UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERIA Y ARQUITECTURA
ESCUELA DE INGENIERIA DE SISTEMAS INFORMATICOS

SISTEMA INFORMÁTICO PARA EL ANÁLISIS DIGITAL DE TOMOGRAFÍAS AXIALES COMPUTARIZADAS CEREBRALES UTILIZANDO REDES NEURONALES ARTIFICIALES COMO APOYO EN EL DIAGNÓSTICO DIFERENCIAL A LOS RADIÓLOGOS DEL MINISTERIO DE SALUD DE LA REPÚBLICA DE EL SALVADOR

PRESENTADO POR:

FELIPE SANTIAGO COLATO MÁRTIR

CARLOS ANTONIO HERNÁNDEZ

LENNIN DAVID HERNÁNDEZ NIETO

ALVARO ERNESTO HERNÁNDEZ ORREGO

MARLON ARMANDO MENJÍVAR MARTÍNEZ

PARA OPTAR AL TITULO DE:

INGENIERO DE SISTEMAS INFORMÁTICOS

CIUDAD UNIVERSITARIA, MARZO DE 2017

UNIVERSIDAD DE EL SALVADOR

RECTOR :

MSc. ROGER ARMANDO ARIAS ALVARADO

SECRETARIO GENERAL :

MSc. CRISTÓBAL HERNÁN RÍOS BENÍTEZ

FACULTAD DE INGENIERIA Y ARQUITECTURA

DECANO :

ING. FRANCISCO ANTONIO ALARCÓN SANDOVAL

SECRETARIO :

ING. JULIO ALBERTO PORTILLO

ESCUELA DE INGENIERIA DE SISTEMAS INFORMATICOS

DIRECTOR :

ING. JOSÉ MARÍA SÁNCHEZ CORNEJO

UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERIA Y ARQUITECTURA
ESCUELA DE INGENIERIA DE SISTEMAS INFORMÁTICOS

INGENIERO DE SISTEMAS INFORMÁTICOS

Título :

SISTEMA INFORMÁTICO PARA EL ANÁLISIS DIGITAL DE TOMOGRAFÍAS AXIALES COMPUTARIZADAS CEREBRALES UTILIZANDO REDES NEURONALES ARTIFICIALES COMO APOYO EN EL DIAGNÓSTICO DIFERENCIAL A LOS RADIÓLOGOS DEL MINISTERIO DE SALUD DE LA REPÚBLICA DE EL SALVADOR

Presentado por :

FELIPE SANTIAGO COLATO MÁRTIR

CARLOS ANTONIO HERNÁNDEZ

LENNIN DAVID HERNÁNDEZ NIETO

ALVARO ERNESTO HERNÁNDEZ ORREGO

MARLON ARMANDO MENJÍVAR MARTÍNEZ

Trabajo de Graduación Aprobado por:

Docente Asesor :

ING. RUDY WILFREDO CHICAS VILLEGAS

San Salvador, marzo de 2017

Trabajo de Graduación Aprobado por:

Docente Asesor :

ING. RUDY WILFREDO CHICAS VILLEGAS

Tabla de contenido

Indice de Ilustraciones

Indice de Tablas

Indice de Ecuaciones

INTRODUCCIÓN

A continuación se presentan los elementos desarrollados a lo largo de la ejecución del trabajo de graduación titulado SISTEMA INFORMÁTICO PARA EL ANÁLISIS DIGITAL DE TOMOGRAFÍAS AXIALES COMPUTARIZADAS CEREBRALES UTILIZANDO REDES NEURONALES ARTIFICIALES COMO APOYO EN EL DIAGNÓSTICO DIFERENCIAL A LOS RADIÓLOGOS DEL MINISTERIO DE SALUD DE LA REPÚBLICA DE EL SALVADOR.

Como parte de estos elementos, se plasman los aspectos que dan origen al desarrollo del proyecto, el planteamiento del problema y el diseño de la investigación realizada para abordar el principal obstáculo que consistio en el desconocimiento de la temática y las tecnologías involucradas en la realización del proyecto.

Seguidamente se presentan los resultados de la investigación realizada sobre temas médicos y técnicos especializados en los Diagnósticos Asistidos por Computadora (CADx) y las Redes Neuronales Artificiales (RNA). Se presenta además el proceso de diseño de pruebas, así como también los resultados de la investigación que dan el aval para la construcción del software que se tiene por objetivo en el proyecto.

Partiendo de los resultados de la investigación y del conjunto de conocimientos y habilidades técnicas recopiladas por cada miembro del equipo en el transcurso de la malla curricular, se formuló el diseño del sistema informático realizando un modelado del negocio, definiendo requerimientos informáticos, operativos, de desarrollo y de implementación valiéndose de técnicas tales como entrevistas, encuestas y reuniones con la contraparte técnica y operativa del Ministerio de Salud de El Salvador(MINSAL).

Finalmente se desarrolló el sistema informático implementando una serie de estándares de desarrollo que garantizan la calidad, sostenibilidad y escalabilidad del sistema.

Se incluye un apartado en el que se presentan las conclusiones obtenidas en la realización del proyecto, además de una sección de recomendaciones y anexos de documentos recopilados a lo largo del desarrollo del proyecto.

OBJETIVOS

Objetivo General

- Desarrollar un Sistema Informático para el Análisis Digital de Tomografías Axiales Computarizadas Cerebrales sin contraste utilizando redes neuronales artificiales como apoyo en el diagnóstico que realizan los radiólogos del MINSAL.

Objetivos Específicos

- Realizar investigación sobre TAC cerebrales, y las tecnologías involucradas en el tratamiento de imágenes así como el uso de las redes neuronales artificiales como apoyo al diagnóstico de TAC cerebrales.
- Realizar el análisis, diseño y programación de un sistema informático que apoye el diagnóstico de TAC cerebrales sin contraste.
- Elaborar el plan de implementación y la documentación necesaria para poner en marcha el sistema propuesto.

CAPÍTULO I: ESTUDIO PRELIMINAR

1.1. Antecedentes

El MINSAL es una de las instituciones con mayor trayectoria en el país. Para el año 2016 la red de servicios cuenta con 30 hospitales. Su objetivo es garantizar el derecho a la salud de toda la población a través de un sistema nacional de salud.

Para poder cumplir con la alta demanda de los servicios que presta el MINSAL, su Dirección de Tecnologías de Información y Comunicaciones (DTIC) ha desarrollado una serie de sistemas informáticos con el objetivo de automatizar procesos y hacer un uso eficiente de los recursos con los que dispone, entre estos sistemas se encuentran:

- MorbiMortalidad (SIMMOW)
- Producción de servicios (SEPS)
- Vigilancia Epidemiológica (VIGEPES)
- Vigilancia VIH-SIDA (SUMEVE)
- Ficha Familiar (SIFF)
- Monitoreo y Abastecimiento (SINAB).

El proyecto más ambicioso que actualmente se está desarrollando es el Sistema Integral de Atención al Paciente (SIAP) que tiene el objetivo de brindar información de sus pacientes, sus citas, análisis, exámenes, etc. Este culminará en el Sistema Único de Información en Salud (SUIS).

Mediante convenios, la DTIC recibe apoyo por parte de las universidades para el desarrollo de sistemas de información para uso interno en hospitales y unidades específicas. Recientemente la Universidad de El Salvador colaboró con el desarrollo del Sistema Informático de Imagenología Digital (SIMAGD), este permite vincular al expediente clínico de cada paciente registrado en el SIAP, con la captura, almacenamiento y recuperación de imágenes médicas; dicho sistema se encuentra en funcionamiento desde septiembre del año 2016 en el Hospital Nacional San Rafael de Santa Tecla. Una de las áreas más beneficiadas con el proyecto es la de radiología, pues es en esta área que se utiliza la tecnología imagenológica para diagnosticar y tratar enfermedades.

1.2. Planteamiento del problema

Actualmente el diagnóstico médico basado en imágenes de anatomías internas del paciente ha permitido que cada vez se logre un diagnóstico en menor tiempo y con mayor certeza. Diagnosticar con base en imágenes, solo se encuentra limitado por los conocimientos completos de la anatomía macroscópica del cuerpo con los que cuenta el especialista. Lo anterior implica que la precisión y capacidad de diagnóstico de éste puede verse afectada por errores de percepción causados por desconocimiento, falta de concentración, cansancio u otro tipo de factores. Una alternativa para disminuir la probabilidad de fallos en diagnósticos consiste en utilizar más de una lectura del caso, siendo éste analizado por varios especialistas, estas lecturas pueden ser conjuntas o independientes, situación que por falta de recursos se vuelve imposible de realizar debido a que la red pública hospitalaria cuenta con poca disponibilidad de radiólogos, quienes son los únicos autorizados a realizar diagnósticos basados en TAC.

El Diagnóstico Asistido por Computadora (CAD por sus siglas en inglés), tiene como objetivo mejorar la precisión y consistencia del diagnóstico, apoyando a los especialistas en la interpretación de imágenes médicas, buscando marcar estructuras anómalas que los médicos deben analizar para alcanzar un diagnóstico basado en los datos del paciente. Los sistemas de asistencia al diagnóstico de TAC, proveen de una rápida y confiable segunda opinión a los radiólogos.

La información proporcionada por una herramienta de tipo CAD se basa en el análisis cuantitativo o morfológico de la imagen médica, esta información es generada a través de la aplicación de algoritmos computacionales sobre la información digital de la imagen, lo que garantiza una segunda opinión basada en una mayor cantidad de datos que los que el especialista percibe a simple vista. (Giger ML, Chan HP, Boone J, 2008).

Los algoritmos computacionales para el análisis de imágenes médicas tienen como objetivo la búsqueda de ciertas características en la imagen que permitan identificar anomalías. Las técnicas de procesamiento de imágenes, reconocimiento de patrones e inteligencia artificial permiten que cada vez se generen nuevos algoritmos y métodos capaces de apoyar el diagnóstico con base en una o varias imágenes médicas de anatomías específicas, uno de estos elementos es el conjunto de paradigmas de aprendizaje y procesamiento automático conocido como redes neuronales artificiales, el cual está inspirado en la forma en que funciona el sistema nervioso.

Antes de poder realizar el desarrollo de un sistema informático que integre procesos de información con redes neuronales artificiales, es necesaria una investigación que brinde conocimientos teórico-prácticos en el área de redes neuronales artificiales y TAC cerebrales, para poder resolver el problema del diagnóstico asistido por computadora.

1.3. Formulación del problema

El proyecto propuesto consiste en desarrollar un sistema informático que apoye al especialista en el diagnóstico basado en TAC cerebrales sin contraste, para lo cual fue necesario llevar a cabo dos grandes etapas las cuales se detallan a continuación.

1.3.1. Formulación de la investigación

Los elementos mínimos necesarios para que la investigación sea de utilidad para el desarrollo del proyecto son: la aplicabilidad de las redes neuronales artificiales para reconocimiento y clasificación de imágenes, y los algoritmos y metodologías para el reconocimiento de las mismas. Para la construcción del software es necesario indagar acerca de las tecnologías de desarrollo en las que se puedan implementar las redes neuronales artificiales.

Además, se necesita llevar a cabo un búsqueda de bancos de imágenes TAC con su respectivo diagnóstico para poder hacer el entrenamiento de la red neuronal y las pruebas sobre la fidelidad de los resultados, también es preciso explorar el funcionamiento del formato estándar DICOM así como su estructura de datos y otros elementos que puedan ser de interés considerando que es el estándar con el que las imágenes TAC se almacenan.

1.3.2. Formulación del desarrollo del sistema

Considerando las limitantes de recursos humanos con los que cuenta el MINSAL y que mayoritariamente son técnicos en radiología los encargados del proceso de toma y gestión de TACs, quienes podrán someter dichas imágenes a análisis en caso de ausencia de un médico, para ello se necesita una herramienta que apoye el proceso de diagnóstico, permitiendo reducir el tiempo de atención del paciente sin afectar la calidad de los diagnósticos emitidos, priorizando y focalizando los recursos, así como reduciendo el desgaste físico y mental que produciría en los radiólogos el atender la misma cantidad de pacientes en la misma cantidad de tiempo.

1.4. Importancia

Actualmente las tecnologías que permiten el funcionamiento de los tomógrafos han evolucionado de tal forma que permiten la obtención de estudios más detallados. Inevitablemente esto se traduce en un problema de información excesiva, ya que se genera una cantidad grande de imágenes por paciente que dificulta al especialista la tarea de visualización y lectura de las series, requiriendo un mayor esfuerzo para realizar su diagnóstico.

En la actualidad existen diversos factores por los cuales puede ocurrir algún error al diagnosticar imágenes médicas, por ejemplo el Accidente Vascular Cerebral (AVC) es una de las principales causas de muerte y discapacidad en algunos países.(Orellana, 2003, p. 93-103). Según un estudio realizado en Bolivia, de un total de 385 pacientes atendidos por algún un cuadro de AVC, 65 evidenciaron un falso diagnóstico (FAVC) es decir se emitió un diagnóstico alternativo o fueron dados de alta, el 71.7% de estos FAVC fue atribuido a causas sistémicas. (Viruez, Vera, Torrez y Bailey, 2011, p. 16-21).

Se ha previsto que el CAD ayudará a reducir esa brecha de falsos diagnósticos, brindando a los médicos especialistas un apoyo en la reducción del tiempo de lectura así como en la mejora de la precisión diagnóstica ya que además de facilitar la clasificación preliminar de los estudios, facilitan la medición de otros parámetros de interés.

1.5. Justificación

La red nacional de salud pública cuenta con un reducido número de radiólogos, por lo que es común encontrar situaciones en las que no se cuenta con la presencia de especialistas para atender emergencias que requieran análisis de TAC, esto principalmente en horarios nocturnos. Adicional a esto, no todos los hospitales de la red nacional cuentan con radiólogos.

Como ejemplo de lo anterior, el Ing. Farid Hernández, encargado de implementar el SIMAGD en el Hospital Nacional San Rafael de Santa Tecla, expresa que dicho hospital tiene a su disposición 3 médicos radiólogos, distribuidos en una jornada de 12 horas, comprendida entre las 6:00 am a 6:00 pm dando como resultado una reducción de la capacidad de atención a emergencias ocurridas en horas nocturnas. La atención de una emergencia puede requerir una TAC cerebral para determinar de manera rápida si el paciente presenta un cuadro de lesión que ponga en peligro su vida. En el 2014, las cifras de fallecidos en el país por traumatismos intracraneales alcanzó los 365 y en el 2015 se tuvo un saldo de 421. (MINSAL, Sistema Informático de MorbiMortalidad).

El sistema propuesto tendrá la capacidad de clasificar aquellos TACs que presentan anomalías, descartando los de menor riesgo y reduciendo el tiempo de atención. El mayor beneficio del sistema se verá reflejado cuando el hospital no cuente con personal médico que brinde diagnósticos. Sin embargo, también será de apoyo a los médicos radiólogos en turno permitiendo enfocar los recursos en aquellos pacientes que los requieran inmediatamente.

Otro aspecto importante del proyecto propuesto es la investigación y aplicación de las redes neuronales artificiales, puesto que este campo ha tenido poca relevancia en los sectores académicos de El Salvador existiendo un reducido número de trabajos de graduación e investigaciones de esta naturaleza, (Alfaro, 2012, p. 57-88), como ejemplo de ello en la Escuela de Ingeniería de Sistemas Informáticos de la Universidad de El Salvador solo se ha desarrollado un trabajo de graduación referente a esta área y la Universidad Centroamericana José Simeón Cañas solo cuenta con 5 trabajos de graduación.

El desarrollo del proyecto propuesto en este documento, aportará al cúmulo de conocimientos y brindará una base teórica y práctica a futuras generaciones de estudiantes en trabajos de graduación o nuevos cursos impartidos por la Escuela de Ingeniería de Sistemas Informáticos.

1.6. Alcances

El desarrollo del proyecto expuesto se divide en dos fases: Una investigativa, con la que se obtendrán los suficientes conocimientos para la construcción de un sistema informático que brinde apoyo al diagnóstico de TAC cerebrales sin contraste y una de desarrollo de dicho sistema. Este hará descubrimientos de TACs con anomalías sin brindar un diagnóstico final, es decir, detectará cuales tienen patrones fuera de lo normal, pero será el médico radiólogo quien deberá emitir el diagnóstico del paciente.

Previo a la finalización del proyecto, se realizará una presentación en las instalaciones del Hospital Nacional San Rafael con el objetivo de garantizar su satisfacción del producto entregado.

Finalmente el proyecto culminará con el plan de implementación dejando documentados los pasos a seguir para su despliegue en producción.

1.7. Limitantes

Existió dificultad para tener un banco de TACs diagnosticadas y de calidad para desarrollar eficazmente el entrenamiento de la RNA y pruebas de software, fueron entregados algunos archivos por el MINSAL, sin embargo; la cantidad fue limitada.

1.8. Diseño de la investigación

1.8.1. Objetivos de la investigación

1.8.1.1. Objetivo general

- Realizar una investigación sobre TAC cerebrales, tecnologías involucradas en el tratamiento de imágenes médicas y redes neuronales artificiales como apoyo al diagnóstico.

1.8.1.2. Objetivos específicos

- Indagar sobre las tecnologías involucradas en la manipulación de imágenes médicas digitales actualmente aplicadas en el MINSAL.
- Investigar sobre redes neuronales artificiales y sus aplicaciones en la medicina.
- Aplicar las redes neuronales artificiales en pruebas que determinen su eficacia para apoyar al diagnóstico de TAC cerebrales.

1.8.2. Alcances de la investigación

- Obtener el conocimiento necesario para la selección de estudios DICOM para las pruebas.
- Adquirir el criterio para corregir errores y establecer eficacia de las redes neuronales en el diagnóstico de TAC.
- Conocer en qué forma las redes neuronales pueden servir como apoyo al diagnóstico de TAC.

1.8.3. Limitantes de la investigación

- Dificultad para encontrar banco de TACs diagnosticados.
- Conocimiento limitado sobre la especialidad de radiología, la cual tiene nivel de doctorado.

1.8.4. Caracterización de la investigación

La caracterización de la investigación es necesaria para establecer los métodos, es decir, la ruta a seguir en el desarrollo de la investigación. Sin embargo, aunque los distintos tipos de investigación están ampliamente documentados, estos presentan ciertos rasgos de

ambigüedad, pues en realidad no existe una metodología estándar de investigación. Por lo tanto, se define la investigación utilizando múltiples criterios presentados a continuación (los criterios para la selección de los distintos tipos se encuentran en los anexos):

Por su profundidad: Será una investigación exploratoria, ya que el objetivo consiste en examinar un tema poco estudiado (Sierra, 2008), como lo es el uso de redes neuronales artificiales como apoyo al diagnóstico de imágenes médicas. Éste está complementado por el alcance del proyecto, el cual establece el nivel de la investigación.

Por su finalidad: La investigación será aplicada, ya que de acuerdo con Bunge (2008), se caracteriza porque el problema que se va a estudiar se elige siempre con un objetivo final definido, y por estar vinculada con el desarrollo de un software tiene como propósito primordial la resolución de problemas prácticos inmediatos. En este caso, su resultado será el desarrollo de pruebas en redes neuronales, las cuales servirán de base para desarrollar un sistema de apoyo al diagnóstico hecho por los radiólogos del MINSAL sobre TAC cerebrales sin contraste en la etapa posterior.

Por su naturaleza: La investigación entra en dos categorías (Sierra, 2008):

- Documental, se basa en documentos y fuentes bibliográficas de validez irrefutable. En este caso materiales impresos como libros sobre radiología, revistas y folletos; además, web especializadas en temas médicos e información de primera mano de radiólogos experimentados.
- Empírica, aquella que implica trabajar con hechos de experiencia directa, como el estudio y las pruebas de las redes neuronales para diagnosticar TAC.

Según su dimensión temporal: La investigación es de corte transversal, ya que se realiza en un solo lapso de tiempo (Hernández, 2008; Sierra, 2008; Méndez, 2006; Torres y Navarro, 2007) que comprenderá la investigación y el desarrollo del software entre los meses de abril y noviembre de 2016.

1.8.5. Enfoque de la investigación

Se compone de cualitativo y cuantitativo, aunque en el caso de ésta investigación abarca a ambos, por lo que es necesario hacer una aclaración de éstos. El concepto de ambos enfoques se conceptualiza de la siguiente manera:

- Enfoque cualitativo, de acuerdo con Hernández (2008) utiliza la recolección de datos sin medición numérica para describir el desarrollo del proyecto, se orienta a profundizar en casos específicos y no a generalizar. Su preocupación no es preponderantemente medir, sino cualificar y describir el fenómeno a partir de sus rasgos determinantes (Bernal, 2006).
- Enfoque cuantitativo, usa la recolección de datos para probar hipótesis con base en la medición numérica y el análisis estadístico para establecer patrones de comportamiento y probar teorías (Hernández, 2008)

De acuerdo con Malhotta y Grover (1998), en las etapas iniciales es conveniente realizar investigaciones exploratorias o descriptivas de tipo cualitativo que permitan generar el

conocimiento necesario para las etapas más avanzadas de la investigación, con base en métodos cuantitativos.

Debido a lo anterior, se determinó que la investigación se realizará en dos fases: La primera fase con enfoque cualitativo, que consistirá en la recolección de conocimiento, lo cual servirá, fundamentalmente, para identificar bancos de TACs, las características del estándar DICOM para su procesamiento y el uso de RNA, la fase posterior con enfoque cuantitativo, cuyo propósito será medir la eficacia de las redes neuronales para diagnosticar TAC cerebrales sin contraste.

1.8.6. Metodología de investigación

La investigación del proyecto será aplicada por su finalidad, es decir, busca generar un desarrollo tecnológico, el método formal que se debe utilizar es el método tecnológico el cual consta de las siguientes etapas (Bunge, 2008):

1. Elección del campo: El proyecto requiere estudiar sobre TAC cerebrales; las herramientas y tecnologías usadas en las imágenes médicas digitales, como el estándar DICOM, y las redes neuronales artificiales aplicadas al diagnóstico médico.
2. Planteamiento del problema práctico: Se realiza a través de un análisis cualitativo del contexto y de la literatura que permita un acercamiento a la situación problemática.
3. Adquisición de conocimientos, antecedentes necesarios para la consecución de los objetivos del trabajo: Se realiza un análisis del contexto y posteriormente un análisis de la literatura que detalle el sustento teórico del proyecto contenido en el marco teórico.
4. Propuesta: Una serie de factores que inciden en el diagnóstico de TACs cerebrales.
5. Descripción detallada del plan: Aplicación del método sistémico para el desarrollo de un modelo de decisión de Van Gigch (2008).
6. Informe de los resultados obtenidos.

Debido a los distintos campos a investigar, se realizará una integración de métodos de investigación; el principal será el método tecnológico, éste servirá de guía para todo el proceso; secundariamente será utilizado el método para desarrollar modelos de decisión de Van Gigch (2008) para el trabajo empírico.

El modelo de decisión de Van Gigch plantea los siguientes pasos:

1. Encontrar las principales variables y parámetros (que describen el evento o fenómeno).
2. Encontrar la relación más plausible entre las variables.
3. Implantar una relación funcional entre las variables.
4. Hipotetizar la forma de la relación funcional.
5. Validar la hipótesis mediante datos empíricos.
6. Realizar la investigación empírica.

7. Estimar los coeficientes de relación funcional postulada en los pasos 3 y 4, y estructurar un modelo representativo.
8. Utilizar el modelo para evaluar nuevas situaciones, alternativas u opciones.

A continuación se presenta un esquema de la multiplicidad de métodos utilizados en el desarrollo de la investigación y que constituyen en su conjunto la metodología.

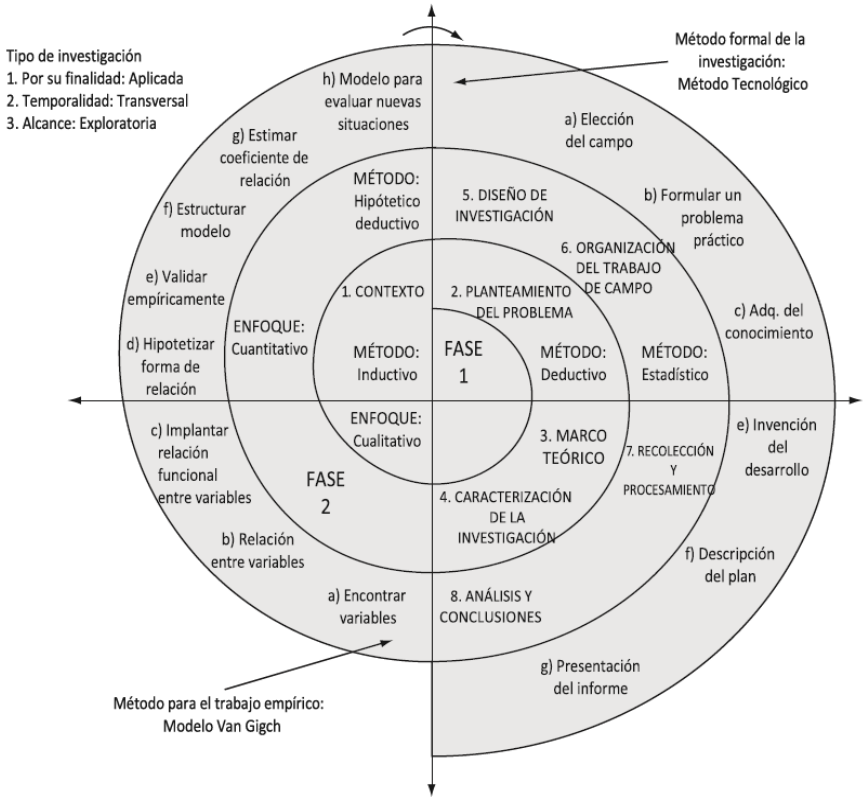


Ilustración 1 - La espiral metodológica

La espiral metodológica representa de manera significativa el proceso de investigación como un todo, que se retroalimenta, considera diferentes momentos, y muestra la forma en que los diferentes métodos se van insertando dentro del proceso conforme la investigación se va desarrollando.

Se presentan diferentes momentos metodológicos en ambas fases del proceso; en la primera fase el método inductivo, se realizarán entrevistas con personal del MINSAL (la DTIC, área de informática y especialistas de radiología); las entrevistas servirán de base para poder clarificar conocimiento y requerimientos del sistema. Posteriormente, se aplicará el método deductivo.

Para la segunda fase, el método de pensamiento será de tipo hipotético-deductivo, y el enfoque de tipo cuantitativo. El método utilizado para el análisis de los datos será el método estadístico; el método de acción para elaborar la propuesta del sistema será el método sistémico para el desarrollo de un modelo de decisión de Van Gigch (2008) que

servirá de guía para la realización de pruebas de la red neuronal, sin embargo, debido a la naturaleza del proyecto no se seguirá a rigor cada uno de los pasos.

1.9. Metodología de desarrollo

Para el desarrollo del sistema informático se utilizará una metodología que contemple el ciclo de vida de desarrollo de software en sus diferentes etapas. A continuación se describe la metodología seleccionada, los criterios utilizados para la selección se encuentran en los anexos.

El **Proceso Racional Unificado** o **RUP** (por sus siglas en inglés de *Rational Unified Process*) es un proceso de desarrollo de software que en conjunto con el Lenguaje Unificado de Modelado (UML) constituye una metodología estándar para el análisis, diseño, implementación y documentación de sistemas orientados a objetos.

En cada una de las fases de RUP se realizan una serie de *artefactos* que sirven para comprender mejor el análisis y diseño, a continuación se listan los artefactos que se espera como resultado de iterar sobre las diferentes fases.

Inicio:

- Documento Visión
- Diagrama de casos de uso
- Especificación de Requisitos
- Diagrama de Requisitos

Elaboración:

- En RUP la arquitectura se organiza en vistas, también conocido como el modelo 4+1 (Kruchten, 1995, p. 45–50). Cada vista tiene asociada una serie de artefactos.

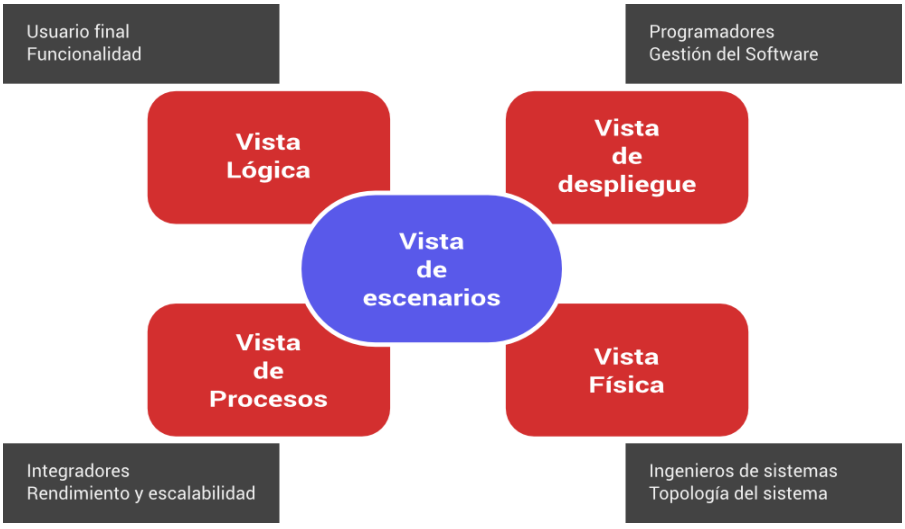


Ilustración 2 - Modelo 4+1 de Kruchten de Kruchten

- Vista Lógica
    - Diagrama de clases
    - Diagrama de comunicación
    - Diagrama de secuencia
    - Modelo E-R (Si el sistema así lo requiere)
- Vista de despliegue
    - Diagrama de componentes
    - Diagrama de paquetes
- Vista de procesos
    - Diagramas de actividad
- Vista física
    - Diagrama de despliegue
- Diseño y desarrollo de casos de uso, o flujos de casos de uso arquitectónicos
- Pruebas de los casos de uso desarrollados, que demuestran que la arquitectura documentada responde adecuadamente a requerimientos funcionales y no funcionales.

Construcción:

- Especificación de requisitos faltantes
- Diseño y desarrollo de casos de uso o flujos de acuerdo con la planeación iterativa
- Pruebas de los casos de uso desarrollados, y pruebas de regresión según sea el caso

Transición:

- Pruebas finales de aceptación
- Puesta en producción
- Estabilización

Todos los conceptos anteriores como los flujos de trabajo, fases e iteraciones se pueden simplificar en el siguiente *Diagrama de esfuerzo en actividades según la fase del proyecto.*

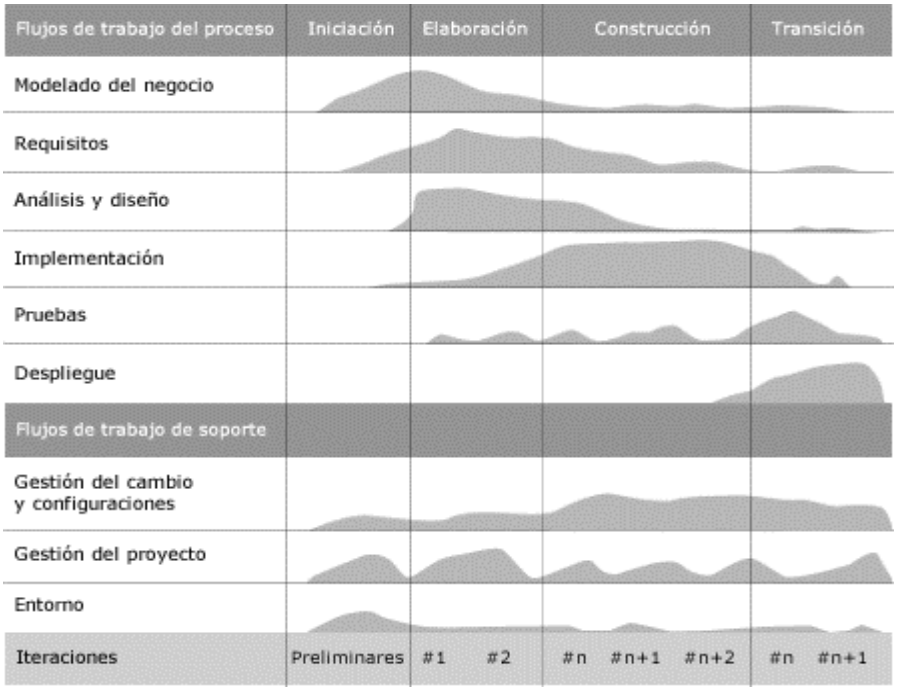


Ilustración 3 - Diagrama de esfuerzo en actividades por fase

1.10. Costeo del proyecto

1.10.1. Modelo de estimación de costos COCOMO

Define tres modelos de estimación que corresponden a las siguientes etapas:

- Etapa 1 (Modelo de Composición de Aplicaciones): Apoya la estimación en etapas tempranas del desarrollo del software donde se conoce poco sobre los requerimientos del mismo.
- Etapa 2 (Modelo de Diseño Temprano): Apoya la estimación tras la definición de requerimientos.
- Etapa 3 (Modelo Post-Arquitectura): Apoya la estimación durante el desarrollo del sistema.

Debido al avance del proyecto se utilizara el Modelo Post-Arquitectura (Etapa 3) para la estimación del costo.

Modelo Post-Arquitectura.

Descripción

Este modelo de estimación se aplica cuando la arquitectura del proyecto esta completamente definida. Suele aplicarse durante el desarrollo y mantenimientos de software incluidos en otros sistemas integrados. COCOMO 2 propone que se pueden identificar elementos de alto nivel que permitan estimar el tamaño del software, para ello

se sugiere estimar el tamaño del producto como Puntos Objeto, que capturan el tamaño en términos de informes, ventanas y componentes de lenguajes de tercera generación.

Primero debe obtenerse el esfuerzo nominal, para el cual los puntos objeto son su entrada de datos, estos se dividen con un ratio de productividad. El esfuerzo nominal calculado se ajusta usando 17 factores multiplicadores de esfuerzo y se divide entre 15, este ultimo paso se ha agregado como un normalizador del esfuerzo.

Estimación de esfuerzo

En esta etapa, el esfuerzo se define por la siguiente ecuación;

$$E_{estimado} = E_{nominal} \times \frac{\sum_{i=1}^{17} EM_i}{15}$$

$$E_{nominal} = \frac{NOP}{PROD}$$

Donde:

- E (estimado): Esfuerzo estimado en personas-mes
- EMi:Factores de costo
- E (nominal): Esfuerzo nominal en personas-mes
- NOP: Nuevos Puntos Objeto
- PROD: razón de productividad

Estimación de NOP

Paso 1: Identificación de tipos de objetos

Lista de ventanas, informes y componentes de tercera generación que abarca la aplicación.

Ventanas:

- Obtener propuesta de diagnóstico.
- Enviar confirmación/denegación de propuesta diagnóstico.

Informes:

- Reporte estadístico de propuestas de diagnóstico.
- Reporte de propuesta de diagnóstico.
- Reporte de falsos positivos/ falsos negativos.

Componentes 3GL:

- Procesamiento de los archivos DICOM.
- Mecanismo de entrenamiento de red neuronal.
- Red neuronal para procesamiento de TACS.

Paso 2: Clasificación de cada objeto según su complejidad

Es necesario etiquetar cada objeto (ventana, informe y componente 3GL) con dificultades simple, media y difícil, para ello se deben tomar en cuenta las siguientes tablas de referencia en el caso de ventanas e informes; en el caso de componentes 3GL se consideran siempre como difícil.

Ventanas			
Número de vistas contenidas	**Número y fuentes de tablas de datos**		
	Total < 4 (<2 srvr, <3 clnt)	Total < 8 (2/3 srvr, 3-5 clnt)	Total >8 (>3 srvr, >5 clnt)
<3	Simple	Simple	Media
3-7	Simple	Media	Difícil
>8	Media	Difícil	Difícil

Tabla 1 - Resumen de la dificultad de las ventanas según tablas de datos

Informes			
Número de vistas Contenidas	**Número y fuentes de tablas de datos**		
	Total < 4 (<2 srvr, <3 clnt)	Total < 8 (2/3 srvr, 3-5 clnt)	Total >8 (>3 srvr, >5 clnt)
0-1	Simple	Simple	Media
2-3	Simple	Media	Difícil
>4	Media	Difícil	Difícil

Tabla 2 - Resumen de la dificultad de los informes según tablas de datos

Paso 3: Cálculo de Puntos Objeto.

Se suman la cantidad de tipos de objetos agrupados por complejidad y luego se multiplican por su respectivo peso.

Tipo Objeto	Complejidad	Peso	Cantidad	Subtotal
Ventana	Simple	1	2	2
	Media	2	0	0
	Difícil	3	0	0
Informe	Simple	2	1	2
	Media	5	1	5
	Difícil	8	1	8
Componente 3GL	Difícil	10	3	30
Total Puntos Objeto (OP)				**47**

Tabla 3 - Cálculo de puntos objeto

Paso 4: Calculo NOP

El NOP se obtiene para estimar el esfuerzo, para su cálculo se toma en cuenta la reutilización de componentes; la ecuación para obtenerlo es la siguiente:

$$NOP = OP * \frac{(100 - \%\, reuso)}{100}$$

Donde:

- OP: Puntos Objetos (sumatoria del cuadro anterior)
- % reúso: porcentaje de objetos de proyectos previos a ser reutilizados en el proyecto actual.

En este caso %reúso es cero, dado que no existen proyectos anteriores de los cuales se pueda reutilizar objetos.

Estimación de PROD

La productividad (PROD), se obtiene a partir de la siguiente tabla:

Experiencia de los desarrolladores	Muy Baja	Baja	Nominal	Alta	Muy Alta
PROD	4	7	13	25	50

Tabla 4 - Estimación de la productividad

Considerando la complejidad que conlleva el desarrollo de un sistema con un componente como una red neuronal artificial, del cual el equipo no posee conocimientos, se concluye que la experiencia de los desarrolladores es baja.

Cálculo de esfuerzo nominal

$$E = NOP / PROD = 47/7 = \mathbf{6.71}$$

El valor del esfuerzo *E* equivale a la cantidad de personas que se necesitan para desarrollar el proyecto; en este caso son 7 personas al mes.

Esfuerzo estimado

Multiplicadores de costo

Factores de costo

- **RELY:** Confiabilidad requerida. Este factor mide la confiabilidad del producto de software a ser desarrollado, esto es, que el producto cumpla satisfactoriamente con la función que debe realizar y respete el tiempo de ejecución que se fijó para el mismo.

- **DATA:** Tamaño de la base de datos El esfuerzo requerido para desarrollar un producto de software está relacionado con el tamaño de la base de datos asociada. Un ejemplo que marca la importancia de esta influencia es el esfuerzo que insume la preparación de los lotes de prueba que se usan en el testeo del producto. El valor de DATA se determina calculando la relación entre el tamaño de la base de datos y el tamaño del programa.

- **CPLX:** Complejidad del producto CPLX analiza la complejidad de las operaciones empleadas en el producto, clasificadas en operaciones: de control, computacionales, dependientes de los dispositivos, de administración de datos y de administración de interfaz de usuario. El nivel que adopta este factor es el promedio del nivel de cada una de las cinco áreas o tipo de operaciones involucradas

- **RUSE:** Requerimientos de reusabilidad, considera el esfuerzo adicional necesario para construir componentes que puedan ser reusadas dentro de un mismo proyecto o en futuros desarrollos. El incremento del esfuerzo se debe a que se incorporan tareas inherentes al reuso, tales como: creación de diseños genéricos de software, elaboración de mayor cantidad de documentación, testeo intensivo para asegurar que las componentes estén debidamente depuradas, etc.

- **DOCU:** Documentación acorde a las diferentas etapas del ciclo de vida Varios modelos de costo de software tienen un factor de costo para representar el nivel de documentación requerida.

- **PVOL:** Volatilidad de la plataforma. Este factor se usa para representar la frecuencia de los cambios en la plataforma subyacente.

- **STOR:** Restricción del almacenamiento principal. Este factor es una función que representa el grado de restricción del almacenamiento principal impuesto sobre un sistema de software.

- **TIME:** Restricción del tiempo de ejecución. Este factor representa el grado de restricción de tiempo de ejecución impuesta sobre el sistema de software. El valor de TIME está expresado en términos de porcentaje de tiempo de ejecución disponible que usará el sistema.

- **ACAP:** Capacidad del analista Se entiende por analista a la persona que trabaja con los requerimientos, en el diseño global y en el diseño detallado. Los principales atributos que deberían considerarse en un analista son la habilidad para el diseño, el análisis, la correcta comunicación y cooperación entre sus pares. En este análisis no se tiene en cuenta el nivel de experiencia.

- **PCAP:** Capacidad del programador Las tendencias actuales siguen enfatizando la importancia de la capacidad de los analistas. Sin embargo, debido a que la productividad se ve afectada notablemente por la habilidad del programador en el uso de las herramientas actuales, existe una tendencia a darle mayor importancia a la capacidad del programador. También se evalúa la capacidad de los programadores para el trabajo en equipo más que para el trabajo individual, resaltando las aptitudes para comunicarse y cooperar mutuamente.

- **PCON:** Continuidad del personal Este factor mide el grado de permanencia anual del personal afectado a un proyecto de software. Los posibles valores que puede adoptar PCON van desde 48% (muy bajo) al 3% (muy alto).

- **AEXP:** Experiencia en la aplicación Este factor mide el nivel de experiencia del equipo de desarrollo en aplicaciones similares.

- **PEXP:** Experiencia en la plataforma COCOMO afirma que existe gran influencia de este factor en la productividad. Reconociendo así la importancia del conocimiento de nuevas y potentes plataformas, interfases gráficas, base de datos, redes, etc.

- **LTEX:** Experiencia en el lenguaje y las herramientas Este factor mide el nivel de experiencia del equipo en el uso del lenguaje y herramientas a emplear. El desarrollo de software, hoy en día, incluye el uso de herramientas que soportan tareas tales como representación de análisis y diseño, administración de la configuración, extracción de documentación, administración de librerías, y chequeos de consistencia. Es por ello que, no sólo es importante la experiencia en el manejo del lenguaje de programación sino también en el uso de estas herramientas, ya que influye notablemente en el tiempo de desarrollo.

- **TOOL:** Uso de herramientas de software Las herramientas de software se han incrementado significativamente desde la década del 70. El tipo de herramientas abarca desde las que permiten editar y codificar hasta las que posibilitan una administración integral del desarrollo en todas sus etapas.

- **SITE:** Desarrollo multisitio La determinación de este factor de costo involucra la evaluación y promedio de dos factores, ubicación espacial (disposición del equipo de trabajo) y comunicación (soporte de comunicación).

- **SCED:** Cronograma requerido para el desarrollo Este factor mide la restricción en los plazos de tiempo impuesta al equipo de trabajo. Los valores se definen como un porcentaje de extensión o aceleración de plazos con respecto al valor nominal. Acelerar los plazos produce más esfuerzo en las últimas etapas del desarrollo, en las que 53 se acumulan más temas a determinar por la escacez de tiempo para resolverlos tempranamente. Por el contrario una relajación de los plazos produce mayor esfuerzo en las etapas tempranas donde se destina más tiempo para las tareas de planificación, especificación, validación cuidadosa y profunda.

Tabla de factores de costo modelo post arquitectura

	Factor	Muy Bajo	Bajo	Normal	Alto	Muy Alto	Extra
Producto	RELY	Inconvenientes insignificantes, que afectan solamente a los desarrolladores	Mínimas pérdidas al usuario, fácilmente recuperables	Pérdidas moderadas al usuario recuperables sin grandes inconvenientes	Pérdida financiera elevada o inconveniente humano masivo	Vida humana en riesgo	
	DATA		DB bytes/Pgm SLOC <10	10<=D/P<100	100<=D/P<1000	D/P >0 1000	
	CPLX						
	RUSE		Ningún componente reusable	Reusable dentro del mismo proyecto	Reusable dentro de un mismo programa	Reusable dentro de una misma línea de productos	Reusable dentro de múltiples líneas de producto
	DOCU	Muchas necesidades del ciclo de vida sin cubrir	Algunas necesidades del ciclo de vida sin cubrir	Necesidades del ciclo de vida cubiertas en su justa medida	Necesidades del ciclo de vida cubiertas ampliamente	Necesidades del ciclo de vida cubiertas excesivamente	
Plataforma	TIME			Uso de <= 50% del tiempo de ejecución disponible	70%	85%	95%
	STOR			Uso de <= 50% del porcentaje total de almacenamiento	70%	85%	95%
	PVOL		Un cambio principal cada 12 meses. Un cambio menor todos los meses	Cambio principal cada 6 meses. Cambio menor cada 2 semanas	Cambio principal cada 2 meses Cambio menor uno por semana	Cambio principal cada 2 semanas. Cambio menor cada 2 días	
Personal	ACAP	15 percentil	35 percentil	55 percentil	75 percentil	90 percentil	
	PCAP	15 percentil	35 percentil	55 percentil	75 percentil	90 percentil	
	PCON	48 % por año	24 % por año	12 % por año	6% por año	3 % por año	
	AEXP	<= 2 meses	<= 6 meses	1 año	3 años	6 años	
	PEXP	<= 2 meses	<= 6 meses	1 año	3 años	6 años	
	LTEX	<= 2 meses	<= 6 meses	1 año	3 años	6 años	
Proyecto	TOOL	Herramientas que permiten editar, codificar, depurar	Herramientas simples con escasa integración al proceso de desarrollo	Herramientas básicas, integradas moderadamente	Herramientas robustas y maduras, integradas moderadamente	Herramientas altamente integradas a los procesos, métodos y reuso	
	SITE Ubicación Espacial	Internacional	Multi-ciudad y multi-compañía	Multi-ciudad o multi-compañía	Misma ciudad o área metropolitana	Mismo Edificio o complejo	Completa-mente Centralizado
	SITE Comuni-cación	Algún teléfono, mail	Teléfonos individuales, FAX	Email de banda angosta	Comunicaciones electrónicas de banda ancha	Comunicaciones electrónicas de banda ancha, ocasionalmente videoconferencia	Multimedia Interactiva
	SCED	75% del nominal	85% del nominal	100% del nominal	130% del nominal	160% del nominal	

5 - Factores de costo Modelo Post-Arquitectura.

Tabla de pesos de factores de costo

Factor	Muy Bajo	Bajo	Normal	Alto	Muy Alto	Extra
RELY	0.82	0.92	1	1.1	1.26	X
DATA	X	0.90	1	1.14	1.28	X
DOCU	0.81	0.91	1	1.11	1.23	X
CPLX	0.73	0.87	1	1.17	1.34	1.74
RUSE	X	0.95	1	1.07	1.15	1.24
TIME	X	X	1	1.11	1.29	1.63
STOR	X	X	1	1.05	1.17	1.46
PVOL	X	0.87	1	1.15	1.30	X
ACAP	1.42	1.19	1	0.85	0.71	X
AEXP	1.22	1.10	1	0.88	0.81	X
PCAP	1.34	1.15	1	0.88	0.76	X
PEXP	1.19	1.09	1	0.91	0.65	X
LTEX	1.20	1.09	1	0.91	0.84	X
PCON	1.29	1.12	1	0.90	0.82	X
TOOL	1.17	1.09	1	0.90	0.78	X
SCED	1.43	1.14	1	1.00	1.00	X
SITE	1.22	1.09	1	0.93	0.86	0.80

6 - Tabla de pesos de factores de costo

Cálculo de costo estimado

Factores de costo aplicados

Factor	Caracterizacion	Valor
RELY	Bajo	0.92
DATA	Bajo	0.90
DOCU	Alto	1.11
CPLX	Extremo	1.74
RUSE	Bajo	0.95
TIME	Normal	1
STOR	Normal	1
PVOL	Bajo	0.87
ACAP	Alta	0.85
AEXP	Muy Baja	1.22
PCAP	Alta	0.88
PEXP	Normal	1
LTEX	Baja	1.09
PCON	Normal	1
TOOL	Alta	0.90
SCED	Alta	1
SITE	Alta	0.93
TOTAL		17.36

7 - Factores de costo aplicados

Esfuerzo estimado

E(estimado) = E(nominal) * Total factores/17 = 6.71 * 17.36/15 = 7.76

Se estima un total de 7.76 ~= 8 personas-mes.

1.10.2. Estimación de costos

A partir de las estimaciones realizadas con el modelo COCOMO se obtuvo el valor del esfuerzo (medido en personas/mes). Para desarrollar el proyecto se cuenta con 1 administrador del proyecto y 4 programadores, con el objetivo de obtener los costos se muestra la siguiente tabla que presenta salarios por hora.

Cargo	Cantidad	Costo por hora *	Costo hora extra**
Administrador de proyecto / Programador	1	$ 8.13	$ 10.16
Programador / Analistas	4	$ 6.25	$ 7.81

Tabla 8 - Tarifa por hora de la mano de obra

* Calculado sobre el sueldo base de $1,000 para el cargo de programador y $1,300 para el cargo de administrador de proyecto.

** Convenios de la OIT sobre trabajo, Pago regular y protección del salario: Convenios 95 (1949) y 117 (1962), Horas extras: Convenio 01 (1919).

Debido a la baja experiencia de los desarrolladores en el campo de las redes neuronales, se ha considerado una jornada laboral de 8 horas por día, es decir; 160 horas mensuales considerando sábado y domingo como días de descanso; debido a la estimación obtenida con el método COCOMO II, se necesitan 8 personas para desarrollar el proyecto, esto implica que hay un excedente de 480 horas mensuales que deben ser distribuidas en el equipo. Cada miembro tendrá una carga de trabajo adicional de 96 horas mensuales, que equivalen a 24 horas semanales; esto modifica la jornada a 11 horas diarias de lunes a viernes y 9 horas el sábado, garantizando el día domingo para descanso. La siguiente tabla resume el costo total del proyecto tomando en cuenta las consideraciones anteriores.

Cargo	Horas / mes	Subtotal Horas /mes	Horas extra /mes	Subtotal Horas extra/mes	Cantidad	Subtotal ($)
Administrador de proyecto / Programador	160	$ 1300	96	$ 975.36	1	$ 2,275.36
Programador / Analistas	160	$ 1000	96	$ 749.46	4	$ 6,997.84
Subtotal/mes						$ 9,273.20
Total (Duración 6 meses)						$ 55,639.20

Tabla 9 - Estimación de los costos de mano de obra

Se estima que el proyecto tendrá un costo total de $ 55,639.20

1.11. Resultados esperados

- Investigación teórico-práctica sobre RNAs y su aplicabilidad en el procesamiento, reconocimiento y clasificación de imágenes médicas.
- Desarrollo de una RNA que tenga la capacidad de poder clasificar aquellos TACs que presentan anomalías.
- Desarrollo de un sistema informático que gestione el análisis digital de TACs y que funcione como interfaz de comunicación entre el médico especialista, el componente de la RNA y los sistemas informáticos del MINSAL.

1.12. Planificación del proyecto

1.12.1. Cronograma de actividades y evaluaciones

Nombre de tarea	Duración	Comienzo	Fin
Trabajo de graduación	196 días	lun 23/05/16	lun 20/02/17
Primera Entrega	73 días	lun 23/05/16	jue 01/09/16
Investigación previa al desarrollo	20 días	lun 23/05/16	vie 17/06/16
Fase Cualitativa	10 días	lun 23/05/16	vie 03/06/16
Recolección de información	10 días	lun 23/05/16	vie 03/06/16
Análisis de información	10 días	lun 23/05/16	vie 03/06/16
Fase Cuantitativa	10 días	lun 06/06/16	vie 17/06/16
Experimentación y pruebas con datos obtenidos	5 días	lun 06/06/16	vie 10/06/16
Elaboración de informe de resultados	10 días	lun 06/06/16	vie 17/06/16
Fase de Iniciación	20 días	lun 23/05/16	vie 17/06/16
Análisis	15 días	lun 23/05/16	vie 10/06/16
Obtención de requerimientos de arquitectura del sistema	15 días	lun 23/05/16	vie 10/06/16
Obtención de macro requerimientos	15 días	lun 23/05/16	vie 10/06/16
Diseño	15 días	lun 30/05/16	vie 17/06/16
Diseño de arquitectura	15 días	lun 30/05/16	vie 17/06/16
Fase de Elaboración	52 días	lun 20/06/16	mar 30/08/16
Análisis	18 días	lun 20/06/16	mié 13/07/16
Obtención de requerimientos	18 días	lun 20/06/16	mié 13/07/16
Validación de requerimientos	10 días	jue 30/06/16	mié 13/07/16
Diseño	41 días	lun 20/06/16	lun 15/08/16
Diseño de arquitectura	15 días	lun 20/06/16	vie 08/07/16
Especificaciones de diseño (estándares de programación)	20 días	lun 20/06/16	vie 15/07/16
Diseño de interfaces	15 días	lun 11/07/16	vie 29/07/16
Diseño de datos	20 días	lun 11/07/16	vie 05/08/16
Diseño de algoritmos de la red neuronal	20 días	mar 19/07/16	lun 15/08/16
Desarrollo (Prototipo 1)	25 días	mar 26/07/16	lun 29/08/16
Pruebas	20 días	mié 03/08/16	mar 30/08/16
Segunda Defensa	0 días	jue 01/09/16	jue 01/09/16
Segunda Entrega	65 días	jue 01/09/16	jue 01/12/16
Fase de Construcción	65 días	jue 01/09/16	jue 01/12/16

Análisis y Diseño	40 días	jue 01/09/16	mié 26/10/16
Correcciones de primera Entrega	39 días	vie 02/09/16	mié 26/10/16
Nuevos requerimientos	31 días	jue 01/09/16	jue 13/10/16
Validación de requerimientos	2 días	vie 14/10/16	lun 17/10/16
Entrega documentación de análisis y diseño	0 días	mié 26/10/16	mié 26/10/16
Desarrollo	65 días	jue 01/09/16	mié 30/11/16
Pruebas Unitarias	65 días	jue 01/09/16	mié 30/11/16
Pruebas de integración de componentes	65 días	jue 01/09/16	mié 30/11/16
Tercera defensa	0 días	jue 01/12/16	jue 01/12/16
Tercera Entrega	41 días	jue 01/12/16	jue 26/01/17
Fase de Transición	38 días	jue 01/12/16	lun 23/01/17
Correcciones de segunda entrega	20 días	jue 01/12/16	mié 28/12/16
Análisis y Diseño	15 días	jue 01/12/16	mié 21/12/16
Desarrollo	23 días	jue 22/12/16	lun 23/01/17
Pruebas	23 días	jue 22/12/16	lun 23/01/17
Despliegue en entorno de pruebas	15 días	mar 03/01/17	lun 23/01/17
Defensa pública	0 días	lun 06/02/17	lun 06/02/17
Preparación de tomos finales	10 días	mar 07/02/17	lun 20/02/17

Tabla 10 - Cronograma de actividades del proyecto

1.12.2. Planificación de recursos

1.12.2.1. Listado de recursos

Recurso humano

Cargo	Cantidad
Administrador de proyecto / Programador	1
Programador / Analistas	4

Tabla 11 - Perfiles requeridos en el proyecto

Recurso Tecnológico

Requerimientos técnicos para el desarrollo:

SOFTWARE

CAFFE Deep Learning Framework:

Pre Requisitos de software:

1. **CUDA (Desarrollado por NVIDIA) para modo GPU versión 7 o superior.**

CUDA es una arquitectura de cálculo paralelo de NVIDIA que aprovecha la gran potencia de la GPU (unidad de procesamiento gráfico) para proporcionar un incremento extraordinario del rendimiento del sistema. Desarrolladores, científicos e investigadores están encontrando innumerables aplicaciones prácticas para esta tecnología en campos como el procesamiento de vídeo e imágenes, la biología y la química computacional, la

simulación de la dinámica de fluidos, la reconstrucción de imágenes de tomografías computarizadas (TC), el análisis sísmico o el trazado de rayos, entre otras. (nvidia, 2016).

2. **BLAS (Basic Linear Algebra Subprograms) via atlas, mkl o OpenBlas.**

BLAS es un estándar que establece un conjunto de rutinas de bajo nivel para realizar tareas de álgebra lineal. Posee diversas implementaciones entre las que se pueden mencionar ATLAS, INTEL MKL y OpenBLAS. (University of Tennessee System, 2016)

3. **Boost versión 1.5 o mayor.**

Boost es un conjunto de librerías escritas en C++.

4. **Protobuf última versión estable.**

Protocol buffers consisten en un conjunto de mecanismos para estructurar datos, escrito por google. (Google Inc, 2016)

5. **Glog última versión estable.**

Implementación en c++ del módulo de logging de google. (Google Inc, 2016)

6. **Gflags última versión estable.**

Gflags: Biblioteca escrita en c++ que implementa el procesamiento de comandos.

7. **Hdf5 última versión estable.**

Paquete python para el manejo del formato hdf5.

8. **Python 2.7 o 3.3+.**
9. **cuDNN Caffe última versión estable.**

Gestor de Bases de Datos:

PostgreSQL 9.5.2 o superior

Servidor PACS DCM4CHE:

Requerimientos de software mínimos:

- JDK 1.4.2
- Sun's Java Advanced Imaging (JAI) Image I/O Tools 1.0_01
- JBOSS
- Visor de contenido.

Requerimientos recomendados:

- JDK 5+
- Sun's Java Advanced Imaging (JAI) Image I/O Tools última versión estable
- JBOSS última versión estable
- Visor de contenido en su última versión estable.

HARDWARE:

Equipo	Cantidad	Requisitos Mínimos	Requisitos Recomendados	Equipo disponible
Servidor	1	• RAM 4GB. • Core i3 de 2.4 GHz. • Disco Duro 2 TB. • Tarjeta de red integrada.	• RAM 8gb. • Core i5 de 2.4 GHz. • Disco duro 2 TB. • Tarjeta de red integrada. • Tarjeta de video dedicada.	• Ram 8GB 1600mhz • Core i5 4690 3.5 Ghz 4 núcleos • SSD 256GB • HDD 2TB • Tarjeta de red integrada • Nvidia gtx760 2gb
PC para desarrollo y documentación	5	• RAM 4gb. • Core i5 de 2.4ghz o equivalente. • HDD de 500gb. • Tarjeta de red integrada.	• RAM 8gb • Core i5 3.5ghz • Disco duro de 1 tb • Tarjeta de red integrada	EQUIPO 1 • Ram 4gb • Core i5 3.3Ghz 4 nucleos • HDD 1TB • Tarjeta de red integrada • Nvidia gtx480 EQUIPO 2 • Ram 8gb • Core i7 2.7Ghz 8 núcleos • HDD 750GB • Tarjeta de red integrada. EQUIPO 3 • Ram 4GB • AMD Radeon 1.67GHz 2 nucleos • Tarjeta de red integrada • HDD 500GB EQUIPO 4 • Ram 6GB • Core i5 2.5GHz • Tarjeta de red integrada • HDD 500 GB EQUIPO 5 • RAM 4GB

				• Core i5 2.5 GHZ • Tarjeta de red integrada • HDD 500GB
Switch	1	• 8 puertos ethernet 10/100mbps • No gestionable	• 8 puertos ethernet 10/100mbps • Gestionable	• 8 puertos ethernet 10/100mbps • No gestionable

Tabla 12 - Listado de hardware requerido en el proyecto

OTRAS HERRAMIENTAS DE DESARROLLO:

- GIT para control de versiones
- PyCharm como IDE
- LibreOffice como software ofimático.
- Weasis integrado a DCM4CHE como visor de imágenes en formato DICOM.

RECURSO HUMANO:

- Un (1) líder de proyecto con experiencia deseable en RUP.
- Cuatro (4) analistas programadores con experiencia deseable en rup, conocimientos de redes neuronales, formato DICOM para el manejo de imágenes radiológicas, conceptos generales acerca de TACs cerebrales.

CRITERIO DE ELECCIÓN DE TECNOLOGÍAS A USAR EN EL DESARROLLO:

En general un criterio fuerte de elección de tecnologías a usar es que fuesen libres, como en el caso de LibreOffice vrs MS Office, otro criterio de peso fue la experiencia previa del equipo con determinadas tecnologías como por ejemplo la elección de Caffe como framework de desarrollo por estar basado en Python, un lenguaje con el equipo posee experiencia y finalmente, algunas herramientas como el caso de DCM4CHE para el servidor PACS sobre las cuales no se tuvo libertad de elección por ser las que usa la contraparte.

Requerimientos técnicos para pruebas:

SOFTWARE:

Red neuronal desarrollada con framework Caffe.

