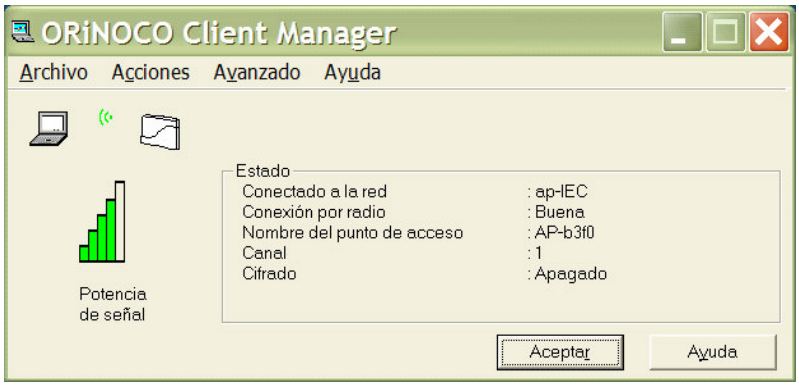

Figura 5.13: Ventana del programa Client Manager

Enseguida hay que configurar el programa para que registre las intensidades de las señales de los puntos de acceso: seleccione **Supervisión en el lugar (Link Test)**, del menú **Avanzado** (**Advanced)**, en la ventana principal del Client Manager y se abrirá la ventana de Supervisión en el lugar (Fig. 5.14).

En la carpeta **Configuración de registro**, de la ventana Supervisión en el lugar, complete el campo del nombre del archivo en el cual se registrarán las mediciones (**Log file name**) con el texto "C:\iec.txt". A continuación seleccione la opción **Conexión automática de datos**(**Automatic data logging**) con periodos de escritura sobre el archivo de **2 segundos**.

Por último, dé clic en el botón **Iniciar conexión**(**Log**). Esto mantendrá al programa Client Manager en estado de registro durante todo el tiempo en que la aplicación consciente del contexto lo necesite. El programa va a registrar sobre el archivo *iec.txt* datos sobre el ambiente de las señales, tales como el nivel de intensidad, el nivel de SNR, el nivel de ruido, y otros.

Si el registro es detenido y se quiere volver a poner en funcionamiento, se debe cerrar y volver a abrir la ventana de Supervisión en el lugar y dar

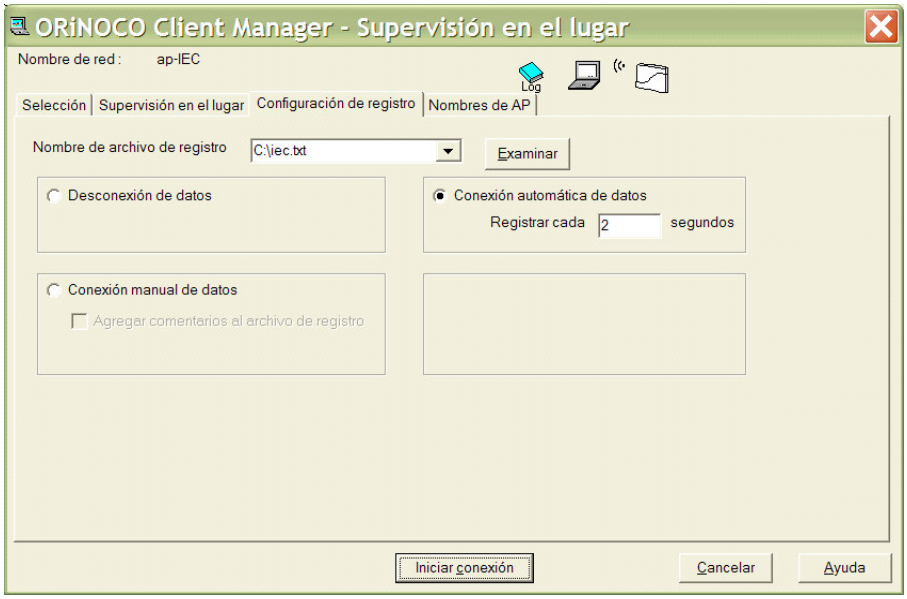

Figura 5.14: Ventana de Supervisión en el lugar

la respuesta de **"Sobreescribir"** el archivo cuando se dé clic en el botón de **Iniciar conexión**.

Ejecutando la Aplicación

Una vez cumplidas las condiciones anteriormente mencionadas, dé clic en el ícono del programa para ponerlo en funcionamiento (Fig. 5.15).

A partir de ese momento puede presionar el botón de **¿Dónde Estoy?** para visualizar en el mapa su posición estimada por la red neuronal, marcada con una X de color rojo (Fig. 5.16).