Gestor de Bases de datos:

PostgreSQL 9.5.2 o superior

Servidor PACS DCM4CHE:

Requerimientos de software mínimos:

- JDK 1.4.2
- Sun’s Java Advanced Imaging (JAI) Image I/O Tools 1.0_01
- JBOSS
- Weasis como visor de contenido.

Requerimientos recomendados:

- JDK 5+
- Sun's Java Advanced Imaging (JAI) Image I/O Tools última versión estable
- JBOSS última versión estable
- Weasis en su última versión estable

HARDWARE:

Equipo	Cantidad	Requisitos Mínimos	Requisitos Recomendados	Equipo disponible
Servidor	1	• RAM 4GB. • Core i3 de 2.4 GHz. • Disco Duro 2 TB. • Tarjeta de red integrada.	• RAM 8gb. • Core i5 de 2.4 GHz. • Disco duro 2 TB. • Tarjeta de red integrada. • Tarjeta de video dedicada.	• Ram 8GB 1600mhz • Core i5 4690 3.5 Ghz 4 núcleos • SSD 256GB • HDD 2TB • Tarjeta de red integrada • Nvidia gtx760 2gb
PC para realizar pruebas	3	• RAM 4gb. • Core i5 de 2.4ghz. • Disco duro de 500GB. • Tarjeta de red integrada.	• RAM 8gb • Core i5 3.5ghz • Disco duro de 2 tb • Tarjeta de red integrada	EQUIPO 1 • Ram 4gb • Core i5 3.3Ghz 4 nucleos • HDD 1TB • Tarjeta de red integrada • Nvidia gtx480 EQUIPO 2 • Ram 8gb • Core i7 2.7Ghz 8 núcleos • HDD 750GB • Tarjeta de red integrada. EQUIPO 3 • Ram 4GB • AMD Radeon 1.67GHz 2 nucleos • Tarjeta de red integrada • HDD 500GB EQUIPO 4

				• Ram 6GB • Core i5 2.5GHz • Tarjeta de red integrada • HDD 500 GB EQUIPO 5 • RAM 4GB • Core i5 2.5 GHZ • Tarjeta de red integrada • HDD 500GB
Switch	1	• 8 puertos ethernet 10/100mbps • No gestionable	• 8 puertos ethernet 10/100mbps • Gestionable	• 8 puertos ethernet 10/100mbps • No gestionable

Tabla 13 - Listado de hardware requerido en la fase de pruebas

RECURSO HUMANO:

- Un (1) líder de proyecto con experiencia deseable en RUP.
- Cuatro (4) analistas programadores con experiencia deseable prueba y entrenamiento de redes neuronales.
- Dos (2) médicos especialistas en el área de radiología.

OTROS RECURSOS TÉCNICOS:
Banco de datos de TACs cerebrales con su respectivo diagnóstico para el entrenamiento de la red neuronal.

Requerimientos técnicos para producción:

SOFTWARE:
Red neuronal desarrollada con framework Caffe y entrenada en fases previas de prueba.

Gestor de Bases de datos:

PostgreSQL 9.5.2 o superior

Servidor PACS DCM4CHE:

Requerimientos de software mínimos:

- JDK 1.4.2
- Sun's Java Advanced Imaging (JAI) Image I/O Tools 1.0_01
- JBOSS
- Weasis como visor de contenido.

Requerimientos recomendados:

- JDK 5+
- Sun's Java Advanced Imaging (JAI) Image I/O Tools última versión estable
- JBOSS última versión estable
- Weasis en su última versión estable.

HARDWARE:

Equipo	Cantidad	Requisitos Mínimos	Requisitos Recomendados
Servidor	1	• RAM 8GB. • Core i5 de 2.4 GHz. • Disco Duro 2 TB. • Tarjeta de red integrada.	• RAM 16gb. • Core i7 de 2.4 GHz. • Disco duro 2 TB. • Tarjeta de red integrada. • Tarjeta de video dedicada.
PC en red con acceso a sistema	A determinar	• RAM 2gb. • Dual Core de 2.4ghz. • Disco duro de 200gb. • Tarjeta de red integrada.	• RAM 4gb • Core i3 2.5ghz • Disco duro de 300gb • Tarjeta de red integrada
Estructura de red interna	1	• 10/100mbps	• 10/100mbps

Tabla 14 - Listado de hardware requerido en producción

Actualmente el MINSAL ya cuenta con el recurso tecnológico necesario para la implementación del sistema propuesto, parte de éste ha sido adquirido para la implementación del SIMAGD y el resto forma parte de su infraestructura tecnológica.

1.12.2.2. Asignación de recursos por actividad

Tarea	Recursos Necesarios
Trabajo de graduación	
Investigación previa al desarrollo	
• Recolección de información • Análisis de información • Experimentación y pruebas de información y datos obtenidos • Elaboración de informe y organización de resultados	• 1 Administrador de proyecto • 4 Analistas • 5 Computadoras
Fase de Iniciación	
Análisis y Diseño	
• Obtención de requerimientos de arquitectura del sistema • Obtención de macro requerimientos según médicos radiólogos • Diseño de arquitectura • Diseño de procedimientos	• 1 Administrador de proyecto • 4 Analistas • 5 Computadoras
Fase de Elaboración	
Análisis	
• Obtención de requerimientos • Especificación de requerimientos • Validación de requerimientos	• 1 Administrador de proyecto • 4 Analistas • 5 Computadoras
Diseño	
• Diseño de arquitectura • Especificaciones de diseño (estándares de programación) • Diseño de interfaces • Diseño de datos • Diseño de algoritmos de la red neuronal	• 1 Administrador de proyecto • 4 Analistas • 5 Computadoras
Primera entrega (Segunda Defensa)	
Fase de Construcción	
Análisis y Diseño	
Correcciones de primera Entrega	• 1 Administrador de proyecto • 2 Analistas • 3 Computadoras
Nuevos requerimientos	• 1 Administrador de proyecto • 2 Analistas • 3 Computadoras
• Desarrollo • Pruebas Unitarias • Pruebas de integración de componentes	• 1 Administrador de proyecto • 4 Analistas • 5 Computadoras
Despliegue en entorno de pruebas	• 1 Administrador de proyecto

	• 4 Analistas • 5 Computadoras • 1 Servidor
Segunda Entrega (Tercera defensa)	
Fase de Transición	
• Análisis • Diseño • Desarrollo • Pruebas • Despliegue en entorno de pruebas	• 1 Administrador de proyecto • 4 Analistas • 5 Computadoras • 1 Servidor
Defensa final (publica)	

Tabla 15 - Asignación de recusos por actividad

CAPÍTULO II: RESULTADO DE LA INVESTIGACIÓN

2.1. Marco teórico

2.1.1. Tomografía Axial Computarizada

El término Tomografía Computarizada (TC), se refiere a un procedimiento computarizado de imágenes por rayos X en el que se proyecta un haz angosto de rayos X a un paciente y se gira alrededor del cuerpo, produciendo señales con intensidades variables que son procesadas por la computadora del tomógrafo para generar imágenes transversales (cortes) del cuerpo. Estos cortes se llaman imágenes tomográficas y contienen información más detallada que los rayos X convencionales. Una vez que se recolectan varios cortes sucesivos, se unen para formar una imagen tridimensional del paciente que facilite la identificación y ubicación de las estructuras básicas y también posibles anomalías presentes.

Con la TC se crean imágenes de cortes transversales de un objeto juntando numerosas imágenes de proyección sobre el objeto en el plano de interés. Una imagen de proyección es una imagen unidimensional en la que el brillo de cada píxel es igual a la absorción de rayos X en la sección del cuerpo u objeto expuesto. Según el plano de orientación existen tres tipos de cortes: axial (barrido en eje Y), coronal (barrido en el eje X) y sagital (barrido en el eje Z).

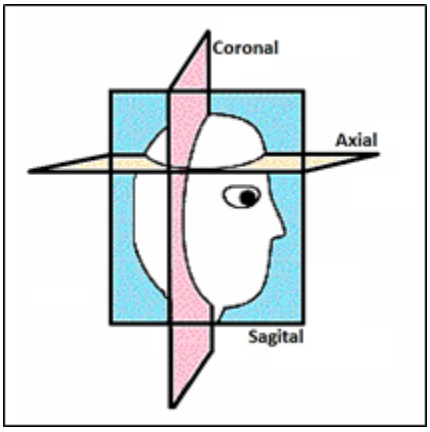


Ilustración 4 - Tipos de cortes según su orientación

Cuando una tomografía computarizada se realiza con orientación axial, el resultante es conocido como Tomografía Axial Computarizada (TAC). (Juan, Aranda, Orallo)

Las TACs de cabeza son un tipo especial de tomografías que son realizadas específicamente a los elementos craneoencefálicos y sirven para detectar:

- Sangraduras, lesiones cerebrales y fracturas de cráneo en pacientes con lesiones en la cabeza.
- Sangraduras causadas por roturas o fisuras de aneurismas en un paciente con dolores de cabeza repentinos.
- Coágulos de sangre.
- Derrames cerebrales.

- Cavidades cerebrales agrandadas (ventrículos) en pacientes con hidrocefalia.
- Enfermedades o malformaciones del cráneo.

En la TAC de cabeza, numerosos haces de rayos X se hacen pasar a través del cráneo y el cerebro en muchos ángulos, y detectores especiales miden la cantidad de radiación absorbida por los diferentes tejidos. El tubo de rayos X gira alrededor del paciente y envía datos desde numerosos ángulos de la cabeza, formando imágenes de secciones transversales (cortes) de la cabeza y el cerebro. Posterior al desarrollo del estudio, un radiólogo o técnico especializado interpreta los resultados del mismo y envía un informe detallado a un médico para emitir un diagnóstico.

Existen distintas generaciones de scanners o tomógrafos que con el tiempo han mejorado la manera en que se realizan las imágenes médicas. Las características principales de estas generaciones se presentan a continuación:

Primera generación	Se realiza una proyección desde un emisor de rayos X hacia un detector en el lado opuesto del gantry (parte del aparato en forma vertical y de anillo). Luego se hace una rotación 1° y se hace una nueva proyección, este proceso se repite hasta llegar a 180°. Las 180 imágenes resultantes se combinan y se obtiene una imagen de corte transversal. El tiempo para la toma de la imagen puede variar entre 4.5 y 5.5 minutos.
Segunda generación	Realiza el mismo proceso de la primera generación, pero con fuentes de rayos X que emiten un haz amplio que abarca muchos detectores que están ubicados en fila recta y al lado opuesto de la fuente (a diferencia de la relación uno a uno de la generación anterior). El tiempo requerido varía entre 20 segundos y 3.5 minutos.
Tercera generación	Se basa en la emisión de rayos X en abanico y permite la rotación de 360°. Los detectores están ubicados en forma curva con un arco entre 30° y 40° permitiendo recibir con más amplitud las proyecciones a través del cuerpo en estudio. El proceso generalmente se realiza en segundos.
Cuarta generación	Sigue la misma estructura de funcionamiento que la tercera generación pero cuenta con una fuente de rayos en abanico con un número de detectores fijo ubicados en forma de anillo, y también de otra fuente de rayos fuera del anillo de detectores.

Tabla 16 - Generaciones de tomógrafos

2.1.1.1. Elementos de equipos de tomografía computarizada

Los elementos o sistemas que componen el equipo de captura de tomografías son:

- El sistema de recolección de datos: gantry.
- El sistema de procesado de datos y de reconstrucción de la imagen: computadora.
- El sistema de visualización y configuración: la consola de control.

2.1.1.2. El gantry

El gantry o carcasa es el cuerpo vertical del tomógrafo que presenta un orificio central en el que se introduce la camilla de exploración con el paciente. Se controla desde la consola y está compuesto por el generador de alta tensión, el tubo de fuente de rayos X, los detectores, los sistemas de adquisición de datos (DAS), los colimadores y por todas las partes mecánicas necesaria para que funcionen los elementos anteriores.

El gantry posee una serie de controles con los que se realiza el ajuste y posicionamiento del paciente, por lo tanto se puede adaptar a la inclinación de la parte corporal bajo estudio. Ciertas regiones anatómicas, como la cabeza y la columna, requieren muchas veces inclinación de la carcasa para obtener una imagen transversal exacta.

El gantry contiene los siguientes elementos:

● Tubo de rayos X (fuente): Dispositivo técnico capaz de producir la radiación ionizante mediante una fuente artificial de alimentación de tipo eléctrico.

● La matriz de detectores: Los detectores miden la energía depositada en ellos después de ser impactados por los fotones de rayos X que han atravesado el cuerpo del paciente. Esta energía la transforman en corriente eléctrica que llegará a la computadora y será cuantificada por un sistema electrónico.

● Sistema de adquisición de datos: Convierte las señales procedentes de los detectores en datos digitales y las transmite a la computadora, ésta es capaz de integrar la información recibida y reconstruir las imágenes de forma casi instantánea.

● Los colimadores: Se usan para diafragmar el haz de rayos X. En general, en la TAC es necesario utilizar la colimación por dos razones:

- Para reducir la dosis que recibe el paciente al disminuir el área de tejido irradiado.
- Para mejorar el contraste de la imagen al disminuir la radiación dispersa.

Todos estos subsistemas se controlan desde la consola y envían datos a la computadora para analizar y generar la imagen.

2.1.1.3. La computadora

Es el soporte técnico que permite llevar a cabo las operaciones de procesado de datos y reconstrucción de la imagen, permitiendo realizar el proceso completo en un corto tiempo. Este es un módulo que está compuesto por tres unidades:

● Unidad de control del sistema: Tiene a su cargo el funcionamiento total del equipo. Su configuración es similar a la de cualquier computadora con su software y hardware asociados.

● Unidad de reconstrucción rápida: Es la encargada de realizar los procedimientos necesarios para la reconstrucción de la imagen a partir de los datos recolectados por el sistema de detección.

● Unidad de almacenamiento de datos e imágenes: Está compuesta por uno o más discos duros donde se realiza el almacenamiento de las imágenes reconstruidas, los parámetros

asociados a la captura de las imágenes y el software de aplicación del tomógrafo. El disco principal de la computadora sólo permite el almacenamiento a corto plazo de las imágenes por lo que los estudios completos deben ser almacenados de forma que sea posible recuperarlos si se considera necesario.

2.1.1.4. La consola de control

Es la responsable de integrar los demás subsistemas y tiene como objetivo programar el estudio que se va a realizar y seleccionar los datos requeridos para la obtención de la imagen. Un estudio de TAC puede realizarse utilizando técnicas de exploración estándar o configurando parámetros que permitan obtener un estudio personalizado. Según el modelo del equipo, se puede contar con una consola destinada para fines técnicos y otra destinada para un médico. La consola técnica cuenta con un monitor de protocolos de estudio en el cual se introducen los datos del paciente, hospital, tipo de estudio a realizar, edad, sexo, etc. Además se cuenta con un monitor de visualización de imágenes. La consola destinada al médico, permite visualizar y modificar valores de brillo, contraste, ampliación de zonas de imagen, etc.

Algunos parámetros configurables en la consola técnica son:

- Intensidad (mA).
- Tensión de pico (Kv).
- Duración de corte.
- Grosor del corte.
- Contraste de la imagen.
- Movimientos de la camilla.
- La ventana.

La ventana se refiere a las escalas de grises o contraste de la imagen. Hace uso de números Hounsfield (en honor de Sir Godfrey Newbold Hounsfield) asociados a los tejidos del cuerpo humano y cuya escala va desde el -1000 (correspondiente al aire) hasta el +1000 (correspondiente al metal) pasando por el 0 que corresponde a la densidad agua, tomada como referencia.

Los valores se dividen en las siguientes características:

- Valores altos: corresponden a una imagen blanca como un hueso por ejemplo.
- Valores bajos: corresponden a una imagen negra como es el caso del aire.
- Valores intermedios: da una imagen con tono de grises con un número máximo de 20 (La visión humana no es capaz de distinguir más).

La pantalla de la consola está dividida en puntos llamados píxeles, que corresponden a una unidad de superficie y tienen una profundidad prefijada con la cual se obtiene una unidad de volumen llamada voxel.

2.1.2. Estándar DICOM y sistema imageneológico

DICOM (Digital Imaging and Communications in Medicine) es el estándar que define los

métodos para la transferencia de imágenes médicas para diagnóstico y la información asociada a ellas, entre equipos de imagenología y sistemas de fabricantes distintos.

El objetivo principal del Comité de Estándares DICOM es crear y mantener estándares internacionales para la comunicación de información biomédica diagnóstica y terapéutica para las disciplinas que utilizan imágenes digitales y datos relacionados. Como resultado, DICOM es utilizado por prácticamente cualquier disciplina médica que utilice imágenes. Esto incluye neurología y neurocirugía, cardiología, endoscopia, mamografía, oftalmología, ortopedia, patología, pediatría, radioterapia, cirugía, odontología y veterinaria.

El estándar DICOM 3.0 (1992) evolucionó de los estándares ACR-NEMA versión 1.0 (1985) y versión 2.0 (1988) desarrollados por ACR (American College of Radiology) y NEMA (National Electrical Manufacturers Association) que en 1983 formaron un comité para desarrollar un estándar que cumpliera los siguientes objetivos:

- Promover la comunicación de imágenes digitales, independientemente del fabricante del equipo.
- Facilitar el desarrollo y expansión de los sistemas de almacenamiento y comunicación de imágenes (PACS) capaces de comunicarse también con otros sistemas de información hospitalaria.
- Permitir la creación de bases de datos de información diagnóstica que pudiesen ser consultadas por una amplia variedad de dispositivos remotos.
- El contenido del estándar DICOM va más allá de la definición del formato de intercambio de imágenes, sus alcances principales son:
- Estructuras de datos (formatos) para imágenes médicas y datos relacionados.
- Servicios orientados a red, como: Transmisión de imágenes, búsqueda de imágenes, impresión y modalidades de integración entre un sistema PACS y un sistema general de información de un hospital (RIS).
- Formatos para intercambio entre medios de almacenamiento.
- Requerimientos de conformidad de los equipos y aplicaciones.

DICOM cubre todas las necesidades de un PACS, las cuales involucran el almacenamiento, transmisión e impresión de imágenes médicas. De esta forma se integran todas las máquinas que forman un PACS, desde los equipos médicos encargados de la obtención de imágenes, hasta las computadoras usados por el personal clínico para visualizar las imágenes.

2.1.2.1. Estructura

En su revisión de 2004 el Estándar DICOM versión 3.1 tiene las siguientes partes:

- PS3.1. Introducción y vista general: La primera parte contiene una panorámica del estándar en sí mismo, con descripción de los principios básicos, como pueden ser los conceptos de SOP Class o Service Class Provider.
- PS3.2. Conformación: Describe la definición de conformación para DICOM, es decir se le solicita a los desarrolladores y vendedores de equipos y sistemas describir claramente cómo es su adhesión al estándar DICOM.
- PS3.3. Definición de los Objetos de Información (IOD): Especifica la estructura y atributos de los objetos que se operan por Clases de Servicio. Estos objetos pueden ser Paciente, Estudio, Serie, Imagen, etc.
- PS3.4. Especificación de las clases de servicios: Se define las funciones que operan sobre los Objetos de información para proporcionar un servicio específico.
- PS3.5. Estructura de datos y codificación: Especifica la codificación de los datos en los mensajes que se intercambian para lograr el funcionamiento de las Clases de Servicio. La principal función es definir el lenguaje que dos aparatos tienen que utilizar en la comunicación.
- PS3.6. Diccionario de datos: Define los atributos de información individuales que representan los datos de todos los IOD definidos en la parte 3.3. También se especifica los valores de algunos de estos atributos.
- PS3.7. Intercambio de Mensajes: Especifica el funcionamiento de los mensajes a intercambiar. Estos mensajes se necesitan para poder utilizar los servicios definidos por las Clases de Servicio (parte 3.4).
- PS3.8. Soporte de las comunicaciones por red para el intercambio de mensajes: Define los servicios y protocolo de intercambio de mensajes (Parte 3.7) directamente en OSI y redes TCP/IP. En el entorno DICOM, el protocolo de comunicación utilizado es el TCP/IP.
- PS3.9. Retirado.
- PS3.10. Medios de almacenamiento y formato de archivos para intercambio de datos: Define los formatos lógicos para guardar la información de DICOM sobre varios medios de comunicación, entre ellos los archivos tratados.
- PS3.11. Perfiles de aplicación para medios de almacenamiento: Define los medios para usuario y vendedores, especificando la sección de medios de comunicación entre los sistemas definidos en la Parte 3.12 y los objetos de información definidos en la parte 3.3.
- PS3.12. Formatos y medios físicos para el intercambio de datos: Las especificaciones de la industria referentes a los medios físicos de almacenamiento o medios de comunicación que estructuran los sistemas de archivos.
- PS3.13. Retirado.
- PS3.14. Estándar para la función de representación de escala de grises: En esta parte se especifica la estandarización de las características de los monitores para la representación en la escala de grises de las imágenes.
- PS3.15. Perfiles de Seguridad: Perfiles de usuario en aplicaciones.
- PS 3.16. Recurso para el mapeo de contenido: Estructuración de documentación de objetos de información DICOM; conjunto de términos codificados para uso en objetos

de información; un léxico de definición de términos para DICOM; traducción de especificaciones del país por medio de código.
- PS3.17. Interpretación de la información: anexos de información y normativa.
- PS3.18. Acceso WEB de objeto persistente DICOM (WADO): Específica la solicitud para el acceso de un objeto persistente DICOM que puede ser expresada en un HTTP URL.

El formato DICOM define cuatro elementos (Jiménez, 2006):

- Preámbulo y prefijo identificativo del archivo.
- Meta-cabecera.
- Cabecera.
- Imagen.

El preámbulo tiene un tamaño fijo de 128 bytes y contiene información sobre el nombre de la aplicación usada para crear el archivo, o información que permita a aplicaciones acceder directamente a los datos de la imagen almacenada en éste. El prefijo consiste en cuatro bytes que contienen una cadena de caracteres DICOM. Esta cadena debe estar codificada siempre con las letras en mayúscula y usando el conjunto de caracteres ISO 8859 G0. El propósito de este prefijo es permitir a las implementaciones diferenciar si un archivo es DICOM o no.

La cabecera y la meta-cabecera de un archivo DICOM consisten en una serie de campos con toda la información necesaria sobre la imágen en cuestión, incluyendo la propia imagen. Al conjunto de toda la información codificada sobre un campo se le conoce con el nombre de Elemento de Datos. Así, tanto la cabecera como la meta-cabecera de un archivo DICOM consisten en una sucesión de elementos de datos.

Un elemento de datos está constituido por los campos:

- Etiqueta del Elemento de Datos: Sirve para identificar cada elemento de datos de forma unívoca. Esta etiqueta está constituida por un número de grupo y un número de elemento. El elemento de datos se suele representar como un vector de dos posiciones, siendo la primera el número de grupo, y la segunda el número de elemento, en hexadecimal con cuatro dígitos. Por ejemplo, si el número de grupo es ocho y el número de elemento es doce, la etiqueta será (0008,000C).
- Representación del Valor: Indica la forma en que se codifica el valor del elemento. Por ejemplo, puede estar codificado como una cadena de caracteres o un entero sin signo.
- Longitud del Valor: Como su nombre indica, es la longitud del campo valor.
- Valor: Codificado según la Representación y Longitud del Valor.

A continuación, se muestra un ejemplo un archivo DICOM con algunos de los elementos que brindan información sobre el estudio.

Metadatos	
(0002|0000) FileMetaInformationGroupLength	192
(0002|0001) FileMetaInformationVersion	00\01
(0002|0002) MediaStorageSOPClassUID	SecondaryCaptureImageStorage[1.2.840.10008.5.1.4.1.1.7]
(0002|0003) MediaStorageSOPInstanceUID	1.3.12.2.1107.5.1.4.29184.30000016060813224128100001029
(0002|0010) TransferSyntaxUID	LittleEndianExplicit[1.2.840.10008.1.2.1]
(0002|0012) ImplementationClassUID	1.3.12.2.1107.
(0002|0013) ImplementationVersionName	SIEMENS_SSV
Identificación de la prueba	
(0008|0005) SpecificCharacterSet	ISO_IR 100
(0008|0008) ImageType	DERIVED\SECONDARY\OTHER\CT_SOMS PROT
(0008|0016) SOPClassUID	1.2.840.10008.5.1.4.1.1.7
(0008|0018) SOPInstanceUID	1.3.12.2.1107.5.1.4.29184.30000016060813224128100001029
(0008|0020) StudyDate	20160608
(0008|0021) SeriesDate	20160608
(0008|0023) ContentDate	20160608
(0008|0030) StudyTime	110031.203000
(0008|0031) SeriesTime	111050.031000
(0008|0033) ContentTime	111050.046000
(0008|0050) AccessionNumber	
(0008|0060) Modality	CT
(0008|0064) ConversionType	WSD
(0008|0070) Manufacturer	SIEMENS
(0008|0080) InstitutionName	HOSPITAL NACIONAL
(0008|0081) InstitutionAddress	Street
(0008|0090) ReferringPhysicianName	
(0008|1010) StationName	CT29184
(0008|1030) StudyDescription	Cabeza^CEREBRO (Adulto)
(0008|103e) SeriesDescription	Protocolo de paciente
(0008|1080) AdmittingDiagnosesDescription	SD VERTIGINOSO
(0008|1090) ManufacturerModelName	Emotion 6 (2007)
Información del paciente	
(0010|0010) PatientName	ERNESTO
(0010|0020) PatientID	35915
(0010|0030) PatientBirthDate	19990608
(0010|0040) PatientSex	M
(0010|1010) PatientAge	017Y
Información sobre la adquisición	
Información de referencias	

Ilustración 5 - Elementos de archivo DICOM

Ilustración 6 - Imagen DICOM

2.1.3 Diagnóstico asistido por computadora

El diagnóstico asistido por computadora (CAD) se define como el proceso de diagnóstico que realiza un radiólogo, teniendo en consideración el resultado de un análisis cuantitativo que hace una computadora sobre imágenes médicas. Según esta definición el sistema informático que denominamos sistema CAD facilita una segunda opinión que permite enfocar la atención del radiólogo sobre ciertas zonas sospechosas de la imagen, mejorando así el rendimiento y la consistencia del diagnóstico y reduciendo el tiempo de lectura de dichas imágenes.

2.1.3.1. Diagnóstico asistido por computadora

Los conceptos de diagnóstico automatizado por computadora y el diagnóstico asistido por computadora están claramente presentes en la actualidad. Una diferencia importante entre ellos es la forma en que se utiliza la salida provista por la computadora en el proceso de diagnóstico. Con CAD, el radiólogo utilizará el resultado como una segunda opinión, son los radiólogos quienes toman las decisiones finales. Por lo tanto, para algunos casos los radiólogos pueden estar de acuerdo o no, con los resultados. Sin embargo, para otros casos en los que el radiólogo tiene alguna duda, se espera que la decisión final se pueda mejorar mediante el uso de estos resultados. Esta mejora es posible, sólo cuando el resultado del equipo es confiable, cuanto mayor sea el rendimiento del equipo, mejor será

el efecto global sobre el diagnóstico final. El nivel de rendimiento del equipo no tiene que ser igual o mayor que el de los radiólogos. Con CAD, la ganancia potencial es debido al efecto sinérgico obtenido mediante la combinación de la capacidad y experiencia del radiólogo y la potencia de la computadora. Debido a estos beneficios multiplicativos, el CAD actualmente ha llegado a ser ampliamente utilizada en situaciones clínicas prácticas. (Kunio, 2007)

Con el diagnóstico automatizado el nivel de rendimiento de los resultados de la computadora está obligado a ser muy alto. Si la precisión para la detección de lesiones es inferior a la precisión media de los médicos, sería difícil justificar el uso de un equipo de diagnóstico automatizado. Además, está presente la dificultad de implementar el diagnóstico automatizado si el número de detecciones de falsos positivos es grande, por ejemplo uno o más falsos positivos por caso, dicha situación implicaría que cada caso debe ser estudiado por un radiólogo y esto ya no cumple con el objetivo del diagnóstico automatizado. (Kunio, 2007).

2.1.3.2. Historia del CAD

A continuación, se presenta una serie de hechos históricos que han permitido el desarrollo del CAD tanto en su filosofía como en su aplicación práctica.

1890 EE.UU - Primeros pasos de la informática médica, se realiza un censo de la población utilizando un sistema de procesamiento de datos con tarjetas perforadas, posteriormente estos programas fueron adaptados para realizar estudios epidemiológicos y de la salud pública.

Década de los 60s - A partir de los años 60 varios grupos de médicos se plantean el uso las computadoras en aplicaciones clínicas, surgiendo de esta forma los primeros sistemas de ayuda al diagnóstico o las primeras historias clínicas automatizadas. Durante esta década también se desarrollaron programas cuyo propósito era la creación de un sistema global de manejo de información hospitalaria, los denominados HIS.

Algunos de los métodos utilizados en esta época, para el apoyo a la toma de decisiones eran los siguientes:

- Modelos matemáticos de procesos fisiológicos.
- Análisis de bases de datos.
- Algoritmos clínicos.
- Métodos de clasificación de patrones.
- Análisis de decisiones.

El éxito que tuvieron los sistemas que aplicaron estos métodos fue limitado, debido a que su base fue puramente matemática y probabilística, era necesario que dichos sistemas realizarán procesos semejantes al razonamiento de los médicos especialistas, es entonces cuando los sistemas basados en técnicas de inteligencia artificial tomaron relevancia como una herramienta para el apoyo en el proceso de diagnóstico.

En esa misma época se realizaron investigaciones sobre los procesos cognitivos involucrados en el razonamiento del médico, estas investigaciones dieron como resultado un conocimiento más detallado sobre los pasos y mecanismos del proceso de diagnóstico

como tal y la forma en que es realizado por los especialistas, paralelamente se estaban desarrollando nuevas técnicas de inteligencia artificial, específicamente en el desarrollo de sistemas expertos.

Principio de los 70s EE.UU - Cuatro universidades de EE.UU. desarrollan sistemas médicos expertos, que no solo serían pioneros en este campo, sino auténticos hitos en la Ingeniería del Conocimiento. Estas universidades eran Standford con MYCIN, Pittsburg con INTERNIST, el Massachusetts Institute of Technology con el PRESENT ILLNESS PROGRAM, y Rutgers con CASNET.

Un problema de la primera generación de sistemas expertos, es que no fueron diseñados para facilitar conocimientos específicos, ni ayudar al médico en el análisis y procesamiento de datos, sino más bien como un reemplazo a los médicos (diagnóstico automatizado por computadora).

Década de los 70s y 80s - Con el desarrollo de los microprocesadores y la introducción de la computadora personal (PC), la informática médica se consolida como disciplina científica independiente, aumentando la cantidad de aplicaciones informáticas disponibles.

Década de los 80s - Surge la segunda generación de sistemas expertos, estos incorporan el concepto de conocimiento profundo, es decir, información añadida sobre las causas y los procesos que provocan una determinada enfermedad, así como nuevas capacidades de razonamiento y diferentes modelos cualitativos, causales, temporales, cuantitativos, etc.

Década de los 90s - Los sistemas médicos expertos desarrollados hasta la fecha no alcanzaron el éxito clínico esperado, aunque sí representaron un gran avance académico y tecnológico.

Surgen nuevos modelos basados en inteligencia artificial, de los cuales destacan las redes neuronales artificiales y el razonamiento basado en casos.

2.1.3.3. Aplicaciones de las RNAs en la medicina

Las redes neuronales artificiales tienen una amplia aplicación en el ámbito médico en general, existen antecedentes de implementaciones exitosas en áreas específicas de la medicina, tales como:

Los sistemas de diagnóstico: Las RNAs se utilizan ampliamente en sistemas de diagnóstico, por lo general para detectar cáncer y problemas cardíacos. Los beneficios del uso de redes neuronales es que no se ven afectados por factores tales como la fatiga, las condiciones de trabajo y el estado emocional.

Análisis bioquímico: Existe una amplia variedad de aplicaciones de química analítica. Por ejemplo, se han utilizado para analizar muestras de sangre y orina, con el fin de determinar niveles de glucosa y detectar condiciones patológicas tales como la tuberculosis.

Análisis de imagen: Las aplicaciones de las RNAs en esta área incluyen la detección de tumores en ultrasonografías, la clasificación de las radiografías de tórax, identificación del

tejido y la clasificación imágenes de resonancia magnética (MRI), la determinación de la edad ósea a partir de imágenes de rayos X, etc.

Desarrollo de fármacos: Las RNAs son utilizadas como herramientas en el desarrollo de fármacos para tratar el cáncer, el SIDA y en el proceso de modelado biomolecular.

2.1.4. Redes neuronales artificiales

2.1.4.1. Reseña histórica

A continuación, se presenta una serie de hitos en la historia que condujeron el desarrollo de las RNAs desde sus inicios hasta la forma de entenderlas de hoy en día.

- 1936 - Alan Turing. Fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación.
- 1943 - Warren McCulloch y Walter Pitts. Fueron los primeros teóricos que concibieron los fundamentos de la computación neuronal, formularon una teoría acerca de la forma de trabajar de las neuronas y finalmente modelaron una red neuronal simple mediante circuitos eléctricos.
- 1949 - Donald Hebb. Fue el primero en explicar los procesos del aprendizaje desde un punto de vista psicológico, desarrollando una regla de cómo el aprendizaje ocurría; Este es el fundamento de la mayoría de las funciones de aprendizaje que se encuentran en una red neuronal.
- 1950 - Karl Lashley. En sus series de ensayos encontró que la información no era almacenada en forma centralizada en el cerebro, sino que era distribuida.
- 1956 - Congreso de Dartmounth. Mencionando frecuentemente para indicar el nacimiento de la inteligencia artificial.
- 1957- Frank Rosenblatt. Comenzó con el desarrollo del perceptrón, la red neuronal más antigua; utilizada como identificador de patrones. Este modelo era capaz de generalizar, es decir, después de haber aprendido una serie de patrones, podía reconocer otros similares, aunque no se le hubiesen presentado en el entrenamiento.
- 1959 - Frank Rosenblatt. Terminó de escribir el libro “Principios de Neurodinámica”. En éste confirmó, que bajo ciertas condiciones, el aprendizaje del perceptrón converge hacia un estado finito (Teorema de Convergencia del Perceptrón).
- 1960 - Bernard Widroff y Marcian Hoff. Desarrollaron el modelo Adaline (ADAptative LINear Elements). Esta fue la primera red neuronal aplicada a un problema real (filtros adaptativos para eliminar ecos en las líneas telefónicas) que se ha utilizado comercialmente durante varias décadas.
- 1961 - Karl Steinbeck. Die Lernmatrix. Red Neuronal para simples realizaciones técnicas (memoria asociativa).
- 1969 - Marvin Minsky/Seymour Papert. En este año casi se produjo la muerte abrupta de las RNAs, ya que Minsky y Papert probaron matemáticamente que el perceptrón no era capaz de resolver problemas relativamente fáciles, tales como el aprendizaje de una función no lineal. Esto demostró que el perceptrón era muy débil, dado que las funciones no lineales son extensamente empleadas en computación y en problemas del mundo real.

- 1974 - Paul Werbos. Desarrolló la idea básica del algoritmo de aprendizaje de Propagación hacia atrás (backpropagation); cuyo significado quedó definitivamente aclarado en 1985.
- 1977 - Stephen Grossberg: Teoría de Resonancia Adaptada (TRA). Esta teoría define una arquitectura de red que se diferencia de todas las demás previamente construidas, simula otras habilidades del cerebro como memoria a largo y corto plazo.
- 1985 - John Hopfield. Provocó el renacimiento de las redes neuronales con su libro "Computación neuronal de decisiones en problemas de optimización".
- 1986 - David Rumelhart/G. Hinton. Redescubrieron el algoritmo de aprendizaje de propagación hacia atrás (bakpropagation).

A partir de 1986, el panorama fue alentador con respecto a las investigaciones y el desarrollo de las redes neuronales y es tan así que en la actualidad son numerosos los trabajos que se realizan y publican cada año, las aplicaciones nuevas que surgen y las empresas que lanzan al mercado productos nuevos, tanto de hardware como de software.

2.1.4.2. Conceptos básicos

Las RNAs son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso biológico. Luego de un proceso de aprendizaje y estabilización, servirán para la resolución de un problema específico. Éstas constan de un conjunto de elementos de procesamiento interconectados entre sí, los cuales procesan información por medio de su estado dinámico como respuesta a entradas externas.

Una RNA trabaja imitando las funciones más elementales del cerebro, teniendo como objetivo la adquisición de alguna de las habilidades básicas de éste, como la de aprender determinados patrones.

En los años 40 surge la idea de que los componentes naturales de la mente son simples abstracciones basadas en el comportamiento de las células nerviosas y que además es posible construir máquinas que, interconectando elementos, imiten el funcionamiento de las neuronas. En 1943 McCulloch y Pitts ponen de manifiesto su optimismo al expresar la idea de que cualquier proceso de entrada/salida puede ser modelado mediante una RNA. A partir de éste esfuerzo, los diferentes avances, tanto tecnológicos como científicos, han permitido expandir las posibilidades en el campo de la investigación y aplicaciones de las RNAs (McCulloch, Pitts, 1943).

A pesar de su simplicidad, en comparación a los modelos biológicos reales, las RNAs conservan algunas propiedades características del cerebro biológico entre las que destacan (Ríos, Pazos, 1991):

1. Habilidad para aprender
2. Capacidad de continuar funcionando aceptablemente a pesar de que se produzcan deterioros o fallos en sus componentes.

Estas características se traducen en ventajas que las redes neuronales ofrecen entre las cuales se incluyen:

Aprendizaje Adaptativo

Las RNAs aprenden a llevar a cabo ciertas tareas mediante un entrenamiento con ejemplos ilustrativos, tales como diferenciar patrones. Estas son adaptables debido a la capacidad de autoajuste de los elementos procesales que componen el sistema y son dinámicas pues son capaces de estar constantemente cambiando para adaptarse a las nuevas condiciones.

En el proceso de aprendizaje, la configuración de la red se ajusta de manera que se obtengan ciertos resultados específicos. Una red neuronal no necesita un algoritmo para resolver un problema, ya que ella puede generar su propia distribución y configuración mediante el aprendizaje.

La función del diseñador es la obtención de la arquitectura de red apropiada, desarrollando un buen algoritmo de aprendizaje que le proporcione a la red la capacidad de discriminar mediante un entrenamiento con patrones.

Auto-organización

Las RNAs emplean su capacidad de aprendizaje adaptativo para auto organizar la información que reciben durante el aprendizaje o la operación. Mientras que el aprendizaje es la modificación de cada elemento procesal, la auto organización consiste en la modificación de la red neuronal completa para llevar a cabo un objetivo específico.

Cuando las redes neuronales se usan para reconocer ciertas clases de patrones, ellas auto organizan la información usada. Por ejemplo, una red que funcione con backpropagation, creará su propia representación característica mediante la cual pueda reconocer ciertos patrones.

Esta auto organización provoca la generalización que consiste en la capacidad de las RNAs para responder apropiadamente cuando se les presentan datos o situaciones a las que no habían sido expuestas anteriormente. Esta característica es muy importante cuando se tiene que solucionar problemas en los cuales la información no es muy clara e incluso cuando está incompleta.

Tolerancia a Fallos

La destrucción parcial de una red conduce a una degradación de su estructura; sin embargo, algunas capacidades de la red se pueden retener, incluso sufriendo un gran daño.

Existen dos aspectos distintos respecto a la tolerancia a fallos:

- Las RNAs pueden aprender a reconocer patrones con ruido, distorsionados o incompletos. Esta es una tolerancia a fallos respecto a los datos.
- Las RNAs pueden seguir realizando su función (con cierta degradación) aunque se destruya parte de ellas.

La razón por la que las redes neuronales son tolerantes a los fallos es que tienen su información distribuida en las conexiones entre neuronas, existiendo cierto grado de redundancia en este tipo de almacenamiento.

Operación en tiempo real

La mayoría de las RNAs pueden operar en un entorno de tiempo real, la necesidad de cambio en los pesos de las conexiones o entrenamiento, luego de la estabilización, es mínimo (Lackes, Mack, Ziola, Ahern, 1998) (Matich, 2001).

2.1.4.3. Perceptrones

Para definir una RNA, su funcionamiento y elementos que la componen, se comenzará explicando un tipo de RNA llamado *Perceptrón.* Éste fue desarrollado entre 1950 y 1960 por el científico Frank Rosenblatt, inspirado en el trabajo de Warren McCulloch y Walter Pitts. Es importante mencionar que en la actualidad es más común usar otros modelos de RNA, siendo más usado el modelo *Sigmoide, el cual* basa su definición en perceptrones por lo que estudiar su estructura y funcionamiento facilitará la futura comprensión de modelos más complejos.

Funcionamiento
Un Perceptrón tiene como entrada una serie de elementos binarios, x1, x2, x3... xn y basado en los valores de estos produce una única salida binaria.

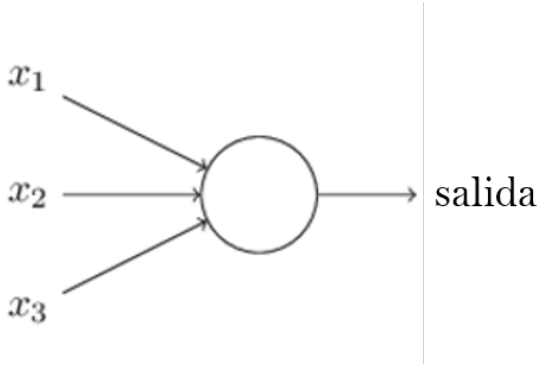


Ilustración 7 - Modelo de Perceptrón

Como se muestra en la figura, el perceptrón tiene 3 entradas llamadas $x_1, x_2, \dots, x_n$. En general, un perceptrón puede tener más entradas. Rosenblatt propuso una regla simple para calcular la salida; Introdujo el concepto de *pesos,* $w_1, w_2, \dots, w_n$, los cuales son números reales que expresan la importancia que cada entrada tiene para la salida; Ésta es determinada comparando la sumatoria de cada entrada por su respectivo peso, con un valor real denominado *umbral*. De tal forma que, si la sumatoria es menor o igual que el umbral, la salida será cero, en caso contrario la salida será uno.

Lo anterior se expresa matemáticamente de la siguiente forma:

$$salida = \begin{cases} 0\ si\ \sum_j w_j\, x_j\ \leq umbral \\ 1\ si\ \sum_j w_j x_j\ > umbral \end{cases}$$

Ecuación 1 - Modelo matemático perceptrón

Red de Perceptrones

Con un sólo perceptrón es imposible modelar la forma en que los humanos toman decisiones por lo que es necesario crear estructuras formadas por varios perceptrones con el objetivo de proveer la capacidad de toma de decisiones en situaciones más complejas.

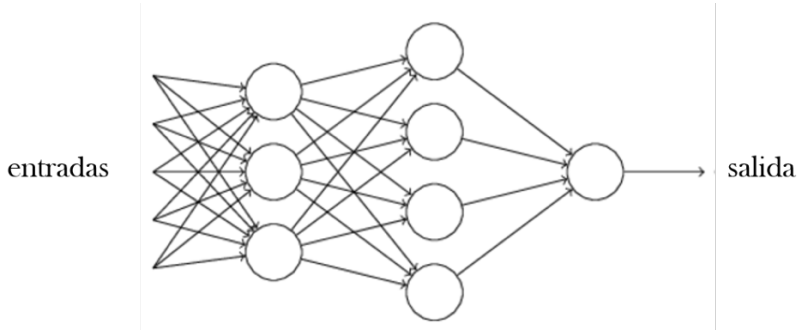


Ilustración 8 - Red de perceptrones

En la RNA representada en la figura, la primera capa de perceptrones realiza tres decisiones simples basadas en el valor y peso de las entradas, cada uno de los perceptrones de la segunda capa realiza una decisión basada en los resultados de la primera capa y el peso de cada entrada. De esta forma un perceptrón en la segunda capa calcula una salida de forma más compleja y a un nivel más abstracto que cualquier perceptrón en la primera capa. Una toma de decisión más compleja se realiza por el perceptrón en la tercera capa. Es de esta forma como una red con muchas capas de perceptrones puede realizar toma de decisiones más sofisticadas.

A diferencia de lo que se muestra en la capa uno, cuando se definió el perceptrón se especificó que cada uno de estos sólo tenía una salida, esto se muestra así en el diagrama denotando que es la misma salida la que sirve de entrada para cada perceptrón de la siguiente capa y no que el mismo perceptrón ha calculado 4 salidas.

Simplificando la expresión matemática de un perceptrón
Para simplificar la

Ecuación 1 se realizan dos cambios en la notación. El primer cambio (1) consiste en escribir $\sum_j w_j x_j$ como un producto punto de $w.x$, donde w y x son vectores cuyos componentes son los pesos y las entradas respectivamente. El segundo cambio es mover el umbral al otro lado de la desigualdad y reemplazarlo por un término denominado *Sesgo del perceptrón* (perceptrón's bias) expresado como b donde $b = -umbral$.

$$(1) salida = \begin{cases} 0 \; si \; w.x \; \leq umbral \\ 1 \; si \; w.x > umbral \end{cases}$$

Ecuación 2 - Expresión matemática simplificada de un perceptrón

$$(2) salida = \begin{cases} 0 \; si \; w.x + b \leq 0 \\ 1 \; si \; w.x + b > 0 \end{cases}$$

Ecuación 3 - Expresión matemática simplificada de un perceptrón 2

El sesgo del perceptrón debe entenderse como una medida de que tan fácil éste puede tener como salida el valor de 1. Un perceptrón con un valor de b muy alto, fácilmente tendrá como salida 1, por otro lado, si el valor de b es muy negativo, el perceptrón difícilmente podrá arrojar 1 como salida.

Como se menciona al inicio de esta sección, este modelo es poco usado en la actualidad y esto se debe a la dificultad que se presenta al entrenar una red modelada con este paradigma. Esta dificultad es debido a que pequeños cambios en los pesos y en los sesgos de los perceptrones producen grandes cambios en las salidas por lo que realizando pequeños ajustes que den determinadas salidas esperadas, se ven afectadas otras salidas que no se desean modificar.

Para poder superar la problemática planteada fue necesario desarrollar un nuevo modelo de RNA llamado *Neurona Sigmoide,* éste es similar al perceptrón, pero ha sido modificado para que pequeños cambios en los pesos y en los sesgos se traduzcan en pequeños cambios en las salidas, lo que representa un hecho crucial para el aprendizaje de este tipo de RNAs.

2.1.4.4. Neuronas sigmoides

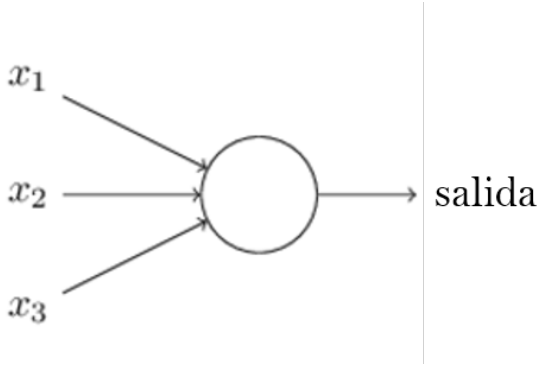


Ilustración 9 - Modelo de Neurona Sigmoide

De la misma forma que un perceptrón, una neurona sigmoide tiene entradas $x_1, x_2, \ldots, x_n$ pero en lugar de tomar valores 0 o 1, estas entradas toman valores *entre* 0 o 1, por ejemplo, 0.638. Así mismo, la neurona sigmoide posee peso para cada una de sus entradas $w_1, w_2 \ldots w_n$ y un valor de sesgo b.

La salida de una neurona sigmoide no es 1 o 0, en su lugar es calculada a través de: $\sigma(w.x + b)$ en donde σ es llamada *Función Sigmoide* y cuya definición matemática es:

$$\sigma(z) \equiv \frac{1}{1 + e^{-z}}$$

Ecuación 4 - Función sigmoide

En donde:

$$z = w.x + b$$

Ecuación 5 - Parámetros función sigmoide

Sustituyendo:

$$\sigma(z) \equiv \frac{1}{1 + e^{-(w.\ x+b)}}$$

Ecuación 6 - Función sigmoide expandida

Para entender la similitud entre la neurona sigmoide y el perceptrón, se debe suponer que $z = w.x + b$ es un número positivo alto; de esa forma $e^{-z} \approx 0$ y por lo tanto $\sigma(z) \approx 1$. En otras palabras, cuando $z = w.x + b$ es un número positivo alto, la salida calculada por la función sigmoide es aproximadamente 1, justo como lo haría un perceptrón. Por otro lado, cuando $z = w.x + b$ es un número negativo alto $e^{-z} \to \infty$ y por lo tanto $\sigma(z) \approx 0$, es decir que cuando z sea un número positivo alto o muy negativo, el comportamiento de una neurona sigmoide se aproxima al de un perceptrón.

Lo anterior se puede demostrar gráficamente a través de las curvas de las funciones que cada modelo usa:

Función Sigmoide

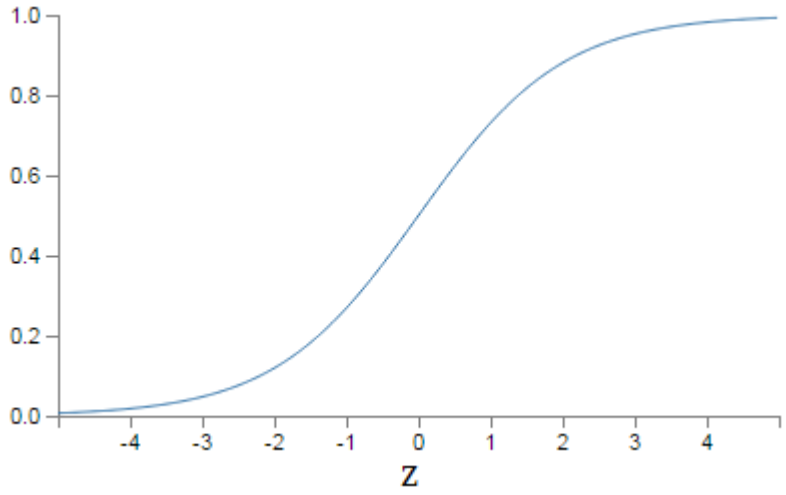


Ilustración 10 - Gráfico de la función sigmoide

Función Perceptrón

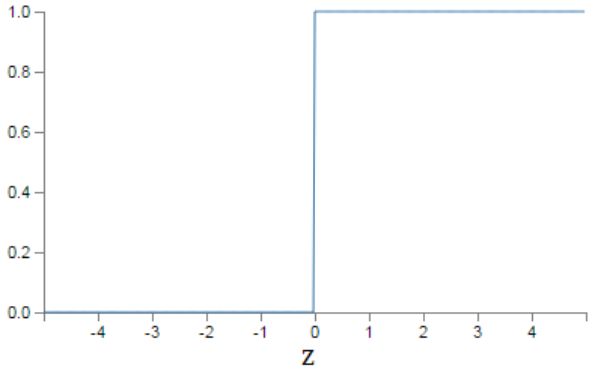


Ilustración 11 - Gráfico de la función perceptrón

Como se aprecia en las gráficas, la función sigma suaviza los bordes de la función perceptrón. Este suavizado es un hecho crucial porque se traduce en que pequeños cambios en los pesos $\Delta\omega_j$ y sesgos Δb producen pequeños cambios en la salida de la neurona. Lo anterior se expresa matemáticamente así:

$$\Delta salida \approx \sum_j \frac{\partial salida}{\partial \omega_j}\Delta\omega_j + \frac{\partial salida}{\partial b}\Delta b$$

Ecuación 7 - Representación matemática cambios de salida

Donde $\boldsymbol{\partial salida/\partial \omega_j}$ y $\boldsymbol{\partial salida/\partial b}$ denotan derivadas parciales de la salida con respecto a $\boldsymbol{\omega_j}$ y $\boldsymbol{b}$, respectivamente. Esta expresión significa que $\Delta salida$ es una función linear de los cambios de pesos y valores del sesgo. La linearidad se traduce en facilidad de realizar pequeños cambios en pesos y sesgos con el objetivo de alcanzar una salida deseada, por lo tanto, las neuronas sigmoides comparten mucho del comportamiento de los perceptrones, pero es mucho más fácil determinar cómo los cambios en pesos y sesgos cambian la salida.

Ejemplo de código de algunos tipos de perceptrones.

```
import math


# FUNCIONES DE ACTIVACION #
def sigmoid(t):
    return 1/(1+(math.e**t))
tangente_hiperbolica = math.tanh
```

```
valor_absoluto = math.fabs

exponencial = math.exp



# PERCEPTRONES #
def perceptron(pesos, entradas, tolerancia, func=None):
    """
    Funcion que modela un perceptron Suma de w*s+b
    w= pesos
    s= entradas
    b= tolerancia
    """
    error = "La cantidad de entradas debe ser igual a la cantidad de salidas"
    assert len(pesos) == len(entradas), error
    sum = 0
    for i in range(0, len(pesos)):
        sum += pesos[i] * entradas[i]
    if func is None:
        return sum+b
    else:
        return func(sum+b)

def perceptron_sigmoide(pesos, entradas, tolerancia):
    return perceptron(pesos, entradas, tolerancia, sigmoid)
def perceptron_tanh(pesos, entradas, tolerancia):
    return perceptron(pesos, entradas, tolerancia, tangente_hiperbolica)
def perceptron_absoluto(pesos, entradas, tolerancia):
    return perceptron(pesos, entradas, tolerancia, valor_absoluto)
def perceptron_exponencial(pesos, entradas, tolerancia):
    return perceptron(pesos, entradas, tolerancia, exponencial)

w = [0.33, 0.6051, 0.242]
s = [1.4, 1.006, 0.71]
b = 0.72
```

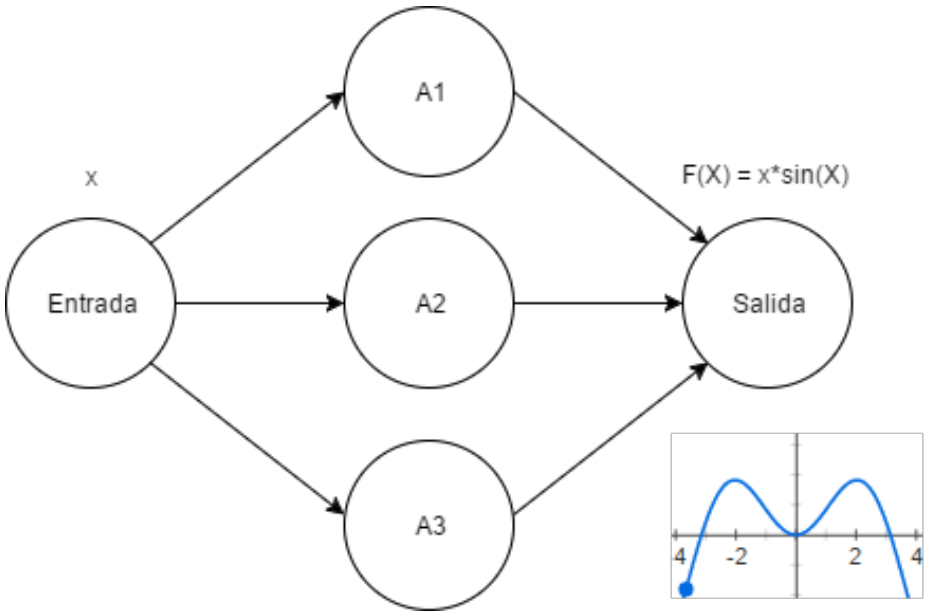
```
print("Pesos: " + str(w))
print("Entradas: " + str(s))
print("Tolerancia: " + str(b))
print("Perceptron " + str(perceptron(w, s, b)))
print("Perceptron Sigmoide " + str(perceptron_sigmoide(w, s, b)))
print("Perceptron Valor Absoluto " + str(perceptron_absoluto(w, s, b)))
print("Perceptron Tangente Hiperbolica " + str(perceptron_tanh(w, s, b)))
print("Perceptron Exponencial " + str(perceptron_exponencial(w, s, b)))
```

```
Python 2.7.10 (default, Jul 14 2015, 19:46:27)
[GCC 4.8.2] on linux
>
Pesos: [0.33, 0.6051, 0.242]
Entradas: [1.4, 1.006, 0.71]
Tolerancia: 0.72
Perceptron 1.9625506
Perceptron Sigmoide 0.123191279139
Perceptron Valor Absoluto 1.9625506
Perceptron Tangente Hiperbolica 0.961283981454
Perceptron Exponencial 7.11745772097
```

12 - Ejecución de código de perceptron

El código mostrado anteriormente, demuestra como funciona un perceptrón de tres entradas. Cabe destacar que este perceptrón funciona como una compuerta AND de tres parámetros, es decir; que su salida será positiva solo si las tres entradas son positivas. Se pueden representar funciones más complejas utilizando una red de perceptrones (Redes Neuronales).

Ejemplo de código que emula la función x*sin(x).
Para ello se modelará la siguiente red de perceptrones:



13 - Red de perceptrones

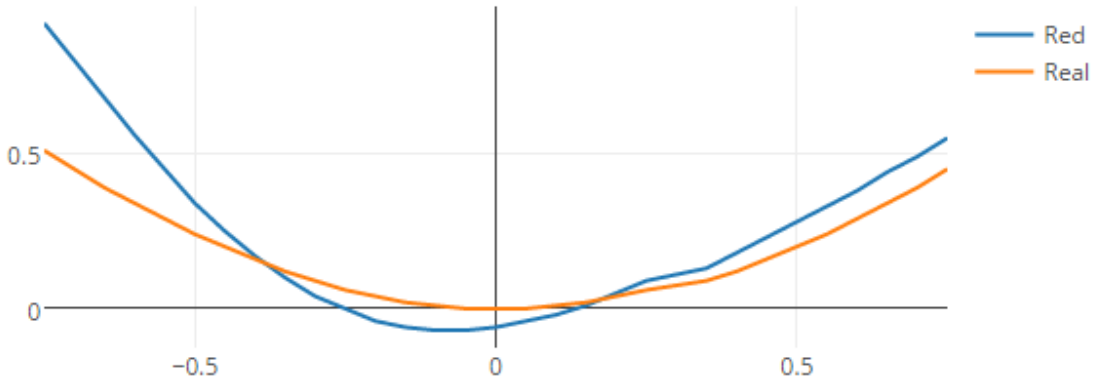
Utilizando como base el código anterior se modelará usando perceptrones con función de activación tangente hiperbólica.

```
#perceptron 1 (entradas)
x = [x * 0.05 for x in range(-15, 15)]
y = [xi*math.sin(xi) for xi in x] #salidas esperadas
y0 = [] #salidas obtenidas

#perceptron A1 (2,1)
w_2_1 = [0.4566516752869121,]
t_2_1 = 0.04860050844775432
#perceptron A2  (2,2)
w_2_2 = [-1.2054833821352962,]
t_2_2 = 4.218003727946144
#perceptrion A3   (2,3)
w_2_3 = [1.6901688851795087,]
t_2_3 = 1.1869345573154395
# perceptron 3,1 (salida)
w_3_1 = [3.1472888555511322, 3.804012936409635, -2.228226892852815]
t_3_1 = -2.1656368745702745

# ejecucion de tratamiento
for xi in x:
  # Salidas capa 1
  o_2_1 = perceptron_tanh(w_2_1, [xi,], t_2_1)
  o_2_2 = perceptron_tanh(w_2_2, [xi,], t_2_2)
  o_2_3 = perceptron_tanh(w_2_3, [xi,], t_2_3)
  # Salidas
  i_3_3 = [o_2_1 ,o_2_2, o_2_3] # entradas capa salida
  o_3_3 = perceptron(w_3_1, i_3_3, t_3_1)
  y0.append(o_3_3)

# impresion de resultados
print("Entradas\tSalida\t\tEsperado")
for i in range(len(x)): print("%.2f\t\t%.2f\t\t%.2f"%(x[i], y0[i], y[i]))
```

```
Python 3.5.2 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
:
Entradas        Salida          Esperado
-0.75           0.92            0.51
-0.70           0.80            0.45
-0.65           0.68            0.39
-0.60           0.56            0.34
-0.55           0.45            0.29
-0.50           0.34            0.24
-0.45           0.25            0.20
-0.40           0.17            0.16
-0.35           0.10            0.12
-0.30           0.04            0.09
-0.25           -0.00           0.06
-0.20           -0.04           0.04
-0.15           -0.06           0.02
-0.10           -0.07           0.01
-0.05           -0.07           0.00
0.00            -0.06           0.00
0.05            -0.04           0.00
0.10            -0.02           0.01
0.15            0.01            0.02
0.20            0.05            0.04
0.25            0.09            0.06
```

14 - Red de perceptrones

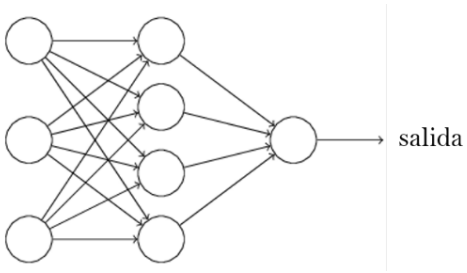
Comparación de valores obtenidos por la red contra valores esperados



*15 - Comparación de valores obtenidos de la red de perceptrones contra reales (f(x) = x*sin(x)).*

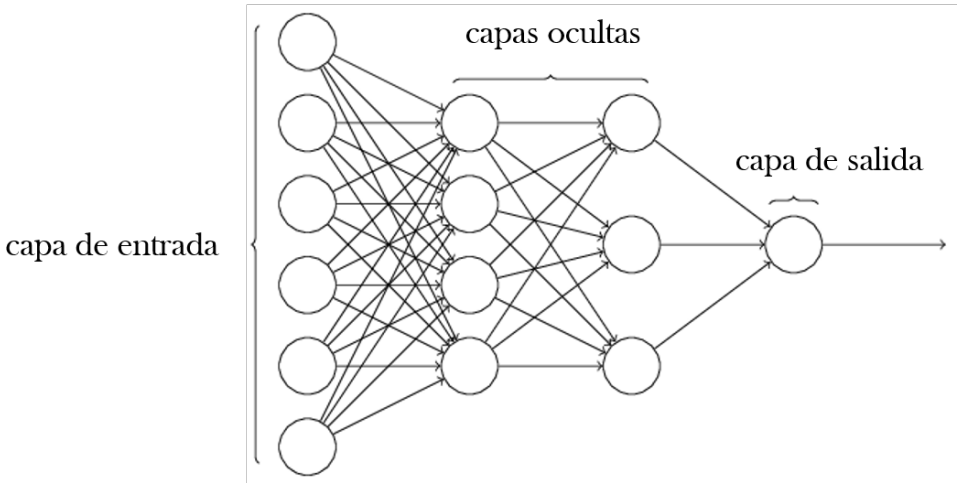
2.1.4.5. Arquitectura de redes neuronales

A continuación, se describe la arquitectura de una red neuronal tomando como base la representada por la figura.



16 - Arquitectura RN

La capa de más a la izquierda es llamada *capa de entrada* y cada neurona de esta capa se conoce como *neurona de entrada*. La capa de más a la derecha conocida como *capa de salida,* contiene a las *neuronas de salida*, en este caso única. La capa del medio es conocida como *capa oculta.* La red de la figura tiene únicamente una capa oculta, pero éstas pueden ser múltiples como en la siguiente figura:



17 - RN con capas ocultas

El diseño de las capas de entrada y salida usualmente es sencillo. Por ejemplo: si se quisiera determinar si una imagen manuscrita representa un número 9 o no, una forma de diseñar la red sería codificar las intensidades de los pixeles de la imagen en entradas a las neuronas. Si la imagen tuviera dimensiones de 64x64 pixeles en escala de grises se tendrían 4,096 entradas para neuronas con la escala de intensidad entre 0 y 1 por pixel. La capa de salida tuviera una única neurona en la que salidas menores a 0.5 indicarían que la imagen de entrada no es un 9 y valores mayores a 0.5 indicarían que la imagen de entrada es un 9.

Por otro lado, el diseño de las capas ocultas es considerablemente más complejo y no es posible formular una lista de reglas a seguir para este proceso. En su lugar, los investigadores de este campo, han desarrollado diseños heurísticos para las capas ocultas que facilitan obtener el comportamiento que se desea en una red.

Redes Neuronales Feedforward

Como su nombre lo indica, en este tipo de redes se empieza con un vector de entradas el cual es equivalente en magnitud al número de neuronas de la primera capa de la red, las cuales procesan dicho vector, elemento por elemento, en paralelo. La información, modificada por los factores multiplicativos de los pesos en cada neurona, es transmitida hacia delante por la red pasando por las capas ocultas para finalmente ser procesada por la capa de salida. Es por eso que este tipo de redes reciben su nombre.

Es importante mencionar que las redes feedforward son las más sencillas en cuanto a implementación y simulación, pero su desempeño es bueno para aplicaciones en los que no se requiera que la red retenga información de eventos pasados como ayuda para evaluar eventos futuros. Cada vector de entrada presentado como entrenamiento para este tipo de redes es una entidad aislada del resto y, al final de dicho periodo de prueba, la red estará lista para comenzar a identificar y clasificar patrones, reconocer imágenes o cualquier otra aplicación que se le quiera dar.

Al iniciar las investigaciones sobre redes neuronales, las redes feedforward fueron las que recibieron más atención de parte de los investigadores porque sus características en cuanto a tiempos de procesamiento hacían viables simulaciones con los equipos computacionales de la época. Comparadas con otras redes, las redes feedforward representaban una opción cuyo balance costo-velocidad y costo-exactitud es tal que da mayor ventaja al costo que a los otros parámetros.

En este tipo de RNAs no existen interconexiones entre capas más allá de la conexión directa hacia adelante para propagar la información. No hay rutas de retroalimentación para desempeñar la función de memoria de la red.

Redes Neuronales Recurrentes

Este tipo de RNAs tienen caminos de retroalimentación entre todos los elementos que las conforman. Una sola neurona está entonces conectada a las neuronas posteriores en la siguiente capa, las neuronas pasadas de la capa anterior y a ella misma a través de vectores de pesos variables que sufren alteraciones con el fin de alcanzar los parámetros o metas de operación.

La complejidad de este tipo de redes es alta en comparación con una red feedforward, ya que en esta última, la red es sólo capaz de transmitir la información hacia las capas siguientes, resultando en un efecto de propagación hacia atrás en el tiempo. Las redes neuronales recurrentes, en cambio, realizan intercambio de información entre neuronas de una manera mucho más compleja y por sus características, dependiendo del tipo de algoritmo de entrenamiento que se elija, pueden propagar la información hacia adelante en el tiempo, lo cual equivale a predecir eventos.

Redes Feedforward vs. Redes Recurrentes

El modelo de Redes Neuronales Feedforward es usualmente más utilizado que el modelo de Redes Neuronales Recurrentes, esto debido a que los algoritmos de aprendizaje para redes recurrentes son menos eficientes que los de redes feedforward. En cualquier caso, las redes recurrentes se acercan más al funcionamiento real de trabajo del cerebro humano y posibilitan la resolución de problemas que con redes feedforward es bastante más complejo resolver.

2.1.4.6. Aprendizaje de una red neuronal

El proceso de aprendizaje de una red neuronal consiste en que, a partir de un conjunto de datos de entrenamiento, la red sea capaz de ajustar los pesos y sesgos de cada neurona a fin de emitir una salida que se concuerde correctamente con la entrada de entrenamiento.

Para ello se definirá lo siguiente:

- x: Entrada de entrenamiento.
- $f(x)$: Salida esperada para una entrada x.
- a: Salida de la RNA.

Partiendo del planteamiento anterior, se necesita un algoritmo que permita encontrar los pesos w y sesgos b a partir de los cuales se calcule la salida a que se aproxime a la salida esperada $f(x)$ para todas las entradas de entrenamiento x. Con el objetivo de cuantificar la medida de cumplimiento de esto se define la **Función de Costo**, también conocida como *loss* o *Función Objetivo*, la cual se expresa matemáticamente así:

$$C(w,b) \equiv \frac{1}{2n}\sum_{x}\|y(x)-a\|^2$$

Ecuación 8 - Función de costo

Donde:

w : Colección de todos los pesos en la red.

b : Colección de todos los sesgos en la red.

n : Número total de entradas de entrenamiento.

a : Salida o Salidas de la RNA cuando x es la entrada.

$y(x)$: Salida esperada para una entrada x.

Está función también es conocida como *Función cuadrática de costo, Error Cuadrático o MSE.* Analizando la forma de la Función Cuadrática de Costo se prevé que $C(w,b)$ nunca tendrá un valor negativo, dado que todos sus términos son no negativos. Además, $C(w,b)$ tenderá a cero cuando $y(x)$ sea aproximadamente igual a la salida a para todas las entradas de entrenamiento x.

El algoritmo de entrenamiento hará un buen trabajo si es capaz de encontrar valores de pesos y sesgos que hagan que la Función Cuadrática de Costo se aproxime a cero; Por otro lado, hará un mal trabajo si el valor de ésta función no se acerca a cero. Por lo tanto,

el objetivo del algoritmo de aprendizaje será minimizar la Función Cuadrática de Costo a partir del ajuste de los pesos y sesgos de la RNA. El algoritmo de aprendizaje planteado a continuación es conocido como *Descenso de Gradiente.*

Descenso de Gradiente

El objetivo al entrenar una RNA es encontrar los valores de pesos y sesgos que minimicen la Función Cuadrática de Costo $C(w,b)$. Esto puede parecer poco complicado a simple vista, pero hay consideraciones que se deben tomar en cuenta como: la interpretación de los valores de pesos y sesgos, la función sigma, la elección de la arquitectura de la red, entre otros. Para la explicación de este algoritmo se concentrará en el problema de minimización de la Función Cuadrática de Costo con el objetivo de facilitar su comprensión para lo cual se presentará una técnica llamada *Descenso de Gradiente* que será usada para resolver el problema de minimización.

Se supondrá que se está tratando de minimizar una función $C(v)$. Donde C puede ser cualquier función dependiente de muchas variables $v = v1, v2, \dots vn$. Nótese que con el fin de simplificar el planteamiento, se ha reemplazado la notación de w y b por v enfatizando que ésta podría ser cualquier función, incluso fuera del área de redes neuronales.

Para minimizar $C(v)$ resulta conveniente imaginar que ésta función depende de solamente dos variables las cuales se llaman $v1\ y\ v2$:

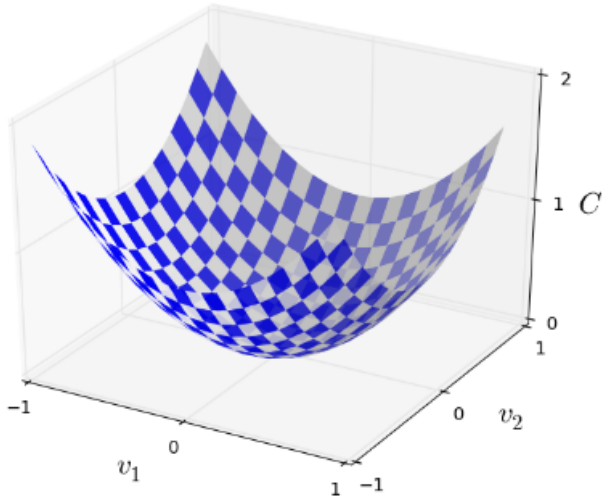


Ilustración 18 - Gráfica gradiente descendiente

El objetivo es encontrar los valores de $v1\ y\ v2$ donde C sea mínimo. Una forma de hacerlo es usar cálculo para tratar de encontrar el mínimo analíticamente, derivando la función y tratando de encontrar valores en donde C es un extremo. Éste método funciona bien cuando C está definida en función de pocas variables, pero cuando es una función de muchas variables el proceso se vuelve muchísimo más complicado. En redes neuronales, las funciones de costo dependen de billones de pesos y sesgos de formas complicadas por lo que usar cálculo para minimizar la función es inefectivo.

Si bien usar cálculo es ineficiente, permite plantear una útil analogía que sugerirá un algoritmo para resolver el problema. Se supondrá que la función, representada gráficamente, tiene la forma de un valle (Similar a la de la ilustración 14). Además, se supondrá que se suelta una pequeña bola desde una de las partes más altas del valle; Naturalmente se espera que esta ruede cuesta abajo hasta la parte con menos altura del valle.

Partiendo de la analogía anterior, se puede utilizar una metodología similar para encontrar el mínimo de la función de costo. Para ello, se escoge de forma aleatoria un punto de inicio de la bola y luego se simula el movimiento de esta, rodando hacia el fondo del valle. Ésta simulación se lleva acabo calculando derivadas y segundas derivadas de la función de costo para determinar la forma de la función de costo y así definir como la bola debería de rodar.

Antes de poder simular el comportamiento de la bola rodando hacia el mínimo del valle, es necesario definir qué ley o leyes físicas se deben tomar en cuenta. Para hacer más precisa ésta interrogante se debe pensar qué sucede cuando la bola se mueve una pequeña distancia $\Delta v1$ en la dirección $v1$ y una pequeña distancia $\Delta v2$ en la dirección $v2$. Esto se puede expresar matemáticamente como:

$$\Delta C \approx \frac{\partial C}{\partial v1}\Delta v1 + \frac{\partial C}{\partial v2}\Delta v2$$

Ecuación 9 - Cambio del valle

Se necesita encontrar valores de $\Delta v1$ y $\Delta v2$ que den como resultado un valor negativo para ΔC, debido a que se ha asumido que la bola rodará hacia el fondo del valle. Para encontrar una forma de determinar valores se define ΔC como el vector de cambios en v, donde:

$$\Delta C \equiv (\Delta v1, \Delta v2)^T$$

Ecuación 10 - Cambio Rep. Matricial

Siendo T la función de trasposición cambiando vectores filas en vectores columnas.

De esta forma se define el gradiente de C como el vector de las derivadas parciales:

$$\nabla C \equiv \left(\frac{\partial C}{\partial v1}, \frac{\partial C}{\partial v2}\right)^T$$

Ecuación 11 - Cambio Rep. Vectorial

Considerando lo anterior, la expresión para ΔC puede ser reescrita como:

$$\Delta C \approx \nabla C \,.\, \Delta v$$

Ecuación 12 - Simplificación Rep. Gradiente

∇C relaciona los cambios de v con los cambios en C y es de esta forma como permite determinar que valores de Δv transformarán a ΔC en negativo. Para llevar a cabo esta tarea se supondrá:

$$\Delta v = -n\nabla C$$

Ecuación 13 - Gradiente

Dónde:

n es un número positivo pequeño conocido como ratio de aprendizaje.

La Ecuación 12 propone que:

$$\Delta C \approx -n\nabla C . \nabla C = -n\|\nabla C\|^2$$

Ecuación 14 - Cambio reescrito

Siendo $\|\nabla C\|^2 \geq 0$. Esto garantiza que $\Delta C \leq 0$. C decrementará si se cambian los valores de v, de acuerdo a la Ecuación 13. Considerando que este es el comportamiento que se busca modelar, la Ecuación 13 define la ley de movimiento para la bola en el algoritmo de gradiente descendente y permite calcular valores de Δv y a continuación cambiar la posición de la bola en la misma cantidad:

$$v \rightarrow v' = v - n\nabla C$$

Ecuación 15 – Decremento de gradiente

A continuación, se debe utilizar esta regla de actualización de nuevo para hacer otro movimiento. Si se sigue haciendo esto, una y otra vez, se seguirá disminuyendo hasta que C sea un mínimo global.

En resumen, la forma en que el algoritmo de descenso de gradiente funciona es calculando repetidamente el gradiente ∇C de tal forma que la bola caiga a través de la pendiente del valle. Gráficamente puede verse así:

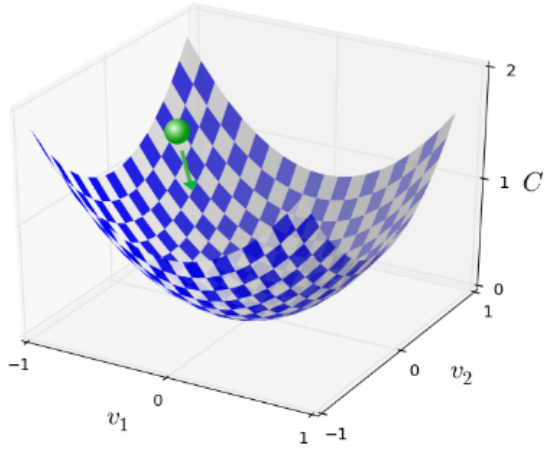


Ilustración 19 - Descenso de Gradiente

Los planteamientos de los párrafos anteriores se han formulado asumiendo que la función de costo C depende de solamente dos variables, pero a continuación se comprobará que funciona de la misma forma cuando C depende de muchas variables. Suponiendo que C es una función dependiente de m variables, $v_1, v_2 \dots v_m$; El cambio en C , ΔC producido por pequeños cambios en $\Delta v = (v_1, v_2 \dots v_m)^T$ es:

$$\Delta C \approx \nabla C \,.\, \Delta v$$

Ecuación 16 - Cambio de C

Donde el vector gradiente ∇C es el vector:

$$\nabla C \equiv \left(\frac{\partial C}{\partial v1}, \dots, \frac{\partial C}{\partial v2}\right)^T$$

Ecuación 17 - Gradiente de C en vector

Al igual que en el caso de dos variables se escoge:

$$\Delta v = -n\nabla C$$

Ecuación 18 - Cambio con dos variables

Por lo tanto, se espera llegar a un mínimo incluso cuando C dependa de muchas variables aplicando repetidamente la regla:

$$v \rightarrow v' = v - n\nabla C$$

Ecuación 19 - Cambio dependiendo de muchas variables

En la práctica, el algoritmo de descenso de gradiente funciona muy bien, y en el área de redes neuronales representa una poderosa manera de minimizar la función de coste, y así permitir el aprendizaje de la red.

2.1.4.7. Algoritmo backpropagation

El algoritmo backpropagation es el método de entrenamiento más utilizado en redes con conexión hacia delante. Es un método de aprendizaje supervisado de gradiente descendente, en el que se distinguen claramente dos fases: primero se aplica un patrón de entrada, el cual se propaga por las distintas capas que componen la red hasta producir la salida de la misma. Esta salida se compara con la salida deseada y se calcula el error cometido por cada neurona de salida. Estos errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de las capas intermedias. Cada neurona recibe un error que es proporcional a su contribución sobre el error total de la red. Basándose en el error recibido, se ajustan los errores de los pesos de cada neurona.

Deducción del algoritmo

El algoritmo propone una actualización iterativa de los pesos de la siguiente manera:

$$W(t+1) = W(t) + \Delta W(t)$$

Ecuación 20 – Actualización de los pesos iterativamente

Donde:

W: Pesos

t: instante en el tiempo.

Si se toma una variación proporcional al gradiente de una función de error $E(w)$ se tiene que:

$$W(t+1) = W(t) - \propto \nabla E[W(t)]$$

Ecuación 21 - Peso profundo

El primer paso de este algoritmo consiste en propagar hacia delante un patrón de entrada Xp y obtener la salida de la red Yp. La salida de la neurona i viene dada según su estado de activación.

Si se considera la función de salida identidad se tiene que:

$$y_i(t) = f_i\big(a_i(t)\big) = a_i(t)$$

Ecuación 22 - Salida de red

Siendo:

$$a_i(t) = f_i\big(h_i(t)\big)$$

Ecuación 23 - Activación

La regla de propagación más simple y utilizada consiste en realizar una suma de las entradas ponderadas con sus pesos sinápticos correspondientes.

$$h_i(t) = \sum_j w_{ij} x_j(t)$$

Ecuación 24 – Regla de Propagación

Se compara la salida obtenida Yp con la salida deseada Dp, obteniéndose un error que viene dado por:

$$e_p = \frac{1}{2} \sum_{k=1}^{M} \big(d_{pk} - y_{pk}\big)^2$$

Ecuación 25 – Expresión de Error

Donde:

K: índice de neurona para las neuronas de la última capa.

M: total de neuronas en la última capa.

El error total de la red está dado por:

$$e = \frac{\sum_{p=1}^{p} e_p}{P}$$

Ecuación 26 - Error total

Donde:

p: índice de ejemplo.

P: número total de ejemplos.

La variación de los pesos será proporcional al gradiente de la función de error:

$$\Delta W_{ji} = -\alpha \frac{\partial e_p}{\partial w_{ji}}$$

Ecuación 27 - Variación de pesos

Aplicando la regla de la cadena a la Ecuación 26 :

$$\frac{\partial e_p}{\partial w_{ji}} = \frac{\partial e_p}{\partial w_j} \frac{\partial h_j}{\partial w_{ji}}$$

Ecuación 28 - Gradiente de pesos

La Ecuación 28 expresa la derivada del error en función de dos derivadas. La derivada del error respecto al potencial resultante h_j indica como varia el error al variar la entrada de la neurona j, mientras que la derivada con respecto al peso sináptico w_{ji} indica como varia la entrada de la neurona j al variar el peso de la conexión que va desde la neurona i hasta la neurona j.

El segundo término de la Ecuación 28 se puede redefinir a partir de la Ecuación 24 de la siguiente manera:

$$\frac{\partial h_j}{\partial w_{ji}} = \frac{\partial \sum_i w_{ji} y_{pi}}{\partial w_{ji}} = y_{pi}$$

Ecuación 29 - Segundo término peso

Escribiendo el primer término de la Ecuación 28 como:

$$\frac{\partial e_p}{\partial h_j} = -\delta_{pj}$$

Ecuación 30 - Primer término

Se tiene que:

$$\frac{\partial h_j}{\partial w_{ji}} = -\delta_{pj} y_{pi}$$

Ecuación 31 - Resultado

Y por lo tanto la Ecuación 27 queda expresada de la siguiente manera:

$$\Delta w_{ij} = -\alpha \delta_{pj} y_{pj}$$

Ecuación 32 - cambio de peso

Para calcular el valor de delta se vuelve a aplicar la regla de la cadena:

$$\delta_{pj} = -\frac{\partial e_p}{\partial h_j} = -\left(\frac{\partial e_p}{\partial y_{pj}} \frac{\partial y_{pj}}{\partial h_j}\right)$$

Ecuación 33 - Delta de peso

El cálculo del segundo término de la Ecuación 33 es simple si se observan las ecuaciones Ecuación 22 y Ecuación 23:

$$\frac{\partial y_{pj}}{\partial h_j} = \frac{\partial f_j(h_j)}{\partial h_j} = f'_j(h_j)$$

Ecuación 34 - Calculo de segundo término

Sin embargo, para el cálculo del primer término de la Ecuación 33 es necesario distinguir entre dos casos diferentes:

La neurona j es una neurona de salida

En este caso podemos obtener el segundo término a partir de la Ecuación 25 ya que el subíndice j es igual al subíndice k:

$$\frac{\partial e_p}{\partial y_{pj}} = \frac{\partial \frac{1}{2} \sum_{j=1}^{M} (d_{pj} - y_{pj})^2}{\partial y_{pj}} = -(d_{pj} - y_{pj})$$

Ecuación 35 - Neurona j es una neurona de salida

Así, la variación de los pesos de una conexión que va hacia la capa externa de la red se calcula como:

$$\Delta w_{ji} = \alpha(d_{pj} - y_{pj}) f_j^{'}(h_j) y_{pi}$$

Ecuación 36 - Variación de pesos

La neurona j es una neurona oculta

En este caso es necesario aplicar nuevamente la regla de la cadena:

$$\frac{\partial e_p}{\partial y_{pj}} = \sum_k \left(\frac{\partial e_p}{\partial h_k} \frac{\partial h_k}{\partial y_{pj}} \right)$$

Ecuación 37 - Regla de cadena

Donde:

k: Es el subíndice de las neuronas que pertenecen a la próxima capa.

La Ecuación 37 se puede reescribir utilizando la Ecuación 24 :

$$\frac{\partial e_p}{\partial y_{pj}} = \sum_k \left(\frac{\partial e_p}{\partial h_k} \frac{\partial \left(\sum_j w_{kj} y_{pj} \right)}{\partial y_{pj}} \right) = \sum_k \left(\frac{\partial e_p}{\partial h_k} w_{kj} \right)$$

Ecuación 38 - Reescritura ecuación 24

Y de acuerdo a la Ecuación 30 se tiene que:

$$\frac{\partial e_p}{\partial y_{pj}} = \sum_k -\delta_{pk} w_{kj} = -\sum_k \delta_{pj} w_{kj}$$

Ecuación 39 - Unión con ecuación 30

Así, la variación de los pesos de una conexión que va desde una capa hacia otra capa de la red que no sea la externa se calcula como:

$$\Delta w_{ji} = \alpha \sum_k \left(\delta_{pk} w_{kj} \right) f_j^{'}(h_j) y_{pi}$$

Ecuación 40 - Variación de pesos

En la implementación del algoritmo, se toma una amplitud de paso que viene dado por la tasa de aprendizaje (α). A mayor tasa de aprendizaje el proceso será más rápido. Sin embargo, si la tasa de aprendizaje es muy alta puede dar lugar a oscilaciones en torno a un mínimo local.

Es posible disminuir el impacto de dichas oscilaciones mediante la adición de un momento (β), quedando la Ecuación 32 expresada de la siguiente manera:

$$\Delta w_{ji}(t+1) = \alpha \delta_{pj} y_{pj} + \beta \Delta w_{ji}(t)$$

Ecuación 41 - Adición de un momento

De esta manera el momento β determina el efecto en el instante t+1 del cambio de los pesos realizado en el instante t. Con este momento se consigue la convergencia de la red en menor número de iteraciones, ya que si la modificación de los pesos en los instantes t y t+1 es en la misma dirección, entonces el descenso por la superficie de error en t+1 es mayor. En cambio, si la modificación en los pesos en los instantes t y t+1 se produce en direcciones opuestas, el paso que se da en t+1 es más pequeño, lo que es adecuado, ya que esto significa que se ha pasado por un mínimo.

Resumiendo, el algoritmo backpropagation queda expresado de la siguiente manera (Bertona, 2005):

$$w_{ji}(t+1) = w_{ji}(t) + \left[\alpha \delta_{pj} y_{pj} + \beta \Delta w_{ji}(t)\right]$$

$$\text{siendo } \delta_{pj} = \begin{cases} (d_{pj} - y_{pj}) f_j'(h_j) & \text{si } j \text{ es una neurona de salida} \\ \left(\sum_k \delta_{pk} w_{kj}\right) f_j'(h_j) & \text{si } j \text{ es una neurona oculta} \end{cases}$$

Ecuación 42 - Algoritmo de backpropagation

2.1.4.8. Mejora del algoritmo de aprendizaje de la red neuronal

Hasta el momento se ha estudiado el funcionamiento del algoritmo de backpropagation y el descenso de gradiente para determinar los pesos y sesgos adecuados, entre otros elementos. A continuación se introducen algunas mejoras con el fin de optimizar el rendimiento de la red, entre las mejoras está la selección de una mejor función de costo conocida como función de costo de entropía cruzada y métodos de regularización que permiten a la red mejorar su capacidad de generalización sobre los datos de entrenamiento.

Función de costo de entropía cruzada

Por lo general se espera que la red neuronal aprenderá rápidamente de sus errores, pero ¿Esto es lo que realmente ocurre en la práctica? Para responder a esta interrogante se plantea el siguiente ejemplo en el cual tenemos una neurona con una sola entrada:

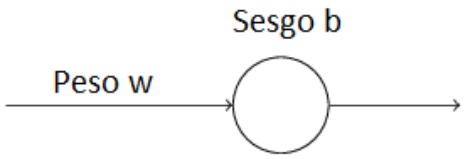


Ilustración 20 - Neurona de una sola entrada

La función de esta red será obtener 0 como salida cuando la entrada sea 1, si bien el problema no amerita la utilización de un algoritmo para ser resuelto, se aplicará el método del descenso de gradiente para encontrar los pesos y sesgos. A continuación se presenta de manera gráfica la ejecución del algoritmo para 300 épocas de entrenamiento. Con un peso y un sesgo iniciales de 0.6 y 0.9 respectivamente obtenemos una salida de 0.82, bastante alejado del resultado esperado.

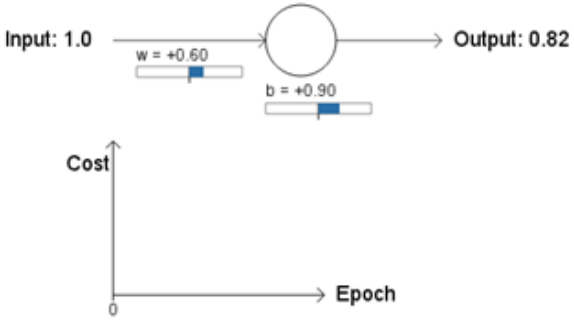


Ilustración 21 - Cambio de costo con respecto a la entrada[1]

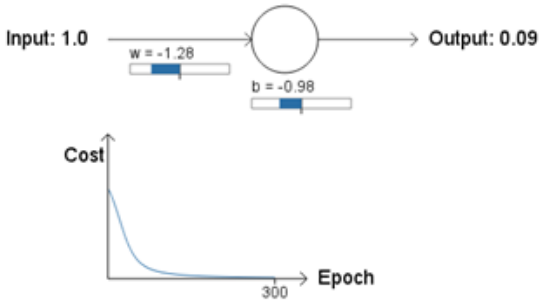


Ilustración 22 - Grafica de costo con respecto al tiempo

[1] Significados de datos presentados en gráficas:

- Input: entrada
- Output: salida
- Cost: costo
- Epoch: Ejecución
- w: peso
- b: sesgo

De la ejecución anterior se puede observar que la red rápidamente aprende nuevos valores para el peso y sesgo, lo cuales dan como resultado una salida de 0.9 para un peso de -1.28 y sesgo de -0.98, si bien no es el 0 esperado es una aproximación bastante buena, ahora se ejecuta el mismo ejemplo pero con un peso y sesgo de 2, lo cual da una salida inicial de 0.98:

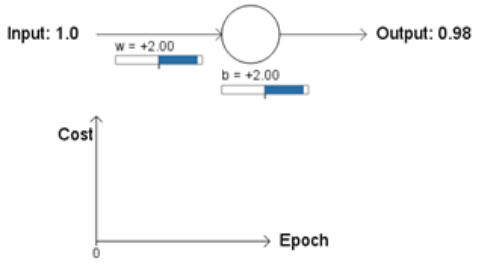


Ilustración 23 - Gráfica de costo con respecto al tiempo 2

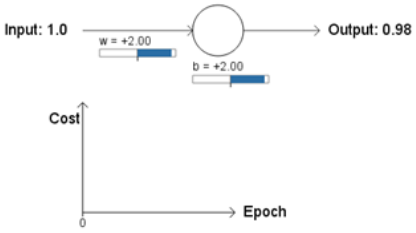


Ilustración 24 - Gráfica de costo con respecto al tiempo 3

Al finalizar el entrenamiento se observa que la aproximación obtenida no es igual de buena que con el primer ejemplo y según la gráfica a la red le tomó mucho tiempo (épocas) determinar los valores apropiados para el sesgo y el peso. Estos resultados sirven para responder la pregunta inicial de la cual podemos concluir que al contrario del proceso de aprendizaje humano en el cual aprendemos más rápidamente de los errores a las redes neuronales les toma más tiempo y esfuerzo hacerlo bajo esas condiciones.

Para poder entender los motivos de esta situación es necesario recordar que nuestra red aprende cambiando el peso y el sesgo a una velocidad determinada por las derivadas parciales de la función de costo ∂C/∂w y ∂C/∂b, decir que el aprendizaje es lento es lo mismo que decir que estas derivadas son demasiado pequeñas.

Cabe mencionar que se está utilizando la función de costo cuadrática definida por:

$$C = \frac{(y-a)^2}{2},$$

Ecuación 43 - Función de costo cuadrática

Donde:

a es la salida de la neurona cuando la entrada es **x=1,** y **y=0** es el resultado esperado.

Reescribiendo la salida de la neurona **a** en términos del peso y sesgo obtenemos: a=σ(z) donde z=wx+b, utilizando la regla de la cadena para derivar con respecto al sesgo y el peso se obtienen las siguientes ecuaciones:

$$\frac{\partial C}{\partial w} = (a-y)\sigma'(z)x = a\sigma'(z)$$
$$\frac{\partial C}{\partial b} = (a-y)\sigma'(z) = a\sigma'(z),$$

Ecuación 44 - Gradiente de costo

La función sigmoide tiene la siguiente forma:

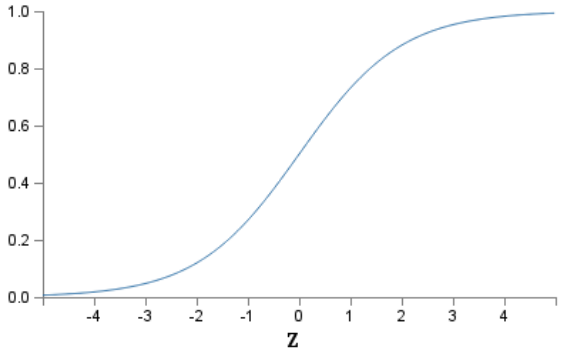


Ilustración 25 - Función sigmoide

Observando la forma de la función y el miembro derecho de las ecuaciones, se puede apreciar que cuando la salida de la neurona (**a**) es cercana a 1 la curva se vuelve un poco más plana y por lo tanto σ′(z) se vuelve más pequeña, esta es la razón por la cual el proceso de aprendizaje se vuelve más lento y esta es una situación común de las redes neuronales, para solucionar dicha situación se sustituye la función de costo por la función de entropía cruzada.

Definición

La función de entropía cruzada está definida por la siguiente ecuación:

$$C = -\frac{1}{n} \sum_x \left[y \ln a + (1-y) \ln(1-a) \right],$$

Ecuación 45 - Función de entropía

Donde:

n es el número total de datos de entrenamiento, la sumatoria es sobre todas entradas de entrenamiento *x,* y la variable *y* representa la salida esperada.

Para comprender el funcionamiento de la función y como esta permite solucionar el problema de la desaceleración del aprendizaje se desarrollará el siguiente ejemplo, en el cual se tiene una neurona con varias variables de entrada (x_n) con sus respectivos pesos (w_n) y sesgo (b).

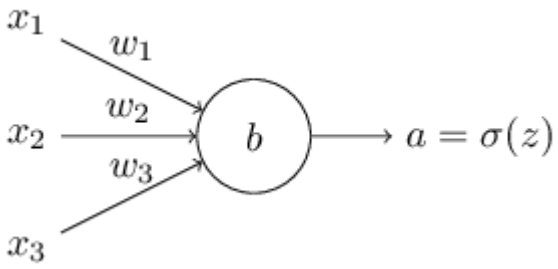


Ilustración 26 - Neurona con tres entradas

Donde:

$z=\sum_j w_j x_j+b$ es la suma ponderada de las entradas.

La ecuación 45 se puede interpretar como una función de costos debido a que posee las propiedades:

- La no negatividad, es decir C > 0, esto se da porque todos los términos individuales de la suma son negativos ya que ambos son los logaritmos de números en el rango de 0 a 1; y hay un signo menos a la parte frontal de la suma.
- Si la salida real de la neurona (*a*) es cercana a la salida esperada para todas la entradas de entrenamiento *x*, entonces la entropía cruzada será cercana a cero. Una comprobación rápida de esto se da cuando, por ejemplo, *y=0* y *a≈0* para alguna entrada *x,* el primer término de la ecuación 45 desaparece por el hecho de que *y=0,* mientras que el segundo término es *ln(1-a)≈0;* una situación similar ocurre cuando *y=1* y *a≈1.*

En resumen, la entropía cruzada es positiva y tiende a cero a medida que la neurona mejora en el cálculo de la salida deseada *a*, para todas las entradas de entrenamiento *x*.

Para comprobar cómo la función de entropía cruzada evita el problema de la desaceleración del aprendizaje es necesario simplificar la ecuación 45**,** para ello se deriva parcialmente con respecto al peso *(w)* y se simplifican los términos, obteniendo la ecuación 46**;** derivando parcialmente con respecto al sesgo *(b)* permite obtener la ecuación 47.

$$\frac{\partial C}{\partial w_j} = \frac{1}{n} \sum_x x_j(\sigma(z) - y).$$

Ecuación 46 - Gradiente de entropía

$$\frac{\partial C}{\partial b} = \frac{1}{n} \sum_x (\sigma(z) - y).$$

Ecuación 47 - Gradiente de entropía 2

En ambas ecuaciones se puede apreciar que se evita la desaceleración de aprendizaje causado por σ'(z) y ya no es tan importante que sea un valor pequeño, en comparación con la ecuación cuadrática análoga. Además la velocidad a la que la neurona aprende el peso es controlado por *σ(z) - y* es decir, por el error en la salida, por lo tanto cuanto mayor sea el error, más rápido aprenderá la neurona.

A continuación se presentan los resultados obtenidos con la nueva función de entropía cruzada, con un peso inicial de 0.60 y un sesgo de 0.90:

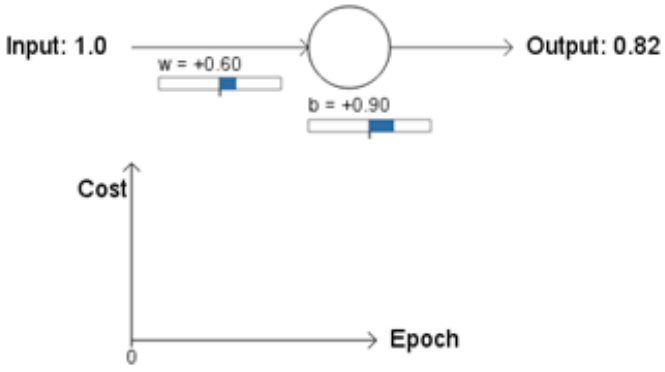


Ilustración 27 - Resultados obtenidos

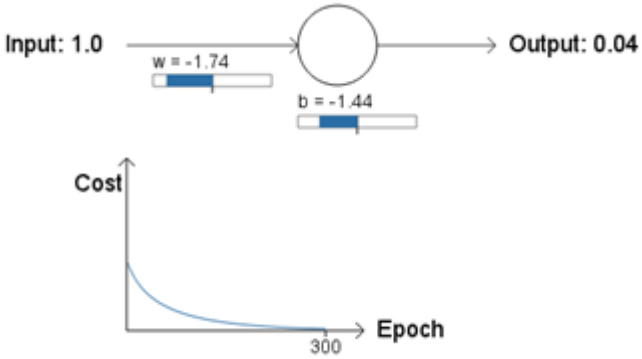


Ilustración 28 - Resultados obtenidos 2

Con un peso y sesgo inicial de 2:

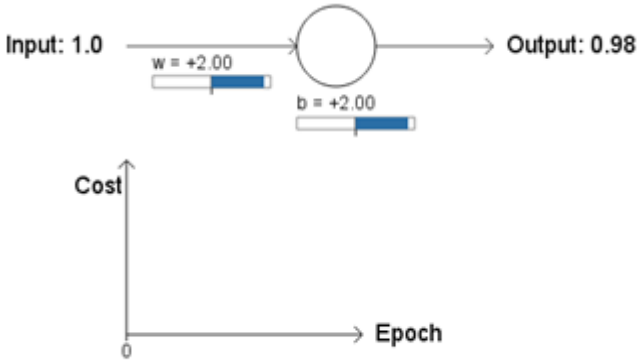


Ilustración 29 - Resultados obtenidos 3

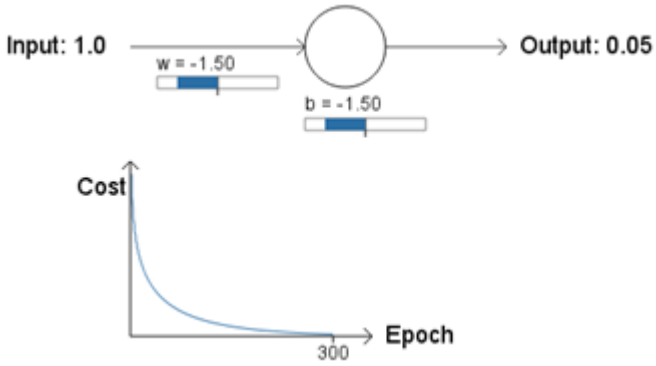


Ilustración 30 - Resultados obtenidos 4

Peso inicial (w)	Sesgo Inicial (b)	Función de costes cuadrática			Función de costos de entropía cruzada		
		Peso final (w)	Sesgo final (b)	Salida	Peso final (w)	Sesgo final (b)	Salida
0.60	0.90	-1.28	-1.98	0.09	-1.74	-1.44	0.04
2.00	2.00	- 0.69	-0.69	0.20	-1.50	-1.50	0.05

Tabla 17 - Comparación de las funciones de costos

Se puede apreciar que con la función de costos de entropía cruzada la neurona aprendió rápidamente a pesar de tener un peso y sesgo inicial equivocado. Las pruebas realizadas hasta el momento son sobre modelos de una sola neurona, pero la función de entropía cruzada también es aplicable a capas de neuronas, la siguiente ecuación describe su comportamiento:

$$C = -\frac{1}{n}\sum_x \sum_j \left[y_j \ln a_j^L + (1-y_j)\ln(1-a_j^L) \right]$$

Ecuación 48 - Entropía cruzada

Esta ecuación es la misma presentada para la definición de la entropía cruzada, solo que ahora se incluye el sumador $\sum_j$ el cual realiza la suma sobre cada salida de las neuronas, para una capa *L*

Sobreajuste

Es una situación común de las RNAs y consiste en que durante el entrenamiento llega un punto en el cual la red deja de aprender y los cambios en la precisión son insignificantes, de manera experimental se ha determinado que una forma de evitarlo es utilizando una gran cantidad de datos de entrenamiento, pero esto no siempre es factible ya que muchas veces dichos datos tienen dificultad o costos

para obtenerlos, por lo cual es importante conocer algunas técnicas para evitar ese estancamiento en el aprendizaje.

Regularización

La idea básica es agregar un término adicional a la función de costo llamado *término de regularización*, a continuación se presenta la ecuación de la función de costo de entropía cruzada con el término de regularización utilizando la técnica de *decadencia del peso o regularización L2.*

$$C = -\frac{1}{n}\sum_{xj}\left[y_j \ln a_j^L + (1-y_j)\ln(1-a_j^L)\right] + \frac{\lambda}{2n}\sum_w w^2$$

Ecuación 49 - Decadencia de peso

Se observa que el término de regularización consiste en la suma de los cuadrados de los pesos por un factor de escala λ/2n donde λ>0 es conocido como parámetro de regularización. La regularización puede ser vista como una forma de compromiso entre la búsqueda de pesos pequeños y la minimización de la función de costo original. La importancia relativa de los dos elementos depende del valor de λ: cuando λ es pequeño preferimos minimizar la función de coste original, pero cuando λ es grande preferimos pesos pequeños.

Criterios para seleccionar hiperparámetros

Se considera como hiperparámetros a aquellos parámetros que nos permiten configurar la RNA, tales como: la tasa de aprendizaje, el parámetro de regularización, la cantidad de épocas de entrenamiento, la cantidad de neuronas ocultas, el peso y sesgo inicial, etc.

La selección arbitraria de estos parámetros puede generar que la RNA no sea capaz de aprender o generalizar con la precisión esperada, tal situación supone que la RNA es deficiente. Para evitar esto se presentan las siguientes técnicas para la selección de los hiperparámetros:

- **Interrupción temprana para determinar el número de épocas:** Consiste en verificar la precisión de la RNA después de cada época de entrenamiento, esto permite determinar en qué punto existe sobreajuste y cuál es la cantidad óptima de épocas.
- **Programar la tasa de aprendizaje:** Es un hecho que al principio de nuestro entrenamiento los pesos serán bastante errados, una forma de corregir esto es usando la misma idea básica de la interrupción temprana; se mantiene

tasa de aprendizaje alta para que el peso varía rápidamente y mejore la precisión, luego con esa aproximación de los pesos podemos reducir la tasa de aprendizaje hasta llegar a un valor aceptable.

- **El parámetro de regularización, λ:** Sobre este punto no hay un criterio claro, algunos investigadores sugieren iniciar con $\lambda = 0$ determinar una tasa de aprendizaje preliminar y estimar el punto de sobreajuste, luego establecer $\lambda = 1$ y reajustar la tasa de aprendizaje; si los resultados aún no son los esperados modificar λ en un factor de 10 y repetir el proceso de ajuste hasta obtener los valores adecuados.

La regla básica es establecer valores preliminares (por lo general altos) y comparar la precisión de la RNA, luego ajustar los parámetros para mejorar la precisión y la cantidad de épocas a utilizar.

2.1.4.9. Teorema de la universalidad de las RNAs

Uno de los hechos más interesantes de las redes neuronales es que se puede usar cualquier función en absoluto para realizar sus cálculos, a pesar de que esta sea una función irregular como la siguiente:

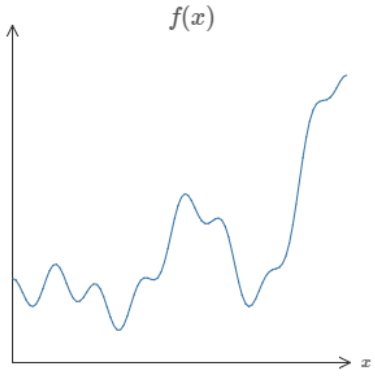


Ilustración 31 – Función irregular

Esto es debido a que una red neuronal, para cada entrada x, calculará la respectiva salida a partir de la función establecida, Este funcionamiento se conserva inclusive cuando se tienen muchas entradas y muchas salidas:

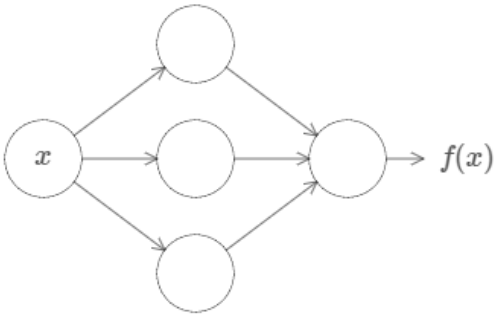


Ilustración 32 - ADG

A partir de lo propuesto en el párrafo anterior, se concluye que las RNAs poseen una característica a la cual se le da el nombre de universalidad la cual consiste en que no importando cual función se quiera tomar como base para la red, existirá una arquitectura que responderá de acuerdo a lo esperado. Este teorema de universalidad se mantiene incluso limitando la arquitectura de las redes a tener sólo una capa oculta entre la capa de entrada y la de salida

2.1.4.10. Redes neuronales profundas

Las consideraciones realizadas hasta este apartado han sido tomando como base una RNA que cuenta con únicamente una capa oculta, además de las capas de entrada y salida.

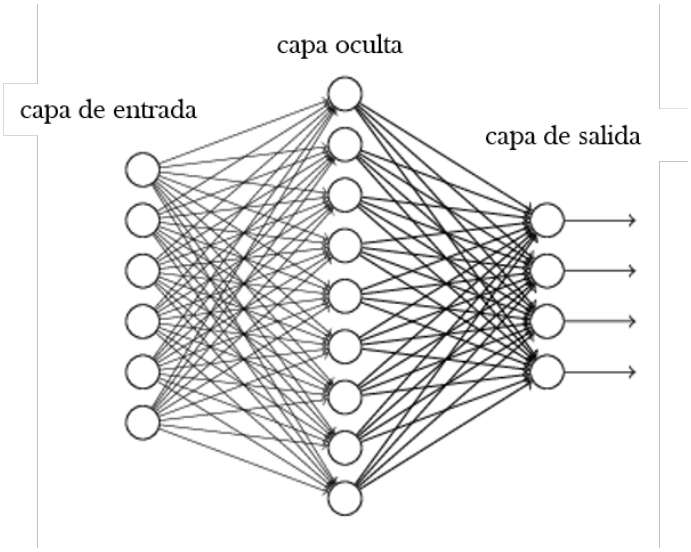


Ilustración 33 - RN con 3 capas

Esta simple arquitectura de red es suficiente para resolver problemas complejos tales como la clasificación de dígitos manuscritos con un porcentaje de acierto de un 98%, pero es natural pensar que, para problemas más complejos, una arquitectura de red más compleja ayude a solucionarlo con un grado alto de precisión.

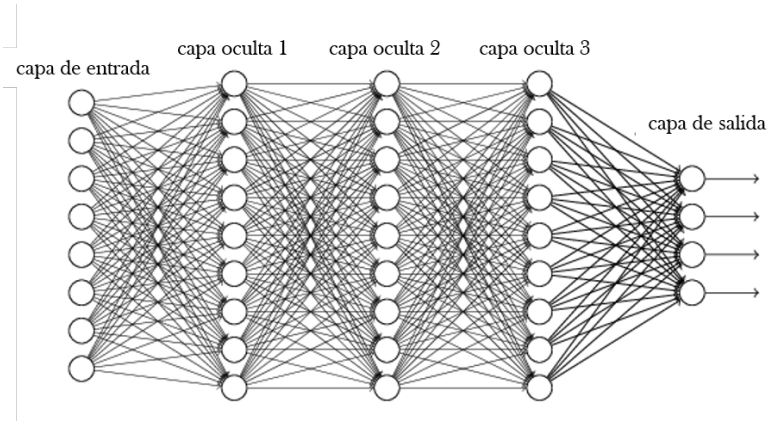


Ilustración 34 - RN profunda

Este tipo de redes utilizan capas intermedias para construir múltiples capas de abstracción. Por ejemplo, si una RNA se está utilizando para reconocimiento de patrones visuales, las neuronas en la primera capa pueden aprender a reconocer los bordes, las neuronas en la segunda capa pueden aprender a reconocer las formas más complejas como triángulos o rectángulos construidos a partir de bordes, las neuronas de la tercera capa pueden aprender a reconocer formas más complejas formadas por triángulos y rectángulos y así sucesivamente, incrementando el grado de complejidad de capa a capa.

Estas múltiples capas de abstracción dan a las redes neuronales la capacidad de aprender a reconocer patrones muy complejos. Esto es comprobado en el libro On the number of response regions of deep feed forward networks with piece-wise linear activations, por Razvan Pascanu, Guido Montúfar, and Yoshua Bengio (2014) y discutido en el libro Learning deep architectures for AI, por Yoshua Bengio (2009).

Al entrenar este tipo de RNA utilizando gradiente descendiente y backpropagation se hace evidente que el rendimiento de estas no es mejor que el de las redes poco profundas de una capa oculta esto es debido a que las distintas capas aprenden a una velocidad distinta lo que genera cuellos de botella al momento de entrenar. Este problema está relacionado con las técnicas de aprendizaje basadas en gradiente descendente por lo cual fue necesario plantear una nueva técnica de aprendizaje para este tipo de RNAs la cual fue titulada Deep Learning.

Gradientes Inestables en Redes Neuronales Profundas
Para entender a qué se debe el fallo del gradiente descendente, se considerará una red neuronal profunda simple, con solo una neurona por capa:

Donde:

w1,w2,… son los pesos, b1,b2,… son los sesgos y C es la función de costo.

Como se detalla en los apartados anteriores, la salida a_j de la j-enésima neurona es $\sigma(z_j)$, donde σ es la función sigmoide de activación y $z_j = w_j a_{j-1} + b_j$ es la entrada y sesgo de la neurona.

Se comenzará inspeccionando el gradiente $\partial C/\partial b_1$:

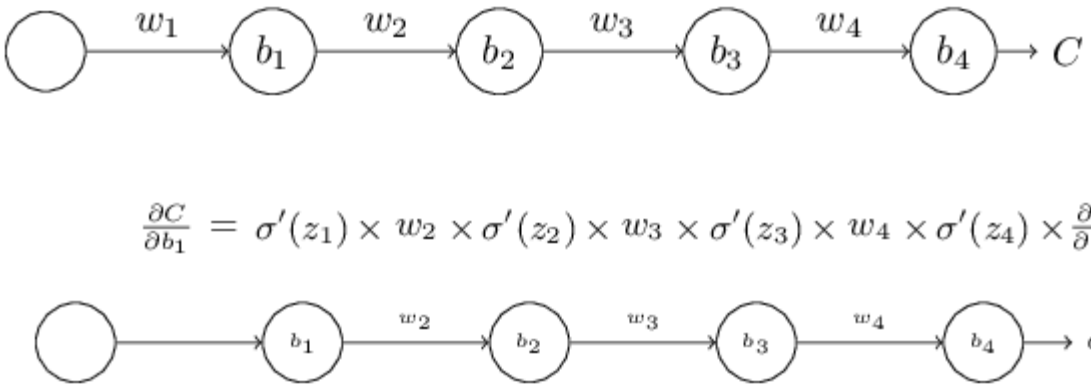


Ilustración 35 - Gradiente de costo

En esta expresión existe un término $\sigma'(z_j)$ por cada neurona en la red, un peso w_j por cada peso en la red y finalmente $\partial C/\partial a_4$ correspondiente a la función de costo. Para entender como se comporta cada uno de estos términos a continuación se detalla en forma gráfica la función σ':

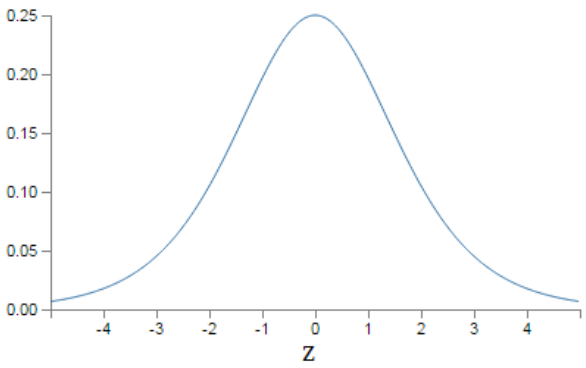


Ilustración 36 - Ecuación normal

La derivada de la función alcanza el máximo cuando es evaluada en 0. Si se usa el enfoque estándar para inicializar los pesos en la red, serán elegidos utilizando una gaussiana con media 0 y desviación estándar 1 por lo tanto los pesos usualmente satisfarán la condición de $|w_j| < 1$ y debido a esto los términos $w_j\sigma'(z_j)$ satisfarán la condición de ser menores a ¼ y por lo tanto el producto de muchos de estos términos decrementará exponencialmente a medida que se aumente la cantidad de estos.

Sumado a lo anterior, el principal problema es que el gradiente en las primeras capas es el producto de los términos de todas las capas posteriores. Cuando la red tiene muchas capas la situación se vuelve inestable y sin algún mecanismo que produzca el equilibrio es muy poco probable que suceda por casualidad.

En resumen, el verdadero problema al entrenar redes neuronales profundas es que sufren de gradiente inestable. Como resultado de esto, si se utilizan técnicas estándar de aprendizaje basado en gradientes, las diferentes capas de la red tienden a aprender a velocidades muy diferentes.

2.1.4.11. Redes convolucionales

Las redes neuronales convolucionales, son muy similares a las redes neuronales ordinarias, puesto que están construidas a partir de neuronas que tienen pesos que se aprenden. Cada neurona recibe algunas entradas, realiza producto escalar y opcionalmente le sigue la no linealidad. Toda la red expresa una única puntuación de la función diferenciable: a partir de los pixeles de la imagen hasta la clasificación. También tienen una función de costo en la última capa (completamente conectada).

La diferencia principal entre las redes neuronales convolucionales y las redes ordinarias radica en la arquitectura, la arquitectura de una red convolucional realiza la suposición que las entradas son imágenes, lo que nos permite codificar ciertas propiedades en la arquitectura, y que realicen el trabajo de procesado más eficientes para implementar y reducir la cantidad de parámetros en la red.

Arquitectura

Como se explicó anteriormente, en las redes neuronales, cada capa está hecha de un conjunto de neuronas, donde cada neurona es completamente conectada con todas las neuronas de la capa anterior, además, todas las neuronas de una sola capa son independientes y no existe ninguna conexión entre ellas. La última capa es a la que llamamos “capa de salidas”. Los problemas de las redes neuronales ordinarias al tratar con imágenes son:

- **Variación de punto de vista.** Un objeto puede estar orientado en muchas formas con respecto a la cámara.
- **Variación de escala.** Las clasificaciones a veces pueden tener variaciones de tamaño (tanto en la realidad como en el tamaño de la imagen).
- **Deformación** Algunos objetos no son cuerpos rígidos y pueden ser deformados en formas extremas.
- **Oclusión** A veces en las imágenes no se pueden apreciar el objeto en su totalidad.
- **Condiciones de iluminación** Los efectos de la iluminación son drásticos cuando se estudia cada pixel.
- **Contexto de fondo** A veces los objetos están mezclados con el entorno, haciéndolos difícil de identificar.
- **Variación entre clases** Algunos objetos pueden tener diferentes tipos con diferentes apariencias.

Las redes neuronales convolucionales aprovechan que las entradas son imágenes, para lo que rigen su arquitectura de una forma más sensible a este tipo de datos. Estas redes tienen neuronas distribuidas en 3 dimensiones (ancho, alto y profundidad).

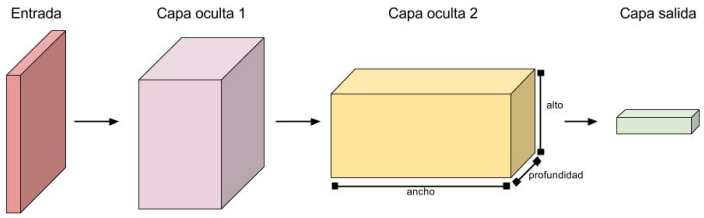


Ilustración 37 - Esquema de red convolucional

Tipos de capa utilizados para crear redes convolucionales.
Una red neuronal convolucional utiliza principalmente cuatro tipos de capa para construir arquitecturas convolucionales:

- Capas convolucionales.
- Capas de pooling.
- Capas de activación.
- Capas completamente conectadas.
- Capas de normalización

La red convolucional transforma la imagen original a través de cada capa, desde los valores originales de pixel hasta la clase final.

Capas convolucionales.

Es la base principal para construir redes neuronales convolucionales, estas realizan la mayor parte de la computación en las RNAs. Consisten en un set de filtros entrenables. Cada uno de estos filtros es espacialmente pequeño, pero se extiende sobre toda la altura y anchura de la imagen y se computa el producto escalar entre las entradas del filtro y la entrada en cada posición.

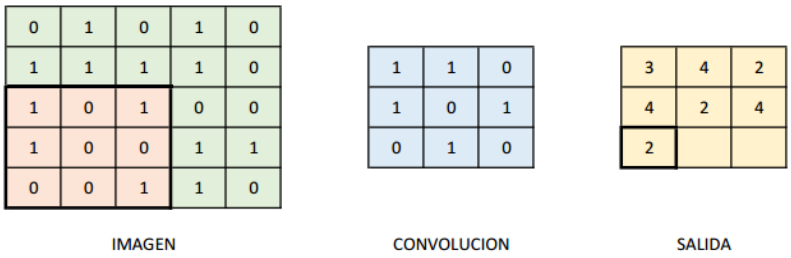


Ilustración 38 - Ejemplo de convolución

A medida se recorre la entrada, se produce un mapa de activación bidimensional que tiene las respuestas de aplicar el filtro en cada posición. La red aprenderá los filtros que se activan cuando se perciben algún tipo de característica visual, como un borde, orientaciones o bloques de colores en las primeras capas, y en capas más profundas elementos visuales más complejos como llantas, luces, rostros, ropa, etc. A partir de ello, se genera un set de filtros entrenables para cada capa convolucional, y cada una generara el mapa de activación bidimensional antes mencionado.

Ilustración 39 - Capa convolucional que reacciona a llantas (Jason Yosinski DEEPVIS)

Cuando se trabaja con entradas de grandes dimensiones (como lo son las imágenes) no es eficiente conectar las neuronas con todas las neuronas de la capa anterior (Full connected), en su lugar se debe conectar cada neurona a una sola región de la capa anterior. La extensión espacial de esta conectividad es un hiper-parámetro llamado **Campo receptivo local** y es equivalente al tamaño del filtro.

Una capa convolucional tiene las siguientes características:

1. Acepta volúmenes de tamaño W x H x D (W: ancho, H: alto, D: canales)
2. Requiere de al menos 4 hiperparámetros:
    a. Cantidad de neuronas o filtros [O]
    b. Extensión espacial (Kernel size) [K]
    c. Zancada (Stride) [S]
    d. Relleno (Padding) [P]
3. Produce un volumen del tamaño w x h x d, donde:
    a. w = (W – K + 2P) / S + 1
    b. h = (H – K + 2P) / S + 1
    c. d = O.

Capas de pooling.

Su función es de reducir el tamaño de la representación para disminuir la cantidad de parámetros y procesos en la red, y también reduce el overfitting. Una capa de pooling opera independientemente en cada trozo de la entrada y la redimensiona utilizando diferentes operadores (por ejemplo, MAX y MEAN).