Faltaría sólo por detallar algunos elementos de la ventana de la aplicación, que se denotan en la Fig. 5.16:

a. **Coordenadas de la posición estimada**. Estas coordenadas están basadas en el plano cartesiano, que se presentó en el capítulo cuatro, para referenciar alguna posición dentro del Instituto de Electrónica y Computación.

b. **Número de mediciones a utilizar**. El programa Client Manager pone un registro nuevo sobre el ambiente en el archivo *iec.txt* cada

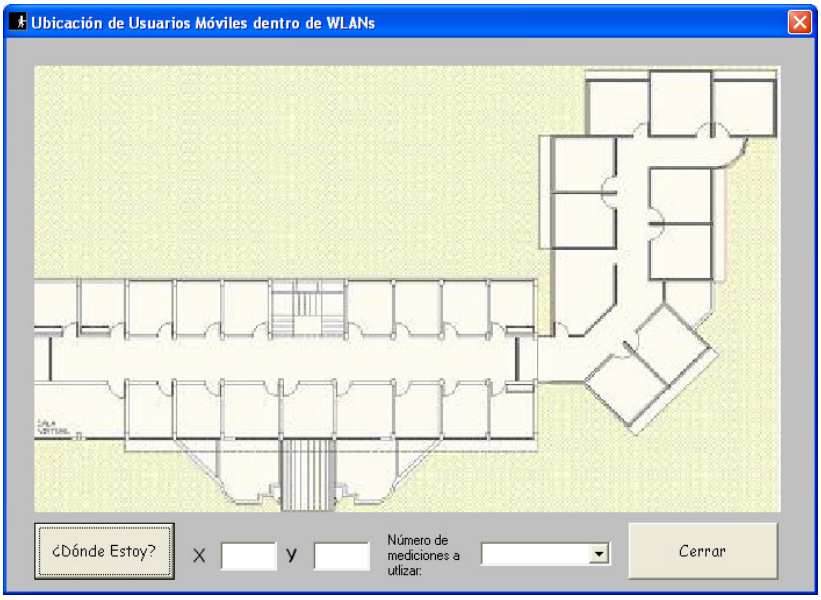

Figura 5.15: Ventana de la aplicación *Mobile.exe*

dos segundos y el módulo de la red neuronal va a promediar las n-últimas mediciones registradas en el archivo para realizar su estimación, parámetro que se le indica en este campo

c. **Cerrar**. Presione este botón cuando desee salir del programa. Al cerrarse la aplicación también se cerrará la ventana de comandos de Matlab que se inició junto con la aplicación.

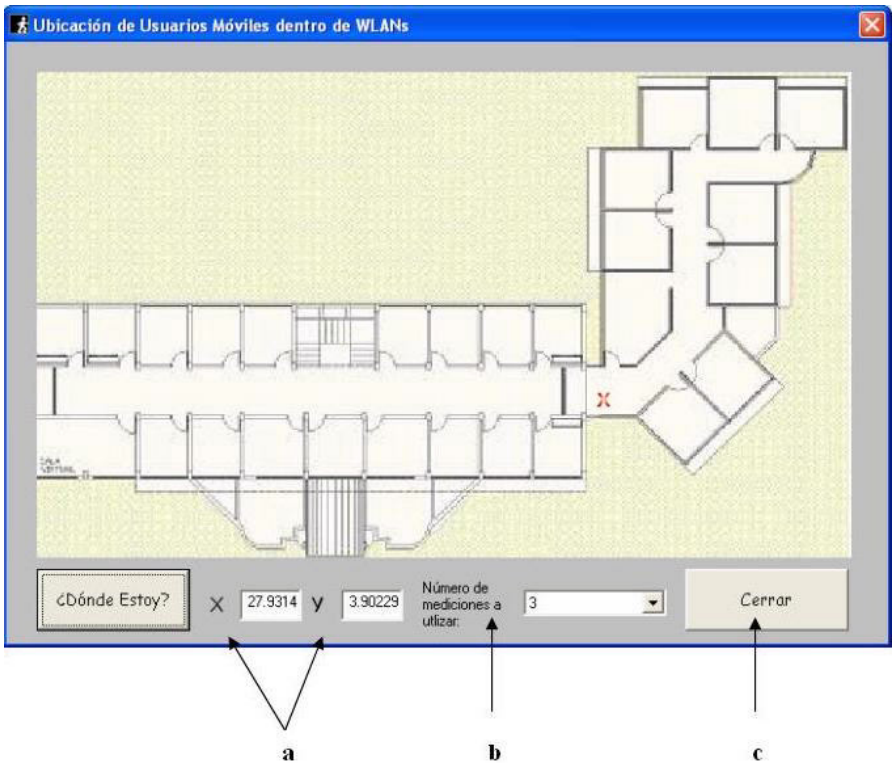

Figura 5.16: La aplicación *Mobile* en ejecución