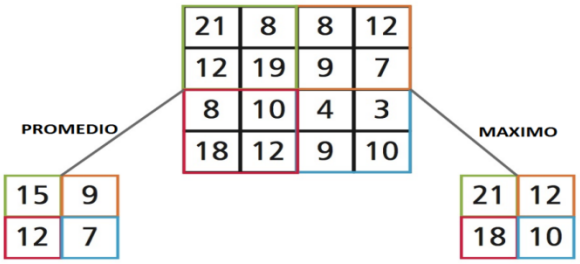


Ilustración 40 - Ejemplo pooling

Una capa de pooling tiene las siguientes características:

1. Acepta volúmenes de tamaño W x H x D (W: ancho, H: alto, D: canales)
2. Requiere de al menos 2 hiperparámetros:
a. Extensión espacial (Kernel size) [K]
b. Zancada (Stride) [S]
3. Produce un volumen del tamaño w x h x d, donde:
a. w = (W – K) / S + 1
b. h = (H – K) / S + 1
c. d = D.
4. No utiliza otros parámetros, puesto que procesa una función constante de la entrada.

Capas de activación.

Las capas de activación generan operaciones que producen una salida del mismo tamaño que sus entradas, normalmente se busca que la activación sea no lineal, puesto que permite que las redes computen problemas no triviales utilizando un número relativamente pequeño de nodos.

Entre las capas de activación se encuentran:

- Rectificación (ReLu).
- Tangente hiperbólica.
- Sigmoidea.
- Otros (valor absoluto, potencia, distribución normal gaussiana, etc.).

Una capa de activación tiene las siguientes características:

1. Entradas n * c * h * w
2. Salidas n * c * h * w

Rectificación (ReLu)

Es una función de activación definida como el máximo entre dos números (0 y x, en este caso “x” es el valor de la entrada)

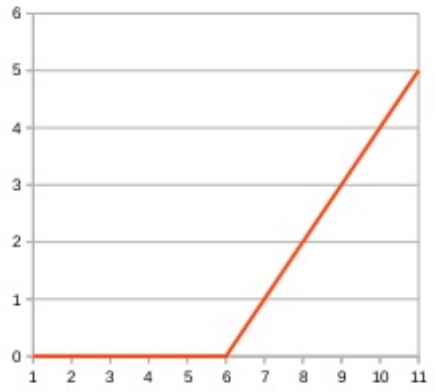


Ilustración 41 - ReLu f(x) = max(0, x-6)

Tangente hiperbólica

Computa la salida como el valor de la tangente hiperbólica de su entrada *"tanh(x)"*

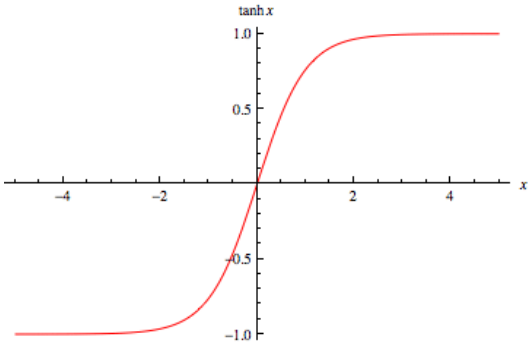


Ilustración 42 - Tangente hiperbólica f(x) = tanh(x)

Sigmoidea.

Computa la salida como el valor de aplicar la función sigmoidea de su entrada "1/(1+e^(-x))"

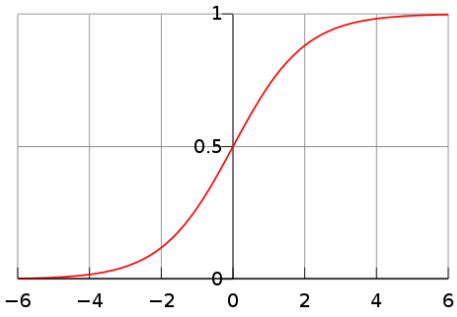


Ilustración 43 - Función sigmoidea f(x) = 1/(1+e^-x)

Capas completamente conectadas.

Las neuronas en una capa completamente conectada tienen conexiones con todas las activaciones generadas en la capa anterior (de la misma forma que trabajan las capas de las RNAs ordinarias).

Capas de normalización

Una red neuronal convolucional suele estar compuesta de diferentes tipos de capa (activación, convolucionales, activación, etc), para generar un tipo de esquema de inhibición. Con esto se buscar a tener la capacidad de que una neurona activada pueda someter a sus neuronas vecinas. De tal forma en que se destaquen más los picos locales, lo que permite generar mayor contraste en una área.

La Normalización de respuesta local (LRN) implementa esta inhibición lateral, lo que es muy útil cuando se utilizan neuronas ReLU. El problema es que las capas ReLU generan activaciones ilimitadas, lo que hace necesaria la normalización; puesto que es necesario detectar características de alta frecuencia con respuestas muy grandes. Si normalizamos contra los vecinos de la neurona activada, se torna más sensitiva al compararse con sus vecinos. También puede sesgar respuestas que son uniformemente grandes de una localidad si todos los valores son grandes.

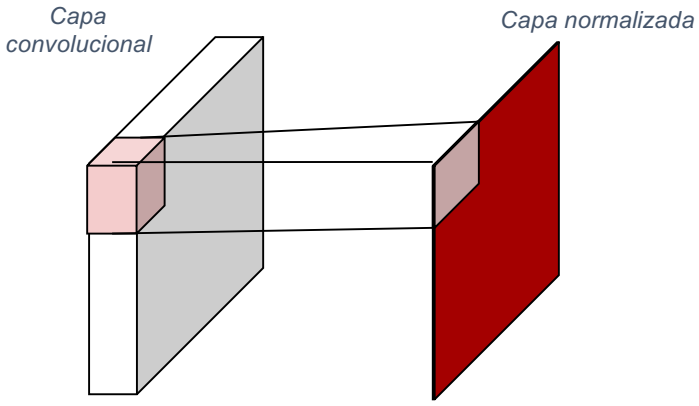


Ilustración 44 - Capa de normalización (LRN)

2.1.5. Framework Caffe

Caffe es un framework de aprendizaje profundo, diseñado para ser expresivo, rápido y modular. Desarrollado por el Centro de Visión y Aprendizaje de Berkeley (BVLC) y por contribuciones de la comunidad.

Las redes profundas son modelos composicionales naturalmente representados como una colección de capas interconectadas que trabajan sobre datos. Caffe define una red capa por capa en su propio esquema modelo. La red define el modelo desde las entradas hasta los costos (loss). Así como la forma en que los datos y sus derivados fluyen sobre la red en las iteraciones hacia delante y atrás. El framework almacena, comunica y

manipula la información como “blobs”, que es un arreglo estándar y sirve de interfaz unificada de memoria para el framework. Las capas son los elementos encargados para definir el modelo y la comunicación.

2.1.5.1. Blobs

Los blobs son contenedores donde los datos son procesados y enviados por Caffe lo que provee de capacidades de sincronización entre el CPU y el GPU. Matemáticamente un blob es un arreglo N-dimensional almacenado usando notación de C. Los blobs se consideran una interfaz de memoria unificada que almacena datos; por ejemplo: un grupo de imágenes, parámetros de modelo, etc.

Normalmente las dimensiones de un blob para un grupo de N imágenes es de N (cantidad de imágenes) x K (canales: 1 grayscale, 3 rgb, 4 rgba) x H (altura) x W (anchura). La memoria en un blob es por filas en disposición, por lo que la última dimensión cambia más rápido. Por ejemplo para un blob de 4 dimensiones, el valor del índice (x, k, h, w) está físicamente almacenado en el índice ((x*K+k)+h)*W+w.

2.1.5.2. Capas

Una capa es la esencia de un modelo y es la unidad fundamental de computación. Encargadas de convolucionar, tomar productos internos, aplicar transformaciones, normalizar, cargar datos, calcular el costo, etc. Una capa toma sus entradas a partir de conexiones inferiores (bottom) y genera sus salidas hacia las conexiones superiores (top).

Cada tipo capa define tres procesos críticos:

- Setup: inicializar la capa y sus conexiones en la inicialización del modelo.
- Forward: procesa la entrada (bottom) para generar la salida (top).
- Backward: genera y almacena el gradiente de la salida (top) y lo envía a la entrada (bottom).

Las capas tienen 2 responsabilidades importantes para la operación de la red como tal: Forward feed que produce una salida a partir de una entrada y Backpropagation que obtiene el gradiente con respecto a la salida y computa el gradiente con respecto a los parámetros y su entrada, que a su vez son enviados a capas anteriores.

Capas de Visión
Usualmente utilizan imDágenes como entradas y producen otras imágenes como salida. Los atributos más importantes en una imagen son su estructura espacial, es decir la altura (h>1) y su anchura (w>1) y la cantidad de canales de color (c). Algunas capas de visión funcionan aplicando operaciones especiales a alguna región de la entrada para producir su correspondiente región en la salida, otras capas pueden ignorar la estructura espacial de la imagen, tratándola como un solo vector con dimensión chw.

Algunas capas de este tipo son:

- Convolucional.
- Pooling.
- Normalización de Respuesta Local (LRN).

Capas de Costo
Se encargan de hacer la computación correspondiente al costo, lo que produce al aprendizaje comparando la salida con su respectivo valor real asignando un costo a minimizar.

Algunas capas de costo implementadas en Caffe son:

- Softmax.
- Suma de cuadrados (Euclídea).
- Entropía cruzada sigmoidea.
- Accuracy.

Capas de activación (Neuronas)
Las capas de activación son operadores de elemento a elemento, teniendo un blob de entrada (bottom) y uno de salida (top).

Algunas capas son:

- ReLu (rectificación lineal)
- Sigmoide.
- Tangente hiperbólica.
- Valor absoluto.
- Potencia
- Probabilidad logarítmica binomial

Capas de datos
Los datos entran a Caffe utilizando estas capas, son la entrada de todas las redes. Estos datos pueden ser a partir de bases de datos eficientes (LevelDB o LMDB), directo de memoria o de archivos de disco duro como HDF5 y formatos comunes de imágenes. Estas capas (en su mayoría) permiten preprocesado de entradas (como sustracción de media, cambio de escala, cortados aleatorios y espejado).

Algunas capas son:

- Database (LevelDB o LMDB).
- MemoryData.
- HDF5.
- Imágenes.

Capas comunes

- Inner Product, capa completamente conectada, usa la entrada como un vector unidimensional y produce otro vector unidimensional de salida.
- Flatter, transforma un vector multidimensional (n*c*h*w) en otro vector de n *(chw).
- Reshape, modifica las dimensiones de su entrada sin cambiar sus datos.
- Concatenación, concatenar sus entradas en una sola salida de n dimensiones.

2.1.5.3. Modelado de redes en Caffe

Los modelos de una red se definen en un texto plano usando un esquema de búfer de protocolo (*.prototxt), y los modelos aprendidos (red con pesos) se definen en un archivo binario (.caffemodel). La red define conjuntamente una función y su gradiente por composición y auto-diferenciación. La composición de las salidas de cada capa computa la función para hacer una tarea determinada, y la composición de cada capa de retroceso calcula el gradiente del costo de aprender una tarea. Los modelos de caffe son máquinas de aprendizaje de extremo a extremo.

Una red es un set de capas conectados en un grafo (un grafo dirigido acíclico DAG). Caffe es el encargado de hacer todos los procedimientos para administrar cualquier DAG de capas para garantizar la exactitud de los pasos hacia delante y hacia atrás (forward feed y backpropagation). Normalmente una red inicia con una capa de datos en lo más bajo y finaliza con una capa de costo (loss) en la parte superior encargada de calcular el costo necesario para realizar una tareas como clasificación o reconstrucción.

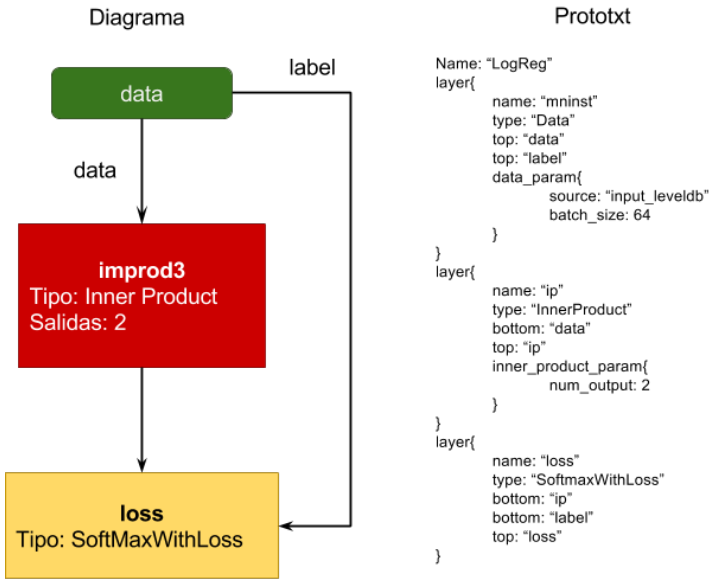


Ilustración 45 - Ejemplo de RN con Caffe

En la imagen anterior se muestra como se define la RNA utilizando la notación de CAFFE, La inicialización:

1. Construye todo el DAG creando los blobs y las capas; y
2. Llama a la función setUp() de cada capa (encargada de inicializar las capas).

Ejemplos de definición de capas (Tomados de la página oficial de caffe)

```
Capas convolucionales
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  param { lr_mult: 1 decay_mult: 1 }
  param { lr_mult: 2 decay_mult: 0 }
  convolution_param {
    num_output: 96
    kernel_size: 11
    stride: 4
    weight_filler {
      type: "gaussian"
      std: 0.01
    }
    bias_filler {
      type: "constant"
      value: 0
    }
  }
}

Capas de Pooling
layer {
  name: "pool1"
  type: "Pooling"
  bottom: "conv1"
  top: "pool1"
  pooling_param {
    pool: MAX
    kernel_size: 3
    stride: 2
```

```
  }
}

Capas ReLU
layer {
  name: "relu1"
  type: "ReLU"
  bottom: "conv1"
  top: "conv1"
}
```

2.2 Marco de pruebas

2.2.1. Definiciones y acrónimos de pruebas

Concepto	Descripción
CONCEPTOS GENERALES	
Python	Lenguaje de programación interpretado.
Pycaffe	Módulo que expone las funciones de CAFFE mediante una interfaz de línea de comandos para el lenguaje de programación python.
Numpy	Módulo de python que posee herramientas para la computación científica.
Script	Conjunto de instrucciones escritas en un determinado lenguaje de programación, cuya ejecución tiene por objetivo realizar una o varias tareas.
Entorno de pruebas	Representa un arreglo o configuración específica del entorno donde se realizará el testing de la RNA, dichas configuraciones incluyen especificaciones de hardware y software, por ejemplo la infraestructura de red, requerimiento mínimo de hardware, versiones específicas de software, etc.
Caso de prueba	Conjunto de condiciones específicas bajo las cuales es sometido a pruebas un determinado software (o elemento del software), difiere del entorno de pruebas en que los casos tienen relación con los parámetros de configuración del software y no del entorno de ejecución
Dataset	Colección de datos, consiste en los conjuntos de imágenes seleccionadas para ser analizadas en las fases de entrenamiento y testing de la RNA.
Archivos .csv	Los archivos CSV (del inglés comma-separated values) son un formato de archivo utilizado para representar datos en forma de tabla, en las que las columnas se separan por coma.
PNG	Formato de imagen; puede tener diferentes canales según su profundidad de color: • Escala de grises (1 canal).

	• Escala de grises y canal alfa (2 canales). • Canales rojos, verde y azul (RGB, 3 canales,). • Canales rojos, verde, azul y alfa (RGB + alfa, 4 canales). Cada variante supone un tratamiento diferente de la imagen para ser procesada por la RNA.
HDF5	Formato de archivo con extensión .h5, diseñado para almacenar y organizar grandes cantidades de datos en formato binario. Los archivos png pueden ser convertidos a este formato, lo cual permite procesarlos como una matriz de datos.
LMDB	Librería de gestión de base de datos basado en árboles B, la base de datos en este formato es de extremadamente alto rendimiento haciendo una gestión eficiente de la memoria.
Hiperparámetros	Corresponden a los valores de configuración de la RNA como la cantidad de imágenes a analizar por iteración (batch), la tasa de aprendizaje, pesos y sesgos iniciales, etc.
FRAMEWORK DE APRENDIZAJE PROFUNDO	
CAFFE	Framework dedicado al desarrollo de Redes Neuronales Convolucionales profundas.
deploy.prototxt	Archivo de configuración que contiene la definición del modelo de la RNA para un ambiente de producción.
solver.prototxt	Archivo que contiene los hiperparámetros de configuración de la RNA en la fase de entrenamiento.
train.prototxt	Archivo que contiene el modelo de la RNA adecuada a los hiperparámetros para las pruebas y el entrenamiento.
Layer	Capa de la RNA que contiene un conjunto de neuronas dedicadas a un fin específico dentro de la Red.
Batch	Colección de imágenes analizadas en una iteración.
Batch_size	Hiperparámetro que define la cantidad de imágenes procesadas en un Batch.
Loss	Costo promedio de procesamiento de un Batch. Es un valor entre 0 y 100, el objetivo del entrenamiento es minimizar dicho valor.
Accuracy	Precisión en la clasificación de imágenes. Tienen un valor entre 0 y 1, el objetivo de la RNA es aumentar este valor.
Learning rate	Hiperparámetro utilizado en la fase de entrenamiento, controla el tamaño del cambio en el peso y sesgo en el algoritmo de entrenamiento.
Solver	Función encargada de realizar el entrenamiento, entre ellas están: - Gradiente descendente estocástico (SDG) - Gradiente acelerado de Nesterov (Nesterov) - Gradiente adaptativo (AdaGrad)
Blob	Arreglo de N dimensiones encargado de almacenar salidas y entradas de capas.
Convolución	Operacion matemática en la que se genera una tercera función a partir de la superposición de una función con la versión trasladada e invertida de otra función.
Peppers	Nombre clave asignado al modelo de la RNA que se propone como solución al problema de clasificación de TACs.

Tabla 18 - Definiciones y acrónimos utilizados en la etapa de pruebas

2.2.2. Requerimientos de la RNA

2.2.2.1. Software

- **CAFFE framework:** El framework posee ciertos requerimientos de hardware y software para su instalación y posterior uso. También es importante mencionar que CAFFE posee dos modos de ejecución:
    - Normal mode: Modo de ejecución por defecto, el cual hace uso de la memoria RAM y el procesador.
    - GPU mode: Utiliza CUDA y las capacidades de la GPU. Este es el modo utilizado para entrenar y probar la RNA.
- **CUDA:** Modelo de computación y programación paralela desarrollado por NVIDIA. Permite un aumento drástico en cuanto a rendimiento de cómputo, aprovechando las capacidades de la unidad de procesamiento gráfico (GPU).
- **BLAS:** Subprogramas básicos de álgebra lineal (de sus siglas en inglés *Basic Linear Algebra Subprograms*).
- **Boost:** Conjunto de librerías de C++ de código abierto.
- **Pycaffe:** El cual posee las siguientes dependencias para su funcionamiento: python 2.7 y numpy >= 1.7

2.2.2.2. Hardware

- **Almacenamiento:** 30 GB de espacio en disco disponible, para almacenar las imágenes de entrenamiento y testing, la RNA y sus dependencias.
- **Procesador:** Intel Core i5 de segunda generación como mínimo.
- **RAM:** 8GB.
- **GPU:** Tarjeta gráfica NVIDIA compatible con CUDA v5.0, v5.5 o v6.0, es recomendable una de que sea compatible con la v7.0+ y con una tasa de capacidad de cálculo >= 3.0 (parámetro definido para cada modelo de tarjeta gráfica). Con memoria integrada de 2GB.

2.2.2.3. Datos de entrenamiento y prueba

Se requieren estudios de TAC cerebrales y no cerebrales en formato DICOM. Es recomendable utilizar el 70% de los datos disponibles para entrenamiento y el 30% restante para pruebas. Se recomienda un mínimo de 10,000 archivos DICOM.

2.2.3. Equipos utilizados en pruebas

ID	EQUIPO	Procesador	Sistema operativo	Gráficos	Memoria
1	DESKTOP THINKCENTRE M82	Intel® Core™ i5 - 3570 (3.40GHz, 8MB Cache)	Antergos Linux 64-bit	Intel® HD Graphics 2500	16GB DDR3

2	LAPTOP TOSHIBA SATELLITE L55	Intel® Core™ i7-4700MQ (3.40GHz, 6MB Cache)	Windows 10 64-bit	Intel® HD Graphics 4600	8GB DDR3
3	DESKTOP 1 NO MODEL	Intel® Core™ i5 - 2500 (3.30GHz, 6MB Cache)	Antergos Linux 64-bit	NVIDIA GTX 760, 2GB Memoria	8GB DDR3
4	DESKTOP 2 NO MODEL	Intel® Core™ i5 - 4690 (3.5GHz, 6MB Cache)	Windows 7 64-bit	NVIDIA GTX 760, 2GB Memoria	8GB DDR3

Tabla 19 - Equipos utilizados en pruebas

2.2.4. Extracción, transformación y carga de datos

2.2.4.1. Extracción

Las imágenes de las TACs son extraídas directamente del tomógrafo en formato DICOM, por lo general son almacenados en medios ópticos o magnéticos, las TAC se componen de una serie de imágenes que conforman un estudio, así mismo pueden haber varias series de imágenes por estudio y un paciente puede tener muchos estudios, los archivos están organizados en la siguiente estructura de carpetas:

|-- PT#

| |-- ST#

| | |-- SE#

| | | |-- ARCHIVOS DICOM

Donde PT# corresponde a un determinado paciente, ST# un estudio y SE# la serie de imágenes del estudio.

El primer paso para preparar las imágenes es clasificarlas con el siguiente criterio:

- 0 => no es TAC Cerebral
- 1 => TAC Cerebral

Dicho proceso de clasificación se realiza mediante la inspección de cada uno de los estudios, para esta tarea se utiliza un visor de DICOM, en este caso Ginkgo CADx.

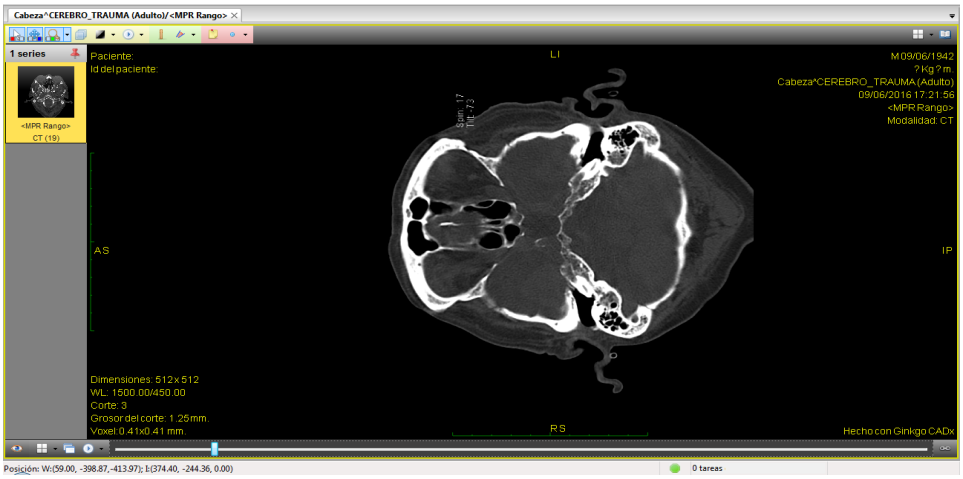


Ilustración 46 - Visor de archivos DICOM, Ginkgo CADx

Como resultado de la clasificación se genera un archivo .csv con dos columnas, en la primera la ruta de todos los archivos y en la segunda el valor de la clasificación correspondiente.

PT7/ST7/SE49/IM16	1
PT7/ST7/SE49/IM17	1
PT7/ST7/SE49/IM18	1
PT7/ST7/SE49/IM19	0
PT8/ST8/SE50/IM0	0
PT8/ST8/SE51/IM0	1
PT8/ST8/SE51/IM1	1
PT8/ST8/SE51/IM2	1
PT8/ST8/SE51/IM3	1
PT8/ST8/SE51/IM4	1

Ilustración 47 - Fragmento del archivo csv con la clasificación de los DICOM

El proceso de entrenamiento requiere de una gran cantidad de archivos DICOM, una técnica utilizada para incrementar esta cantidad consiste en realizar modificaciones significativas al archivo original y crear copias con estos cambios, de esta manera se puede incrementar la cantidad de archivos DICOM disponibles según sea la cantidad de modificaciones realizadas.

Inicialmente se cuenta con 5,508 archivos DICOM, correspondientes a 21 pacientes, 24 estudios y 144 series. Mediante un script de python se modificó la rotación de la matriz de píxeles del archivo DICOM a 90°, 180° y 270°; con lo cual se obtuvieron 22,032 archivos

en total. Cada uno de los nuevos archivos coincide con la clasificación del archivo original por lo tanto es necesario generar los archivos csv correspondientes.

PT12/ST14/SE86/IM268_90	0
PT12/ST14/SE86/IM269_90	0
PT12/ST14/SE86/IM270_90	0
PT12/ST14/SE86/IM271_90	0
PT12/ST14/SE86/IM272_90	0
PT12/ST14/SE86/IM273_90	0
PT13/ST15/SE87/IM0_90	0
PT13/ST15/SE88/IM0_90	1
PT13/ST15/SE88/IM1_90	1
PT13/ST15/SE88/IM2_90	1

PT12/ST14/SE86/IM268_180	0
PT12/ST14/SE86/IM269_180	0
PT12/ST14/SE86/IM270_180	0
PT12/ST14/SE86/IM271_180	0
PT12/ST14/SE86/IM272_180	0
PT12/ST14/SE86/IM273_180	0
PT13/ST15/SE87/IM0_180	0
PT13/ST15/SE88/IM0_180	1
PT13/ST15/SE88/IM1_180	1
PT13/ST15/SE88/IM2_180	1

PT12/ST14/SE86/IM268_270	0
PT12/ST14/SE86/IM269_270	0
PT12/ST14/SE86/IM270_270	0
PT12/ST14/SE86/IM271_270	0
PT12/ST14/SE86/IM272_270	0
PT12/ST14/SE86/IM273_270	0
PT13/ST15/SE87/IM0_270	0
PT13/ST15/SE88/IM0_270	1
PT13/ST15/SE88/IM1_270	1
PT13/ST15/SE88/IM2_270	1

48 - Fragmento del archivo csv con la clasificación de los DICOM

2.2.4.2. Transformación

La RNA recibe como entrada un formato binario en el cual se encuentra ordenada e indexada la información contenida en la matriz de píxeles del archivo DICOM. Por lo tanto hay que realizar un pre-procesamiento del archivo DICOM de la siguiente manera:

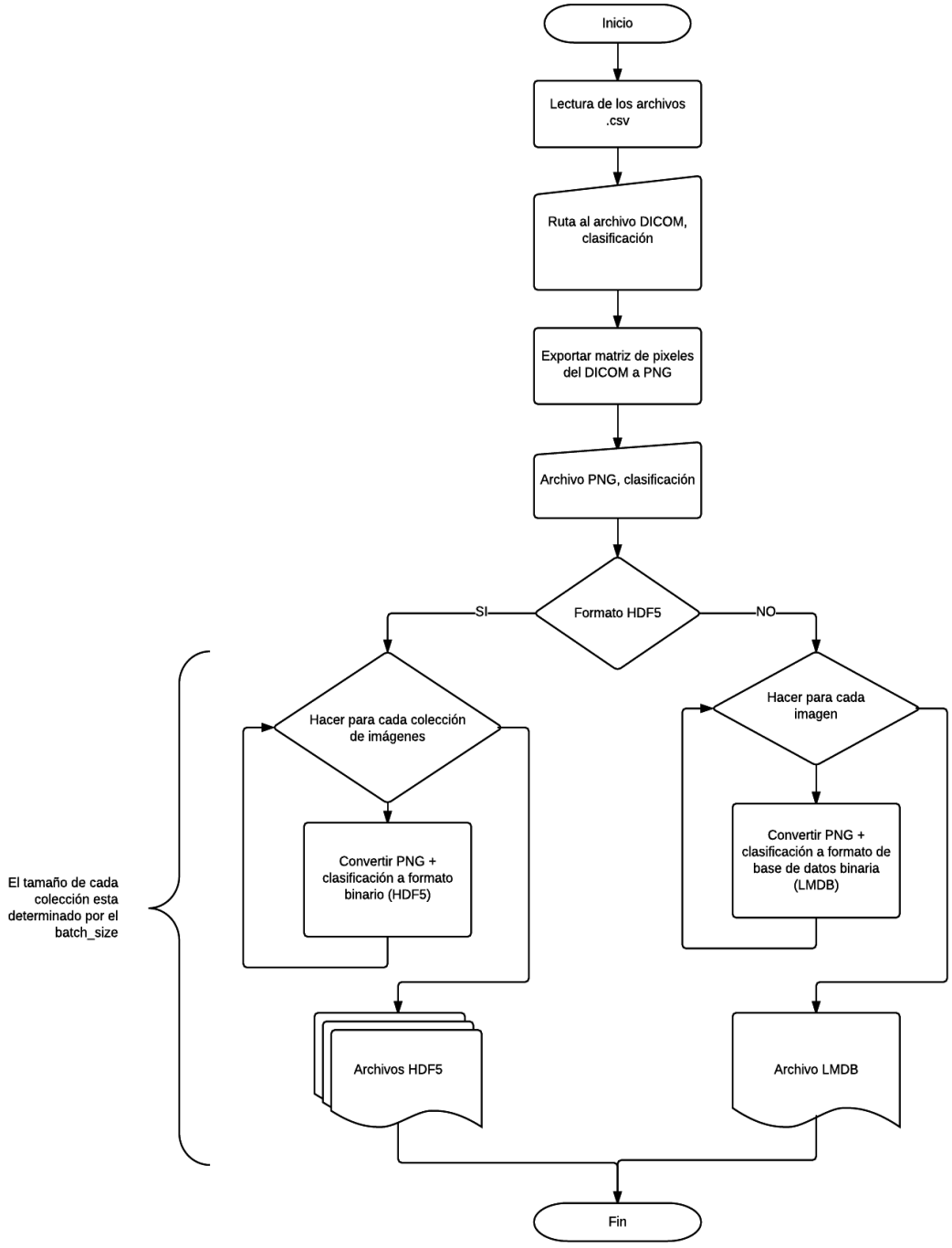


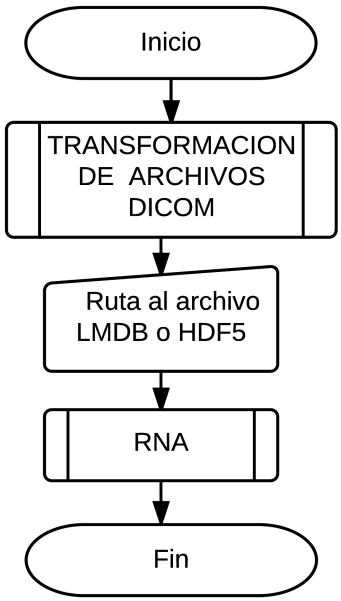
Ilustración 49 - Proceso de transformación de los archivos DICOM

En primera instancia se utilizó el formato h5, una de las características de utilizar este formato es que se genera un archivo que contiene tantas imágenes como se ha definido en el batch_size, posteriormente se cambió el formato de archivo a LMDB el cual es un archivo binario de base de datos que contiene todas las imágenes; ambos formatos almacenan las imágenes con su respectiva clasificación.

2.2.4.3. Carga de datos

Finalizado el proceso de conversión de los archivos DICOM al formato requerido por la RNA se inicia el proceso de carga de los datos. En el archivo de configuración de la RNA *train.prototxt* se define la ubicación del archivo y el formato que posee.

El proceso resumido se puede apreciar en el siguiente diagrama:



Executable File | 374 lines (373 sloc)

```
1  name: "peppers"
2  layer {
3    name: "data"
4    type: "Data"
5    top: "data"
6    top: "label"
7    include {
8      phase: TRAIN
9    }
10   data_param {
11     source: "train_lmdb"
12     batch_size: 12
13     backend: LMDB
14   }
15 }
```

50 - Proceso general de carga de datos a la RNA y train.prototxt

2.2.5. Especificación de datos

Inicialmente solo se disponían 5,508 archivos DICOM, mediante un proceso de rotación de la matriz de píxeles se logró cuadruplicar esta cantidad dando como resultado 22,032 archivos DICOM disponibles para la etapa de entrenamiento y testing. La siguiente tabla resume la distribución de los archivos para cada etapa.

Cantidad total	Entrenamiento (70%)		Pruebas (30%)	
22,032	15,298		6,734	
	Cerebrales	No Cerebrales	Cerebrales	No Cerebrales
	4589	10709	2020	4714

Tabla 20 - Distribución de los datos de pruebas

Durante el entrenamiento la RNA recibe como entrada un archivo de base de datos binario (LMDB), este archivo contiene todas las imágenes con su respectiva clasificación, el testing de la RNA puede realizarse de dos maneras:

1. Por batch, en este caso es necesario convertir los DICOM al formato LMDB y ese archivo es pasado como parámetro.
2. Individual, se convierte la matriz de píxeles del archivo DICOM a PNG y esa imagen es pasada como parámetro.

2.2.6. Diseño de pruebas

2.2.6.1. Escenarios

ID	Escenario	Descripción
Es1	Conversión de imágenes a PNG	Comprende la extracción de estudios en formato DICOM de los discos compactos, el agrupamiento en una sola carpeta y la anonimización utilizando la herramienta Ginkgo CADx 3.7.1. Además de la conversión de los archivos a formato PNG utilizando un script de python.
Es2	Conversión de imágenes a base de datos binaria .h5	Transformar las imágenes a archivos H5 para ser utilizadas por la RNA.
Es3	Conversión de imágenes a LMDB	Transformar las imágenes a un archivo LMDB para ser utilizado por la RNA.
Es4	Desarrollo RNA	Comprende el desarrollo de la RNA, así como los cambios que se le hicieron en el proceso.
Es5	Entrenamiento de RNA	Entrenamiento de la RNA para que reconozca los TACs cerebrales, lo hace por medio de análisis de patrones en un 70% del dataset.
Es6	Test de la RNA entrenada	Test de la RNA entrenada sobre el 30% del dataset, la salida de la red es comparada con su clasificación en el archivo csv para saber si tuvo éxito.
Es7	Test manual de la RNA entrenada	Test de la RNA entrenada con una imagen del dataset.

Tabla 21 - Escenarios de pruebas

2.2.6.2. Eventos

N° Evento	Evento	Escenario	Descripción
A1	Finalización exitosa del escenario		La conversión de las imágenes finaliza exitosamente.

A2	Suspensión en proceso	Es1, Es2, Es3	La conversión de las imágenes se detuvo.
A3	Proceso fallido		La conversión de las imágenes falla.
A4	Proceso no iniciado		La conversión de las imágenes no puede iniciar debido a un error.
B1	Desarrollo exitoso	Es4	El desarrollo se ejecuta normalmente.
B2	Desarrollo con errores		El desarrollo presenta errores y no puede ejecutarse.
C1	RNA entrenada	Es5	La RNA finaliza el entrenamiento exitosamente.
C2	RNA no entrenada		La RNA presenta un error al entrenarla.
C3	Entrenamiento no iniciado		La RNA presenta un error al iniciar el proceso de entrenamiento.
D1	Test realizado	Es6, Es7	El test de la RNA es realizado exitosamente.
D2	Test no realizado		El test de la RNA no se realizó debido a un error.

Tabla 22 - Eventos de pruebas

2.2.6.3. Diseño casos de pruebas

Fecha	Escena-rio	Datos utiliza-dos	Iteracio-nes	Accura-cy	Tiempo (horas)	Evento resul-tante	Comen-tario

Tabla 23 - Formato para los casos de pruebas

2.2.6.4. Procedimiento

Como punto inicial, se definirán las variables e hipótesis a demostrar en las pruebas, aunque no se siga el método científico sino el método tecnológico, esto con el fin de comprender claramente el procedimiento que se siguió:

Variables:

- TAC: Archivo de imagen en formato DICOM con metadata, el cual debe ser pasado a formato PNG y luego a un archivo LMDB para su procesamiento en lotes.

- RNA: Sistema de interconexión de neuronas que colaboran entre sí para producir un estímulo de salida. Su objetivo es procesar imágenes médicas.

Hipótesis:

- Entre más imágenes se tienen para entrenamiento y test, es más efectivo el análisis de la RNA.
- Aumentar el número de iteraciones aumenta el tiempo de entrenamiento y test.
- El número de iteraciones influye directamente en el accuracy (efectividad) de la RNA para el análisis de imágenes.
- Bajar el valor del learning rate reduce el error en el escenario de entrenamiento y pruebas, sin embargo en las fases iniciales de entrenamiento; es necesario un valor no tan pequeño para disminuir los errores de clasificación y el overfitting.
- La RNA puede ser entrenada para diferenciar entre TACs cerebrales y los otros tipos de TACs o archivos DICOM.
- La RNA puede ser entrenada para encontrar anomalías en TACs cerebrales.

En la ilustración 43, se muestra a grandes rasgos el procedimiento que se siguió para cumplir con los objetivos de la investigación y comprobar las hipótesis descritas anteriormente. El primer paso es la que más tiempo tomó, debido a que el MINSAL debió seguir un proceso burocrático para proporcionar dichos datos, debido a la privacidad de los pacientes. Una vez se tenían los archivos se procedió a su clasificación y se rotaron 90 grados, dando como resultado cuatro imágenes por cada DICOM. Luego se procedió a su conversión a formato PNG, y en dicho formato se pasó inicialmente a archivos H5. Los H5 fueron sustituidos posteriormente por un solo archivo LMDB.

Se creó la primera versión de la RNA utilizando el framework Caffe, se procedió al entrenamiento, test y evaluación. Cuando los resultados no fueron satisfactorios se repitieron ciertos pasos dependiendo la falla, si era error de dataset se procesó nuevamente los archivos DICOM y si fallaba en el entrenamiento se hacían cambios a las especificaciones y características de la RNA, de esta manera se llegó a la versión final.

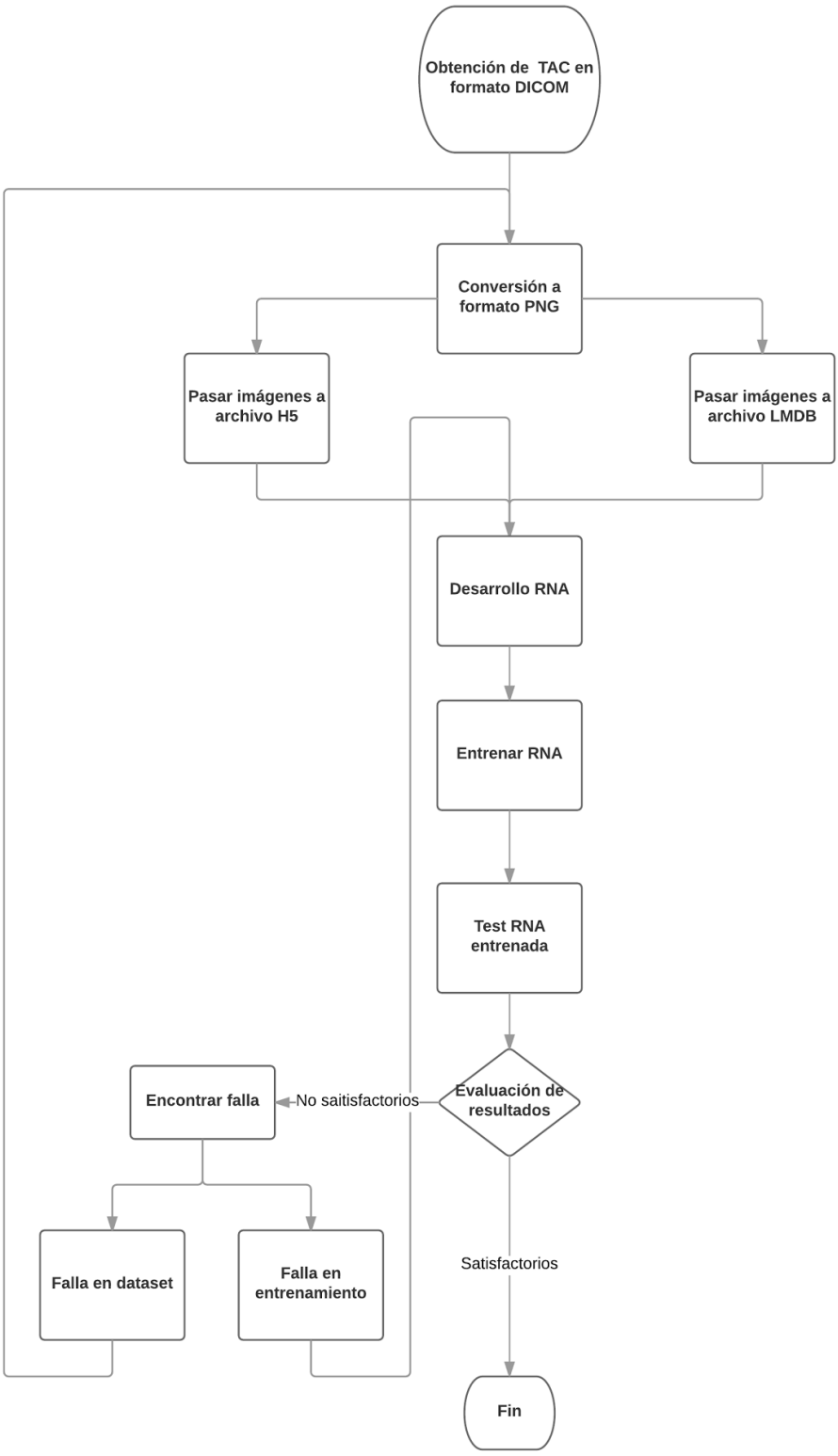


Ilustración 51 Procedimiento general

2.3. Resultados de pruebas

2.3.1. Detalles de aprendizaje

2.3.1.1 Detalle de casos de pruebas

Fecha	Id Equipo	Esce-nario	N° Datos	Iteracio-nes	Accuracy	Tiempo (horas)	Evento resultan-te	Comen-tario
13/07/16 20/07/16	1	Es1	--	--	--	--	A1	Conversió n inicial
21/07/16 04/08/16	1	Es4	--	--	--	--	B1	RNA v.1
05/08/16	2	Es5	15,298	900	--	18	C1	Entrenami ento inicial
05/08/16	2	Es6/Es 7	6,734	--	60%	--	D1	No clasificó bien
06/08/16 10/08/16	1	Es4	--	--	--	--	B1	Se cambió estructura de la RNA
10/08/16	1	Es1	--	--	--	--	A1	No se detectaba n todas las caracterísi cas
10/08/16 11/08/16	1	Es2	--	--	--	--	A1	Se detectaron errores en la conversión
11/08/16 12/08/16	1	Es5	15,298	4000	--	14	C2	El Loss tiene valor Nan
12/08/16 14/08/16	1	Es4	--	--	--	--	B1	RNA v.2
15/08/16	1	Es5	15,298	--	--	--	C3	Error de memory corruption al iniciar
15/08/16	1	Es4	--	--	--	--	B1	Se cambió estructura de la RNA
15/08/16 16/08/16	3	Es5	15,298	40,000	--	4:10	C1	Se entrena RNA con GPU
16/08/16	3	Es6/Es 7	6,734	--	75%	--	D1	Error de clasificació n
16/08/16	1	Es4	--	--	--	--	B1	Se agregaron 2 capas más de cada tipo
16/08/16 17/08/16	3	Es5	15,298	30,000	--	4:22	C1	Se redujeron las iteraciones debido a limitacione

								s de memoria
17/08/16	3	Es6/Es7	6,734	--	58%	--	D1	El porcentaje no es muy confiable
17/08/16 18/08/16	1	Es4	--	--	--	--	B1	RNA v.3
19/08/16 20/08/16	1	Es3	--	--	--	--	A1	Se cambió la forma de cargar las imágenes
21/08/16	4	Es5	15,298	50,000	--	6:22	C1	Entrenamiento con nueva RNA v.3
21/08/16	4	Es6/Es7	6,734	--	50%	--	D1	Test con porcentaje no confiable
22/08/16	4	Es5	15,298	200,000	--	10:41	C1	Se continúa el entrenamiento RNA v. 3
22/08/16	4	Es7	6,734	--	53%	--	D1	Test con porcentaje mejorado
23/08/16 24/08/16	4	Es5	15,298	360,000	--	18:14	C1	Se ejecuta el último entrenamiento.
24/08/16	4	Es6	6,734	--	54%	--	D1	Último test ejecutado
24/08/16	4	Es5	15,298	15,000	--	0:55	C1	Entrenamiento RNA v.1 con Nesterov para comparar modelos
24/08/16	4	Es6	6,734	--	53%	--	D1	Test RNA v.1 con Nesterov
24/08/16	4	Es5	15,298	15,000	--	0:40	C1	Entrenamiento RNA v.1 con SGD para comparar modelos
24/08/16	4	Es6	6,734	--	53%	--	D1	Test RNA v.1 con SGD
25/08/16	4	Es5	15,298	15,000	--	0:57	C1	Entrenamiento RNA v.2 con Nesterov para comparar modelos
25/08/16	4	Es6	6,734	--	53%	--	D1	Test RMA v.2 con Nesterov
25/08/16	4	Es5	15,298	15,000	--	0:41	C1	Entrenamiento RNA v.2 con SGD para comparar modelos

25/08/16	4	Es6	6,734	--	53%	--	D1	Test RNA v.2 con SGD
25/08/16	4	Es5	15,298	15,000	--	1:11	C1	Entrenamiento RNA v.3 con Nesterov para comparar modelos
25/08/16	4	Es6	6,734	--	53%	--	D1	Test RNA v.3 con Nesterov
25/08/16	4	Es5	15,298	15,000	--	0:44	C1	Entrenamiento RNA v.3 con SGD para comparar modelos
25/08/16	4	Es6	6,734	--	53%	--	D1	Test RNA v.3 con SGD

Tabla 24 - Bitácora de las pruebas realizadas

2.3.1.2 Selección de RNA

La selección de un modelo de RNA que se adecuará al problema de clasificación de imágenes se realizó en dos etapas:

Selección teórica

Uno de los objetivos primordiales de la investigación realizada fue obtener los conocimientos necesarios para realizar una estrategia válida y viable de cómo resolver el problema de clasificación de imágenes. Dicha investigación presenta la evolución natural que la forma de entender y construir las RNAs ha tenido desde sus orígenes en 1943 hasta la época actual.

El primer modelo descartado es el de perceptrones, la teoría infiere que si bien es el modelo que sirve como punto de inicio al resto de avances que le siguieron, este tiene fallos grandes que se resumen en que pequeños cambios en las entradas de cada neurona, se traducen en grandes cambios en sus salidas, por lo que entrenar redes complejas de este tipo se vuelve una tarea demasiado complicada.

El siguiente modelo estudiado es el de neuronas sigmoides que mejoran el principal problema que los perceptrones tenían, estas neuronas suavizan la forma en que cambios en las entradas afectan las salidas lo cual representa una mejora sustancial al modelo anterior. El problema con este modelo resulta al agregar complejidad a la arquitectura de la red, a medida que se van agregando capas, el entrenamiento se hace cada vez más difícil y requiere mayor capacidad de hardware. De este modelo se rescata la función sigmoide como función de cálculo de salidas que es utilizada en modelos posteriores.

Con el auge cada vez mayor de RNAs profundas, se evidenció que las formas de entrenamiento y modelos, desarrollados hasta ese momento, resultaban poco eficientes y se hizo necesario desarrollar nuevos paradigmas. El surgimiento de las Redes Convolucionales fue una respuesta a este problema y es el modelo con más aceptación

en la actualidad, teniendo especial aplicación en problemas de reconocimiento de imágenes, de audio y escritura, tanto así, que compañías importantes a nivel mundial como Google y Facebook cuentan con investigaciones e implementaciones de estas.

Una importante implementación desarrollada por Google, específicamente por el Google Brain Team, es TensorFlow, la cual es una biblioteca de software de código abierto para aprendizaje de máquinas en diversos tipos de tareas perceptuales y comprensión de lenguaje. Actualmente es usado para investigación y producción de docenas de equipos diferentes encargados de desarrollar productos de Google como reconocimiento de lenguaje, gmail, google photos y motor de búsqueda.

De esta forma, la investigación concluye en que se debe usar una Red Convolucional como solución al problema de clasificación. En el siguiente apartado se explicará los esfuerzos que se realizaron para determinar, de forma experimental, que arquitectura de red convolucional ofrecía mejores resultados.

Selección experimental

La selección de una arquitectura adecuada de la RNA se llevó a cabo a través de una serie de pruebas que se realizaron a diferentes desarrollos, los cuales se construyeron en forma de versiones, es decir, realizando modificaciones a la versión predecesora. Esto con el fin de dar soporte a patrones más complejos, con lo que se reduce el tiempo de respuesta y optimiza el uso de los recursos.

Se escogieron las tres versiones de RNA más representativas de las seis versiones probadas, esto para ejemplificar los cambios a manera de hitos, las cuales son mostradas en los diagramas de la siguiente sección. La RNA v.1 es la desarrollada inicialmente, la RNA v.2 es representativa de los cambios intermedios a la estructura. Finalmente la red utilizada en los entrenamientos y pruebas finales es la RNA v.3.

A continuación se presentan las principales características de las configuraciones sujetas a prueba:

RNA v.1

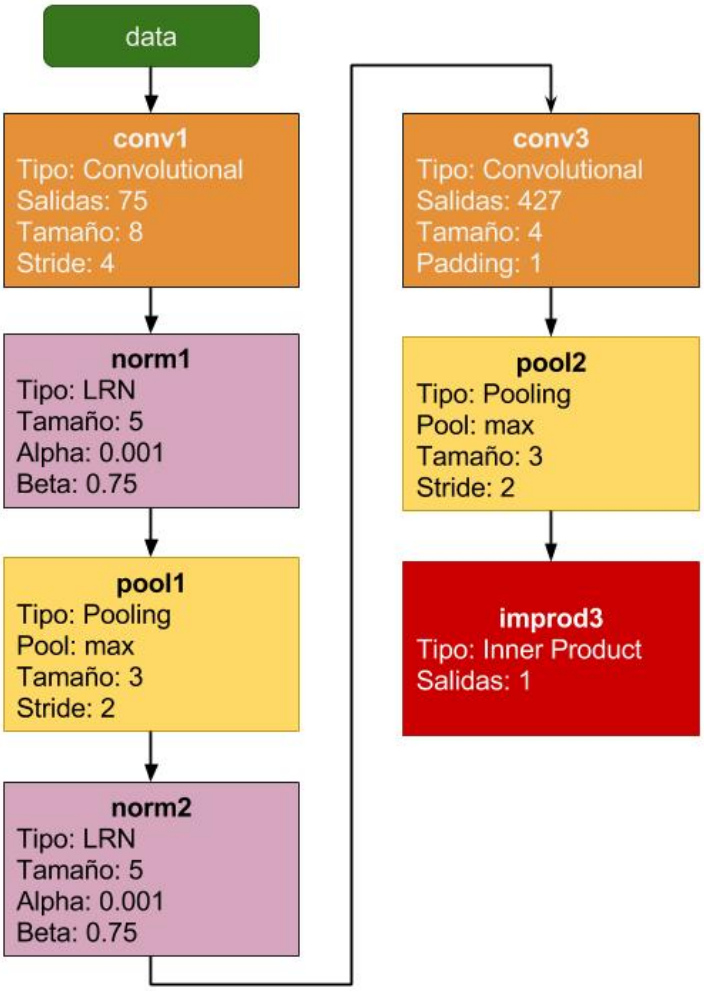


Ilustración 52 - RNA v.1[2]

[2] La explicación del funcionamiento de cada capa se realiza en la sección 2.1.4.11

RNA v.2

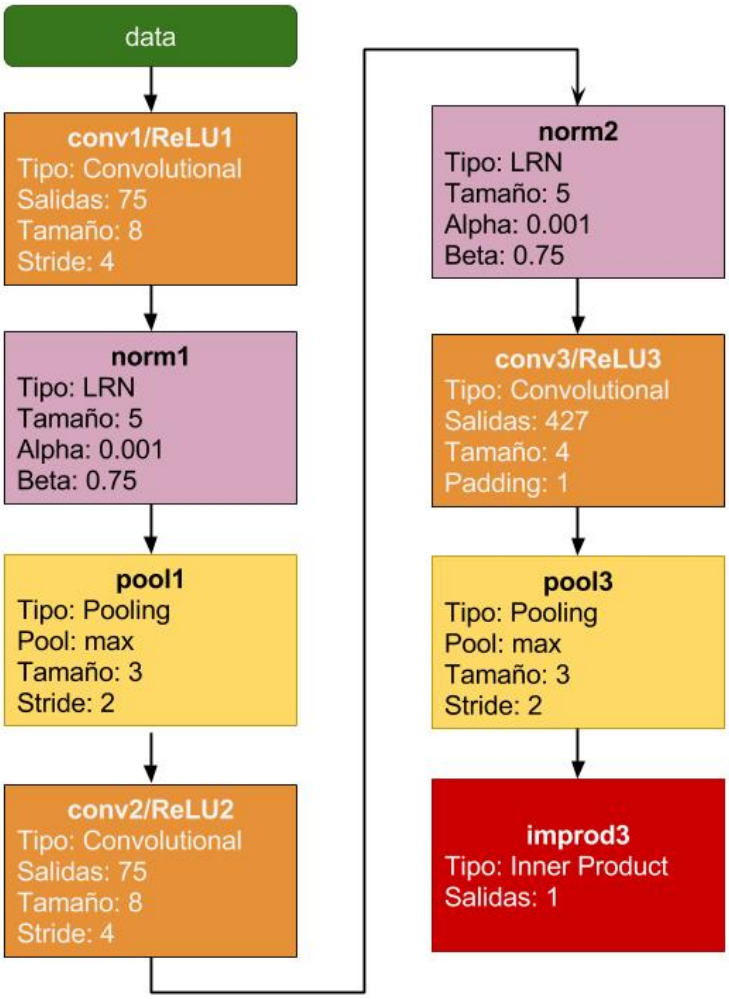


Ilustración 53 - RNA v.2[3]

[3] La explicación del funcionamiento de cada capa se realiza en la sección 2.1.4.11

RNA v.3

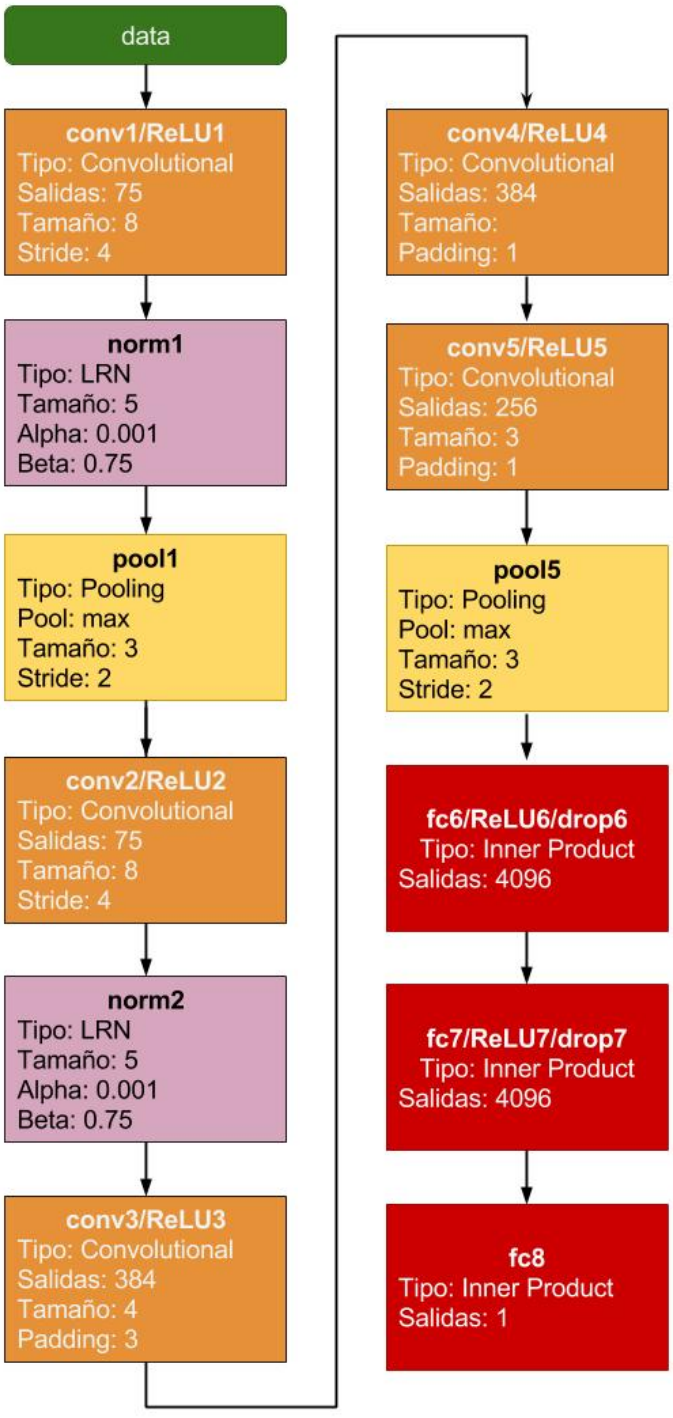


Ilustración 54 - RNA v.3[4]

[4] La explicación del funcionamiento de cada capa se realiza en la sección 2.1.4.11

2.3.1.3 Comparación de modelos

Con el objetivo de mostrar el comportamiento de las diferentes versiones de RNA desarrolladas, al finalizar la investigación se entrenaron en un mismo equipo y dejando varios parámetros estáticos. Aunque es necesario aclarar que las estadísticas de estas comparaciones no necesariamente muestran cuál versión de la red es la más eficaz.

Cada una de estas versiones se entrenó con 2 algoritmos diferentes Gradiente Descendente Estocástico (SDG) y Gradiente descendiente acelerado de Nesterov (Nesterov). Para cada uno de estos algoritmos se realizaron 15,000 iteraciones, guardando estados cada 5000 iteraciones para realizar pruebas a cada modelo.

Para los procesos de entrenamiento se dejó fijo el tamaño del Batch a 12 debido a que es la mayor cantidad de imágenes que las GPU (GTX760) usadas soportan; además se fijó el ratio de aprendizaje (Learning Rate) considerando que para el número de iteraciones con las que se realizaría el entrenamiento su variación no sería relevante. El resto de variables que se podrían considerar se dejaron fijas para todos los procesos de entrenamiento debido a restricciones de hardware lo que se traducía en procesos de entrenamiento demasiados largos.

Comparación con algoritmo Nesterov y 5000 iteraciones				
	RNA v1	RNA v2	RNA v3	Valor destacado
Tiempo (HH:mm)	0:19	0:20	0:22	0:19
Loss	87.3913	87.3751	87.3361	87.3361
Test accuracy	0.51231	0.533411	0.533979	0.51231

Tabla 25 - Resultados con el algoritmo de Nesterov a 5000 iteraciones

Comparación con algoritmo Nesterov y 10000 iteraciones				
	RNA v1	RNA v2	RNA v3	Valor destacado
Tiempo (HH:mm)	0:29	0:35	0:37	0:29
Loss	88.33614	87.3751	87.3351	87.3351
Test accuracy	0.533119	0.53313	0.533911	0.533119

Tabla 26 - Resultados con el algoritmo de Nesterov a 10000 iteraciones

Comparación con algoritmo Nesterov y 15000 iteraciones				
	RNA v1	RNA v2	RNA v3	Valor destacado
Tiempo (HH:mm)	0:55	0:57	1:11	0:55
Loss	87.44662	87.3751	87.3361	87.3361
Test accuracy	0.53391	0.53371	0.533913	0.53371

Tabla 27 - Resultados con el algoritmo de Nesterov a 15000 iteraciones

Comparación con algoritmo SGD y 5000 iteraciones				
	RNA v1	RNA v2	RNA v3	Valor destacado
Tiempo (HH:mm)	0:17	0:19	0:17	0:17
Loss	87.7559	87.3753	87.3364	87.3364
Test accuracy	0.53788	0.53381	0.53381	0.53381

Tabla 28 - Resultados con el algoritmo SGD a 5000 iteraciones

Comparación con algoritmo SGD y 10000 iteraciones				
	RNA v1	RNA v2	RNA v3	Valor destacado
Tiempo (HH:mm)	0:29	0:37	0:31	0:29
Loss	88.33614	87.3363	87.3362	87.33614
Test accuracy	0.533119	0.53383	0.53392	0.533119

Tabla 29 - Resultados con el algoritmo SGD a 10000 iteraciones

Comparación con algoritmo SGD y 15000 iteraciones				
	RNA v1	RNA v2	RNA v3	Valor destacado
Tiempo (HH:mm)	0:40	0:41	0:44	0:40
Loss	87.4462	87.3411	87.3362	87.3362
Test accuracy	0.53391	0.53381	0.53378	0.53378

Tabla 30 - Resultados con el algoritmo SGD a 15000 iteraciones

2.3.2. Estadísticas de comparaciones

2.3.2.1 Ocurrencias de eventos

NÚMERO DE OCURRENCIAS DE EVENTOS											
ESCENARIO	A1	A2	A3	A4	B1	B2	C1	C2	C3	D1	D2
Es1	2	0	0	0	0	0	0	0	0	0	0
Es2	1	0	0	0	0	0	0	0	0	0	0
Es3	1	0	0	0	0	0	0	0	0	0	0
Es4	0	0	0	0	6	0	0	0	0	0	0
Es5	0	0	0	0	0	0	12	1	1	0	0
Es6	0	0	0	0	0	0	0	0	0	11	0
Es7	0	0	0	0	0	0	0	0	0	5	0

Tabla 31 - Resumen de ocurrencia de eventos de pruebas

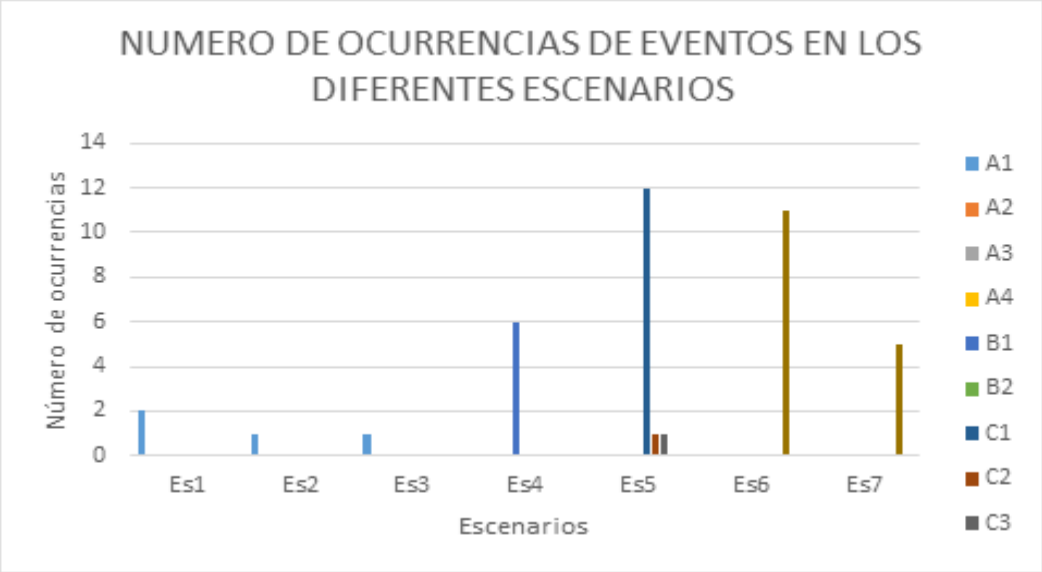


Ilustración 55 - Número de ocurrencias de eventos en los diferentes escenarios

ESCENARIO	EFECTIVIDAD
Es6	Es7
60.00	60.00
75.00	75.00
58.00	58.00
50.00	50.00

54.00	53.00
53.00	-
53.00	-
53.00	-
53.00	-
53.00	-
53.00	-
55.91	59.20

Tabla 32 - Comparativa entre escenarios y efectividad de las pruebas

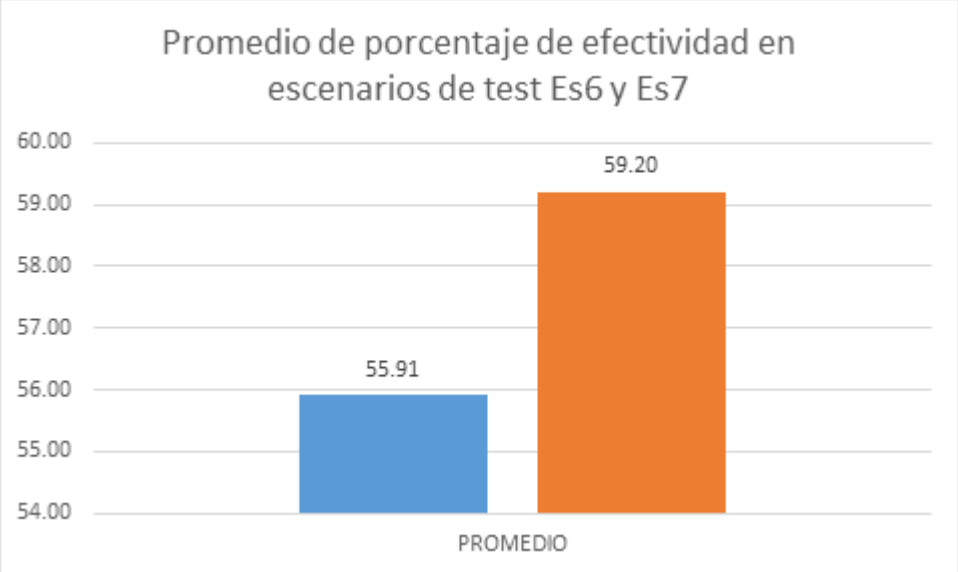


Ilustración 56 - Promedio de porcentaje de efectividad en escenarios de test Es6 y Es7

2.3.2.2 Porcentajes de precisión.

RNA v1		
ITERACIONES	TEST PRECISIÓN NESTEROV	TEST PRECISIÓN SGD
5000	0.51231	0.53788
10000	0.533119	0.533119
15000	0.53391	0.53391

Tabla 33 - Comparativa del precisión con el algoritmo de Nesterov y SGD en RNA v1

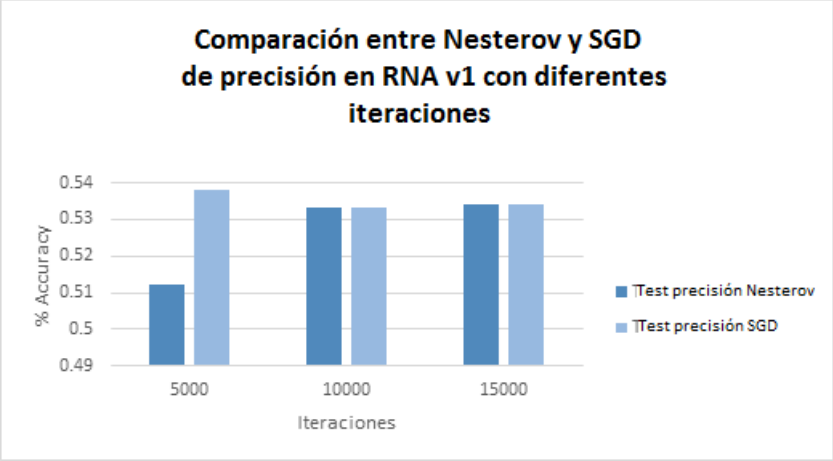


Ilustración 57 - Comparativa de accuracy en RNA v1

RNA V2		
ITERACIONES	TEST PRECISIÓN NESTEROV	TEST PRECISIÓN SGD
5000	0.533411	0.53381
10000	0.53313	0.53383
15000	0.53371	0.53381

Tabla 34 - Comparativa del precisión con el algoritmo de Nesterov y SGD en RNA v2

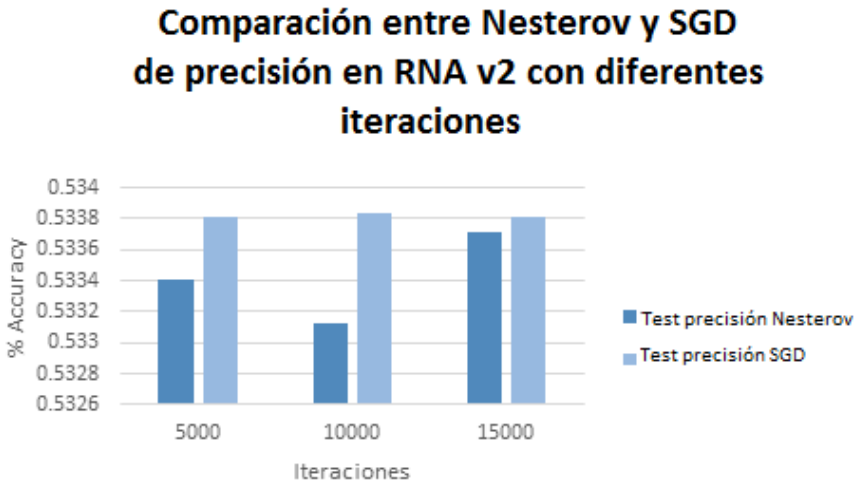


Ilustración 58 - Comparativa de accuracy en RNA v2

RNA V3		
ITERACIONES	TEST PRECISIÓN NESTEROV	TEST PRECISIÓN SGD
5000	0.533979	0.53381
10000	0.533911	0.53392
15000	0.533913	0.53378

Tabla 35 - Comparativa del accuracy con el algoritmo de Nesterov y SGD en RNA v3

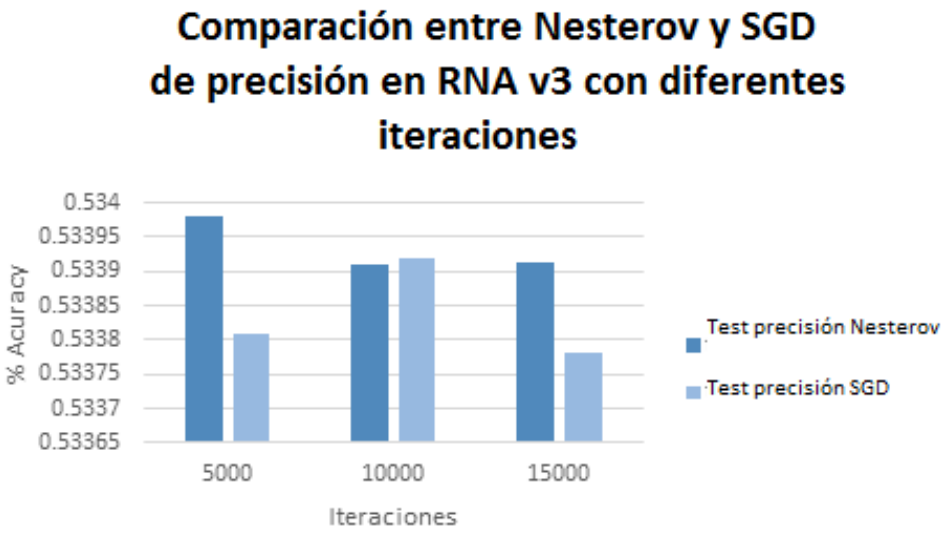


Ilustración 59 - Comparativa de precisión en RNA v3

	PROMEDIOS	
RNA	PRECISIÓN NESTEROV	PRECISIÓN SGD
RNA v1	0.5264463	0.5349697
RNA v2	0.533417	0.5338167
RNA v3	0.53393433	0.53383667

Tabla 36 - Comparativa del precisión de las diferentes versiones de la RNA

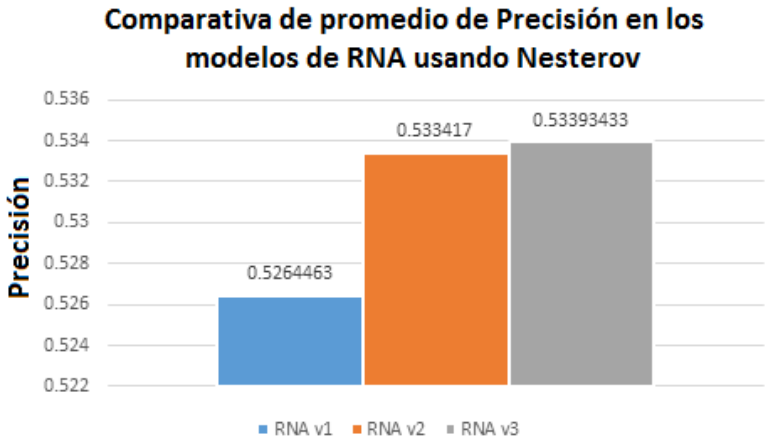


Ilustración 60 - Comparativa de promedios de precisión en los modelos de RNA

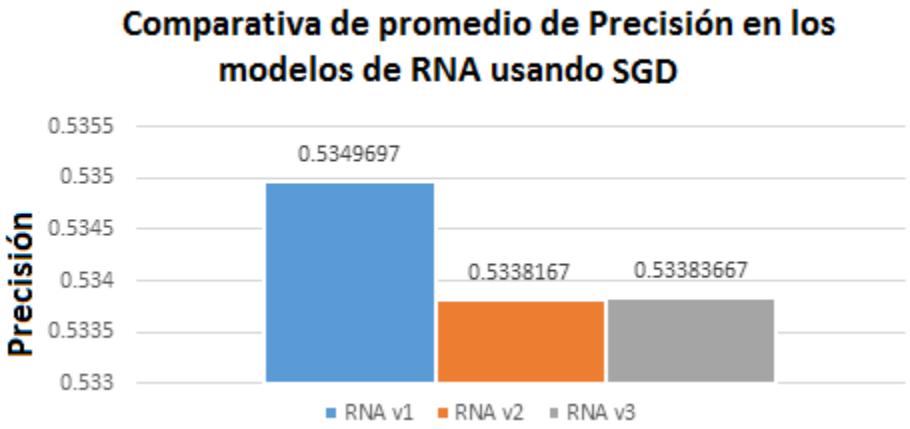


Ilustración 61 - Comparativa 2 de promedios de precisión en los modelos de RNA

CAPÍTULO III: ANÁLISIS

3.1. Modelado del negocio

3.1.1 Situación actual

3.1.1.1. Descripción general

El Ministerio de Salud (MINSAL) es una de las instituciones con mayor trayectoria en el país, para el año 2016 la red de servicios cuenta con 30 hospitales repartidos en los 14 departamentos de El Salvador. Su principal objetivo es garantizar el derecho a la salud de toda la población a través de un sistema nacional de salud.

Para poder cumplir con la alta demanda de los servicios que presta el MINSAL, la Dirección de Tecnologías de Información y Comunicaciones (DTIC) del MINSAL ha desarrollado una serie de herramientas de software; cada una con el objetivo de automatizar procesos y hacer un uso eficiente de los recursos que dispone. El proyecto más ambicioso que actualmente se está desarrollando es el Sistema Integral de Atención al Paciente (SIAP) que tiene el objetivo de brindar información de sus pacientes, sus citas, análisis, exámenes, etc. Este culminará en el Sistema Único en Salud (SUS).

Una de las áreas más beneficiadas con el proyecto es la de radiología, la cual utiliza la tecnología imagenologíca para diagnosticar y tratar enfermedades. El MINSAL cuenta también con el equipo para realizar TACs con los que se tiene ventaja sobre los rayos X convencionales en cuanto al detalle de las imágenes y además se puede revelar lesiones lo suficientemente rápido para salvar vidas. Aunque actualmente, no todos los hospitales de la red nacional cuentan con radiólogos o con este equipo especializado.

La DTIC, ha recibido recientemente el apoyo de la Universidad de El Salvador con el desarrollo del Sistema Informático de Imagenología Digital (SIMAGD) que permite vincular al expediente clínico de cada paciente registrado en el SIAP con la captura, almacenamiento y recuperación de imágenes médicas, siendo el Hospital Nacional San Rafael de Santa Tecla el primero en ponerlo en producción.

En el SIMAGD está modelada la lógica de todos los procesos administrativos de la unidad de radiología (como programación de citas, asignación de recursos de la unidad) y también la administración de archivos de almacenamiento de radiografías, tomografías, fluoroscopías, mamografías, entre otras (DICOM).

3.1.1.2. Perspectiva del producto

SIADTACC estará compuesto por dos elementos que tendrán funciones específicas y delimitadas, estos son:

Frontend: Compuesto por interfaces gráficas y demás elementos que servirán como punto de acceso a la red neuronal. Estos elementos se integrarán con SIMAGD y será desde esta aplicación que se pueda acceder a él. Específicamente la integración se realizará a través Weasis, visor de archivos DICOM que SIMAGD implementa. A partir del desarrollo de un plugin para dicho visor, se lanzarán peticiones REST a las instancias del servidor de SIADTACC el cual las gestionará y retornará los resultados de los procesamientos de imágenes para ser presentados al cliente.

Backend: Compuesto por un balanceador de carga, 3 instancias de Gunicorn como servidores http, una base de datos gestionada por Postgres y la red neuronal la cual recibirá como entrada archivos DICOM proporcionados por SIMAGD, los analizará y brinda como salida un reporte de resultados.

Adicional a estos elementos se deben de considerar las interacciones que SIADTACC tendrá con SIMAGD, siendo este último el encargado de proveer los archivos DICOM para su posterior procesamiento.

Estas relaciones se aprecian a detalle en la Ilustración 72 - Diagrama de componentes

3.1.1.3. Funcionalidad del producto

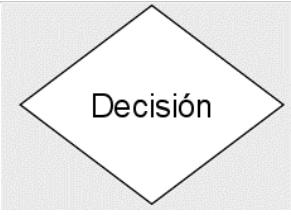
- Recepción, Procesamiento y Envío de archivos DICOM: SIADTACC será capaz de acceder a los archivos DICOM previa petición del visor Weasis, estos serán procesados y enviados como entrada en la red neuronal.
- Análisis de DICOM por la RNA: Luego de recibir el archivo DICOM ya procesado, la red neuronal se encargará de analizarlo a través de sus capas.
- Clasificación de TAC y envío: El procesamiento del archivo DICOM ofrecerá como resultado la clasificación entre normal y anormal de un estudio específico. Posteriormente este resultado en forma de reporte, se enviará al Frontend del sistema.

3.1.2 Modelado del proceso actual

En el modelado del proceso actual, se detalla el flujo de trabajo, los procesos principales, los roles y las relaciones dentro de la unidad de negocio. Al construir un sistema informático, se puede usar el modelado del proceso de la unidad para conocer y documentar que hace la institución.

3.1.2.1. Nomenclatura del workflow

Para modelar el proceso de la unidad de imagenología se utilizan los siguientes elementos:

Símbolo	Descripción
	Representa el inicio y fin de las actividades del workflow
	Representa una bifurcación en el flujo normal de las actividades partiendo de una decisión ante determinado evento.

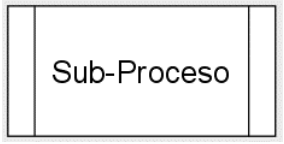
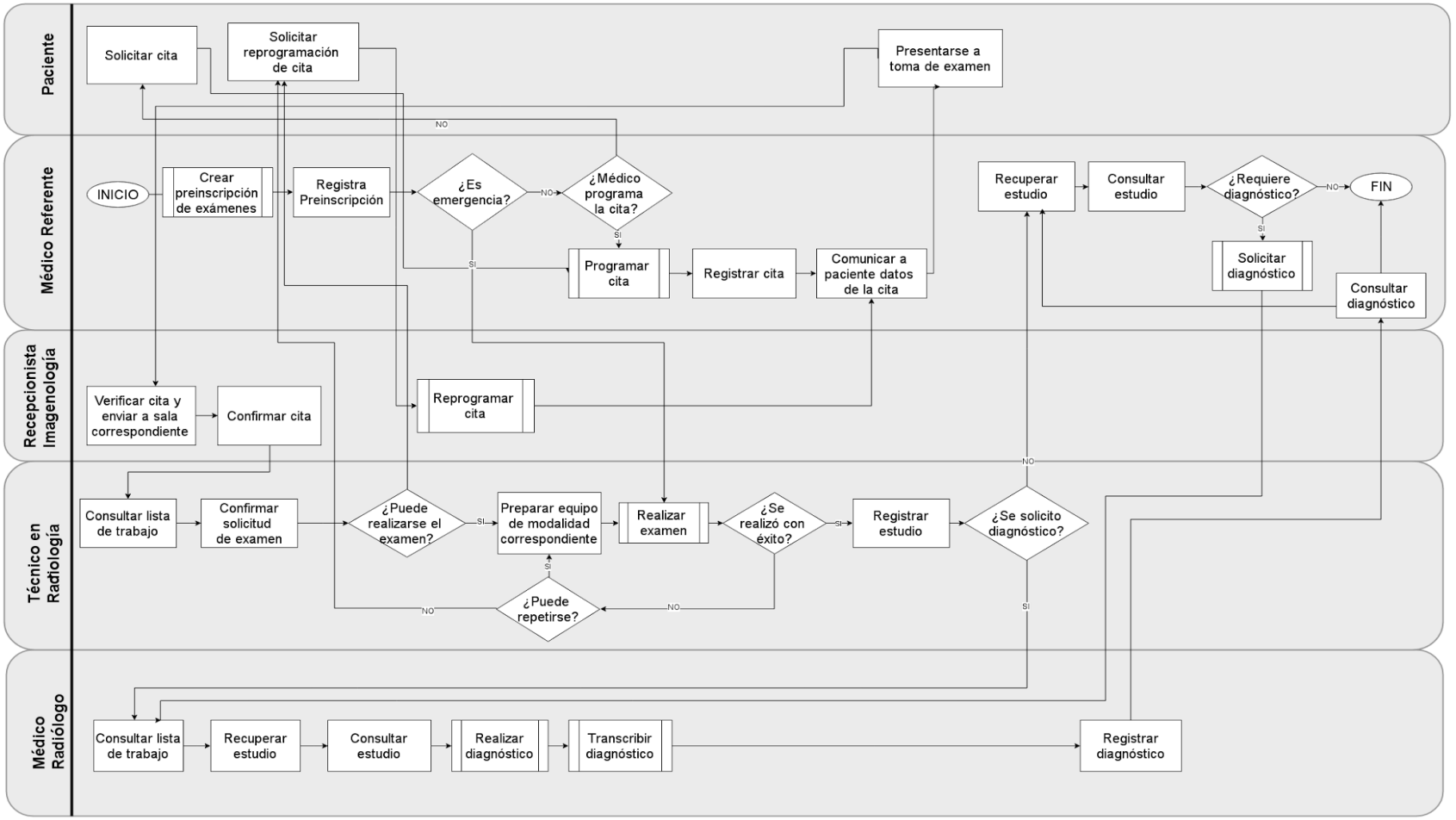
	Representa una actividad del workflow.
	Representa un sub-proceso, es decir, un conjunto de actividades de que poseen un fin común.
	Representa un carril (lane), el cual tiene por objetivo agrupar las actividades, sub-procesos y decisiones que debe realizar determinado actor dentro de workflow.

Tabla 37 - Nomenclatura del diagrama de workflow

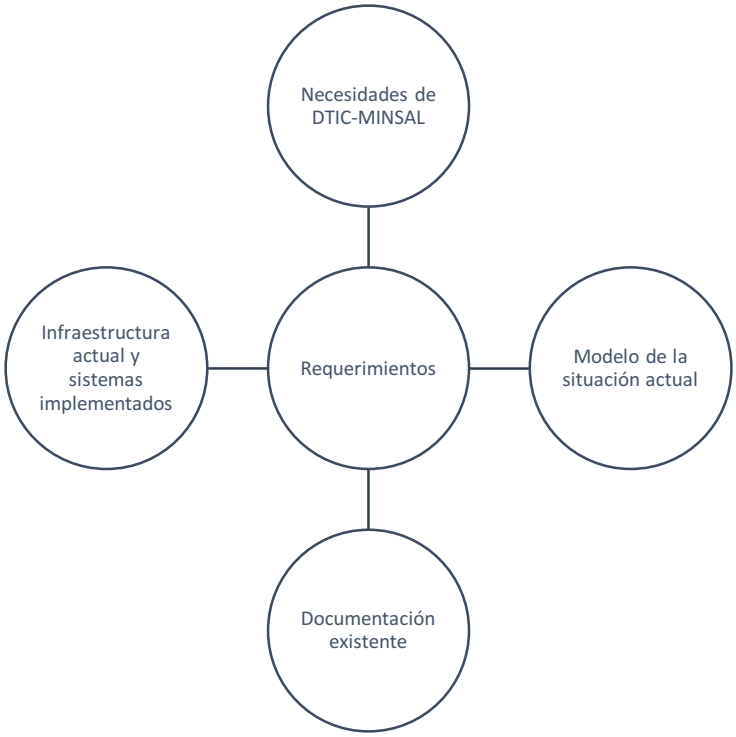
3.1.2.2. Diagrama del workflow



62 - Diagrama del workflow

3.2 Metodología para la determinación de requerimientos

Las fuentes de requerimientos de este proyecto se muestran el siguiente diagrama:



63 - Fuentes de requerimientos

Gran parte de la funcionalidad de SIADTACC, depende de la implementación tecnológica actual del MINSAL, es por eso que el modelo de la situación actual y la infraestructura fueron puntos clave para la determinación de requerimientos.

Además, se contó con la documentación de SIMAGD la cual detalla gran parte de los componentes a integrar con SIADTACC.

Las funciones más especializadas nacieron del planteamiento de necesidades de la contraparte y se determinaron mediante entrevistas no estructuradas en las cuales se intercambiaron ideas por parte de los usuarios y el equipo de desarrollo. Las ideas y datos importantes fueron apuntados, luego depurados y analizados para formar un listado de requerimientos formales y ser aprobados por la DTIC. El detalle de las estas sesiones puede consultarse en el ANEXO 1.

Por otra parte, otra fuente de información fue la comunicación vía correo electrónico. Se mantuvo un flujo constante de correos electrónicos con solicitudes y respuestas. Las bitácoras de esta vía de comunicación pueden consultarse en el ANEXO 2.

La información recolectada en las entrevistas y correos electrónicos sirvió para determinar requerimientos preliminares. Para esto se utilizó un formulario que tiene la estructura detallada en el ANEXO 3.

3.3. Requerimientos informáticos

Describen las funciones o características que debe poseer el sistema informático, estos fueron proporcionados por los usuarios del negocio, que corresponden al personal de la DTIC del MINSAL, personal técnico y del área de imagenología del Hospital Nacional San Rafael; dichos requerimientos están clasificados en dos tipos:

1. **Requerimientos funcionales:** Describen las características específicas del sistema, los procesos, procedimientos y usuarios involucrados para lograr el objetivo del requerimiento.
2. **Requerimientos no funcionales:** Están relacionados con características que pueden limitar o afectar el rendimiento general y están asociados con la calidad del sistema, algunos ejemplos de este tipo de requerimientos son: tiempos de ejecución y repuesta, espacio de almacenamiento, interfaces de usuario, robustez del sistema, disponibilidad de equipo, mantenimiento, seguridad, portabilidad, estándares, etc.

En la siguiente tabla se encuentra el listado de los requerimientos funcionales y no funcionales del proyecto.

Identificador	Tipo	Requerimiento
RF001	Funcional	Acceso a archivos DICOM almacenados en el servidor PACS
RF002	Funcional	Clasificación de TAC cerebrales
RF003	Funcional	Entrenamiento de la RNA
RF004	Funcional	Mecanismo de feedback (retroalimentación)
RNF001	No funcional	Infraestructura de red funcional
RNF002	No funcional	Sistema operativo Debían GNU/Linux
RNF003	No funcional	Mono sesión
RNF004	No funcional	Usabilidad
RNF005	No funcional	Documentación de la RNA
RNF006	No funcional	Notificaciones y navegabilidad
RNF007	No funcional	Visor DICOM Weasis

Tabla 38 - Listado de requerimientos funcionales y no funcionales

El formato utilizado para la descripción de los requerimientos es el siguiente.

Identificador	Identificador del requerimiento
Requerimiento	Nombre del requerimiento
Descripción	Descripción detallada del requerimiento
Prioridad	Nivel de precedencia en el desarrollo del proyecto
Alcance	Indica todo lo que incluirá el requerimiento

Tabla 39 - Formato para describir los requerimientos

3.3.1. Requerimientos funcionales

Identificador	RF001
Requerimiento	Acceso a archivos DICOM almacenados en el servidor PACS.
Descripción	El sistema deberá poseer la capacidad de acceder a los archivos DICOM de los TACs almacenados en el servidor PACS.
Prioridad	Alta
Alcance	Se tomarán en cuenta únicamente los TAC cerebrales sin contraste y los estudios deben estar almacenados en el PACS.

Tabla 40 - Descripción del requerimiento RF001

Identificador	RF002
Requerimiento	Clasificación de TAC cerebrales.
Descripción	El sistema deberá poseer la capacidad de clasificar los TAC cerebrales en dos criterios, los que poseen alguna anomalía y los que no.
Prioridad	Alta
Alcance	Se tomarán en cuenta únicamente los TAC cerebrales sin contraste y los estudios deben estar almacenados en el PACS.

Tabla 41 - Descripción del requerimiento RF002

Identificador	RF003
Requerimiento	Entrenamiento de la RNA.
Descripción	El sistema debe poseer un procedimiento para realizar el entrenamiento de la RNA tomando en consideración los resultados del feedback realizado por los radiólogos.
Prioridad	Alta
Alcance	El entrenamiento será realizado forma periódica (cada año).

Tabla 42 - Descripción del requerimiento RF003

Identificador	RF004
Requerimiento	Mecanismo de feedback.
Descripción	El sistema debe poseer un procedimiento de retroalimentación que permita evaluar el desempeño de la RNA con el fin de realizar un reentrenamiento y mejorar su precisión.
Prioridad	Alta
Alcance	El feedback será una calificación otorgada por el radiólogo la cual emitirá en cada uso del sistema.

Tabla 43 - Descripción del requerimiento RF004

3.3.2. Requerimientos no funcionales

Identificador	RNF001
Requerimiento	Infraestructura de red funcional.
Descripción	El sistema debe funcionar en un entorno web por lo tanto es necesario una infraestructura de red que soporte la arquitectura cliente-servidor de 3 capas.
Prioridad	Alta
Alcance	Se debe utilizar la infraestructura de red que posee el MINSAL y limitar el acceso al sistema únicamente a esta.

Tabla 44 - Descripción del requerimiento RNF001

Identificador	RNF002
Requerimiento	Sistema operativo GNU/Debian Linux.
Descripción	La RNA y el sistema informático debe ser capaz de funcionar sobre el sistema operativo Debian GNU/Linux, dado que este es el utilizado por el MINSAL.
Prioridad	Alta
Alcance	Se debe utilizar la última versión de la rama estable, actualmente es la Debian Jessie 8.6

Tabla 45 - Descripción del requerimiento RNF002

Identificador	RNF003
Requerimiento	Mono sesión.
Descripción	El sistema no soporta concurrencia en una sola instancia, debe poner un balanceador para manejar múltiples peticiones.
Prioridad	Media
Alcance	N/A

Tabla 46 - Descripción del requerimiento RNF003

Identificador	RNF004
Requerimiento	Usabilidad y navegabilidad.
Descripción	El sistema debe ser fácil de utilizar, intuitivo y agradable a la vista.
Prioridad	Media
Alcance	Se deben seguir las guías de estilo del SIMAGD, DCM4CHEE y Weasis según sea el componente desarrollado.

Tabla 47 - Descripción del requerimiento RNF004

Identificador	RNF005
Requerimiento	Documentación de la RNA.
Descripción	Debe contar con manuales de implementación, uso y entrenamiento de la RNA estructurados adecuadamente.
Prioridad	Alta
Alcance	N/A

Tabla 48 - Descripción del requerimiento RNF005

Identificador	RNF006
Requerimiento	Notificaciones.
Descripción	Se deben proporcionar mensajes orientados al usuario final, con el objetivo de describir y guiar respecto al uso correcto de las características del sistema.
Prioridad	Media
Alcance	N/A

Tabla 49 - Descripción del requerimiento RNF006

Identificador	RNF007
Requerimiento	Visor DICOM Weasis.
Descripción	El sistema debe acoplarse al visor weasis que utiliza el sistema SIMAGD.
Prioridad	Alta
Alcance	La versión del visor debe ser la 2.0.3 en conjunto con el Servidor PACS

Tabla 50 - Descripción del requerimiento RNF007

3.4. Requerimientos operativos

3.4.1. Requerimientos de mantenimiento

N°	Descripción
1	Se hará un entrenamiento anual a la red neuronal para mejorar la cantidad de aciertos.
2	Llevar un control sobre aciertos y errores de clasificación de la red neuronal.

Tabla 51 - Listado de requerimientos de mantenimiento

3.4.2. Requerimientos de seguridad

N°	Descripción
1	No debe dar accesos a las tomografías de los pacientes.
2	Debe garantizar los mismos niveles de seguridad del SIMAGD.
3	Limitar el acceso al sistema únicamente a la estructura de red del MINSAL.

Tabla 52 - Listado de requerimientos de seguridad

3.4.3. Requerimientos operativos de recurso humano

N°	Descripción
1	Técnicos de la unidad informática del MINSAL.
2	Técnicos de las unidades de radiología de los hospitales del MINSAL.
3	Médicos radiólogos de los hospitales del MINSAL.

Tabla 53 - Listado de requerimientos operativos de recurso humano

3.5. Requerimientos de desarrollo

3.5.1. Requerimientos de hardware de desarrollo

Equipo	Cantidad	Especificaciones
Servidor	1	● RAM 8GB 1600mhz ● Core i5 4690 3.5 GHz 4 núcleos ● SSD 256GB ● HDD 2TB ● Tarjeta de red integrada ● Nvidia gtx760 2gb
PC para desarrollo y documentación	5	EQUIPO 1 ● RAM 4gb ● Core i5 3.3Ghz 4 núcleos ● HDD 1TB ● Tarjeta de red integrada ● Nvidia gtx760 EQUIPO 2 ● RAM 8gb ● Core i7 2.7Ghz 8 núcleos ● HDD 750GB ● Tarjeta de red integrada. EQUIPO 3 ● RAM 4GB ● AMD Radeon 1.67GHz 2 núcleos ● Tarjeta de red integrada ● HDD 500GB EQUIPO 4 ● RAM 6GB ● Core i5 2.5GHz ● Tarjeta de red integrada ● HDD 500 GB EQUIPO 5 ● RAM 4GB ● Core i5 2.5 GHZ ● Tarjeta de red integrada ● HDD 500GB
Switch	1	● 8 puertos ethernet 10/100mbps ● No gestionable

Tabla 54 - Requerimientos de hardware de desarrollo

3.5.2. Requerimientos de software de desarrollo

N°	Nombre	Versión	Descripción
1	Debian/Linux	8.6 Jessie 64bits	Sistema operativo con repositorios del Ministerio de Salud de El Salvador
2	PostgreSQL	9.4+165+deb8u1	Gestor de base de datos
3	Python	2.7.3	Lenguaje de programación

4	Java (JDK)	1.6.0_38+	Lenguaje de programación, última versión disponible en los repositorios de Debian Jessie (hasta la fecha).
5	Apache Server	2.2.22-13+deb7u1+	Servidor web HTTP de código abierto.
6	JBOSS	4.2.3 GA	Servidor de aplicaciones Java EE de código abierto implementado en la especificación Java EE. Utilizado para ejecutar DCM4CHEE y sus componentes.
7	DCM4CHEE	2.18.0	Colección de aplicaciones de código abierto y utilidades para las instituciones de salud.
8	Weasis	2.0.3	Visor web de archivos en formato DICOM
9	CAFFE	0.9999	Caffe es un framework para el desarrollo de aplicaciones de deep learning, posee una interfaz de línea de comandos para el lenguaje de programación python (pycaffe).
10	CUDA	7+	Modelo de computación y programación paralela desarrollado por NVIDIA. Permite un aumento drástico en cuanto a rendimiento de cómputo, aprovechando las capacidades de la unidad de procesamiento gráfico (GPU).
11	Pycharm	2016.2.3	IDE para el lenguaje de programación Python
12	NetBeans	8.2+	IDE para el lenguaje de programación JAVA
13	Nginx	1.6.2-5+deb8u2+	Servidor web/proxy inverso, ligero de alto rendimiento y un proxy para protocolos de correo electrónico (IMAP/POP3)
14	Gunicorn	19.6+	Servidor HTTP de Python WSGI.
15	Django	1.10.2	Framework web de Python de alto nivel que fomenta el desarrollo rápido, diseño limpio y pragmático.
16	GIT	1:2.1.4-2.1+deb8u2+	Sistema de control de versiones
17	Sun's Java Advanced Imaging (JAI) Image I/O Tools	1.0_01	

Tabla 55 - Requerimientos de software de desarrollo

3.5.3. Requerimientos de recurso humano de desarrollo

Cantidad	Cargo	Descripción
2	Analistas/Programadores JAVA	Debe realizar actividades de análisis, diseño, codificación en el lenguaje de programación JAVA y construcción del sistema informático; además de la realización de pruebas de la solución.
1	Analistas/Programadores Python	Debe realizar actividades de análisis, diseño, codificación en el lenguaje de programación Python y construcción del sistema informático; además de la realización de pruebas de la solución.
1	Desarrollador de Red Neuronal	Debe realizar actividades de diseño y codificación de la RNA utilizando el framework CAFFE; además de la realización de pruebas y entrenamiento.
1	Administrador de proyecto	Debe realizar la coordinación del equipo como planificación, asignación y seguimiento de las actividades.

Tabla 56 - Requerimientos de recurso humano para actividades de desarrollo

3.6. Requerimientos de implementación

Los requerimientos necesarios para la puesta en producción del software SIADTACC están divididos en tres grupos, los cuales se detallan a continuación:

3.6.1. Requerimientos de hardware de implementación

Se tienen dos tipos de equipo, los servidores donde se instalará la RNA y el SIADTACC y las computadoras que se utilizarán como cliente en las estaciones de trabajo. A continuación, se detalla las especificaciones de cada uno de ellos.

Equipo	Cantidad	Especificaciones
Servidor para la RNA (VPS)	1	● Procesador: 6 núcleos ● **RAM:** 16GB 1600mhz ● Disco Duro: 200GB ● Tarjeta de red Ethernet integrada.
Servidor para el SIADTACC	1	● **Procesador:** 1 Intel Xeon E5-2620, 2.4GHz, 6C/12T, 15 MB Cache ● **RAM:** 16 GB (2x8GB), 2133 MT/s, Dual Rank RDIMM ● **Controladora de discos duros:** Controladora RAID 0, 1, 5, 6, 10 60, 512MB de Cache, con capacidad para 8 HD, Hot Swap.

		● **Discos duros:** 2 x 500GB 7.2K RPM Serial-Attach SCSI 3Gbps 3.5in (Configurados en RAID 1), Hot-plug. ● Adaptador de red a 1 Gigabit cuádruple puerto, cobre, PCIe-4.

Tabla 57 - Requerimientos de hardware para los servidores de implementación

Equipo	Cantidad	Especificaciones
Computadora cliente	20	● **Procesador:** Intel Dual Core PDC 3.2GHZ G3250 S-1150 3M. ● **RAM:** 4 GB DDR3 1333. ● **Disco duro:** 500 GB 7500 RPM 32MB CACHE SATA. ● Tarjeta de red Ethernet integrada.

Tabla 58 - Características de las computadoras cliente

3.6.2. Requerimientos de software de implementación

Los requerimientos de software en el entorno de producción son los mismos que en el entorno de desarrollo.

3.6.3. Requerimientos de recurso humano de implementación

Cantidad	Cargo	Descripción
4	Analistas programadores	Encargados de realizar las actividades de instalación y configuración de los componentes del SIADTACC. Es recomendable que 2 analistas sean parte del equipo de desarrollo del proyecto y los otros 2 de la DTIC del MINSAL.
1	Administrador de proyecto	Responsable de coordinar las actividades del plan de implementación, es necesario que tenga experiencia con el manejo del SIMAGD.

Tabla 59 - Requerimientos de recurso humano para actividades de implementación

CAPÍTULO IV: DISEÑO

4.1. Diseño de la RNA

4.1.1. Notación del diagrama de la RNA

Símbolo	Descripción	Significado
	Flecha.	Vincula dos capas, el origen de la fecha es el bottom blob y el final de la flecha es el top blob
	Rectángulo verde.	Capa de entrada de datos.
	Rectángulo morado.	Capa de normalización
	Rectángulo anaranjado.	Capa convolucional
	Rectángulo amarillo.	Capa de pooling
	Rectángulo rojo.	Capa de producto escalar (capa completamente conectada)
ReLU <<N>>	Texto “/ReLUX” posterior al nombre de una capa	Capa ReLU, reemplaza valores de salida de la capa anterior.
Drop <<N>>	Texto “/dropX” posterior al nombre de una capa	Capa de dropout, reemplaza valores de salida de la capa anterior.

Tabla 60 - Nomenclatura del diagrama de la RNA

4.1.2. Diagrama de la RNA

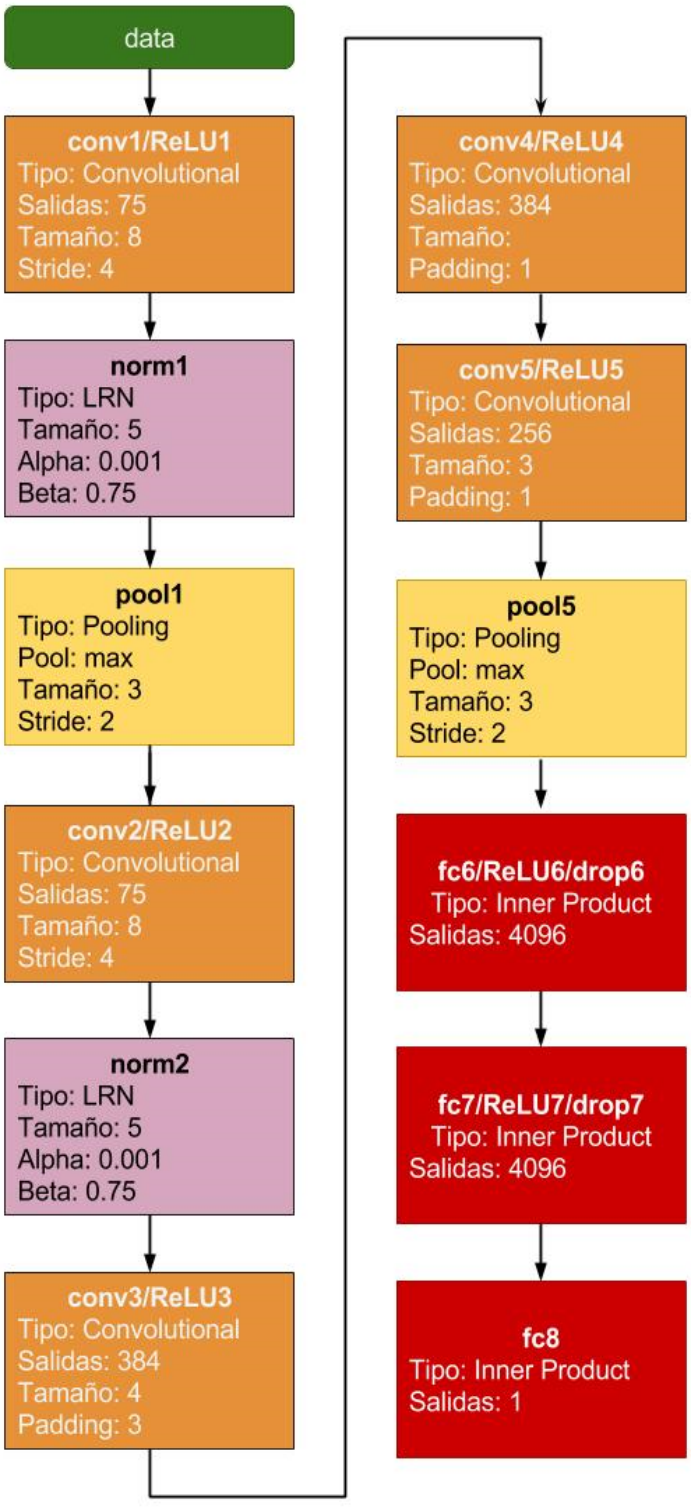


Ilustración 64 - Diagrama de RNA

4.2. Diseño del sistema

4.2.1. Arquitectura del sistema informático

4.2.1.1. Arquitectura cliente/servidor

La arquitectura cliente-servidor propone repartir la funcionalidad de la aplicación en al menos dos elementos diferenciados, interconectados entre sí a través de una infraestructura de red. De acuerdo a la función realizada, estos reciben el nombre de cliente y servidor.

Estos dos elementos pueden residir en equipos distintos o ser distinguidos de forma lógica en un solo equipo. Cualquiera sea el caso el funcionamiento se basa en la idea de centralizar el procesamiento y datos y volverlo accesible a distintos clientes conectados, que de forma concurrente envían peticiones al servidor y reciben respuestas con los recursos solicitados.

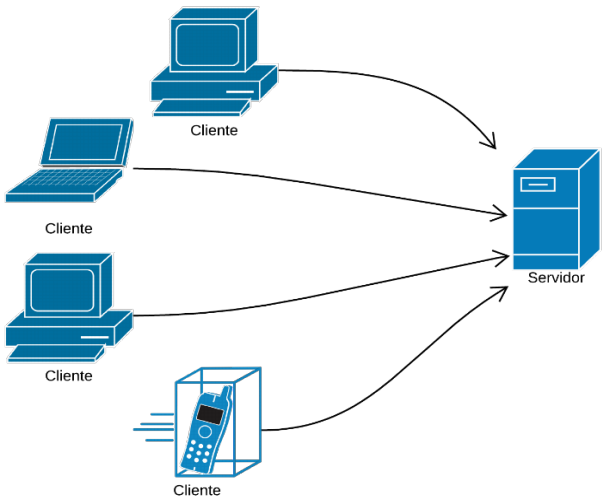


Ilustración 65 - Arquitectura cliente servidor

4.2.1.2. Cliente/servidor de 3 capas

Esta variación de la arquitectura cliente/servidor, separa la aplicación en 3 capas atendiendo a la funcionalidad que realiza cada elemento de software, estas son las siguientes:

- Capa cliente: al igual que en la arquitectura cliente/servidor es el punto de acceso de la aplicación y se encarga en gestionar peticiones realizadas al servidor y darle tratamiento a las respuestas que provengan de este.

- Capa de aplicación: Encargada de implementar la lógica del negocio, recibe peticiones de la capa cliente y realiza peticiones a la capa de datos para poder servir las peticiones aceptadas, transforma y le de tratamiento a los datos atendiendo la lógica del negocio.
- Capa de datos: generalmente implementada a través de un sistema gestor de base de datos, es un almacén de datos que garantiza la integridad de estos y los pone a disposición de la capa de aplicación para ser servidos a la capa cliente.

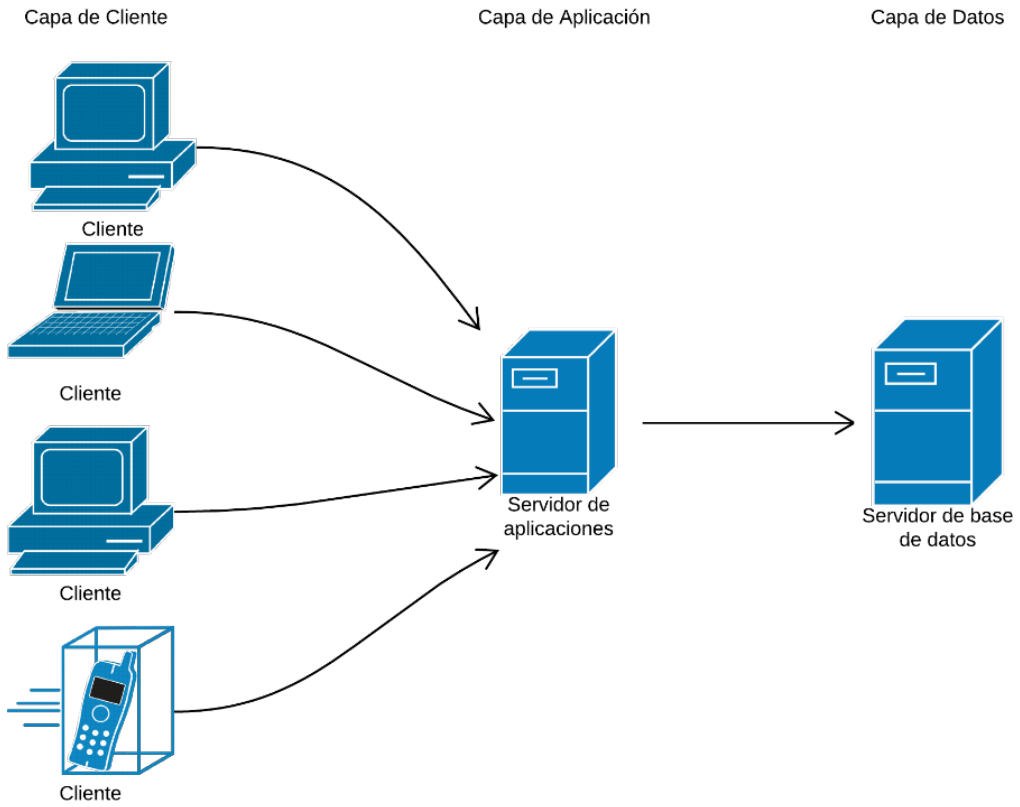


Ilustración 66 - Arquitectura de 3 capas

4.2.1.3. Arquitectura SIADTACC

La arquitectura del sistema se basará en la arquitectura cliente servidor de 3 capas, a continuación, se presenta en forma de diagrama los elementos que compondrán cada capa y seguido de esto se explica cada uno de estos:

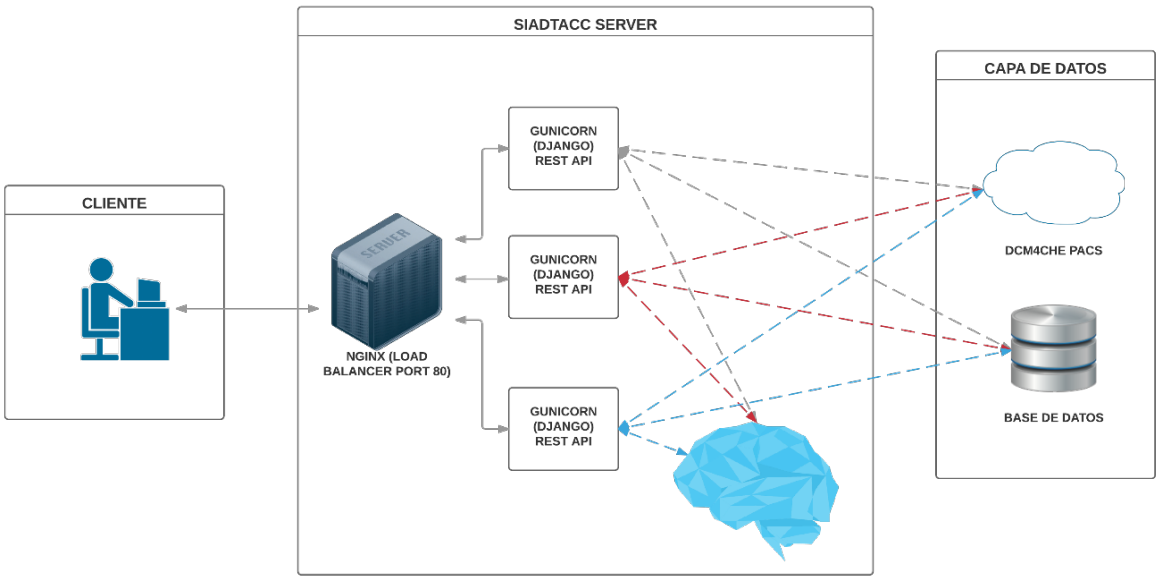


Ilustración 67 - Arquitectura de SIADTACC

Cliente: Como su nombre lo indica, este elemento será el encargado de iniciar todo el procesamiento que SIADTACC desarrollará, para luego consumir los resultados que éste ofrezca; se implementará en el visor web de archivos DICOM denominado WEASIS que a su vez forma parte del Sistema de Imagenología Digital del Ministerio de Salud; será desarrollado como un plugin de dicho visor a partir del cual serán lanzadas las peticiones REST a SIADTACC.

SIADTACC Server

Será el núcleo de la aplicación y estará compuesto por 3 elementos diferenciables:

NGINX

Como balanceador de carga, su papel consistirá en recibir peticiones desde el cliente y redirigirlas a los diferentes métodos del API REST que se implementarán en 3 diferentes instancias de Gunicorn (Servidor HTTP) a través del framework Django, además enviará las respuestas de éstas al cliente.

GUNICORN

Gunicorn consiste en un HTTP WSGI para Python, provee de una interfaz simple y universal entre distintas tecnologías desarrolladas con Python. En el servidor de SIADTACC se ejecutarán 3 instancias de Gunicorn, para mejorar la disponibilidad, disminuir los tiempos de bloqueo y mejorar la concurrencia.

REST API

Se utilizará una aplicación web que implemente un API REST para la comunicación del cliente con la red neuronal, para la programación se utilizará el framework DJANGO Python. En esta aplicación se centralizarán las operaciones del software a entregar, estas son la comunicación con la RNA, comunicación con el sistema gestor de base de datos Postgres

y comunicación con DCM4CHE PACS que será desde donde se leerán las imágenes radiológicas; además de la comunicación entre estos elementos.

RNA: Implementada en el framework Caffe, la red neuronal será el elemento que realizará la función elemental del sistema; será la encargada de darle un correcto procesamiento a las imágenes radiológicas, analizarlas y emitir un resultado a partir de las características propias de cada una; se comunicará con el resto de elementos a través de la REST API implementadas en Django Python.

4.2.1.4. Capa de datos

Base de Datos: Implementada a través del Sistema Gestor de Base de Datos (SGBD) Postgres, proveerá de persistencia al sistema, almacenará detalles de la arquitectura de la RNA, así como archivos de configuración de esta.

DCM4CHE PACS: Sistema de archivado y transmisión de imágenes (PACS por sus siglas en inglés), implementado a través de DCM4CHE, su papel consistirá en recibir peticiones desde una API y retornar los archivos DICOM para su posterior tratamiento y procesamiento por parte de la RNA.

4.2.2. Casos de uso

4.2.2.1. Actores

Actor	Descripción
Médico radiólogo	Encargado de emitir diagnósticos a partir de estudios de radiología.
Técnico de radiología	Encargado de realizar el proceso de toma de estudios del paciente.
Administrador de sistemas	Se encarga de realizar un entrenamiento periódico de la RNA.

Tabla 61 - Listado de actores del sistema

4.2.2.2. Lista de casos de uso

Código	Nombre
CU001	Lectura TAC desde servidor PACS
CU002	Procesamiento de TAC
CU003	Análisis de TAC por la RNA
CU004	Elaboración y envío de reporte de resultados
CU005	Presentación de Reporte de Resultados
CU006	Envío y recepción de Feedback
CU007	Entrenamiento de RNA

Tabla 62 - Listado de casos de uso

4.2.2.3. Diagramas de casos de uso

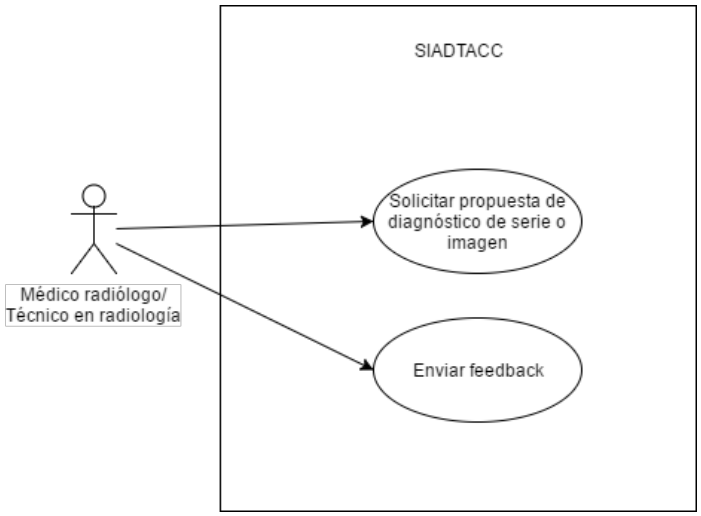


Ilustración 68 - Casos de uso Solicitud de propuesta de diagnóstico y Envío de feedback

4.2.2.4. Descripción de casos de uso

CU001: Solicitar propuesta de dianóstico de serie o imagen

CU001	Solicitar propuesta de dianóstico de serie o imagen
Requisitos asociados	RF001 Acceso a archivos DICOM almacenados en el servidor PACS.
Descripción	El usuario debe de ser capaz de solicitar una propuesta de diagnóstico a través del visor de imágenes médicas WEASIS, esta solicitud se puede hacer en base a una imagen o a una serie de imágenes.
Precondición	Para que el caso de uso inicie, es necesario que el usuario haya iniciado sesión en SIMAGD y se encuentre dentro del visor Weasis visualizando una tomografía Cerebral sin contraste.
Flujo Normal Solicitud basada en corte	1. El usuario accede al menú del plugin y hace clic en la opción “Analizar corte”. 2. El plugin envía una petición al sistema en la que se especifica el id del paciente, estudio e imagen de tal forma que esta sea reconocible inequívocamente. 3. El sistema recibe la petición del plugin en la que se especifica el id del paciente, estudio e imagen de tal forma que esta sea reconocible inequívocamente. 4. El sistema solicita al servidor PACS el estudio basado en la información recibida en el paso anterior. 5. El servidor PACS retorna las imágenes solicitadas.

	6. La RNA analiza la imagen y determina una propuesta de diagnóstico. 7. La propuesta de diagnóstico es enviada al visor.
Flujo Alternativo A Solicitud basada en serie	1. El usuario accede al menú del plugin y hace clic en la opción “Analizar serie”. 2. El plugin envía una petición al sistema en la que se especifica el id del paciente, estudio de tal forma que este sea reconocible inequívocamente. 3. El sistema recibe la petición del plugin en la que se especifica el id del paciente, estudio de tal forma que este sea reconocible inequívocamente. 4. El sistema solicita al servidor PACS el estudio basado en la información recibida en el paso anterior. 5. El servidor PACS retorna las imágenes solicitadas. 6. La RNA analiza la imagen y determina una propuesta de diagnóstico. 7. La propuesta de diagnóstico es enviada al visor.
Flujo Alternativo B	2.A El plugin no se puede conectar con el sistema. 3.A El plugin muestra una pantalla al usuario detallando el problema.
Flujo Alternativo C	4.B El sistema no puede conectar con el servidor PACS 5.B El sistema retorna al plugin información acerca del error. 6.B El plugin muestra una pantalla al usuario detallando el problema.
Postcondiciones	Finalizado el caso de uso, el sistema tendrá listo un determinado estudio para su procesamiento y posterior análisis por parte de la RNA, este se almacenará en una carpeta temporal que será limpiada finalizado el flujo de trabajo.
Frecuencia	Múltiples ejecuciones a diario.

Tabla 63 - Descripción del caso de uso CU001

CU002: Enviar Feedback

CU002	Enviar Feedback
Requisitos asociados	RF004 Mecanismo de feedback.
Descripción	El sistema debe de ser capaz de recibir retroalimentación de parte de los usuarios con respecto a la calidad de sugerencias de diagnóstico emitidas, almacenarla y luego utilizarla para procesos de entrenamiento posteriores.
Precondición	Para que el caso de uso inicie, es necesario que el usuario haya iniciado sesión en SIMAGD y se encuentre dentro del visor Weasis visualizando una tomografía Cerebral sin contraste.
Flujo Normal	1. El plugin implementado en el visor Weasis muestra una pantalla de recepción con un formulario de aceptación o rechazo de la sugerencia de diagnóstico. 2. El usuario ingresa su valoración de la propuesta de diagnóstico emitida y envía el formulario. 3. El sistema recibe y almacena la valoración para su posterior uso.

Flujo Alternativo A	2.A El usuario decide cerrar el formulario de envío de valoración.
Postcondiciones	Finalizado el caso de uso, el sistema contará con una nueva valoración para usarla en el proceso de reentrenamiento.
Frecuencia	Múltiples ejecuciones a diario.

Tabla 64 - Descripción del caso de uso CU006

4.2.3. Diagramas de secuencia

4.2.3.1. Nomenclatura del diagrama de secuencia

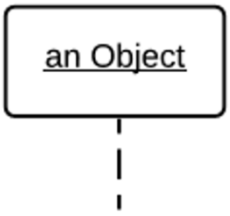
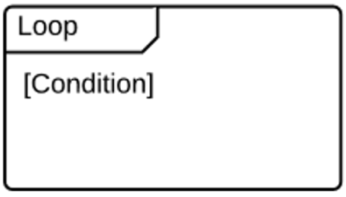
Símbolo	Nombre	Descripción
	Línea de vida	Línea vertical que representa la secuencia de eventos que se producen en un participante durante una interacción, mientras el tiempo avanza por la línea. Este participante puede ser una instancia de una clase, un componente o un actor.
	Activación	Los cuadros de activación representan el tiempo que un elemento necesita para completar una tarea.
	Mensajes	Son flechas que representan comunicaciones entre objetos. Las medias flechas representan mensajes asincrónicos. Los mensajes asincrónicos son enviados desde un objeto que no va a esperar una respuesta del receptor para continuar con sus tareas.
	Loop	Una repetición o loop en un diagrama de secuencias, es representado como un rectángulo. La condición para abandonar el loop se coloca en la parte inferior entre corchetes [].

Tabla 65 - Nomenclatura del diagrama de secuencia

4.2.3.2. Diagramas de secuencia

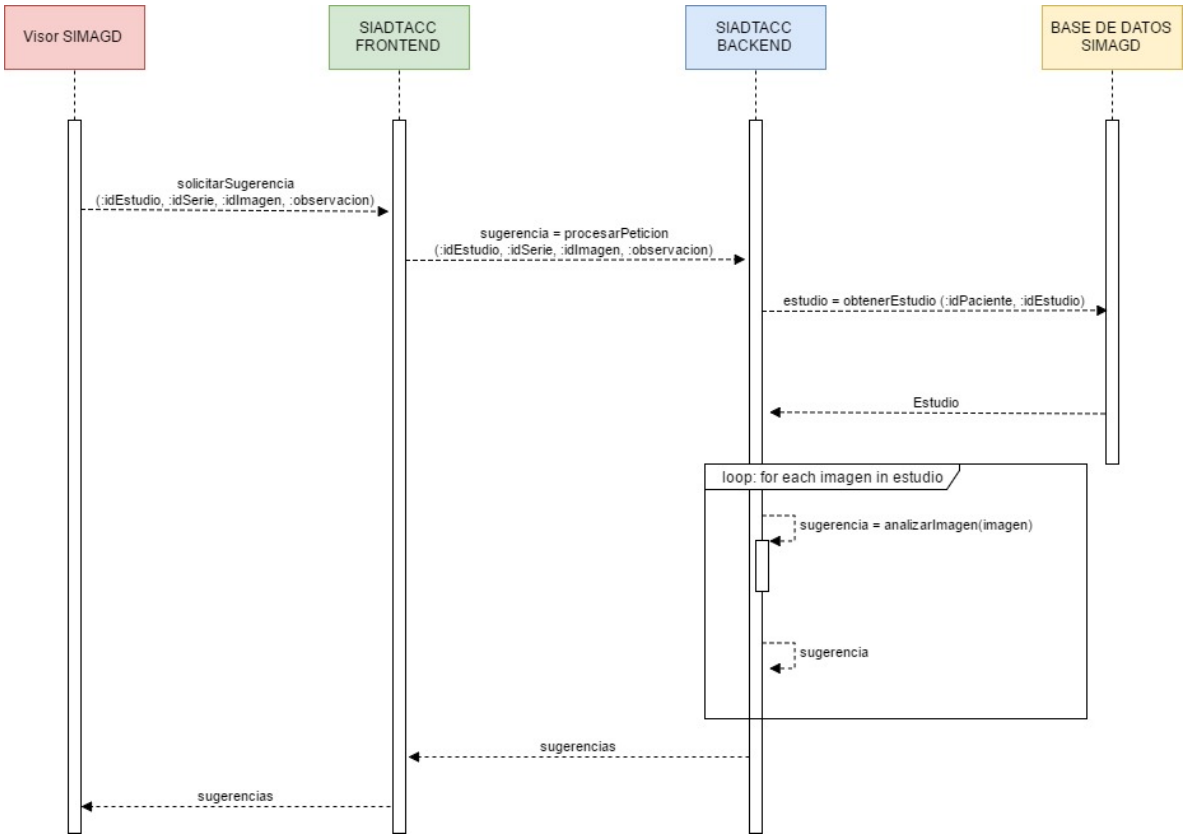


Ilustración 69 - Diagrama de secuencia para análisis de serie y corte

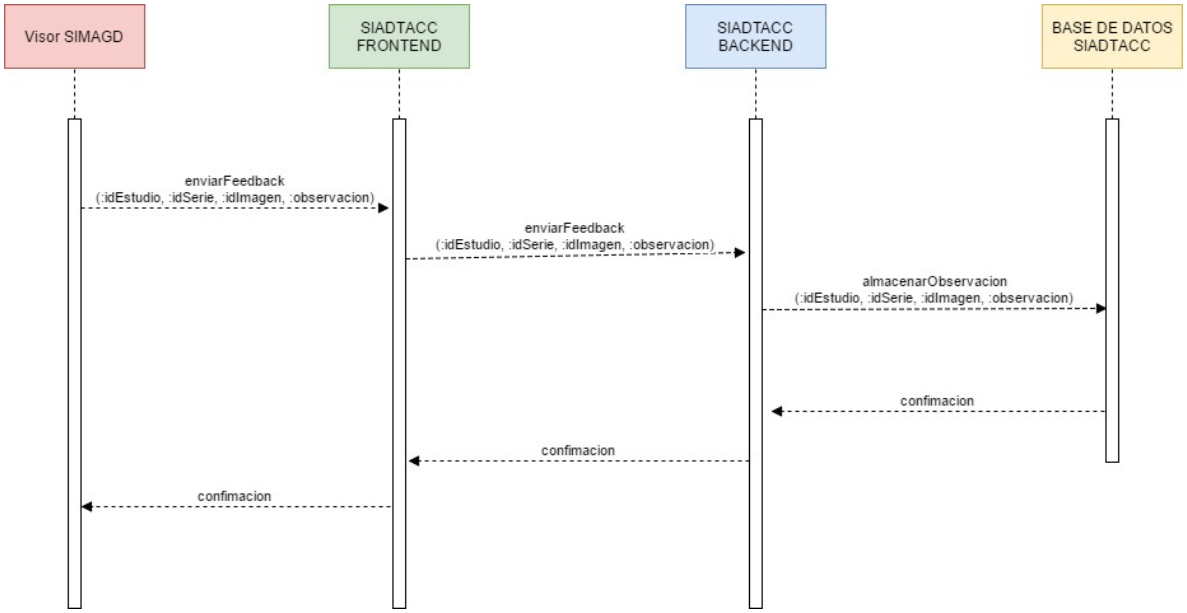
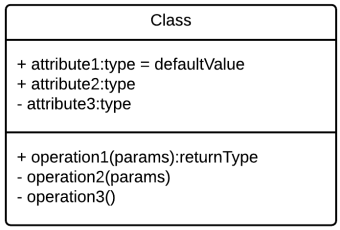
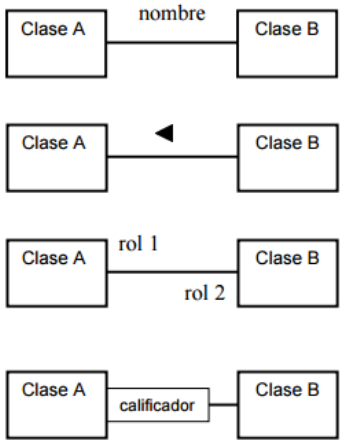


Ilustración 70 - Diagrama de secuencia para feedback

4.2.4. Diagrama de clases

4.2.4.1. Nomenclatura del diagrama de clases

Símbolo	Nombre	Descripción
	Clase	Las clases se representan con rectángulos divididos en tres áreas: la superior contiene el nombre de la clase, la central contiene los atributos y la inferior las acciones.
	Asociaciones	Las asociaciones son las que representan a las relaciones estáticas entre las clases. El nombre de la asociación va por sobre o por debajo de la línea que la representa. Una flecha rellena indica la dirección de la relación. Los roles se ubican cerca del final de una asociación. Los roles representan la manera en que dos clases se ven entre ellas. No es común el

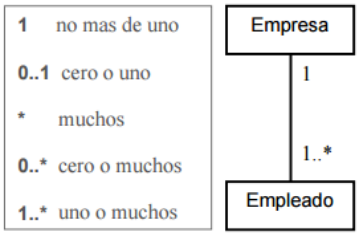
		colocar ambos nombres, el de la asociación y el de los roles a la vez. Cuando una asociación es calificada, el símbolo correspondiente se coloca al final de la asociación, contra la clase que hace de calificador.
	Multiplicidad	Las notaciones utilizadas para señalar la multiplicidad se colocan cerca del final de una asociación. Estos símbolos indican el número de instancias de una clase vinculadas a una de las instancias de la otra clase. Por ejemplo, una empresa puede tener uno o más empleados, pero cada empleado trabaja para una sola empresa solamente.

Tabla 66 - Nomenclatura del diagrama de clases

4.2.4.2. Diagrama de clases del sistema informático

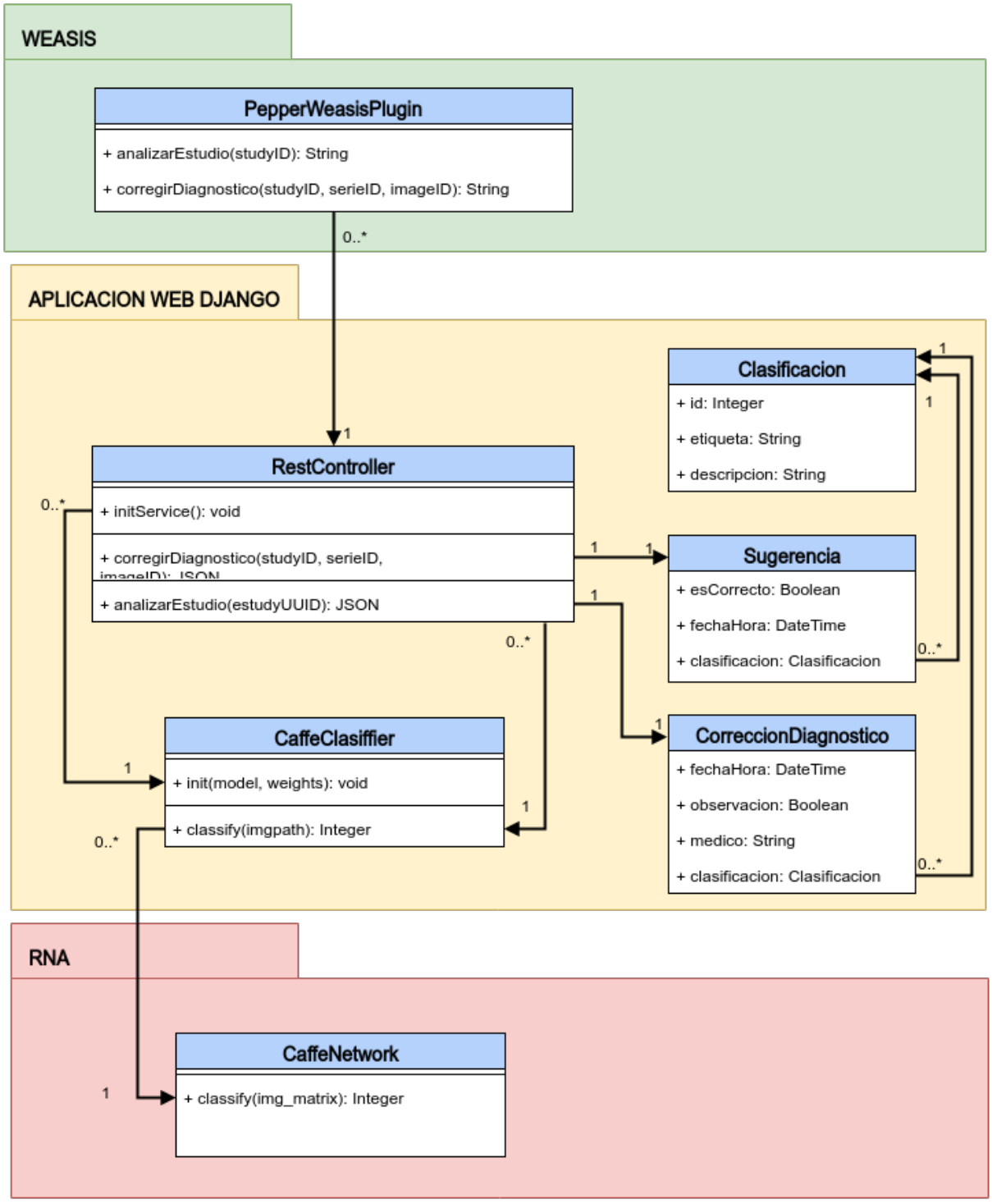


Ilustración 71 - Diagrama de clases

4.2.5. Diagrama de componentes

4.2.5.1. Nomenclatura del diagrama de componentes

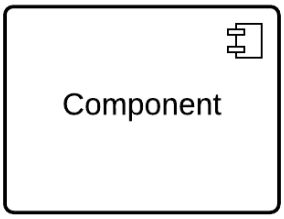
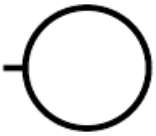
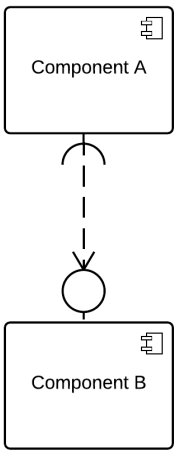
Símbolo	Nombre	Descripción
	Componente	Un componente es un bloque de construcción física del sistema.
	Interfaz requerida	Representa un grupo de mensajes o llamadas que envía el componente a otros componentes o sistemas externos. El componente está diseñado para acoplarse a componentes que proporcionan al menos estas operaciones. El puerto tiene una interfaz como su tipo.
	Interfaz provista	Representa un grupo de mensajes o llamadas que implementa un componente y que pueden usar otros componentes o sistemas externos. Un puerto es una propiedad de un componente que tiene una interfaz como su tipo.
	Dependencia	Se usa para indicar que una interfaz necesaria en un componente se puede satisfacer mediante una interfaz proporcionada en otro.

Tabla 67 - Nomenclatura del diagrama de componentes

4.2.5.2. Diagrama de componentes del sistema informático

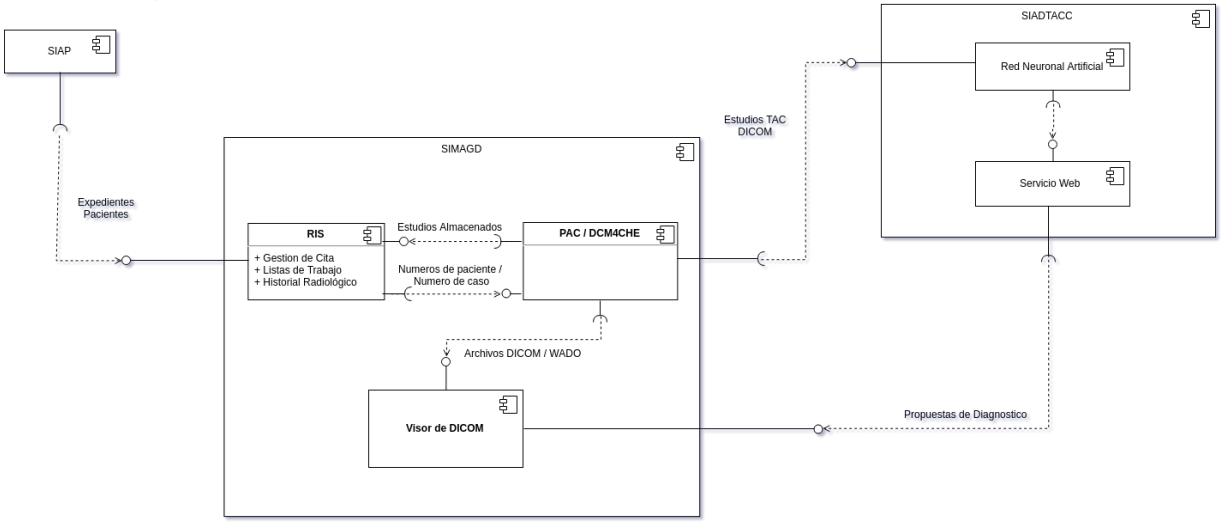


Ilustración 72 - Diagrama de componentes

4.2.6. Diagrama de despliegue

4.2.6.1. Nomenclatura del diagrama de despliegue

A continuación, se describen algunos de los elementos utilizados en el diagrama de despliegue. Para una especificación más detallada consultar la documentación oficial de UML. (Object Management Group, 2015)

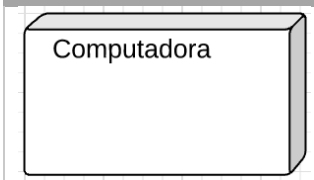
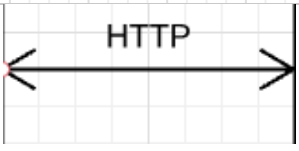
Símbolo	Nombre	Descripción
	Nodo	Es un elemento de hardware o software. Se representa con la forma de una caja en tres dimensiones.
	Asociación	Representa una ruta de comunicación entre los nodos.

Tabla 68 - Nomenclatura del diagrama de despliegue

4.2.6.2. Diagrama de despliegue del sistema informático

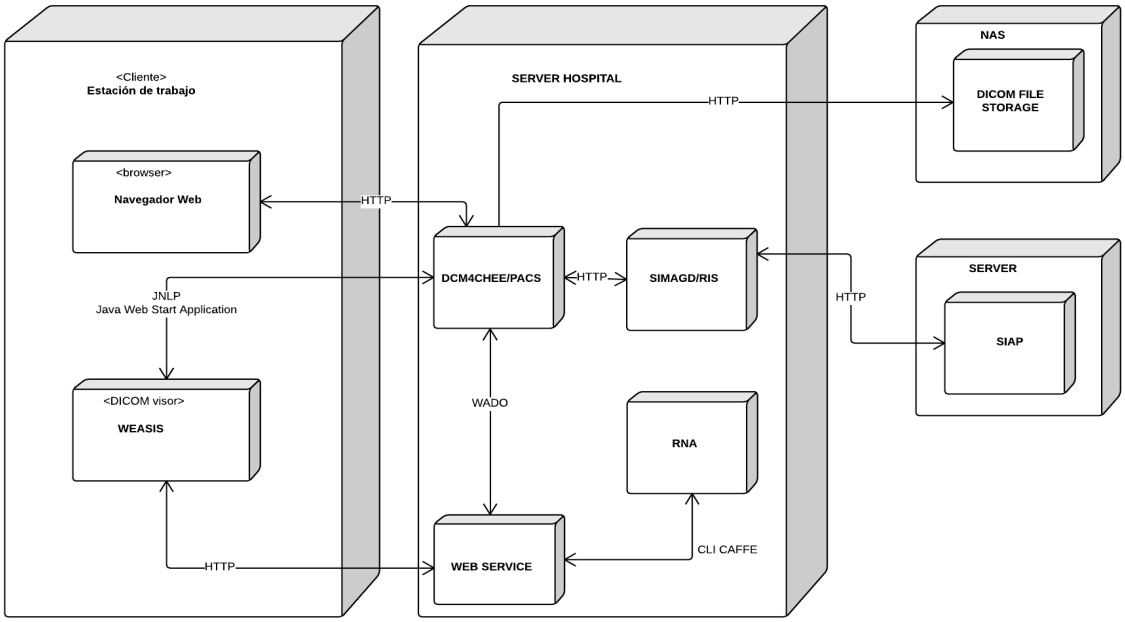


Ilustración 73 - Diagrama de despliegue

4.2.7. Diseño de base de datos

4.2.7.1. Modelo entidad-relación

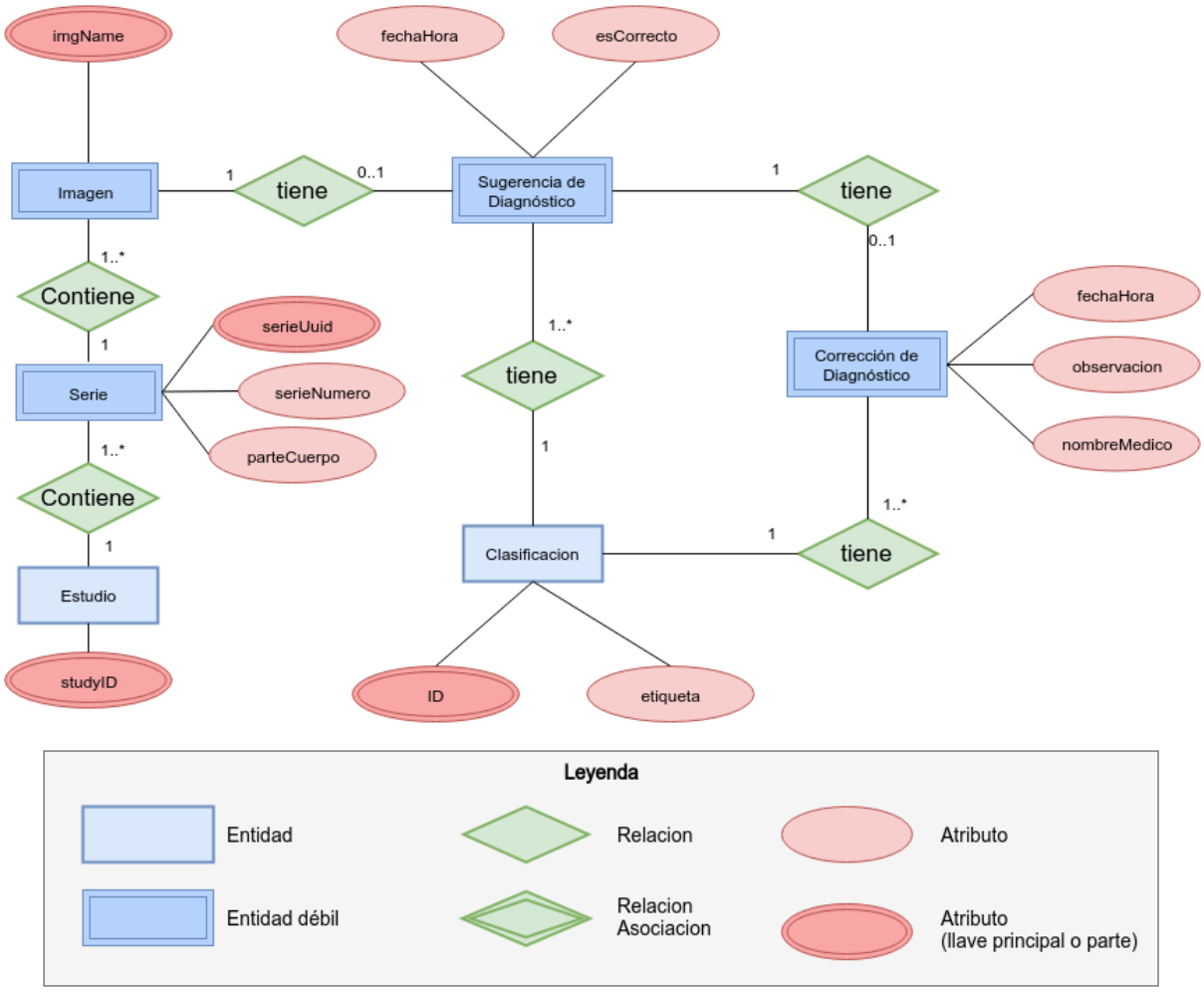


Ilustración 74 - Modelo Entidad Relación

4.2.7.2 Modelo conceptual

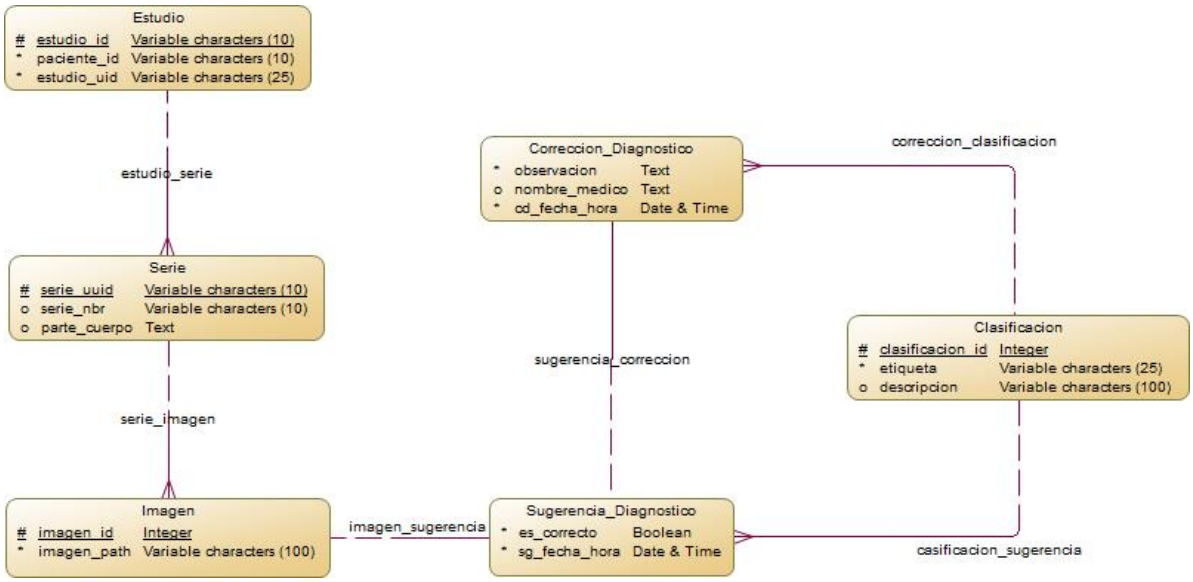


Ilustración 75 - Modelo Conceptual

4.2.7.2 Modelo lógico

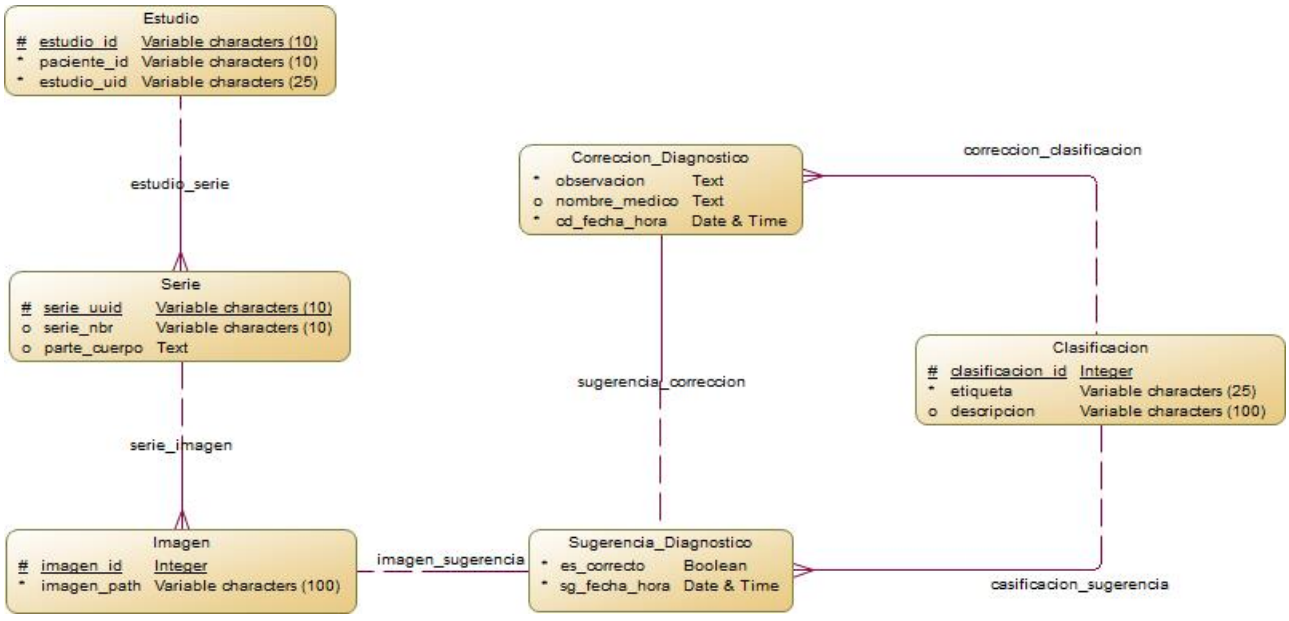


Ilustración 76 - Modelo Conceptual

4.2.7.2 Modelo físico

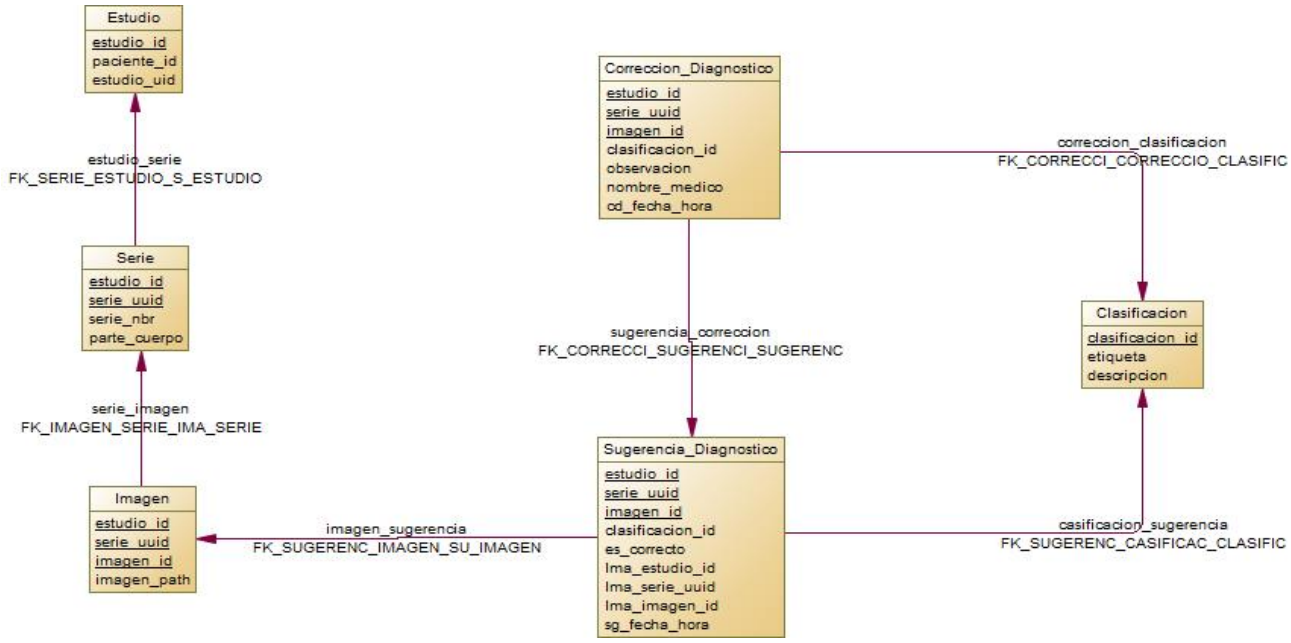


Ilustración 77 - Modelo Físico

4.2.8. Diseño de pruebas.

4.2.8.1 Aproximación

En esta sección se describe los tipos de pruebas a utilizar para el sistema SIADTACC, cada una de ellas presenta un diferente formato para el registro de resultados.

Pruebas unitarias
Pruebas por cada unidad funcional, que no necesariamente es equivalente a un requerimiento. La funcionalidad es aprobada si cumple con lo escrito en las especificación de requerimientos. Tiene por objetivo detectar errores en los datos, lógica o algoritmos.

CÓDIGO PRUEBA	#
Descripción	
Objetivo de prueba	
Caso de uso	
Tipo de prueba	UNITARIA
Precondición	
Componente	
Entradas	
Resultados esperados	
Resultados obtenidos	

Tabla 69 - Formato para la realización de pruebas unitarias

Pruebas de integración
Las pruebas de integración se realizan a todo el sistema como un conjunto y se distribuyen en todos los requerimientos solicitados. Se debe detallar qué casos de uso cumple, con rutas de éxito y fallo definidas en la sección de Casos de uso. Tiene por objetivo detectar errores de interfaces y relaciones entre los componentes.

CÓDIGO PRUEBA	#
Descripción	
Objetivo de prueba	
Tipo de prueba	INTEGRACIÓN
Precondición	
Resultados esperados	
Resultados obtenidos	

Tabla 70 - Formato para la realización de pruebas de integración

4.2.8.2 ORDEN DE EJECUCIÓN DE PRUEBAS

Las pruebas se ejecutarán en el siguiente orden:

1. Mandar a análisis una serie de TAC:

   Proceso que utiliza un plugin hecho al visor Weasis para mandar a análisis una serie.

2. Conexión de servicio web a WADO:

   Conexión desde el servicio web hacia el servidor de imágenes médicas mediante protocolo WADO.

3. Conexión de servicio web a RNA:

   Conexión que permite el análisis de los estudios alojados en el servidor PACS.

4. Análisis de RNA:

   Análisis de TACs para dar sugerencias de diagnóstico.

5. Retorno de respuesta a visor Weasis:

   Se retorna una respuesta al visor Weasis y se muestra el reporte.

CAPÍTULO V: DESARROLLO

5.1. Estándares de desarrollo

5.1.1. Tecnologías a utilizar

El listado completo de las tecnologías con sus respectivas versiones a utilizar se encuentra en la sección de los requerimientos de software de desarrollo en la Sección 3.5.2 Requerimientos de software de desarrollo.

5.1.2. Estándares de base de datos

Criterio	Descripcion
Generalidades	PostgreSQL posee una guía de estilos que recopila las buenas prácticas para implementar y documentar los distintos objetos de la base de datos (The PostgreSQL Global Development Group, 1996-2016), en esta sección se describen algunos de los elementos más relevantes, para mayor detalle revisar la guía completa, aunque como regla general utilizar sintaxis snake_case para nombrar los diferentes objetos, es decir todas las palabras en minúsculas y separadas por guion bajo.
Comentarios	Se recomienda utilizar comentarios en múltiples líneas, como el siguiente ejemplo: /*---------- * Start of comment * end of comment *---------- */
Idioma	Utilizar el idioma inglés para nombrar los objetos de base de datos.
Tablas y relaciones	Utilizar nombres en singular ejemplo: student, customer, state.
Nombres de campos y variables	Utilizar nombres completos, no abreviaciones, por ejemplo: middle_name, no mid_nm. Además, no utilizar palabras reservadas, como user, lock, schema, etc. para una lista completa revisar la siguiente documentación oficial (The PostgreSQL Global Development Group, Appendix C. SQL Key Words,1996-2016).
Llave primaria	El identificador de cada tabla será nombrado simplemente como id.
Llave foránea	Deben ser compuesta de el nombre de la tabla a la que hace referencia y el sufijo id, separada por guion bajo, ejemplo team_id, person_id, customer_id.
Índices	Deben ser compuesta de el nombre de la tabla a la que hace referencia y el sufijo id, separada por guion bajo, ejemplo team_id, person_id, customer_id.
Restricciones	Asignar el nombre del campo al que pertenece la restricción y un sufijo que indique el tipo de restricción, ejemplo: email_pkey, code_fkey.
Otros	Otros objetos como triggers, procedimientos, etc. Serán nombrados según el apartado de nombres de campos y variables.

71 - Estándares de la Base de Datos

5.1.3. Estándares de programación lenguaje Python

El framework de deep learning CAFFE posee un módulo el cual puede ser incluido en los programas escritos en python y utilizar las funciones del framework.

Existe una guía de estilo oficial para el desarrollo de aplicaciones en python (Python Software Foundation, 2001-2016), esta es PEP8, en este apartado se describirán los elementos más relevantes, para mayor detalle revisar la documentación oficial.

Criterio	Descripcion
Codificación de archivos	Para python 2 se utiliza la codificación ASCII y para python 3 UTF-8.
Sangrado (Indentación)	Utilizar 4 espacios por cada nivel, es preferible el uso de espacios y no de tabuladores.
Longitud máxima por línea	Se recomienda utilizar un máximo de 79 caracteres por línea, si la sentencia excede el límite se puede utilizar la pleca invertida (\) al final de la línea y continuar en la siguiente.
Líneas en blanco	Se permite el uso de líneas en blanco dentro de las clases para separar las definiciones de los métodos y dentro de los métodos para separar las secciones lógicas.
Importar módulos	Se recomienda hacer la importación de un módulo por línea, evitar lo siguiente: *import sys, os.* Las importaciones deben ser agrupadas de la siguiente manera: 1. Librerías estándar 2. Librerías desarrolladas por terceros 3. Librerías locales
Comentarios	Para comentarios de una línea se utiliza el símbolo de numeral (#), se recomienda colocar el comentario al final de la sentencia y este debe ser claro y descriptivo, ejemplo: x = x + 1 # Comentario Los comentarios de múltiple línea se envuelven en tres pares de comillas dobles (“””), se recomienda además que se coloquen las comillas en una sola línea para el cierre del bloque de comentario, ejemplo: """Return a boolean Optional comments. """
Convenciones de nombres	• **Nombre de módulos:** Los nombres deben ser cortos, en minúsculas y las palabras separadas por guiones bajos (*snake_case*). • **Nombre de paquetes:** Los nombres deben ser cortos, en minúsculas, no se recomienda el uso de guiones bajos para separar las palabras. • **Nombre de clases:** Se utiliza la convención *UpperCamelCase,* es decir las iniciales de cada palabra en mayúscula y sin espacios para separar las palabras.

	• **Nombre de excepciones:** Se utiliza la convención *UpperCamelCase* y se agrega el sufijo *Error* al nombre de la excepción. • **Nombre de funciones y variables globales:** Todo en minúsculas y las palabras separadas por guiones bajos en caso de ser necesario. • **Aplica para todos los anteriores:** La longitud mínima para el nombramiento de variables, constantes, clases, modulos, etc; será de 5 caractéres y máximo 30. • **Constantes:** Todos los caracteres en mayúsculas con y las palabras separadas por guiones bajos en caso de ser necesario.

72 - Estándares Lenguaje de programación Python

5.1.4. Estándares de programación lenguaje Java

Weasis, el visor de archivos en formato DICOM y el servidor PACS (DCM4CHEE), están escritos en el lenguaje de programación Java, se ha tomado como referencia la guía de estilo para Java escrita por Google (Google Inc. 1998 - 2016), a continuación, se describen los elementos más relevantes contenidos en dicha guía

Criterio	Descripcion
Nombres de archivos	Los archivos tendrán extensión .java
Codificación de archivos	La codificación será en formato UTF-8.
Estructura de los archivos	Se recomienda que los archivos posean la siguiente estructura 1. Tipo de licenciamiento o copyright (en caso de poseer). 2. Definición del paquete. 3. Sentencias de Importación. 4. Definición de la clase
Llaves opcionales	En las estructuras if, else, for, do y while, las llaves son opcionales cuando las sentencias a ejecutar sean de una sola línea.
Sangrado (Indentación)	Utilizar 2 espacios por cada nivel, es preferible el uso de espacios y no de tabuladores.
Utilizar una sentencia por línea	Evitar escribir más de una sentencia por línea incrementa la legibilidad y mantenimiento del código.
Líneas en blanco	Se permite el uso de líneas en blanco en las siguientes situaciones: ● Entre dos miembros consecutivos de la clase: campos, constructores, métodos, clases anidadas, etc. ● En el cuerpo de los métodos para separar en secciones la lógica del método.
Comentarios	Para los comentarios de una sola línea se recomienda utilizar la doble pleca (//), ejemplo: // Comentario

	Para los comentarios de múltiple línea se recomienda el uso de los comentarios de bloque, es decir apertura el comentario con una pleca y asterisco (/*) y cerrarlo con un asterisco y pleca (*/); además colocar un asterisco al inicio de cada línea, ejemplo: /* * Comentario */
Convenciones de nombres	• **Nombre de paquetes:** Todo escrito en minúsculas, no separar las palabras. ejemplo: com.example. deepspace, evitar lo siguiente: com.example. deepSpace o com.example. deep_space. • **Nombre de clases:** Se usará la convención UpperCamelCase, es decir las iniciales de cada palabra en mayúscula y sin espacios para separar las palabras. • **Nombre de métodos:** Se usará la convención lowerCamelCase, es decir las iniciales de cada palabra en mayúscula, excepto la primera inicial y sin espacios para separar las palabras, ejemplo: sendMessage, calculateAverage, etc. Es recomendable utilizar verbos para identificar las acciones de los métodos. • **Nombre de constantes:** Todas las letras en mayúsculas y palabras separadas por guion bajo. • **Nombre de parámetros y variables:** Se utilizará la convención lowerCamelCase. • **Aplica para todos los anteriores:** La longitud mínima para el nombramiento de variables, constantes, clases, modulos, etc; será de 5 caractéres y máximo 30.

73 - Estándares de programación en lenguaje Java

5.2. Pruebas

5.2.1. Alcance de pruebas

- Las pruebas fueron realizadas en ambiente de desarrollo, tal como se describe en el apartado de estrategia de pruebas (Seccion 5.2.3).
- Las pruebas unitarias fueron realizadas por el encargado de cada parte del desarrollo, tal como se describe en el apartado de estrategia de pruebas (Seccion 5.2.3)

5.2.1.1. Cuadro resumen de las pruebas

Cuadro resumen de las pruebas	
Componentes a ser probados	● Plugin del visor Weasis ● Servicio web ● Conexión vía WADO ● Funcionalidad de RNA

Objetivos de las pruebas	En estos componentes se realizarán pruebas para validar: ● Verificar conectividad de los componentes. ● Efectuar de manera correcta cada uno de los procesos. ● Usabilidad para el usuario. ● Operación del servicio web y la respuesta que devuelve. ● Secuencia lógica de las funcionalidades.
Detalle del orden de ejecución de los componentes	Los componentes se ejecutarán en el siguiente orden: ● Mandar a análisis una serie de TAC. ● Conexión de servicio web a WADO. ● Conexión de servicio web a RNA ● Análisis de RNA ● Retorno de respuesta a visor Weasis.
Responsabilidad de la prueba	Las pruebas son responsabilidad del equipo de desarrollo encargado del proyecto “Sistema informático para el análisis digital de tomografías axiales computarizadas cerebrales utilizando redes neuronales artificiales como apoyo en el diagnóstico diferencial a los radiólogos del Ministerio de Salud de la República de El Salvador.

Tabla 74 - Resumen de las pruebas realizadas

5.2.1.2. Requerimientos incluidos

Requerimiento	Caso de uso asociado	Tipo	Nivel Criticidad (Bajo, Medio, Alto)
Acceso a archivos DICOM	Lectura TAC desde servidor PACS.	Funcional	Alto
Clasificación de TAC cerebrales	Análisis de TAC por la RNA.	Funcional	Alto
Mecanismo de feedback	Envío y recepción de feedback.	Funcional	Alto
Entrenamiento de la RNA	Entrenamiento de la RNA	Funcional	Alto

Tabla 75 - Requerimientos incluidos en las pruebas

5.2.1.3. Casos de uso incluidos

Código	Caso de uso	Descripción
CU001	Lectura TAC desde servidor PACS	El sistema debe de ser capaz de conectarse al servidor PACS implementado por SIMAGD para la obtención de archivos DICOM asociados a pacientes específicos. La lectura de TACs desde el servidor PACS será llevada a cabo por un conjunto de servicios web empaquetados en una API y publicados en una instancia del servidor Gunicorn.

CU002	Procesamiento de TAC	El sistema debe de ser capaz de darle un correcto tratamiento a los archivos DICOM para que estos se vuelvan legibles a la RNA. El sistema se hará cargo de correr un script que dado un archivo DICOM, desempaquete las imágenes en un formato que sea legible por la RNA.
CU003	Análisis de TAC por la RNA	El sistema debe de ser capaz de proporcionar las imágenes de forma adecuada a la RNA para que esta pueda analizarlas. Ejecutado el caso de uso CU002, el sistema ejecutará el proceso de análisis de las imágenes procesadas en el caso de uso anterior.
CU006	Envío y recepción de Feedback	El sistema debe de ser capaz de recibir retroalimentación de parte de los usuarios con respecto a la certeza de sugerencias de diagnóstico emitidas, almacenarla y luego utilizarla para procesos de entrenamiento posteriores.

Tabla 76 - Casos de uso incluidos en las pruebas

5.2.1.4. Casos de uso excluidos

Código	Caso de uso	Descripción
CU004	Elaboración y envío de reporte de resultados	El sistema debe de ser capaz de elaborar un informe de resultados basado en las conclusiones que la RNA arroje finalizado el análisis.
CU005	Presentación de Reporte de Resultados	El sistema debe de ser capaz de mostrar el informe de resultados del análisis realizado por la RNA.
CU007	Entrenamiento de RNA	El sistema debe de ser capaz de valiéndose del feedback recibido por los usuarios, permitir un proceso de reentrenamiento de la RNA de forma periódica.

Tabla 77 - Casos de uso excluidos en las pruebas

5.2.2. Entorno y configuración de pruebas

Para el proceso de pruebas del proyecto se utilizó el siguiente entorno, tomando en consideración las características del hardware que posee el Ministerio de Salud y los requerimientos mínimos de las tecnologías a utilizar:

a. Servidor PACS
    i. RAM 8gb
    ii. Core i7 2.7Ghz 8 núcleos
    iii. HDD 750GB
    iv. Tarjeta de red integrada.
b. Servidor RNA

i. RAM 8gb
ii. Core i7 2.7Ghz 8 núcleos
iii. HDD 750GB
iv. Tarjeta de red integrada.
v. Nvidia gtx760

c. Equipo cliente
i. RAM 4gb
ii. Core i5 3.3Ghz 4 núcleos
iii. HDD 1TB
iv. Tarjeta de red integrada

d. Base de datos Postgresql
i. Versión 9.4+165+deb8u1. El cual se encuentra en el mismo servidor PACS.

5.2.2.1. Componentes del sistema

Componente	Criterios de pruebas	Descripción
Plugin Weasis	Facilidad de uso	El usuario debe tener siempre claros y entendibles todos los procesos que tiene a su disposición en el entorno del sistema.
	Look & Feel	Apariencia y sensación amigables para el usuario.
Servicio web/RNA	Persistencia	Se debe contar con capacidad para almacenar datos y utilizarlos en momentos posteriores de manera consistente e íntegra.
No funcionales	No funcionales	Cumplimiento de requerimientos no funcionales planteados en la especificación de requerimientos del software
SIADTACC	Funcionalidad	El sistema debe cumplir satisfactoriamente todos los requerimientos planteados en fases anteriores del desarrollo del proyecto.

Tabla 78 - Detalle de los componentes del sistema

5.2.2.2. Criterios de aprobación o rechazo

Los siguientes criterios han sido definidos tomando en consideración únicamente el funcionamiento adecuado de los componentes del sistema y basados en la experiencial del equipo en el desarrollo de sistemas informáticos, cabe destacar el el funcionamiento de la red neuronal y sus resultados son tratados en el apartado 2.3 Resultados de pruebas en el capítulo Resultado de Investigación.

- **Graves:**
  - No brindar un resultado posterior a la solicitud de análisis de series de imágenes.
  - Presentar resultados no correspondientes al paciente del análisis solicitado.
  - Falla en inicio de sesión.
  - Fallos en proceso de feedback.

- **Medios:**
    - Tiempo de respuesta demasiado alto.
    - Reporte de análisis con mala estructura en la presentación de información.
- **Leves:**
    - Dificultades de operación.
    - Presentación de datos con palabras mal escritas que están ya registradas en la información de los archivos DICOM.

El proyecto es aprobado con todas las pruebas realizadas y se puede permitir un 90% de aceptación lo que implica que un 90% de las pruebas deben tener resultado exitoso y sin errores. Además, el resultado del 10% de pruebas restante no debe ser comprendido en criterios graves, pero sí en medios y leves que deben ser solventados posteriormente.

5.2.3. Estrategia de pruebas

La estrategia de pruebas a utilizar se plasma en la sección de diseño de pruebas definida anteriormente.

5.2.4. Resultados de pruebas

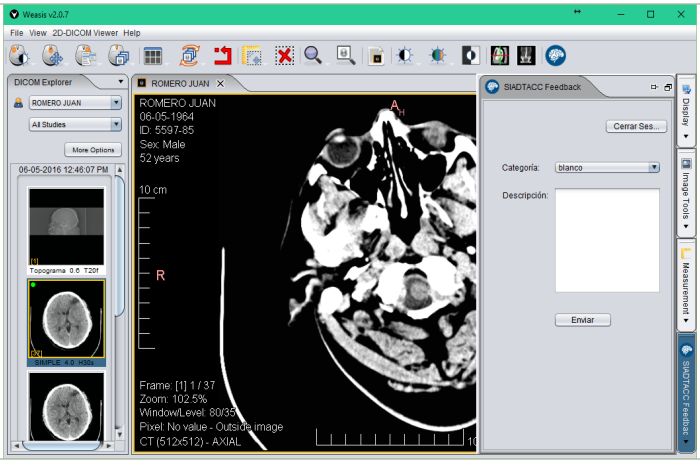
CÓDIGO PRUEBA	01
Descripción	Se envían credenciales en el plugin de Weasis para autenticarse con el web service.
Objetivo de prueba	Verificar el correcto funcionamiento del sistema al momento de autenticarse
Caso de uso	CU006
Tipo de prueba	Unitaria
Precondición	Tener abierto el visor Weasis con un estudio cargado
Componente	Weasis Plugin
Entradas	User: prb01 Password: prb_123123
Resultados esperados	Autenticación exitosa
Resultados obtenidos	

Tabla 79 - Resultado de la prueba 01

CÓDIGO PRUEBA	02
Descripción	Obtención de sugerencias de diagnóstico
Objetivo de prueba	Comprobar la correcta obtención de sugerencias
Caso de uso	CU003
Tipo de prueba	Unitaria (JSON)
Precondición	No es necesario que el usuario esté con una sesión activa, pero sí debe conocer que estudio, series y objetos desea clasificar.
Componente	Servicio Web REST (POST /rest-api/estudio/sugerencia/)

Entradas	{
	"series": [
	{
	"seriesUID": "1.3.12.2.1107.5.1.4.18375892400000010",
	"objects": [
	"1.3.12.2.1106051837589240000011"
	]
	},
	{
	"seriesUID": "1.3.12.2.11079024352800000039",
	"objects": [
	"1.3.12.2.1107.5.0016060519024352800000188",
	"1.3.12.2.1107.6060519024352800000183",
	"1.3.12.2.606051902435280000178"
	]
	},
	{
	"seriesUID": "1.3.12.606051902435280000000",
	"objects": [
	"1.3.10160605190243528000000023",
	"1.3.12.2.1000160605190243528000000021",
	"1.3.7.3000001606051902435280000018",
	"1.3.12.2.1107.6051902435280000016",
	"1.3.12.2.1107.5.1.4.6624352800000008",
	"1.3.12.2.1106060519024352800000029"
	]
	}
	],
	"studyUID": "1.3.12.2.1107.5.10518460717200000002"
	}

Resultados esperados	[
	{
	"imagen": {
	"studyUID": "1.3.12.2.1107.5.1.4.6617200000002",
	"seriesUID": "1.3.12.2.1107.5.1.4.6613400000010",
	"objectUID": "1.3.12.2.1107.5.1.4.66135892400000011",
	"nombre": "1.3.12.2."
	},
	"clasificacion": {
	"id": 3,
	"etiqueta": "TAC Cerebral Anormal",
	"descripcion": "TAC C A"
	},

	"es_correcto": true
	},
	{
	"imagen": {
	"studyUID": "1.3.12.2.1107.5.1.4460717200000002",
	"seriesUID": "1.3.12.2.17.30518375892400000010",
	"objectUID": "1.3.12.2.1107.5.1.4.6675892400000011",
	"nombre": "1.3.12.2."
	},
	"clasificacion": {
	"id": 3,
	"etiqueta": "TAC Cerebral Anormal",
	"descripcion": "TAC C A"
	},
	"es_correcto": true
	}
	]

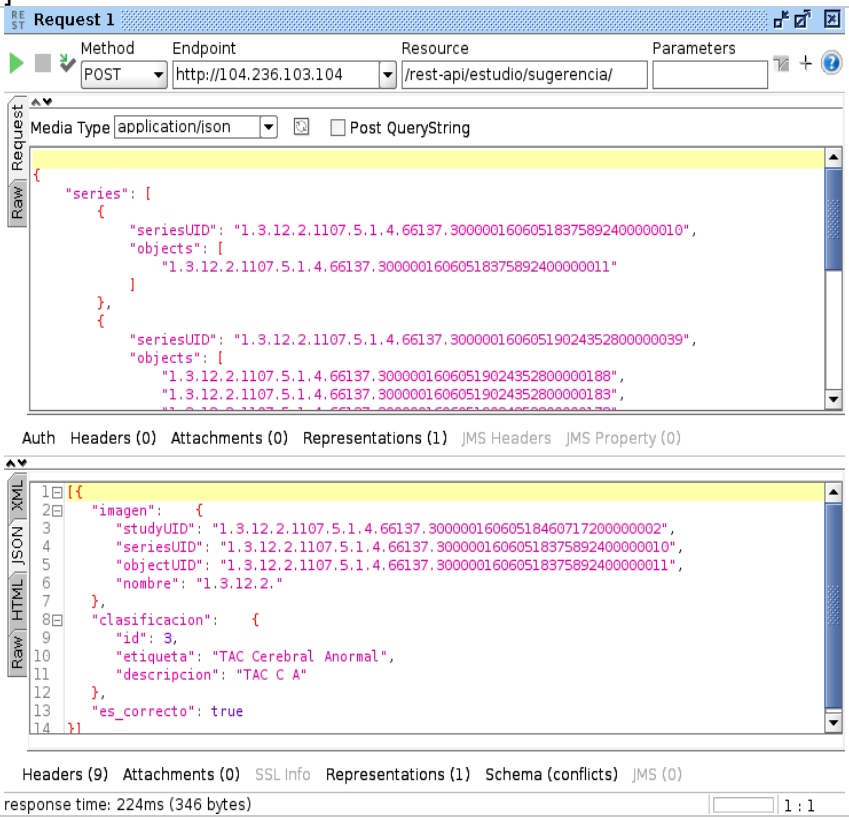
Resultados obtenidos	

Tabla 80 - Resultado de la prueba 02

CÓDIGO PRUEBA	03
Descripción	Envío de corrección de sugerencia, cuando el usuario vea una clasificación hecha por el sistema que no sea correcta, utiliza este método para almacenar dicha corrección.
Objetivo de prueba	Comprobar que el correcto funcionamiento del envío de feedback.
Caso de uso	CU006
Tipo de prueba	JSON

Precondición	No es necesario que el usuario esté con una sesión activa, pero sí debe conocer que estudio, series y objetos desea clasificar.
Componente	Servicio Web REST (POST a /rest-api/correccion)

Entradas	{
	"clasificacion_correcta": {
	"id": 0,
	"etiqueta": "Imagen en blanco",
	"descripcion": "Imagen en blanco"
	},
	"observacion": "observacion asdasdasda",
	"imagen": {
	"studyUID": "1.3.12.2.60518460717200000002",
	"seriesUID": "1.3.1001606051837589240000010",
	"objectUID": "1.3.0000160605183758924000000011",
	"nombre": "1.307"
	}
	}

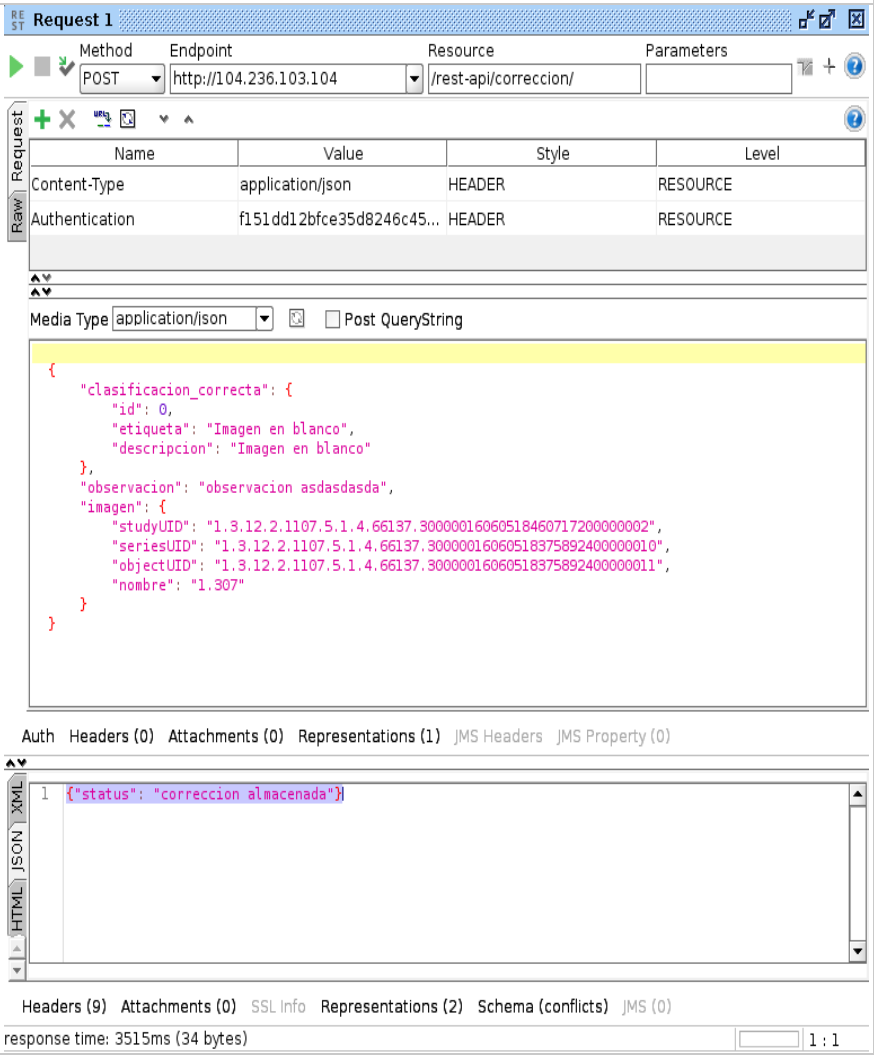
Resultados esperados	{"status": "correccion almacenada"}
Resultados obtenidos	

Tabla 81 - Resultado de la prueba 03

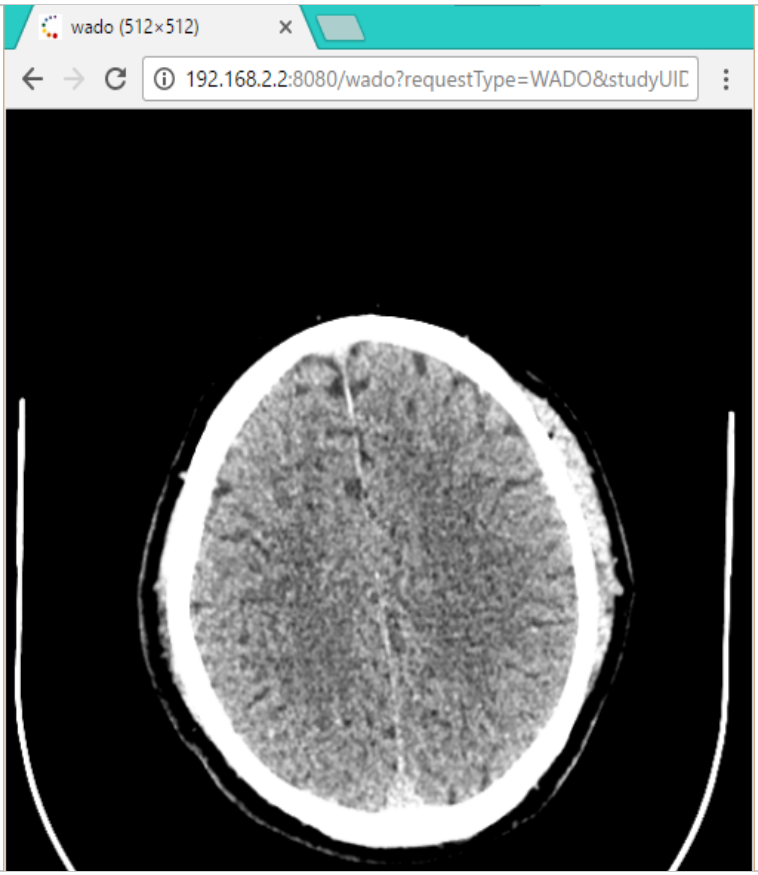
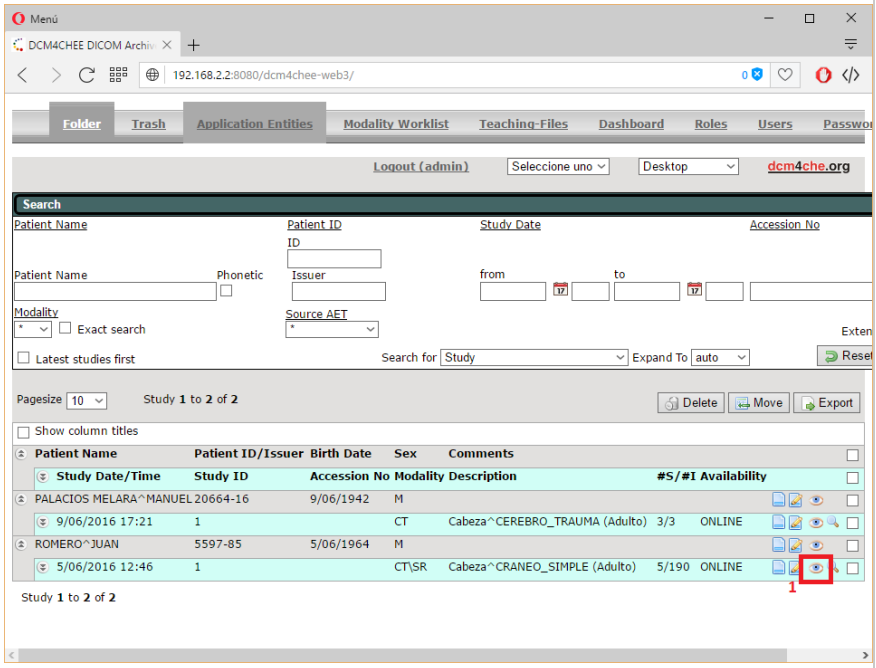
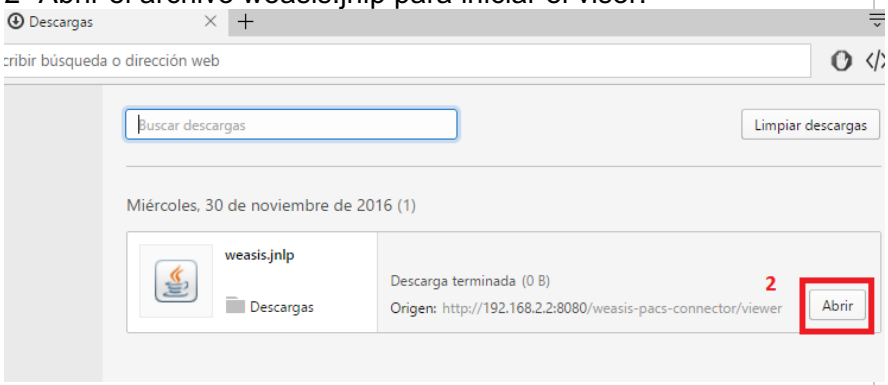
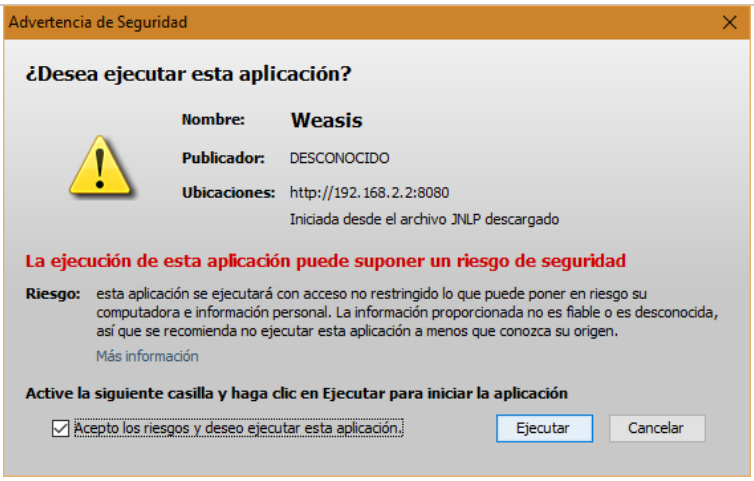
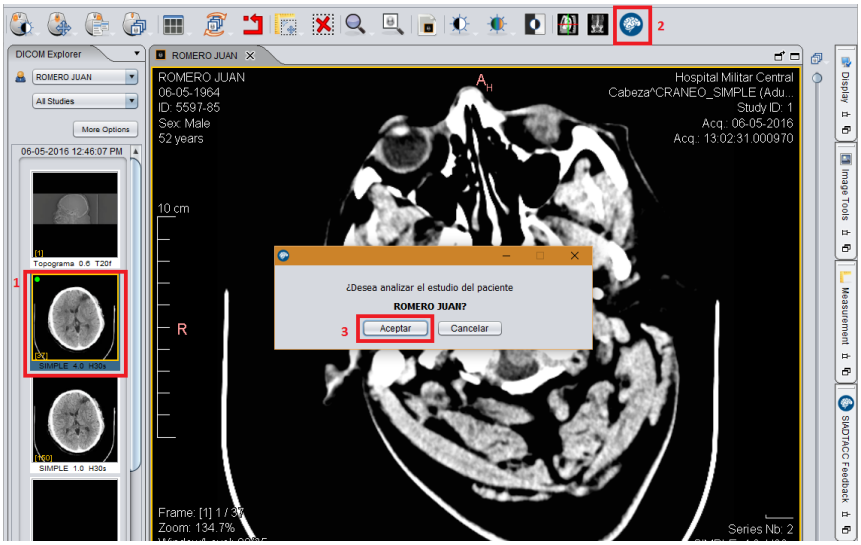
CÓDIGO PRUEBA	04
Descripción	Se envía una petición al protocolo WADO para obtener imágenes del servidor.
Objetivo de prueba	Comprobar que el protocolo WADO esté funcionando
Caso de uso	CU001/CU002
Tipo de prueba	Unitaria (Imagen PNG)
Precondición	Se debe conocer los valores del studyUID, seriesUID y ObjectUID(SOPUID)
Componente	Servicio Web REST (POST a /rest-api/correccion)
Entradas	StudyUID=1.3.12.2.1107.5.1.4.66137.300000160605184607172000000001 SeriesUID=1.3.12.2.1107.5.1.4.66137.3000001606051902435280000000039 ObjectUID=1.3.12.2.1107.5.1.4.66137.3000001606051902435280000000142 HTTP GET /wado?requestType=WADO&studyUID=1.3.12.2.1107.5.1.4.66137.300000160605184607172000000001&seriesUID=1.3.12.2.1107.5.1.4.66137.300000160605190243528000000039&objectUID=1.3.12.2.1107.5.1.4.66137.300000160605190243528000000142&contentType=image/png
Resultados esperados	Imagen PNG de dicom
Resultados obtenidos	

Tabla 82 - Resultado de la prueba 04

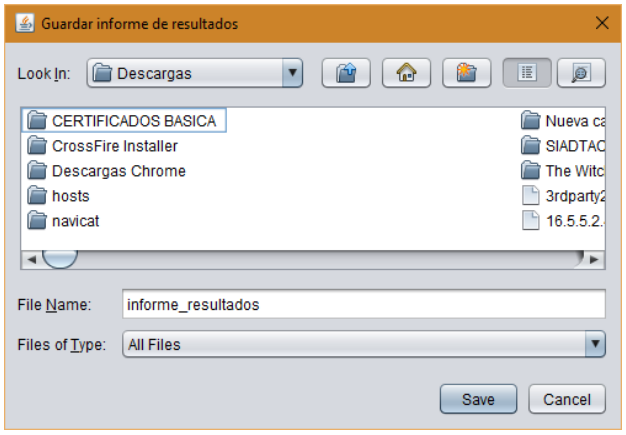
CÓDIGO PRUEBA	05
Descripción	Se comprueba el funcionamiento de todo el sistema.
Objetivo de prueba	Comprobar el funcionamiento del sistema
Tipo de prueba	INTEGRACIÓN
Precondición	DCM4CHEE Funcionando
Resultados esperados	Generación de reporte de sugerencias y envío de correcciones.
Resultados obtenidos	1- Obtención del estudio desde el dcm4chee  2- Abrir el archivo weasis.jnlp para iniciar el visor.  3- Aceptar los mensajes de seguridad de java.



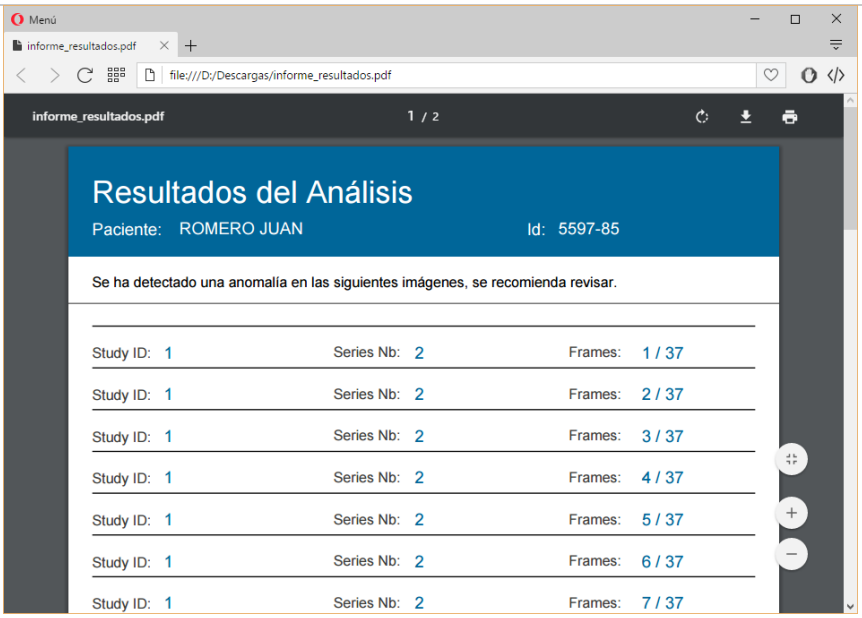
4- En el visor seleccionar una serie (debe aparecer el puntito verde sobre la serie), dar clic en el icono de SIADTACC y aceptar la acción a realizar.



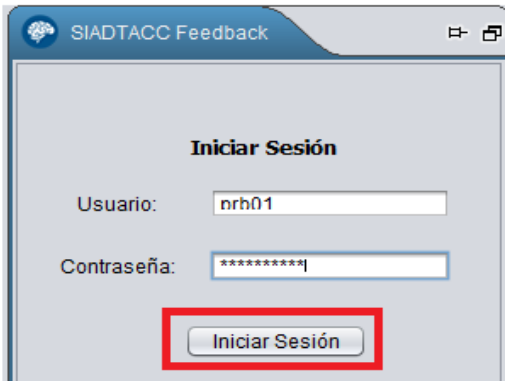
5- Esperar que concluya la petición de sugerencia, y guardar el reporte del estudio



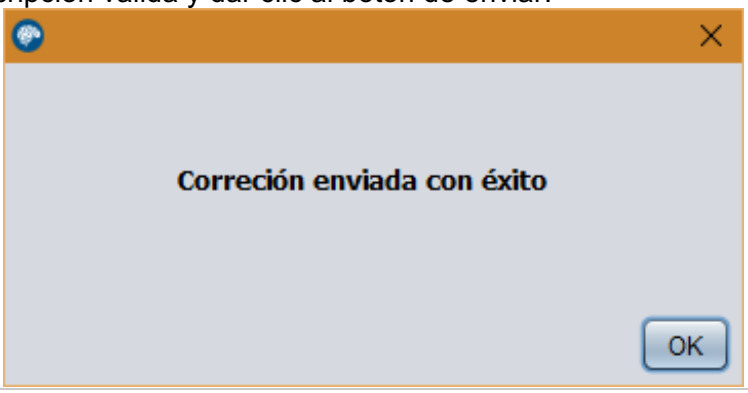
6- Abrir el reporte y visualizarlo.



7- Enviar una sugerencia de diagnóstico. Iniciar sesión con credenciales válidas.



8- Seleccionar la clasificación correcta de la lista, escribir una descripción válida y dar clic al botón de enviar.



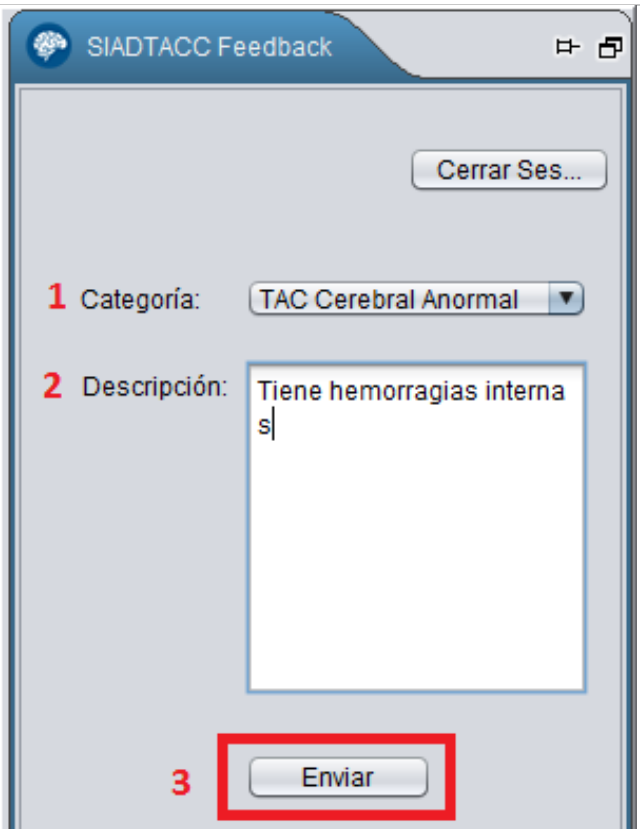
	 9- Dar clic a aceptar, para dar finalizada el ciclo completo de las sugerencias y correcciones

Tabla 83 - Resultado de la prueba 05

CAPÍTULO VI: DOCUMENTACIÓN

6.1. Manual de usuario

El presente documento permite describir y entender entorno gráfico y operatividad de SIADTACC. Se abordan a detalle los pasos que se deben seguir para el manejo general las funciones disponibles.

SIADTACC es amigable y de fácil uso ya que funciona de manera conjunta con Weasis; un entorno de trabajo ya conocido por los usuarios, permite analizar y clasificar imágenes realizadas en cualquier momento, lo que colabora de gran manera al proceso de aprendizaje del sistema mismo.

Usuarios. Existen dos tipos de usuarios para el SIADTACC:

- Técnicos en radiología: Se encargan de realizar la captura y procesamiento de imágenes médicas, además de hacer una revisión preliminar de éstas.
- Médicos radiólogos: Emiten el diagnóstico de imágenes médicas, tomando como partida la revisión preliminar realizada por el técnico en radiología.
- Administrador del sistema: Encargado de realizar todas las actividades relacionadas a la gestión del sistema, entre las cuales están: realizar backups, migraciones de datos, agregar categorías de clasificación de las imágenes, etc.

Ambos tipos de usuarios pueden aportar sugerencias de diagnóstico con el fin de refinar la clasificación de imágenes luego de un proceso de entrenamiento usando como punto de partida estas sugerencias.

El análisis y clasificación de imágenes se realiza en el entorno del visor Weasis con la incorporación de un complemento (plugin) ubicado en la parte superior con el botón y la pestaña en el lado derecho del visor como se indica en la Ilustración 76.

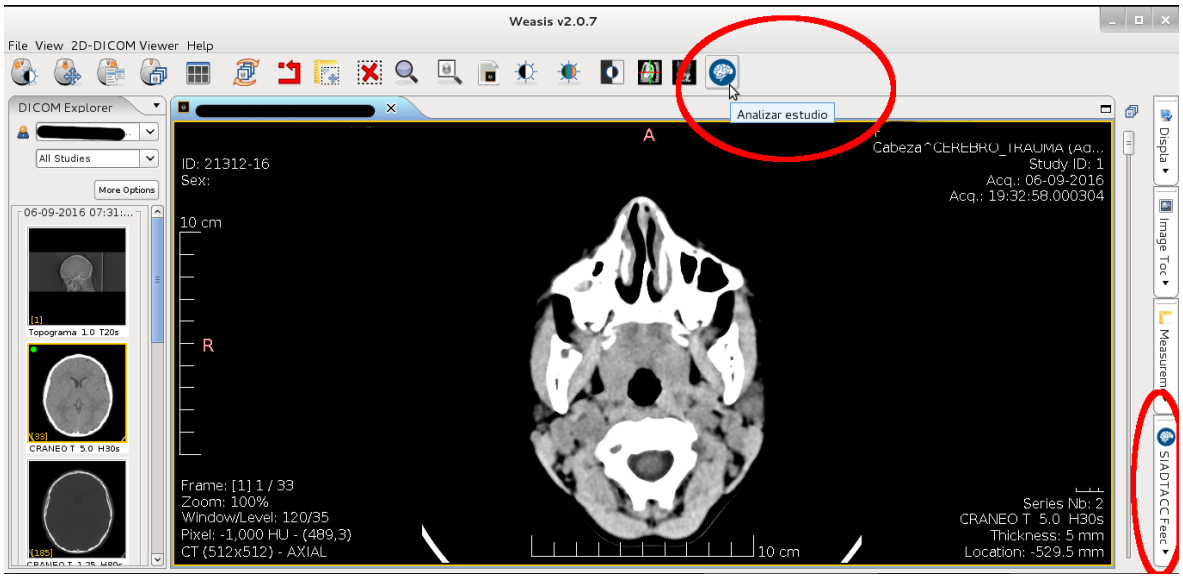


Ilustración 78 - Visor Weasis y plugin de SIADTACC para análisis y clasificación de imágenes.

ANALISIS

Cuando las imágenes a analizar estén cargadas en Weasis, se procede a presionar el botón ubicado en la parte superior del visor (Ilustración 75), e inmediatamente se despliega un cuadro de dialogo para la confirmación del análisis. Se ponen a disposición el análisis del corte en que se encuentre actualmente el visor, al análisis de la toda la serie y la generación de un archivo PDF con los resultados del análisis (Ilustración 76).

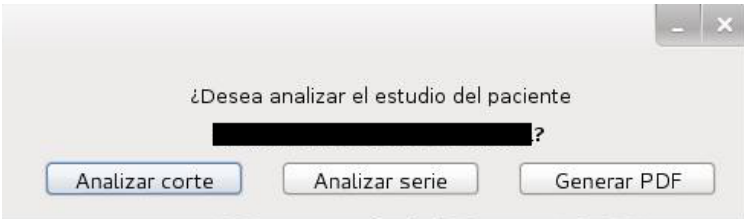


Ilustración 79 - Confirmación de análisis.

Posterior a la confirmación del análisis, se despliega una ventana con información de las imágenes con anomalía y tres elementos para interactuar (Ilustración 77 e Ilustración 78):

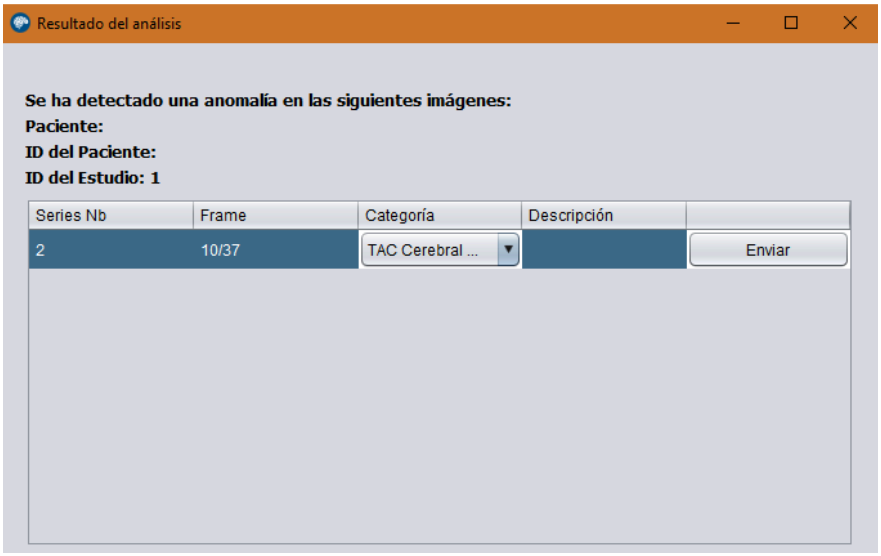


Ilustración 80 - Despliegue de resultados de análisis.

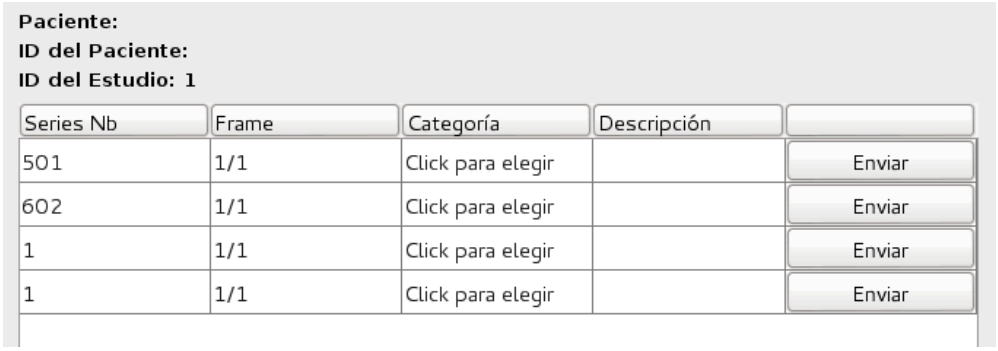


Ilustración 81 - Despliegue de resultados de análisis.

- Categoría: En la que se puede seleccionar entre Blanco (para imágenes pertenecientes a huesos), Negro (para imágenes con ausencia de anatomía cerebral), TAC Cerebral Normal y Anormal y un TAC no cerebral.
- Descripción: Campo en el que se puede agregar observaciones específicas al resultado de la imagen en cuestión.
- Enviar: Botón para guardar los resultados del análisis y la clasificación asignada.

Estos campos son requeridos por lo que se despliegan diálogos en los que se solicita la información (Ilustración 79 e Ilustración 80).

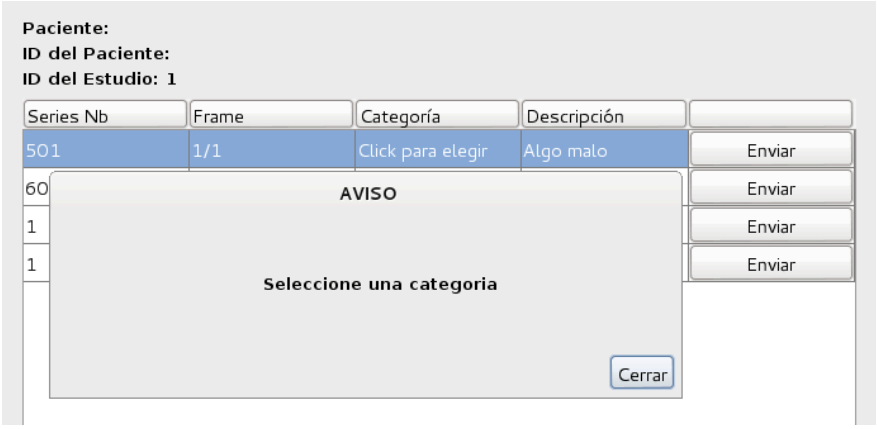


Ilustración 82 - Solicitud de información requerida.

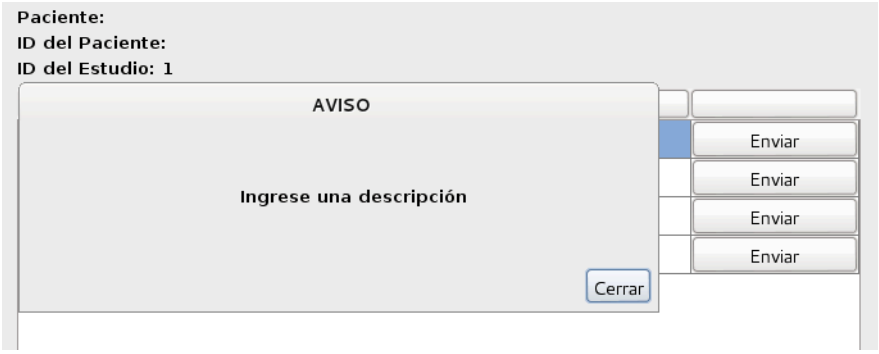


Ilustración 83 - Solicitud de información requerida.

Luego de ingresar la categoría y descripción a los resultados, se despliega una confirmación del registro al presionar el botón Enviar (Ilustración 81).

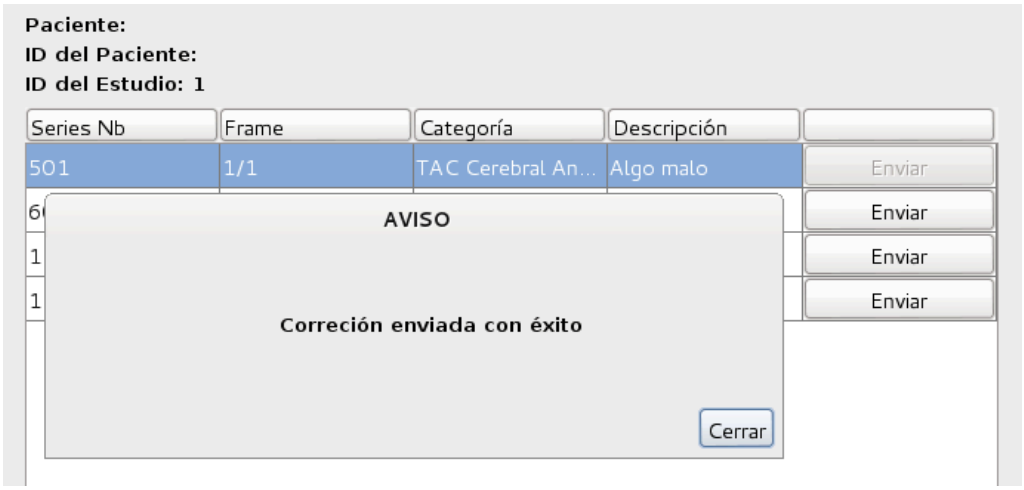


Ilustración 84 - Confirmación de registro.

CLASIFICACIÓN (ENVÍO DE CORRECCIONES).

Accedemos a esta función desde la pestaña en el lado derecho del visor (Ilustración 75) y posteriormente se despliega una pestaña con los elementos de interacción descritos anteriormente (Ilustración 82) y también con diálogos de Solicitud de información requerida y confirmación de registro el presionar el botón Enviar (Ilustración 83).

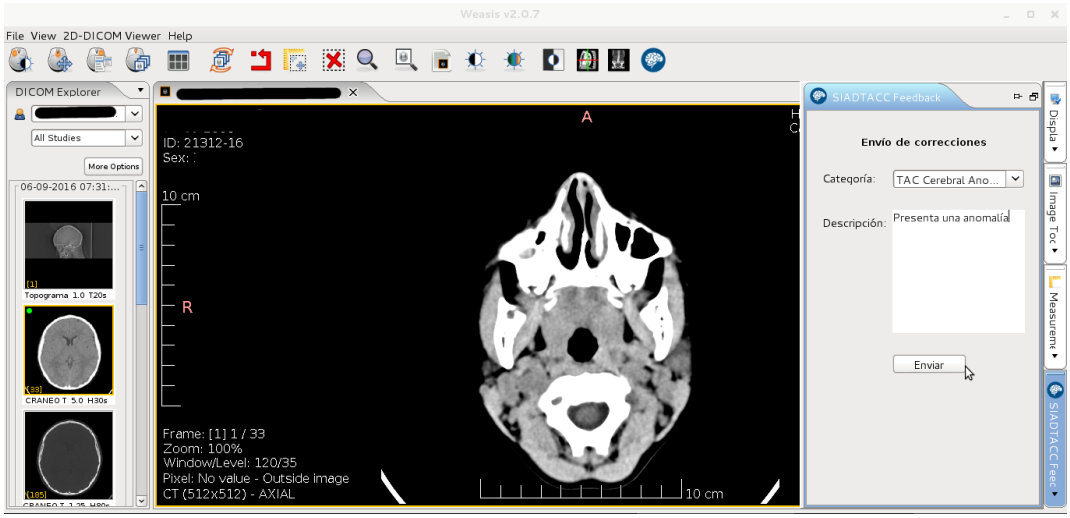


Ilustración 85 - Clasificación (envío de correcciones).

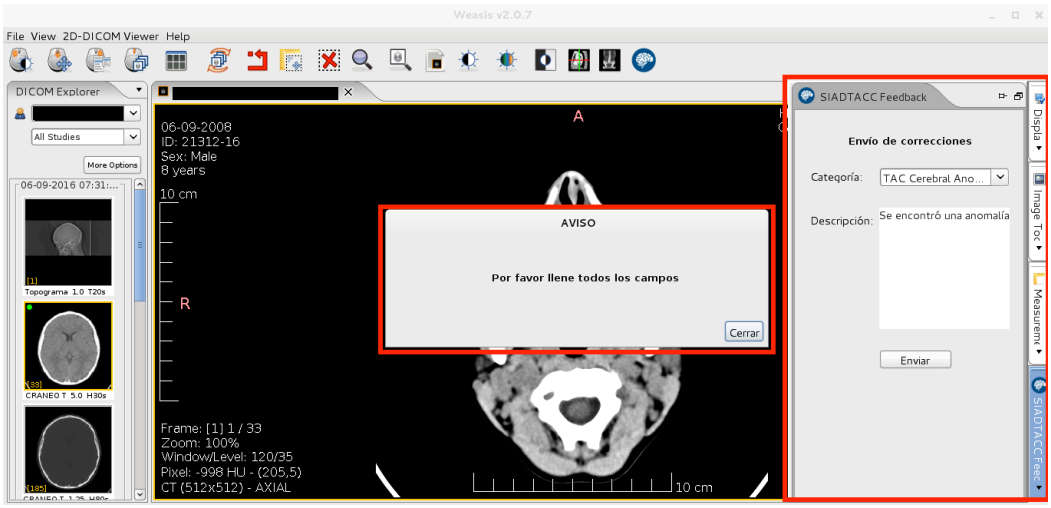


Ilustración 86 - Solicitud de información requerida.

CREACION DE OPCIONES DE SUGERENCIA

El encargado de crear mas opciones de sugerencia de diagnóstico es el administrador del sistema previo de una petición de los médicos radiólogos, por lo tanto esta interfaz es exclusiva para su uso.

1- Abrir el navegador y visitar la url administrativa del mismo (http://url_siadtacc/admin)

2- Ingresar las credenciales de superusuario.

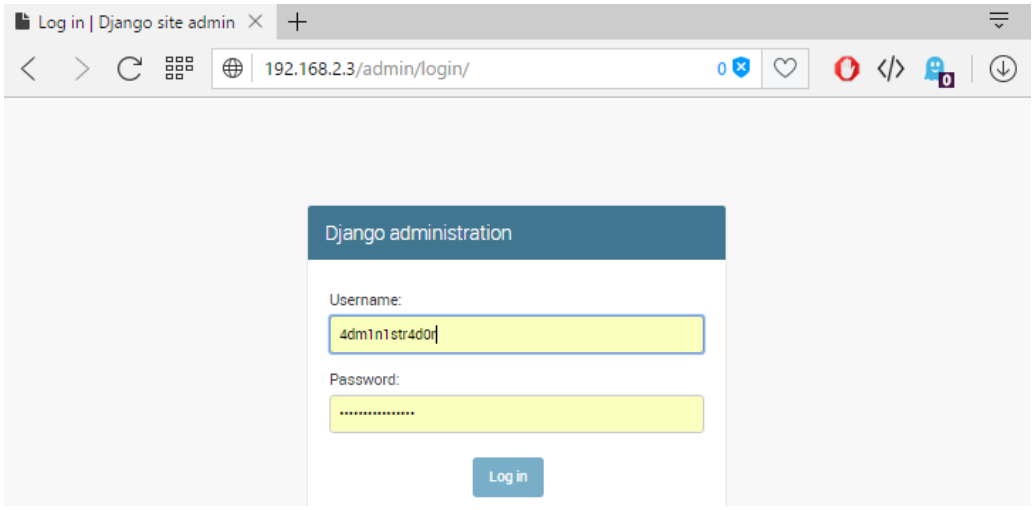


Ilustración 87 - Inicio de sesion superusuario siadtacc

3- Acceder a la subsección de Clasificaciones.

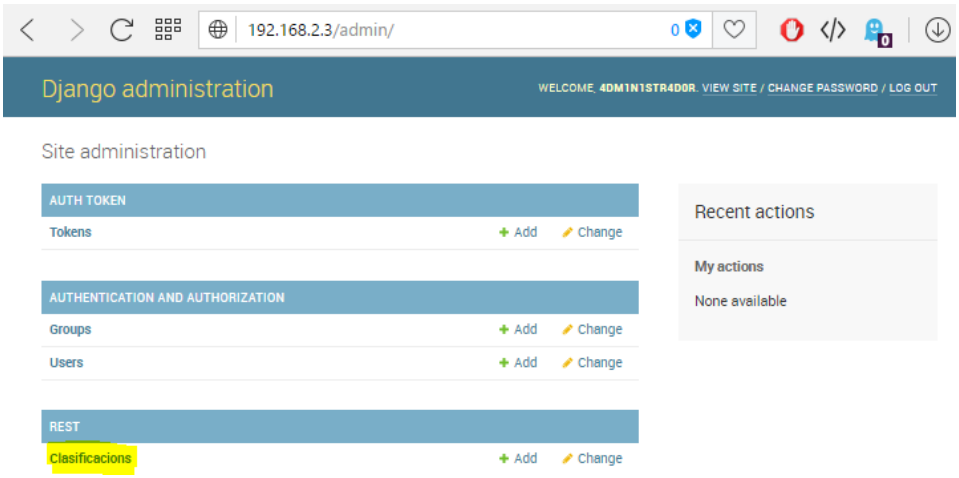


Ilustración 88 - Consola administrativa

4- Dar clic a agregar clasificación.

Ilustración 89 - Listado de clasificaciones (I)

5- Completar información y dar click a guardar.

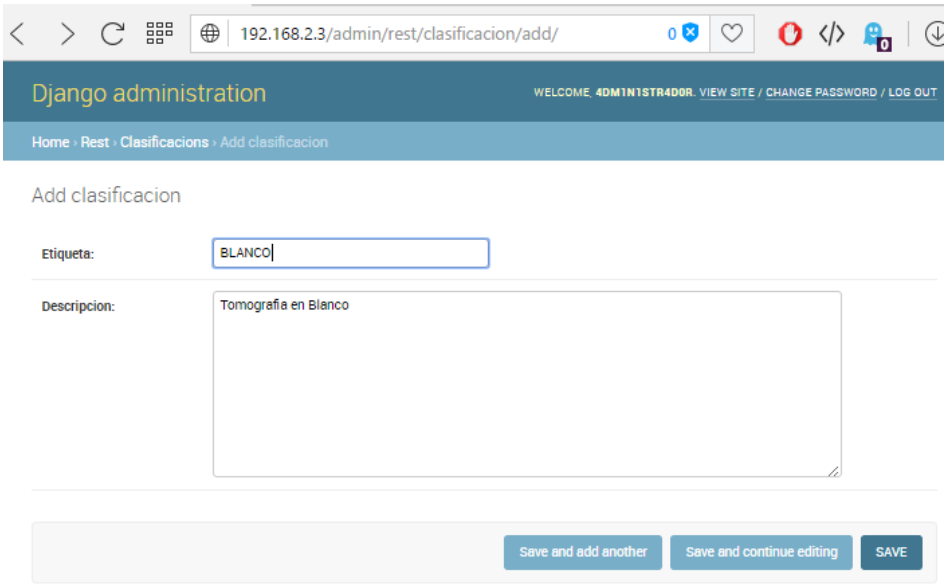


Ilustración 90 - Agregando nueva clasificacion

6- Verificar que la nueva clasificación sea almacenada correctamente.

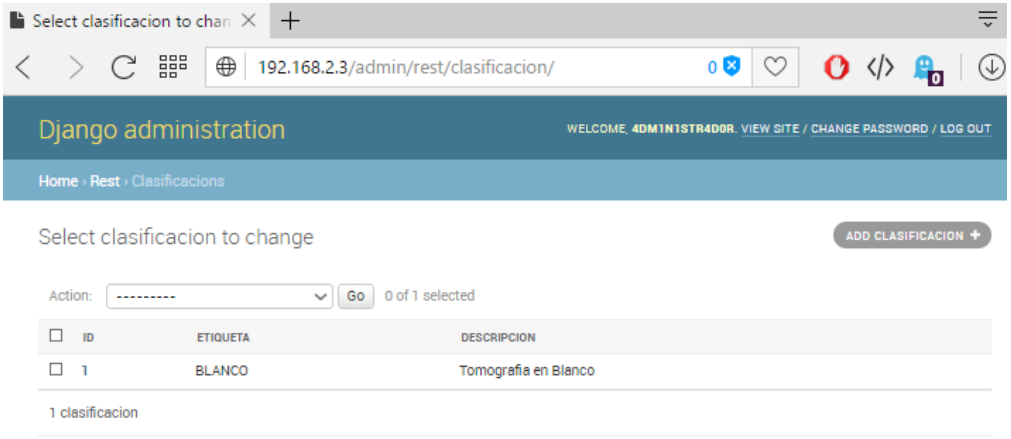


Ilustración 91 - Listado de clasificaciones (II)

6.2. Manual técnico

6.2.1 Flujo general del sistema

A continuación se presenta en forma de diagrama el flujo general del sistema:

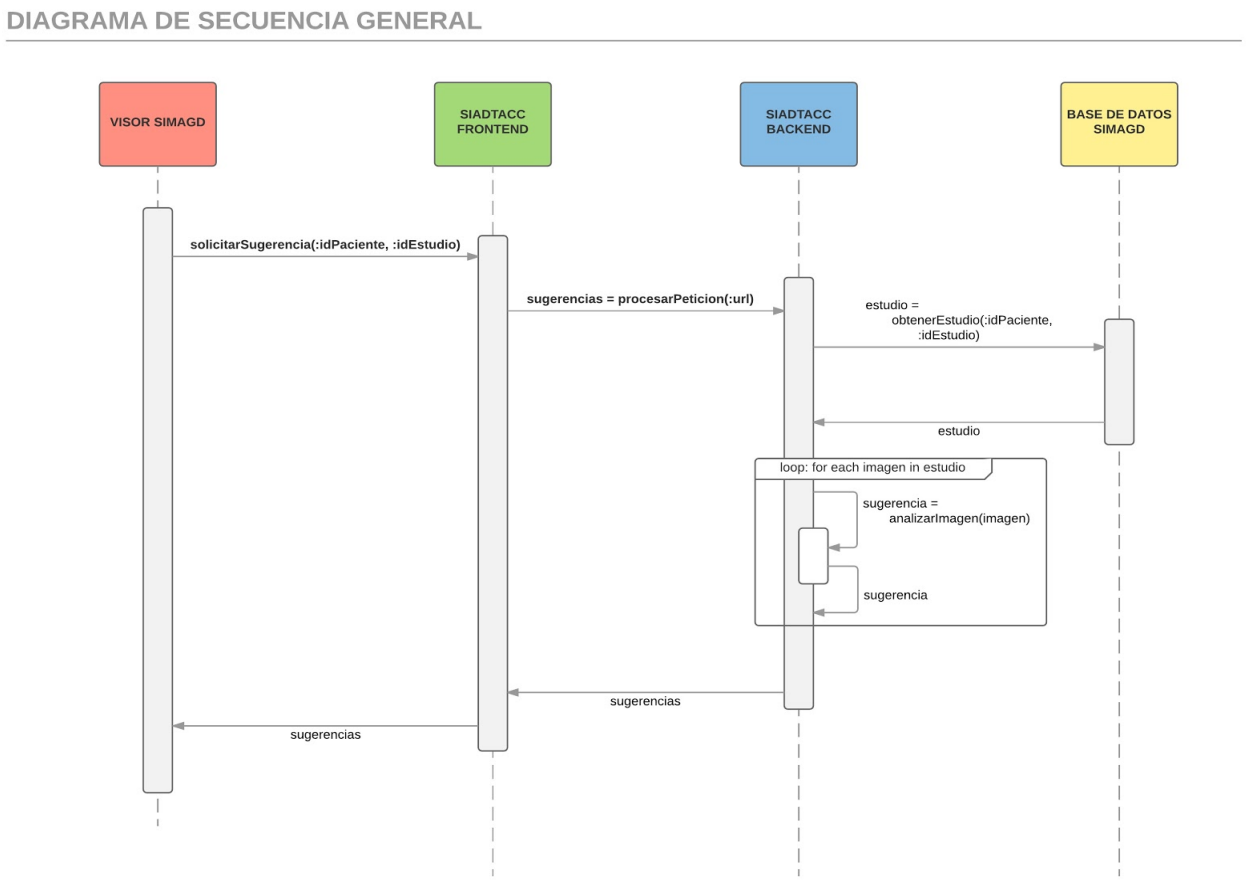


Ilustración 92 - Diagrama de secuencia general

6.2.2 Indicaciones sobre cómo acceder al software

El componente con el que interactúan los usuarios consiste de un plugin desarrollado para el visor de archivos dicom WEASIS, el cual a su vez está incluido en el PACS DCM4CHEE, a continuación, se listan los pasos para acceder a cada uno de los distintos componentes.

6.2.2.1 Acceder al visor WEASIS

El PACS funciona en un ambiente web, por lo tanto es necesario acceder mendiante un navegador web; en primer lugar hay que iniciar sesión y para ello se ingresan las credenciales en el formulario correspondiente.

Ilustración 93 - Formulario de inicio de sesión

Después de ingresar al sistema, se debe realizar la búsqueda de los estudios del paciente sobre los cuales se desea realizar un análisis; se puede filtrar los resultados utilizando los parámetros de búsqueda (1) o realizando una búsqueda general (2). En la siguiente imagen se pueden observar dichos controles.

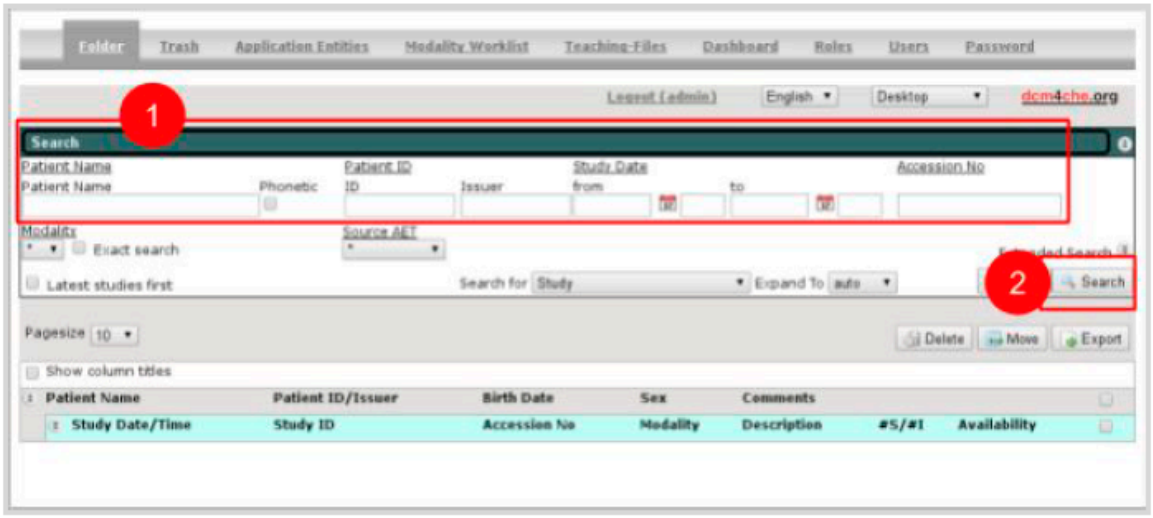


Ilustración 94 - Búsqueda de pacientes en el PACS

Después se presenta un listado con todos los pacientes que coincidan con los criterios de búsqueda aplicados, el PACS gestiona múltiples estudios y series para cada paciente; para poder examinar a detalle la información de cada paciente se debe presionar el botón con de flecha vertical (1) para expandir o colapsar la información, para visualizar el contenido de un estudio en particular, es necesario hacer click en el botón con forma de ojo (2).

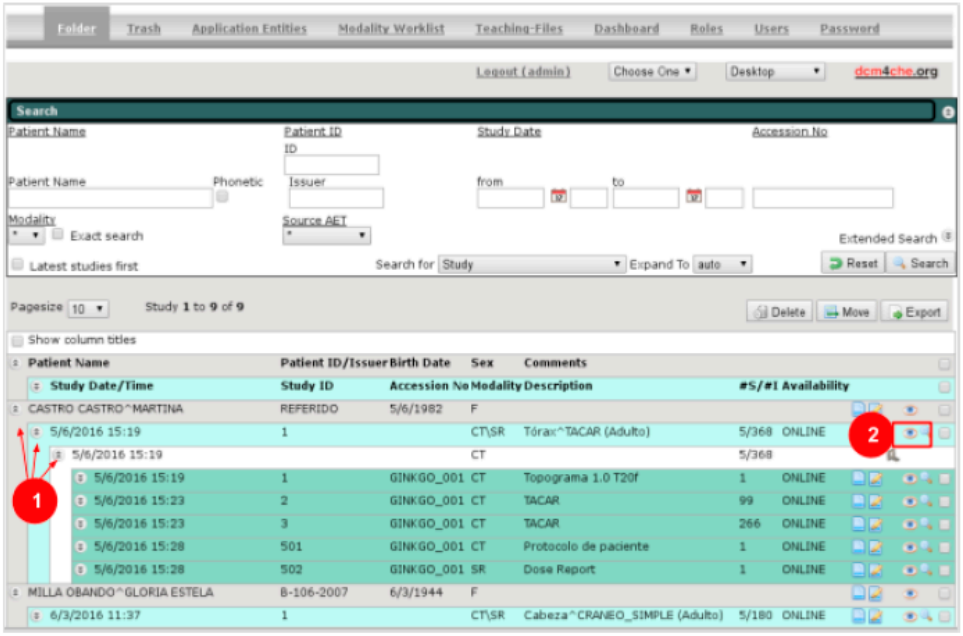


Ilustración 95 - Presentación de los resultados

Cuando se presiona el botón con icono de ojo, se descarga un archivo de extensión jnlp como se observa en la Ilustración 88. Es necesario abrir este archivo lo cual ejecutará el visor Weasis (Ilustración 89).

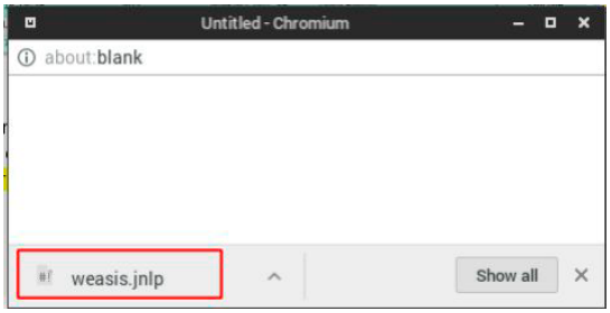


Ilustración 96 Descarga de archivo jnlp

Ilustración 97 - Pantalla de inicio de visor Weasis

6.2.2.2 Acceder a las funciones del plugin

Finalizado el proceso de carga de los plugins del visor se muestra la pantalla de principal con todas las funciones disponibles, como se observa en la siguiente imagen. Las funciones del plugin son accesibles de manera directa.

La opción para realizar el analisis del estudio actual se encuentra en la barra principal (1) en la parte superior del panel de visualización y la opción para enviar sugerencias a la derecha del panel de visualización (2).

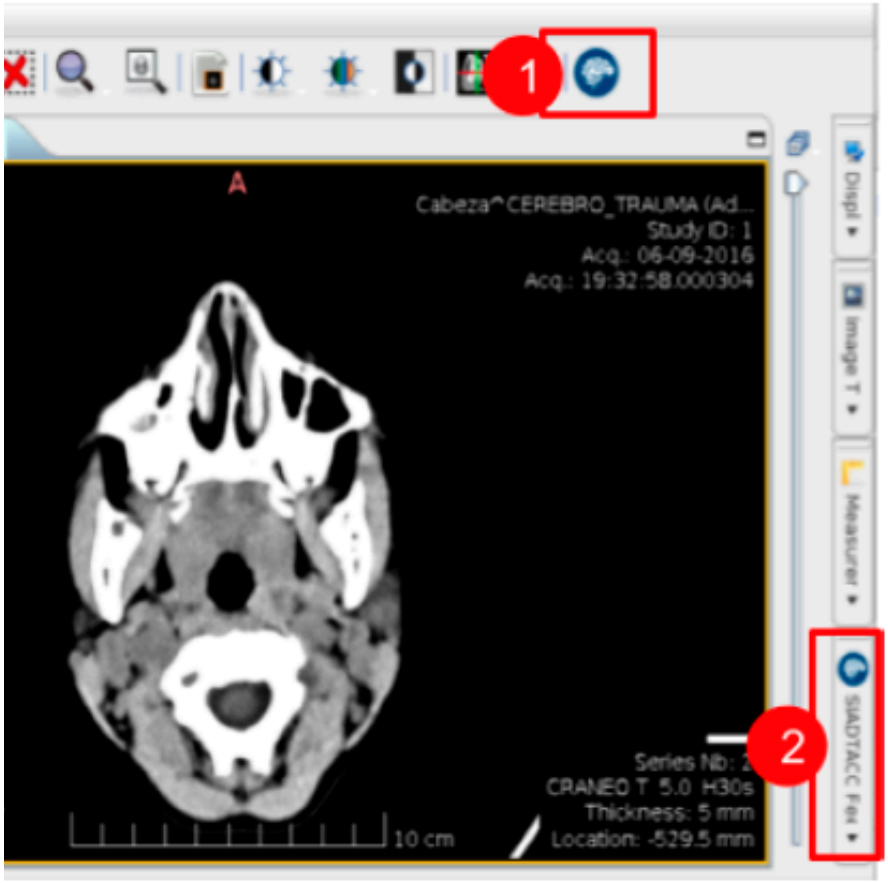


Ilustración 98 - Pantalla principal del visor Weasis

El detalle de cada una de las funciones disponibles en el plugin se encuentra en el manual de usuario, consultar dicho documento para consultas al respecto.

6.2.3 Requerimientos para la implementación

Los requerimientos necesarios para la puesta en producción del software SIADTACC están divididos en tres grupos, los cuales se detallan a continuación:

6.2.3.1 Requerimientos de hardware

Se tienen dos tipos de equipo, los servidores donde se instalará la RNA y el SIADTACC y las computadoras que se utilizarán como cliente en las estaciones de trabajo. Se han tomado en consideración las características del hardware que posee el Ministerio de Salud y los requerimientos mínimos de las tecnologías a utilizar.

Equipo	Cantidad	Especificaciones
Servidor para la RNA (VPS)	1	• Procesador: 6 núcleos • **RAM:** 16GB 1600mhz • Disco Duro: 200GB • Tarjeta de red Ethernet integrada.
Servidor para el SIADTACC	1	• **Procesador:** 1 Intel Xeon E5-2620, 2.4GHz, 6C/12T, 15 MB Cache • **RAM:** 16 GB (2x8GB), 2133 MT/s, Dual Rank RDIMM • **Controladora de discos duros:** Controladora RAID 0, 1, 5, 6, 10 60, 512MB de Cache, con capacidad para 8 HD, Hot Swap. • **Discos duros:** 2 x 500GB 7.2K RPM Serial-Attach SCSI 3Gbps 3.5in (Configurados en RAID 1), Hot-plug. • Adaptador de red a 1 Gigabit cuádruple puerto, cobre, PCIe-4.
Computadora cliente	20	• **Procesador:** Intel Dual Core PDC 3.2GHZ G3250 S-1150 3M. • **RAM:** 4 GB DDR3 1333. • **Disco duro:** 500 GB 7500 RPM 32MB CACHE SATA. • Tarjeta de red Ethernet integrada.

Tabla 84 - Requerimientos de hardware en la etapa de implementación

6.2.3.2 Requerimientos de software

Los requerimientos de software en el entorno de producción son los mismos que en el entorno de desarrollo.

N°	Nombre	Versión	Descripción
1	Debian/Linux	8.6 Jessie 64bits	Sistema operativo con repositorios del Ministerio de Salud de El Salvador
2	PostgreSQL	9.4+165+deb8u1	Gestor de base de datos
3	Python	2.7.3	Lenguaje de programación
4	Java (JDK)	1.6.0_38+	Lenguaje de programación, última versión disponible en los repositorios de Debian Jessie (hasta la fecha).
5	Apache Server	2.2.22-13+deb7u1+	Servidor web HTTP de código abierto.
6	JBOSS	4.2.3 GA	Servidor de aplicaciones Java EE de código abierto implementado en la

			especificación Java EE. Utilizado para ejecutar DCM4CHEE y sus componentes.
7	DCM4CHEE	2.18.0	Colección de aplicaciones de código abierto y utilidades para las instituciones de salud.
8	Weasis	2.0.3	Visor web de archivos en formato DICOM
9	CAFFE	0.9999	Caffe es un framework para el desarrollo de aplicaciones de deep learning, posee una interfaz de línea de comandos para el lenguaje de programación python (pycaffe).
10	CUDA	7+	Modelo de computación y programación paralela desarrollado por NVIDIA. Permite un aumento drástico en cuanto a rendimiento de cómputo, aprovechando las capacidades de la unidad de procesamiento gráfico (GPU).
11	Pycharm	2016.2.3	IDE para el lenguaje de programación Python
12	NetBeans	8.2+	IDE para el lenguaje de programación JAVA
13	Nginx	1.6.2-5+deb8u2+	Servidor web/proxy inverso, ligero de alto rendimiento y un proxy para protocolos de correo electrónico (IMAP/POP3)
14	Gunicorn	19.6+	Servidor HTTP de Python WSGI.
15	Django	1.10.2	Framework web de Python de alto nivel que fomenta el desarrollo rápido, diseño limpio y pragmático.
16	GIT	1:2.1.4-2.1+deb8u2+	Sistema de control de versiones
17	Sun's Java Advanced Imaging (JAI) Image I/O Tools	1.0_01	

85 - Requerimientos de software

6.2.3.3 Requerimientos de recurso humano

A continuación se lista los perfiles y la cantidad de personas necesarias para el desarrollo del proyecto.

Cantidad	Cargo	Descripción
4	Analistas programadores	Encargados de realizar las actividades de instalación y configuración de los componentes del SIADTACC. Es

		recomendable que 2 analistas sean parte del equipo de desarrollo del proyecto y los otros 2 de la DTIC del MINSAL.
1	Administrador de proyecto	Responsable de coordinar las actividades del plan de implementación, es necesario que tenga experiencia con el manejo del SIMAGD.

Tabla 86 - Requerimientos de recurso humano la etapa de implementación

6.2.4 Modelos Lógico y Físico de datos

6.2.4.1 Modelo lógico

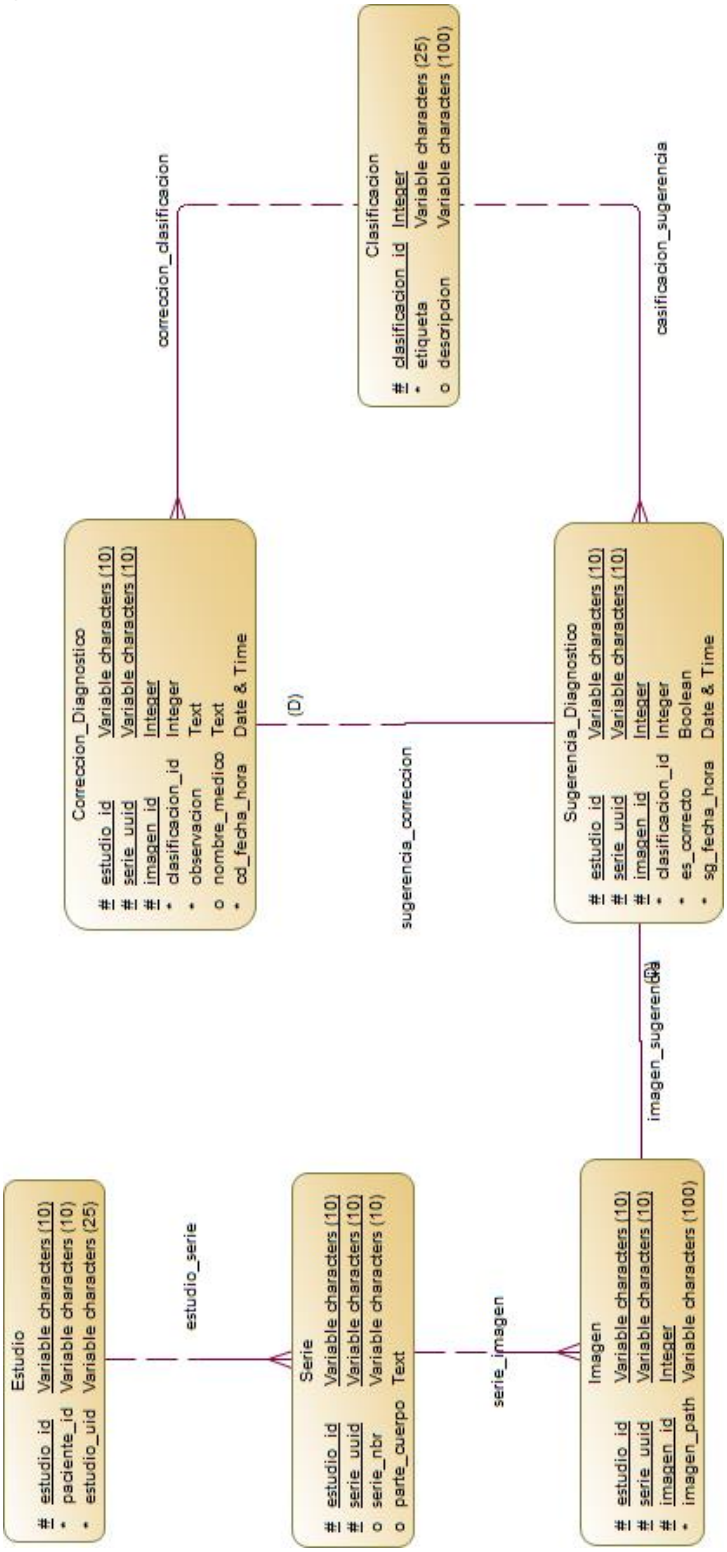


Ilustración 99 - Diagrama de secuencia general

6.2.4.2 Modelo físico

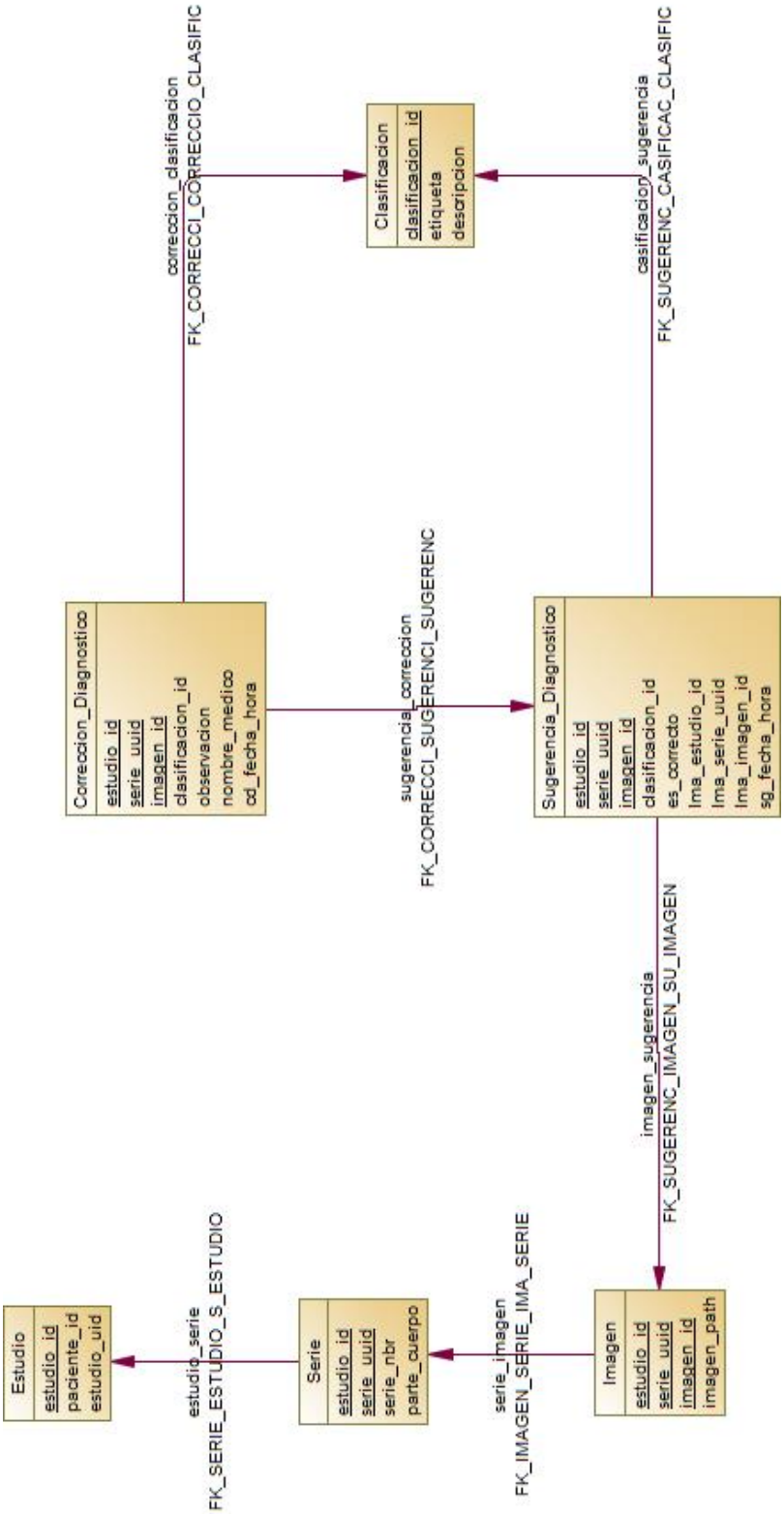


Ilustración 100 Modelo físico

6.2.5 Diccionario de Datos

6.2.5.1 Lista de tablas en orden alfabético

1. Clasificacion
2. Correccion_Diagnostico
3. Estudio
4. Imagen
5. Serie
6. Sugerencia_Diagnostico

6.2.5.2 Descripción de la base de datos

Para la definición de los tipos de datos y sus respectivos tamaños se han tomado en consideración las especificaciones técnicas del gestor de base de datos PostgreSQL.

Tabla: Clasificación **Descripción:** Tabla que contiene las diferentes clasificaciones posibles de una imagen.					
Lista de columnas					
Nombre de columna	Tipo de dato	Tamaño	Descripción		
clasificacion_id	Integer	32	Identificador de la clasificación que puede tener una imagen.		
Etiqueta	Varchar	25	Identificador nemotécnico de una clasificación.		
Descripción	Varchar	100	Descripción a detalle de una clasificación.		
Referencias hacia la tabla					
1. clasificacion_sugerencia 2. correccion_clasificacion					
Lista de índices de la tabla					
Nombre	Único	Cluster	Primario	Foráneo	Columnas
CLASIFICACION_PK	X	X	X		clasificacion_id

Tabla 87 - Descripción de la base de datos

Tabla: Correccion_Diagnostico **Descripción:** Almacena las observaciones y la clasificación real que un radiólogo realiza sobre una clasificación hecha por la red neuronal.			
Lista de columnas			
Nombre de columna	Tipo de dato	Tamaño	Descripción
estudio_id	Varchar	10	Identificador del estudio
serie_uuid	Varchar	10	Identificador de la serie
imagen_id	Integer	32	Identificador de la imagen.
clasificacion_id	Integer	32	Identificador de la clasificación correcta
observacion	Text	N/A	Observacion sobre el analisis realizado por la rna
nombre_medico	Text	N/A	nombre del médico que hace la corrección
cd_fecha_hora	DateTime	8	Fecha y hora de la corrección
Referencias desde la tabla			
1. correccion_clasificacion. 2. sugerencia_correccion.			
Lista de índices de la tabla			

Nombre	Único	Cluster	Primario	Foráneo	Columnas
CORRECCION_DIAGNOSTICO_PK	X	X	X		estudio_id serie_uuid imagen_id
CORRECCION_CLASIFICACION_FK				X	estudio_id serie_uuid imagen_id

Tabla 88 - Descripción de la tabla Correccion_Diagnostico

Tabla: Estudio **Descripción:** Contenedor de series					
Lista de columnas					
Nombre de columna	Tipo de dato	Tamaño	Descripción		
estudio_id	Varchar	10	Identificador del estudio		
paciente_id	Varchar	10	Identificador del paciente		
estudio_uuid	Varchar	25	Uuid del estudio		
Referencias hacia la tabla					
1. estudio_serie.					
Lista de índices de la tabla					
Nombre	Único	Cluster	Primario	Foráneo	Columnas
ESTUDIO_PK	X	X	X		estudio_id

Tabla 89 - Descripción de la tabla Estudio

Tabla: Imagen **Descripción:** Contiene los identificadores de la imagen en el PACS.					
Lista de columnas					
Nombre de columna	Tipo de dato	Tamaño	Descripción		
estudio_id	Varchar	10	Identificador del estudio		
serie_uuid	Varchar	10	Identificador de la serie		
imagen_id	Integer	32	Identificador de la imagen		
imagen_path	Varchar	100	Dirección de almacenamiento.		
Referencias hacia la tabla					
1. serie_imagen.					
Lista de índices de la tabla					
Nombre	Único	Cluster	Primario	Foráneo	Columnas
IMAGEN_PK	X	X	X		estudio_id serie_uuid imagen_id

Tabla 90 - Descripción de la tabla Imagen

Tabla: Serie **Descripción:** Contenedor de imágenes, según parte del cuerpo a la cual se realizo el estudio					
Lista de columnas					
Nombre de columna	Tipo de dato	Tamaño	Descripción		
estudio_id	Varchar	10	Identificador del estudio		
serie_uuid	Varchar	10	Identificador de la serie		
serie_nbr	Varchar	10	Número de la serie del estudio		
parte_cuerpo	Text	N/A	Parte del cuerpo que se ha capturado en la serie		
Referencias hacia la tabla					
1. serie_imagen					
Referencias desde la tabla					
1. estudio_serie					
Lista de índices de la tabla					
Nombre	Único	Cluster	Primario	Foráneo	Columnas
SERIE_PK	X	X	X		estudio_id serie_uuid
ESTUDIO_SERIE_FK				X	estudio_id

Tabla 91 - Descripción de la tabla Serie

Tabla: Sugerencia_Diagnostico Descripción:			
Lista de columnas			
Nombre de columna	Tipo de dato	Tamaño	Descripción
estudio_id	Varchar	10	Identificador del estudio

serie_uuid	Varchar	10	Identificador de la serie
imagen_id	Integer	32	Identificador de la imagen
clasificacion_id	Integer	32	Identificador de la clasificación correcta
es_correcto	Boolean	1	Bandera que valida la clasificación hecha por la RNA
cd_fecha_hora	DateTime	8	Fecha y hora en que se realizó la clasificación

Referencias hacia la tabla

1. sugerencia_correccion

Referencias desde la tabla

1. imagen_sugerencia
2. casificacion_sugerencia.

Lista de índices de la tabla

Nombre	Único	Cluster	Primario	Foráneo	Columnas
SUGERENCIA_DIAGNOSTICO_PK	X	X	X		estudio_id serie_uuid imagen_id
CASIFICACION_SUGERENCIA_FK				X	clasificacion_id

Tabla 92 - Descripción de la tabla Sugerencia_Diagnostico

6.2.5.3 Lista de referencias

Nombre Referencia	Tabla Padre	Columnas Tabla Padre	Tabla Hija	Columnas Tabla Hija
estudio_serie	Estudio	estudio_id	Serie	estudio_id
serie_imagen	Serie	estudio_id serie_uuid	Imagen	estudio_id serie_uuid
imagen_sugerencia	Imagen	estudio_id serie_uuid imagen_id	Sugerencia_Diagnostico	estudio_id serie_uuid imagen_id
sugerencia_correccion	Sugerencia_Diagnostico	estudio_id serie_uuid imagen_id	Correccion_Diagnostico	estudio_id serie_uuid imagen_id
casificacion_sugerencia	Clasificacion	clasificacion_id	Sugerencia_Diagnostico	clasificacion_id
correccion_clasificacion	Clasificacion	clasificacion_id	Correccion_Diagnostico	clasificacion_id

Tabla 93 - Listado de referencias entre tablas de la base de datos

6.2.6 Procesos de mantenimiento del sistema

6.2.6.1 Compilar plugin para Weasis

- Crear una nueva carpeta que servirá como workspace.
- Dentro de esa nueva carpeta clonar el repositorio del visor Weasis

```
git clone git://github.com/nroduit/Weasis.git
```

- Nos movemos a la carpeta recién creada (Weasis) y nos cambiamos a la ultima rama estable del weasis

```
git checkout 2.0.7
```

- Despues ejecutamos lo siguiente (siempre en el directorio raíz de Weasis)

```
mvn clean install
```

- Finalizado el proceso, nos movemos a la carpeta weasis-dicom; dentro de ella clonamos el proyecto correspondiente al plugin.

```
git clone git@bitbucket.org:wesis_plugin/weasis-siadtacc-plugin.git
```

- En el pom.xml dentro de la carpeta del proyecto asegurarse que el ***groupId*** del plugin sea ***org.weasis***, que la **versión del parent sea 2.0.7** y que el **relative path** este debidamente referenciado al weasis-parent/pom.xml, como se muestra a continuación.

```
<parent>
  <artifactId>weasis-parent</artifactId>
```

```
    <groupId>org.weasis</groupId>
    <version>2.0.7</version>
    <relativePath>../../weasis-parent/pom.xml</relativePath>
  </parent>
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.weasis</groupId>
  <artifactId>weasis-siadtacc-plugin</artifactId>
  <packaging>bundle</packaging>
  <name>Siadtacc custom plugin [${project.artifactId}]</name>
```

- En el archivo ***conf/ext-config.properties*** (dentro del plugin) verificar que la siguiente linea este comentada:

  #org.osgi.framework.startlevel.beginning=120

- Por ultimo ya se puede compilar el plugin, esto generá un archivo .jar en la carpeta *target/*:

```
mvn install
```

El siguiente paso es generar el archivo .war que empaqueta nuestro plugin (archivo .jar generado) y luego añadirlo al weasis. Para poder añadir los diferentes plugins desarrollados es necesario generar un archivo .war llamado weasis-ext.war que sera colocado junto al archivo weasis.war en la carpeta server/default/deploy del DCM4CHEE.

El equipo de desarrollo weasis provee un proyecto con la estructura básica para generar el archivo .war, solo basta con clonar el siguiente repositorio: https://github.com/nroduit/weasis-plugins-war-buildery agregar nuestro plugin personalizado (archivo .jar) en el directorio raíz del proyecto:

La estructura del proyecto weasis-plugins-war-builder es la siguiente:

+- -weasis-ext/ | +- -WEB-INF/

| +- - - web.xml | +- -conf/

| +- - - ext-config.properties | +- -**plugin1.jar** | +- -**plugin2.jar**

- En necesario clonar el proyecto del war-builder en el mismo nivel que nuestro plugin y llamarlo weasis-ext:
- git clone https://github.com/nroduit/weasis-plugins-war-builder.git weasis-ext
- En el archivo ***pom.xml*** del weasis-ext configurar la sección correspondiente al plugin colocar el groupId, artifactId y versión, como se muestra a continuación.

```
<artifactItems>
  <artifactItem>
    <groupId>org.weasis</groupId>
    <artifactId>weasis-siadtacc-plugin</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <type>jar</type>
  </artifactItem>
</artifactItems>
```

- En el archivo ***src/main/webapp/conf/ext-config.properties*** (del weasis-ext) asegurarse que la siguiente linea tenga el nombre y versión de nuestro plugin (aproximadamente la linea 22, sección Add additional plugins)

  *felix.auto.start.81= \ ${weasis.codebase.ext.url}/weasis-siadtacc-plugin-0.0.1-SNAPSHOT.jar *

Finalizada la configuración del weasis-ext, procedemos a empaquetarlo; para que el DCM4CHEE no lance una excepción de seguridad es necesario firmar el .war; para eso necesitamos generar un certificado, ejemplo:

```
keytool -genkey -keyalg RSA -alias tesis_ia_2016 -keystore ~/.keystore/weasis_test.jks -storepass “t3s1s&2016” -validity 360 -keysize 2048
```

Para empaquetar el archivo .war ejecutamos el siguiente comando (desde el root del weasis-ext) :

```
mvn clean package -Djarsigner.alias="tesis_ia_2016" -Djarsigner.storepass="t3s1s&2016" -Djarsigner.keystore=" ~/.keystore/weasis_test.jks"
```

El proceso anterior genera el archivo .war en la carpeta target (***target/weasis-ext.war***). Con esto ya tenemos el weasis-ext empaquetado con nuestro plugin (.jar), solo hace falta publicarlo en la carpeta ***/server/default/deploy/*** del dcm4chee.

Cabe mencionar que es necesario utilizar la version 5.0.X del weasis-pacs-connector, el cual se descarga de este enlace. Se deben descargar los archivos ***weasis-pacs-connector.war*** y ***dcm4chee-web-weasis.jar*** y colocarlos en la carpeta ***/server/default/deploy/*** del dcm4chee.

6.2.6.2 Debug del proyecto en Eclipse

Para realizar el proceso de eliminación de posibles errores (debugging) en el plugin desarrollado es necesario realizar los siguientes pasos:

1. Abrir Perspectiva de GIT Window>Perspective>Open Perspective>Other>GIT
2. Clonar repositorio desde "Clone a GIT repository"
3. En el nuevo dialogo copiar la URL git://github.com/nroduit/Weasis.git
4. Clic en Siguiente y Finalizar
5. Desde la perspectiva de Java EE: File>Import
6. En el nuevo dialogo seleccionar: Maven>Existing Maven Projects Y abrir el directorio que clonaste.
7. Se abren un montón de proyectos y da un error que no afecta al final.
8. Selecciónas el proyecto que se llama weasis-framework y le das install.
9. Configurar la ejecución: Abrir Run>Debug configurations

Crear una "new Java Application" En la Tab Main en el project poner "weasis-launcher" En la Main class poner "org.weasis.launcher.WeasisLauncher"

1. En la pestaña Arguments:

Para ejecutar weasis necesitás tener una carpeta con algún dicom, podés bajarlo de los que están en el drive.

en Program arguments tenés que poner la ruta del dicom que te va a abrir: Como ejemplo: $dicom:get -l "D:\images\dicom" En linux es algo como $dicom:get -l "/home/user/Images/MRIX LUMBAR" en VMarguments:

-Xms32m -Xmx512m -

Dmaven.localRepository="C:/Users/Marlon%20Menjivar/.m2/repository" -Dgosh.args="- sc telnetd -p 17179 start"

En donde -Dmaven.localRepository tenés que hacer que apunte al directorio del repositorio de Maven

1. Aplicás los cambios y ya podés darle RUN.
2. Para debug te vas a la pestaña de debug y ahí tiene que salir la nueva configuración que has hecho para debuguear.

6.2.6.3 Debug del weasis en dcm4chee

En la home del usuario se crea una carpeta oculta .weasis/ dentro de ella esta la carpeta preferences/username/default; alli se encuentra el archivo weasis.properties, su contenido debe quedar como este:

```
#Mon Nov 14 18:49:48 CST 2016
locale.lang.code=en
org.apache.sling.commons.log.file.size=10MB
weasis.confirm.closing=false
weasis.look=com.sun.java.swing.plaf.gtk.GTKLookAndFeel
org.apache.sling.commons.log.file.number=5
org.apache.sling.commons.log.level=TRACE
weasis.show.disclaimer=false
org.apache.sling.commons.log.file.activate=true
weasis.download.immediately=true
weasis.version.release=2.0.7
locale.format.code=es_SV
```

Al reiniciar weasis encontraremos un archivo en la siguiente ruta .weasis/log/default.log, para revisarlo en tiempo de ejecución utilizamos el siguiente comando:

```
tailf ~/.weasis/log/default.log
```

Algunas configuraciones de ese archivo se pueden realizar mediante la interfaz grafica del weasis, en: File > Preferences > General

6.3. Manual de instalación

6.3.1 Instalación y configuración de backend.

6.3.1.1 Verificar instalación de python 2 y sus dependencias.

```
$ su
# aptitude install python-skimage python-pip
```

```
administrador@debian:~$ su
Password:
root@debian:/home/administrador# aptitude install python-skimage python-pip
No packages will be installed, upgraded, or removed.
0 packages upgraded, 0 newly installed, 0 to remove and 63 not upgraded.
Need to get 0 B of archives. After unpacking 0 B will be used.

root@debian:/home/administrador#
```

Ilustración 101 - Instalación dependencias python

6.3.1.2 Instalación de caffe.

```
$ su
# aptitude install git
```

```
root@debian:/home/administrador# aptitude install git
The following NEW packages will be installed:
  git git-man{a} liberror-perl{a} rsync{a}
0 packages upgraded, 4 newly installed, 0 to remove and 63 not upgraded.
Need to get 4,942 kB of archives. After unpacking 24.3 MB will be used.
Do you want to continue? [Y/n/?] y
Get: 1 http://ftp.br.debian.org/debian/ jessie/main liberror-perl all 0.17-1.1 [22.4 kB]
Get: 2 http://ftp.br.debian.org/debian/ jessie/main git-man all 1:2.1.4-2.1+deb8u2 [1,267 kB]
Get: 3 http://ftp.br.debian.org/debian/ jessie/main git amd64 1:2.1.4-2.1+deb8u2 [3,262 kB]
Get: 4 http://ftp.br.debian.org/debian/ jessie/main rsync amd64 3.1.1-3 [390 kB]
Fetched 4,942 kB in 7s (641 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 163746 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17-1.1_all.deb ...
Unpacking liberror-perl (0.17-1.1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.1.4-2.1+deb8u2_all.deb ...
Unpacking git-man (1:2.1.4-2.1+deb8u2) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.1.4-2.1+deb8u2_amd64.deb ...
Unpacking git (1:2.1.4-2.1+deb8u2) ...
Selecting previously unselected package rsync.
Preparing to unpack .../rsync_3.1.1-3_amd64.deb ...
Unpacking rsync (3.1.1-3) ...
Processing triggers for man-db (2.7.0.2-5) ...
Processing triggers for systemd (215-17+deb8u5) ...
Setting up liberror-perl (0.17-1.1) ...
Setting up git-man (1:2.1.4-2.1+deb8u2) ...
Setting up git (1:2.1.4-2.1+deb8u2) ...
Setting up rsync (3.1.1-3) ...
Processing triggers for systemd (215-17+deb8u5) ...

root@debian:/home/administrador#
```

Ilustración 102 - Instalacion de git

```
$ git clone https://github.com/BVLC/caffe.git
```

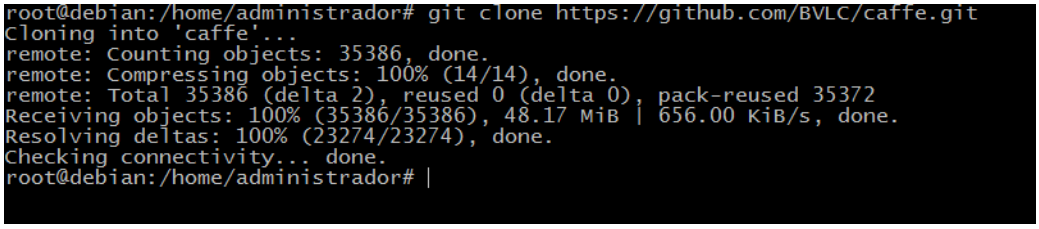
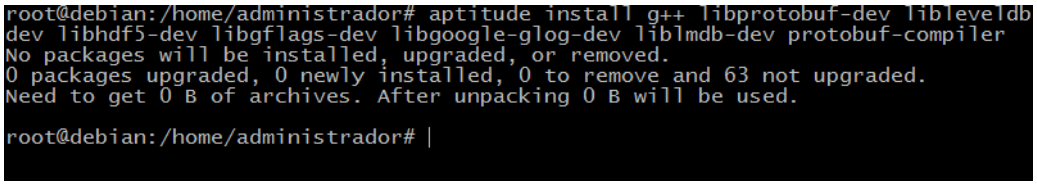


Ilustración 103 - Clonado de repositorio

```
# aptitude install g++ libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libboost-all-dev libhdf5-dev libgflags-dev libgoogle-glog-dev liblmdb-dev protobuf-compiler libopenblas-dev libatlas-base-dev

# pip install protobuf
```



104 - Instalacion de dependencias de caffe I

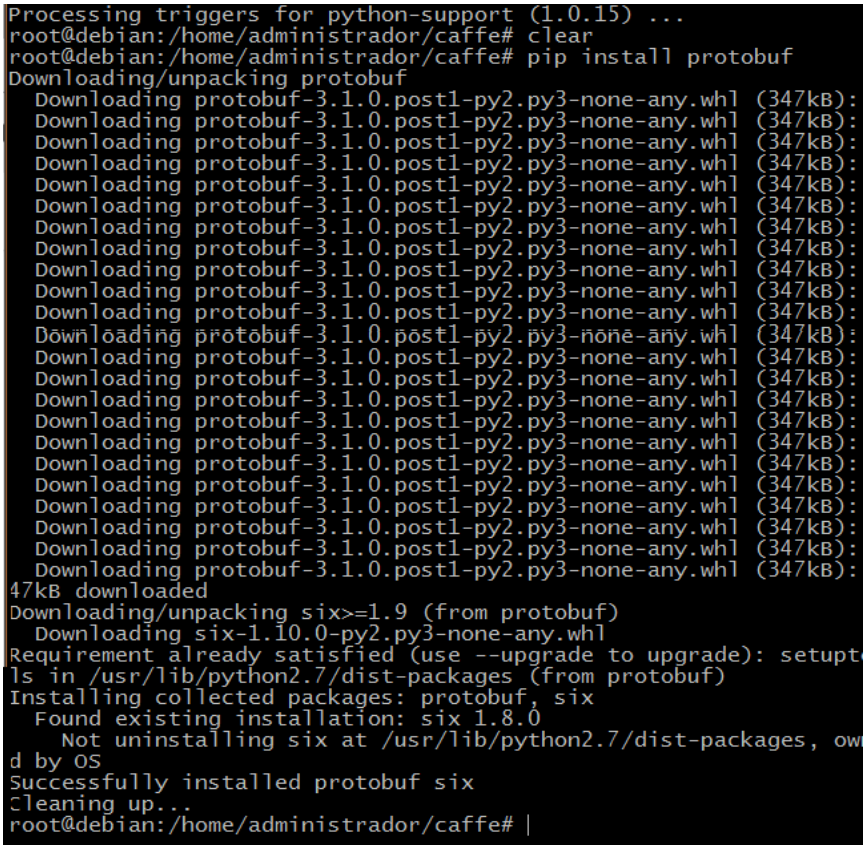


Ilustración 105 - Instalacion de dependencias de caffe II

```
# cd /usr/lib/x86_64-linux-gnu
# sudo ln -s libhdf5_serial.so.8.0.2 libhdf5.so
# sudo ln -s libhdf5_serial_hl.so.8.0.2 libhdf5_hl.so
```

```
root@debian:/usr/lib/x86_64-linux-gnu# cd /usr/lib/x86_64-linux-gnu
root@debian:/usr/lib/x86_64-linux-gnu# sudo ln -s libhdf5_serial.so.8.0.2 libhdf5.so
root@debian:/usr/lib/x86_64-linux-gnu# sudo ln -s libhdf5_serial_hl.so.8.0.2 libhdf5_hl.so
root@debian:/usr/lib/x86_64-linux-gnu#
```

Ilustración 106 - Vinculación de dependencias

```
# cd (dirección donde se clono el directorio de caffe )
```

Crear dos vínculos simbólicos en el directorio fuente de caffe

```
# ln -s /usr/lib/x86_64-linux-gnu/libhdf5.so .
# ln -s /usr/lib/x86_64-linux-gnu/libhdf5_hl_cpp.so libhdf5_hl.so
# cp Makefile.config.example Makefile.config
```

```
root@debian:/home/administrador# cd caffe
root@debian:/home/administrador/caffe# cp Makefile.config.example Makefile.config
```

Ilustración 107 - Compilacion de caffe I

```
# nano Makefile.config
```

Editar el archivo, descomentar la linea que dice CPU_ONLY y reemplazar las configuraciones siguientes:

INCLUDE_DIRS := $(PYTHON_INCLUDE) /usr/local/include /usr/include/hdf5/serial

LIBRARY_DIRS := $(PYTHON_LIB) /usr/local/lib /usr/lib /usr/lib/x86_64-linux-gnu

```
  GNU nano 2.2.6                File: Makefile.config

## Refer to http://caffe.berkeleyvision.org/installation.html
# Contributions simplifying and improving our build system are welcome!

# cuDNN acceleration switch (uncomment to build with cuDNN).
# USE_CUDNN := 1

# CPU-only switch (uncomment to build without GPU support).
CPU_ONLY := 1

# uncomment to disable IO dependencies and corresponding data layers
# USE_OPENCV := 0
# USE_LEVELDB := 0
# USE_LMDB := 0

# uncomment to allow MDB_NOLOCK when reading LMDB files (only if necessary)
#       You should not set this flag if you will be reading LMDBs with any
#       possibility of simultaneous read and write
# ALLOW_LMDB_NOLOCK := 1

                              [ Read 113 lines ]
^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

Ilustración 108 - Compilación de caffe II

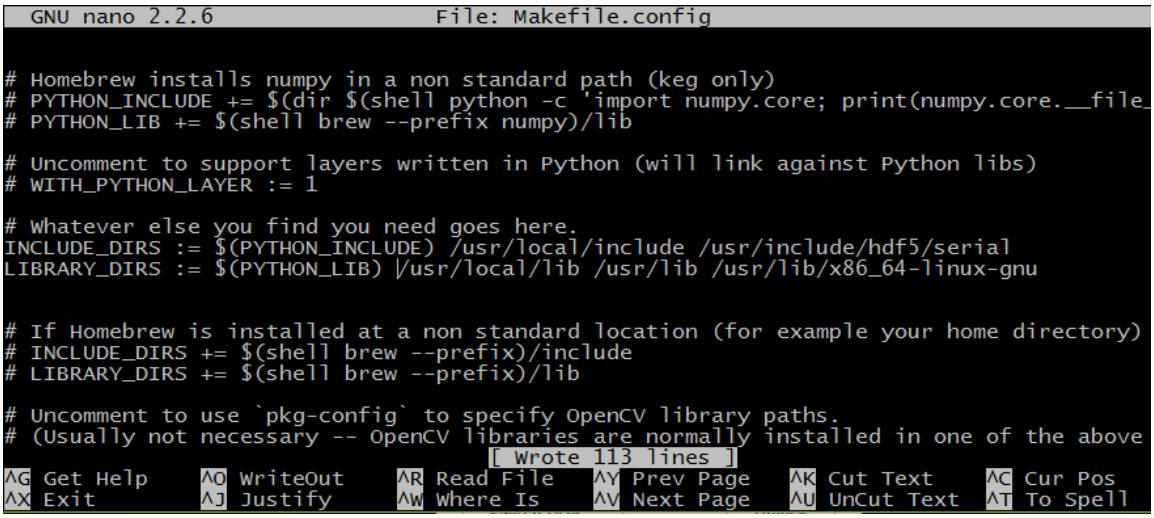


Ilustración 109 - Compilación de caffe III

```
make all
```

```
root@debian:/home/administrador/caffe# make all
LD -o .build_release/lib/libcaffe.so.1.0.0-rc3
CXX tools/upgrade_solver_proto_text.cpp
CXX/LD -o .build_release/tools/upgrade_solver_proto_text.bin
CXX tools/compute_image_mean.cpp
CXX/LD -o .build_release/tools/compute_image_mean.bin
CXX tools/convert_imageset.cpp
CXX/LD -o .build_release/tools/convert_imageset.bin
CXX tools/net_speed_benchmark.cpp
CXX/LD -o .build_release/tools/net_speed_benchmark.bin
CXX tools/device_query.cpp
CXX/LD -o .build_release/tools/device_query.bin
CXX tools/train_net.cpp
CXX/LD -o .build_release/tools/train_net.bin
CXX tools/finetune_net.cpp
CXX/LD -o .build_release/tools/finetune_net.bin
CXX tools/extract_features.cpp
CXX/LD -o .build_release/tools/extract_features.bin
CXX tools/upgrade_net_proto_binary.cpp
CXX/LD -o .build_release/tools/upgrade_net_proto_binary.bin
CXX tools/test_net.cpp
CXX/LD -o .build_release/tools/test_net.bin
CXX tools/caffe.cpp
CXX/LD -o .build_release/tools/caffe.bin
CXX tools/upgrade_net_proto_text.cpp
CXX/LD -o .build_release/tools/upgrade_net_proto_text.bin
CXX examples/cpp_classification/classification.cpp
CXX/LD -o .build_release/examples/cpp_classification/classification.bin
CXX examples/siamese/convert_mnist_siamese_data.cpp
CXX/LD -o .build_release/examples/siamese/convert_mnist_siamese_data.bin
CXX examples/cifar10/convert_cifar_data.cpp
CXX/LD -o .build_release/examples/cifar10/convert_cifar_data.bin
CXX examples/mnist/convert_mnist_data.cpp
CXX/LD -o .build_release/examples/mnist/convert_mnist_data.bin
```

Ilustración 110 - Compilación de caffe IV

```
make pycaffe
```

```
root@debian:/home/administrador/caffe# make pycaffe
CXX/LD -o python/caffe/_caffe.so python/caffe/_caffe.cpp
touch python/caffe/proto/__init__.py
PROTOC (python) src/caffe/proto/caffe.proto
root@debian:/home/administrador/caffe#
```

Ilustración 111 - Compilación de caffe V

```
# make distribute
```

```
root@debian:/home/administrador/caffe# make distribute
# add proto
cp -r src/caffe/proto distribute/
# add include
cp -r include distribute/
mkdir -p distribute/include/caffe/proto
cp .build_release/src/caffe/proto/caffe.pb.h distribute/include/caffe/proto
# add tool and example binaries
cp .build_release/tools/upgrade_solver_proto_text.bin .build_release/tools/compu
_imageset.bin .build_release/tools/net_speed_benchmark.bin .build_release/tools/
t.bin .build_release/tools/finetune_net.bin .build_release/tools/extract_feature
inary.bin .build_release/tools/test_net.bin .build_release/tools/caffe.bin .buil
tribute/bin
cp .build_release/examples/cpp_classification/classification.bin .build_release/
n .build_release/examples/cifar10/convert_cifar_data.bin .build_release/examples/
# add libraries
cp .build_release/lib/libcaffe.a distribute/lib
install -m 644 .build_release/lib/libcaffe.so.1.0.0-rc3 distribute/lib
cd distribute/lib; rm -f libcaffe.so;   ln -s libcaffe.so.1.0.0-rc3 libcaffe.so
# add python - it's not the standard way, indeed...
cp -r python distribute/python
root@debian:/home/administrador/caffe#
```

Ilustración 112 - Compilación de caffe VI

```
# nano /<dirección de home de usuario/.bashrc
```

agregar la línea:

```
export PYTHONPATH=/(direccion de caffe)/python
```

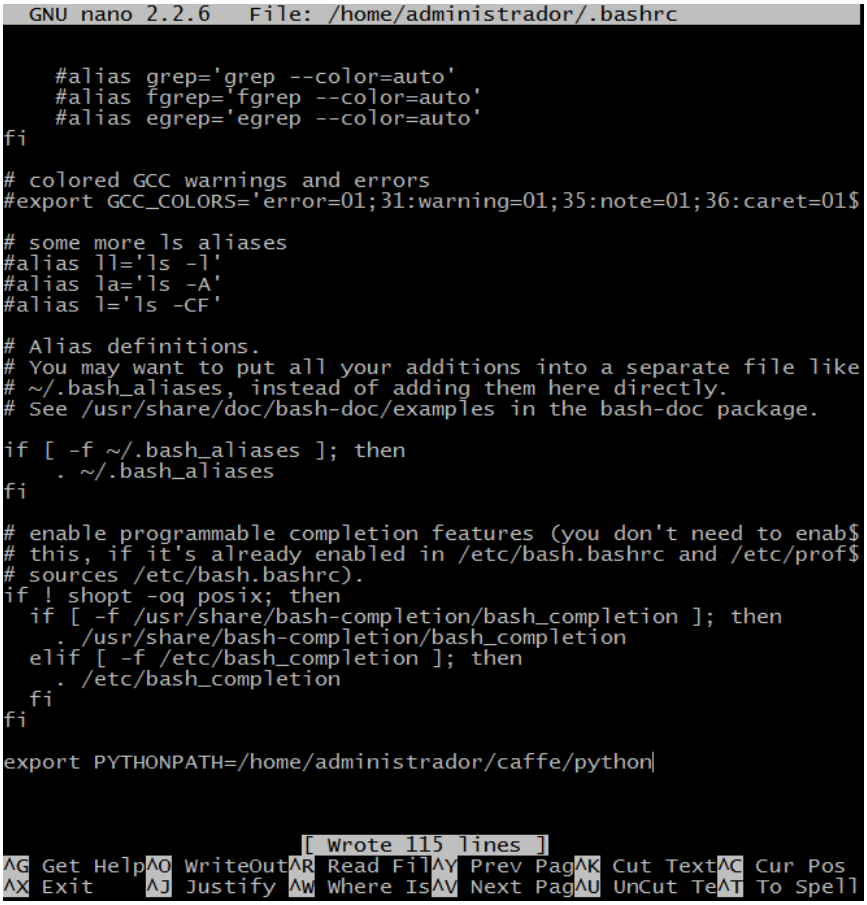


Ilustración 113 - Configuración de pycaffe

```
$ source /<direccion de home de usuario/.bashrc
$ python
>> import caffe
```

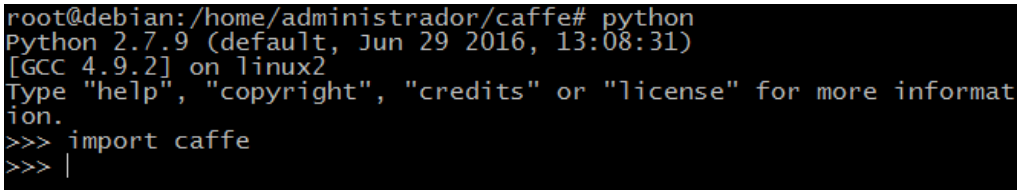


Ilustración 114 - Prueba de pycaffe

6.3.1.3 Instalación de postgresql

```
# aptitude install postgresql postgresql-contrib
```

```
root@debian:/home/administrador/caffe# aptitude install postgresql
 postgresql-contrib
The following NEW packages will be installed:
  libpq5{a} postgresql postgresql-9.4{a}
  postgresql-client-9.4{a} postgresql-client-common{a}
  postgresql-common{a} postgresql-contrib
  postgresql-contrib-9.4{a}
0 packages upgraded, 8 newly installed, 0 to remove and 63 not upg
raded.
Need to get 5,716 kB of archives. After unpacking 28.7 MB will be
used.
Do you want to continue? [Y/n/?] y
Get: 1 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in libpq5 amd64 9.4.10-0+deb8u1 [125 kB]
Get: 2 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in postgresql-client-common all 165+deb8u2 [73.7 kB]
Get: 3 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in postgresql-client-9.4 amd64 9.4.10-0+deb8u1 [1,081 kB]
Get: 4 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in postgresql-common all 165+deb8u2 [200 kB]
Get: 5 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in postgresql-9.4 amd64 9.4.10-0+deb8u1 [3,681 kB]
Get: 6 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in postgresql all 9.4+165+deb8u2 [52.2 kB]
Get: 7 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in postgresql-contrib-9.4 amd64 9.4.10-0+deb8u1 [452 kB]
Get: 8 http://ftp.br.debian.org/debian/ jessie-proposed-updates/ma
in postgresql-contrib all 9.4+165+deb8u2 [52.2 kB]
```

Ilustración 115 - Instalación de PostgreSQL

6.3.1.4 Instalación de gunicorn.

Como se muestra en la Ilustración 65 - Arquitectura de SIADTACC, es necesario la instalación de gunicorn como middleware para la ejecución de la aplicación web django.

```
# pip install gunicorn gevent
```

```
root@debian:/home/administrador# pip install gunicorn gevent
Requirement already satisfied (use --upgrade to upgrade): gunicorn in /usr/local/lib/python2.7/dist-packages
Downloading/unpacking gevent
  Downloading gevent-1.2.1.tar.gz (2.8MB): 2.8MB downloaded
  Running setup.py (path:/tmp/pip-build-plFom0/gevent/setup.py) egg_info for package gevent
    /usr/lib/python2.7/distutils/dist.py:267: UserWarning: Unknown distribution option: 'cffi_modules'
      warnings.warn(msg)
  Running setup.py install for greenlet
    building 'greenlet' extension
    x86_64-linux-gnu-gcc -pthread -DNDEBUG -g -fwrapv -O2 -Wall -Wstrict-prototypes -fno-strict-aliasing -D_FORTIFY_SOUR
CE=2 -g -fstack-protector-strong -Wformat -Werror=format-security -fPIC -I/usr/include/python2.7 -c greenlet.c -o build/
temp.linux-x86_64-2.7/greenlet.o
    x86_64-linux-gnu-gcc -pthread -shared -Wl,-O1 -Wl,-Bsymbolic-functions -Wl,-z,relro -fno-strict-aliasing -DNDEBUG -g
 -fwrapv -O2 -Wall -Wstrict-prototypes -D_FORTIFY_SOURCE=2 -g -fstack-protector-strong -Wformat -Werror=format-security
-Wl,-z,relro -D_FORTIFY_SOURCE=2 -g -fstack-protector-strong -Wformat -Werror=format-security build/temp.linux-x86_64-2.
7/greenlet.o -o build/lib.linux-x86_64-2.7/greenlet.so

Successfully installed gevent greenlet
Cleaning up...
root@debian:/home/administrador#
```

Ilustración 116 - Instalación de gunicorn

6.3.1.5 Instalación de nginx y supervisor

Como se muestra en la Ilustración 65 - Arquitectura de SIADTACC, es necesario la instalación de nginx como balanceador de carga entre las instancias de gunicorn.

```
# aptitude install supervisor nginx
```

```
root@debian:/home/administrador# aptitude install supervisor nginx
The following NEW packages will be installed:
  nginx nginx-common{a} nginx-full{a} python-meld3{a} supervisor
0 packages upgraded, 5 newly installed, 0 to remove and 63 not upgraded.
Need to get 895 kB of archives. After unpacking 2,896 kB will be used.
Do you want to continue? [Y/n/?] y
Get: 1 http://security.debian.org/ jessie/updates/main nginx-common all 1.6.2-5+
deb8u4 [88.1 kB]
Get: 2 http://security.debian.org/ jessie/updates/main nginx-full amd64 1.6.2-5+
deb8u4 [430 kB]
Get: 3 http://security.debian.org/ jessie/updates/main nginx all 1.6.2-5+deb8u4
[72.6 kB]
Get: 4 http://ftp.br.debian.org/debian/ jessie/main python-meld3 amd64 1.0.0-1
37.0 kB]
Get: 5 http://ftp.br.debian.org/debian/ jessie/main supervisor all 3.0r1-1 [267
kB]
Fetched 895 kB in 5s (161 kB/s)
Preconfiguring packages ...
Selecting previously unselected package nginx-common.
(Reading database ... 169134 files and directories currently installed.)
Preparing to unpack .../nginx-common_1.6.2-5+deb8u4_all.deb ...
Unpacking nginx-common (1.6.2-5+deb8u4) ...
Selecting previously unselected package nginx-full.
Preparing to unpack .../nginx-full_1.6.2-5+deb8u4_amd64.deb ...
```

Ilustración 117 - Instalación de nginx y supervisor

6.3.1.6 Instalacion de SIADTACC

$ git clone https://github.com/lennin92/peppers-v2.git

$ su

aptitude install python-psycopg2 python-pandas python-numpy

pip install –r requirements.txt

```
root@debian:/home/administrador/peppers-v2# aptitude install python-psycopg2 python-pandas python-numpy
No packages will be installed, upgraded, or removed.
0 packages upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B of archives. After unpacking 0 B will be used.

root@debian:/home/administrador/peppers-v2# pip install -r requirements.txt
Requirement already satisfied (use --upgrade to upgrade): Django in /usr/local/lib/python2.7/dist-packages
irements.txt (line 1))
Requirement already satisfied (use --upgrade to upgrade): djangorestframework in /usr/local/lib/python2.7/
(from -r requirements.txt (line 2))
Requirement already satisfied (use --upgrade to upgrade): Pillow in /usr/lib/python2.7/dist-packages (from
```

Ilustración 118 - Instalación requerimientos de siadtacc

6.3.1.7 Ejecución de SIADTACC.

Para que se pueda ejecutar SIADTACC debe haber una configuración valida antes (ver sección de configuración)

cd (dirección del proyecto de siadtacc)

python manage.py runserver 0.0.0.0:8000

```
administrador@debian:~/peppers-v2$ python manage.py runserver 0.0.0.0:8000
Performing system checks...

WARNING: Logging before InitGoogleLogging() is written to STDERR
W0118 23:18:07.205420 24100 _caffe.cpp:135] DEPRECATION WARNING - deprecated use of Python interface
W0118 23:18:07.205694 24100 _caffe.cpp:136] Use this instead (with the named "weights" parameter):
W0118 23:18:07.206028 24100 _caffe.cpp:138] Net('/home/administrador/peppers-v2/model/peppers.prototxt',
me/administrador/peppers-v2/model/peppers.caffemodel')
System check identified no issues (0 silenced).
January 18, 2017 - 23:18:07
Django version 1.10.5, using settings 'peppers.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

Ilustración 119 - Ejecución de siadtacc I

Abrir el navegador y visitar la dirección del servidor en el puerto 8000

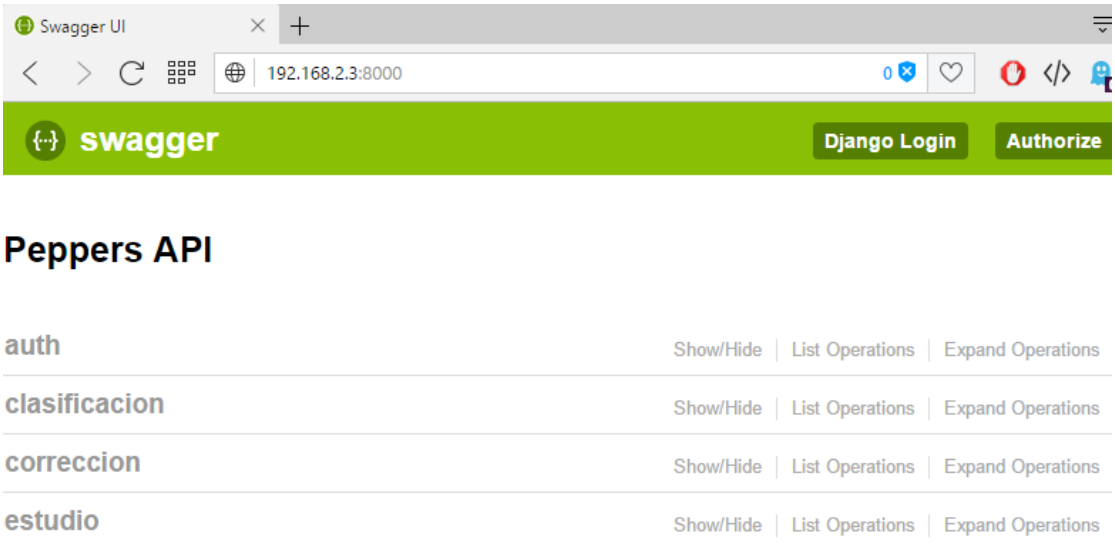


Ilustración 120 - Ejecucion de siadtacc II

Cancelar la ejecución del servidor (Ctrl + C)

6.3.1.8 Configuración general

Para configurar los parámetros de conexión a la base de datos y al servidor dcm4chee se debe editar el archivo settings.py en la carpeta peppers dentro del proyecto siadtacc.

Base de datos

La configuración de la base de datos se realiza en la sección DATABASE, para cambiar su configuración, se debe sustituir los valores.

- NAME: nombre de la base de datos
- USER: usuario de la base de datos
- PASSWORD: contraseña de la base de datos
- HOST: dirección de la base de datos
- PORT: puerto de la base de datos
- ENGINE: el conector de la base de datos (el valor que se encuentra predeterminado no debe cambiar, puesto que es el conector de postgresql)

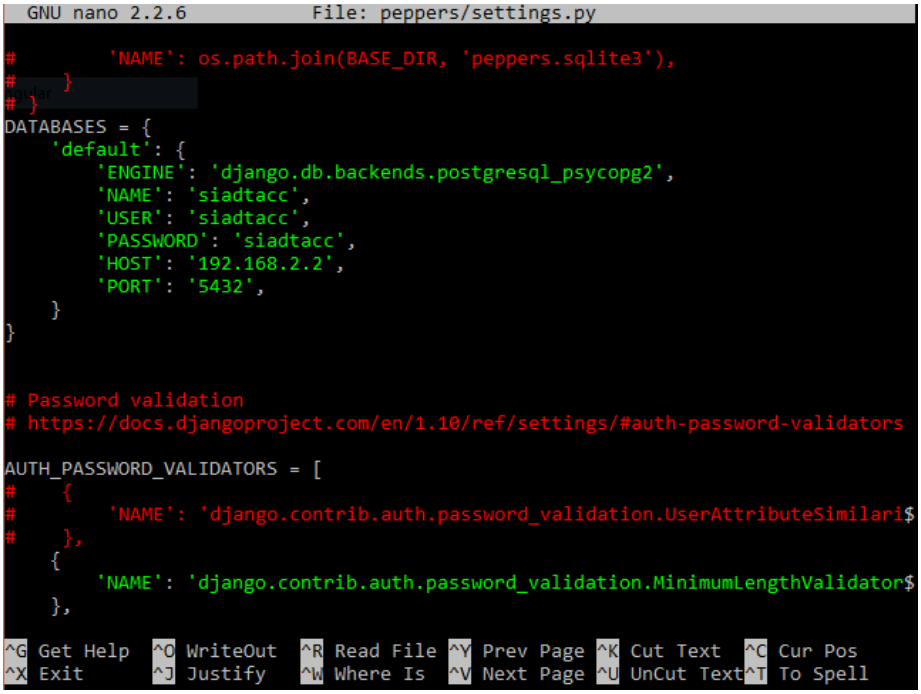


Ilustración 121 - Configuración siadtacc I (Base de datos)

Creacion de super usuario

El rol de super usuario permite acceder a la consola localizada en /admin, esta consola da acceso a la administración de algunas entidades como los usuarios y las clasificaciones. Para crear un super usuario seguir los siguientes pasos:

```
cd /direccion del Proyecto siaddtac

python manage.py createsuperuser # llenar la informacion que se solicita
```

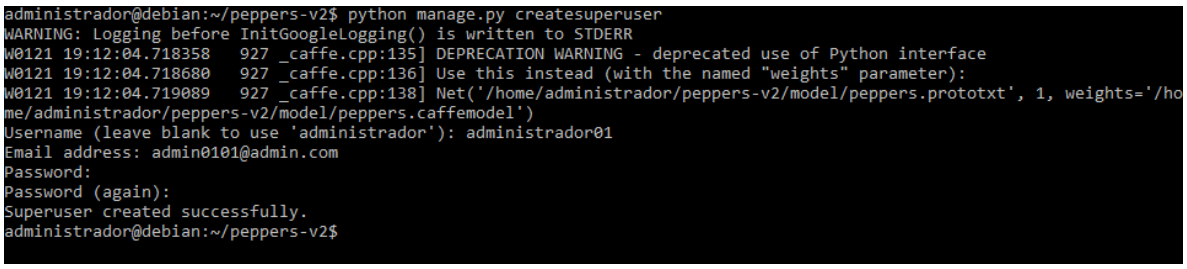


Ilustración 122 - Creacion de superusuario

Conexión con DCM4CHEE

La configuración de la conexión con el dcm4Chee se realiza en las opciones:

- DCM4CHEE_HOSTDIR: Es una cadena de texto que contiene la dirección del servidor de dcm4chee.
- DICOM_PNG_URL_PATTERN: Esta cadena de texto contiene los valores necesarios para obtener una imagen en formato png para un elemento especifico

con objectUID, seriesUID y studyUID (No es necesario cambiar, a menos que se cambie de PACS).

- DICOM_TMP_PATH: Contiene la dirección donde se descargan los archivos a ser analizados por la red neuronal, cabe destacar que se debe hacer una eliminación manual cada cierto periodo de tiempo, es recomendable que este valor tenga acceso a mucho espacio en disco duro)

Configuración de la RNA

Los siguientes valores deben modificarse para poder realizar una conexión valida con la RNA.

- CAFFE_DIR: Dirección donde se encuentra los binarios de python para caffe (normalmente donde se compilo caffe es la carpeta llamada python).
- CAFFE_MODEL: Se debe escribir la dirección completa al archivo prototxt que defina la red neuronal.
- CAFFE_WEIGHTS: Se debe escribir la dirección completa al archivo caffemodel que define los pesos de la red neuronal.
- CAFFE_GPU: (True o False) Si es True implica que se utilizara la tarjeta gráfica nvidia para realizar cálculos (suele ser más rápido); Si es False se utilizara el CPU (si no se tiene tarjeta gráfica Nvidia con instrucciones CUDA se debe tener este valor).

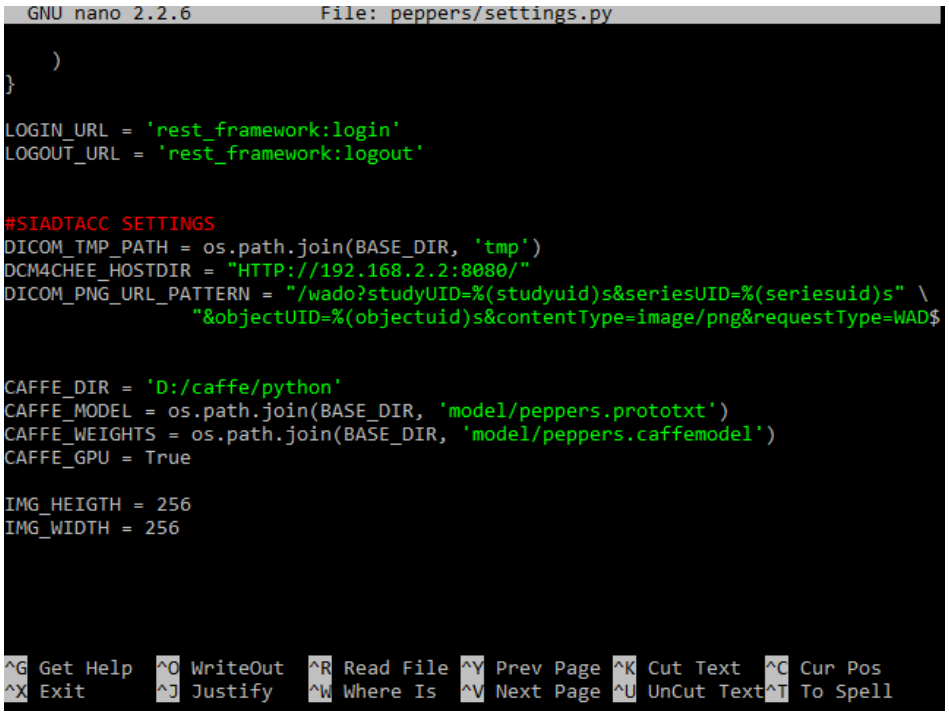


Ilustración 123 - Configuración siadtacc II (RNA)

Configuración de gunicorn

Al editar el archivo “scripts/start_gunicorn.sh” se deben establecer los parámetros siguientes a sus valores correctos:

- LOG_FILE: Establece el archivo donde se escribe el log del servidor.
- LOG_LEVEL: Establece el nivel del log.
- NAME: Nombre del proceso de la aplicación.
- BIND_SOCKET: Puede ser socket unix o una dirección del servidor, establece que puerto estará escuchando y respondiendo dentro del servidor.
- USER: Establece el usuario que se utilizará para iniciar la aplicación.
- WORKERS: Cantidad de trabajadores se recomienda usar la función 2 * cantidad de núcleos + 1, Gunicorn debería necesitar entre 4 y 12 worker para manejar miles de peticiones por segundo.
- WORKER_TYPE: Tipo de worker, se recomienda usar “gevent” por que mantiene las conexiones abiertas por más tiempo, ideal en este caso para evitar que se cierren mientras la red procesa una o varias imágenes.
- DJANGO_WSGI_MODULE: módulo wsgi de django, no se recomienda cambiar el valor de este.
- PROJECT_DIR: dirección absoluta donde está alojado el proyecto. nano scripts/start_gunicorn.sh

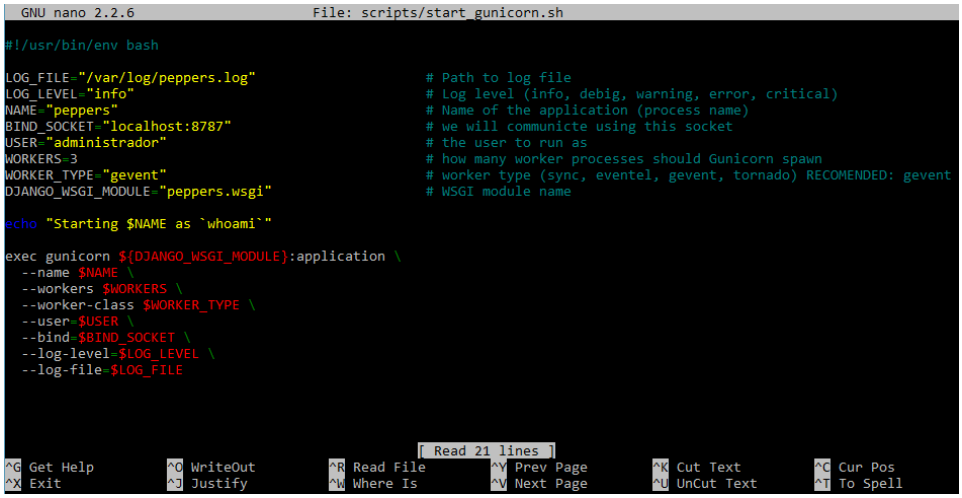


Ilustración 124 - Configuración Gunicorn

```
chmod +x scripts/start_gunicorn.sh
```

Para probar se puede ejecutar el script.

```
scripts/start_gunicorn.sh
```

Configuración de supervisor

Supervisor es un sistema cliente servidor que permita a sus usuarios monitorear y controlar un numero de proceso en sistemas operativos UNIX.

Se debe editar el archivo “scripts/peppers.conf”, este contiene la configuración de supervisor para el proceso de siadtacc.

- command: debe contener la ruta absoluta al archivo start_gunicorn.sh

- user: usuario a correr el script.
- stdout_logfile: archivo que se utilizara de salida de errores estándar.
- redirect_stderr: se establece que la salida estándar de errores se redirige a supervisor
- envoriment: se establecen el juego de caracteres a usar en la salida de mensajes.

```
nano scripts/peppers.conf
```

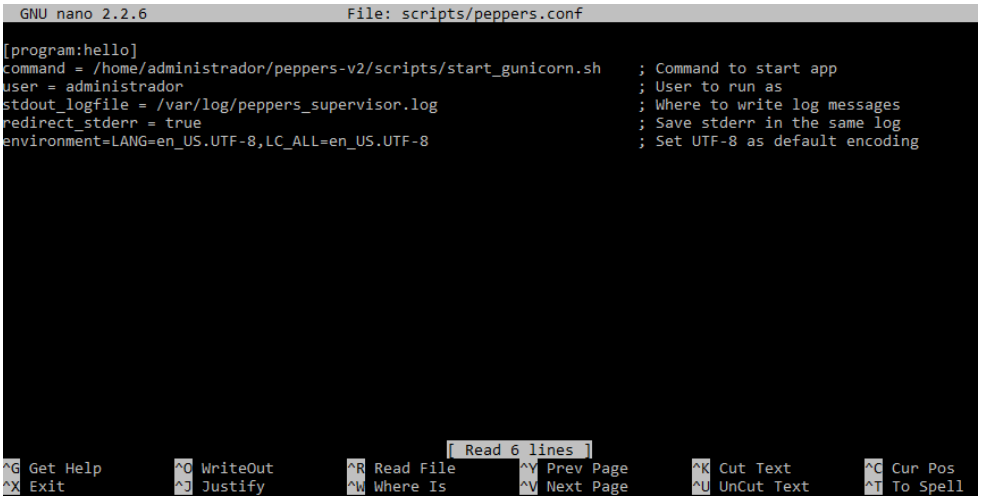


Ilustración 125 - Configuración supervisor I

Copiar el archivo scripts/peppers.conf a /etc/supervisor/conf.d/peppers.conf

```
cp scripts/peppers.conf /etc/supervisor/conf.d/peppers.conf
supervisorctl reread
supervisorctl update
supervisorctl status peppers
```

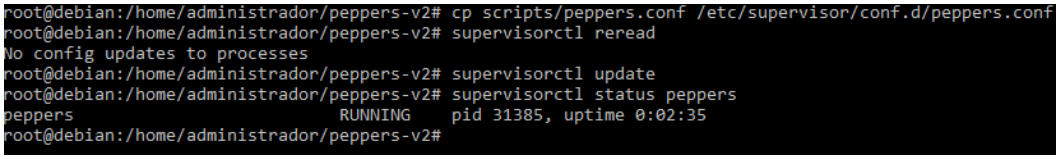


Ilustración 126 - Configuración supervisor II

Configuración de nginx.
Nginx es un servidor Http que sirve de interfaz entre el servidor y clientes externos a este, para configurar la conexión basta con editar el archivo “scripts/peppers.nginx.conf” que contiene la información de conexión entre nginx y gunicorn.

```
nano scripts/peppers.nginx.conf
```

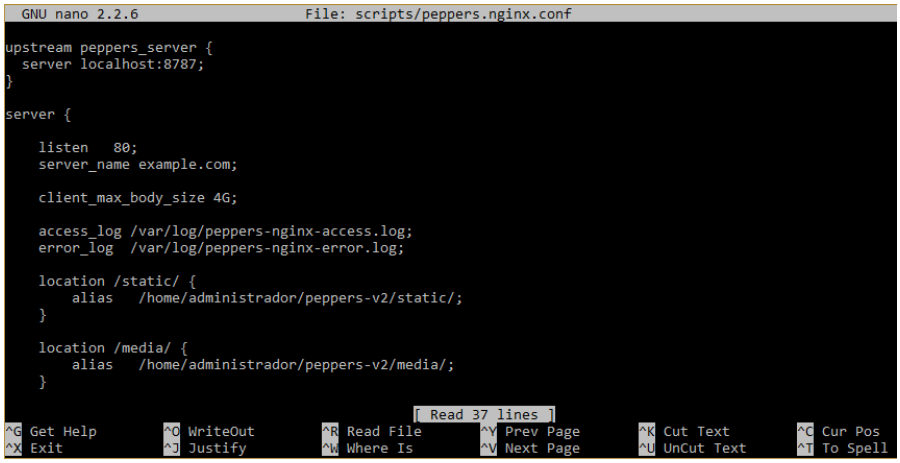


Ilustración 127 - Configuración nginx I

```
cp scripts/peppers.nginx.conf /etc/nginx/sites-available/peppers
ln –s /etc/nginx/sites-available/peppers /etc/nginx/sites-enabled/peppers
python manage.py collectstatic
systemctl restart nginx
```

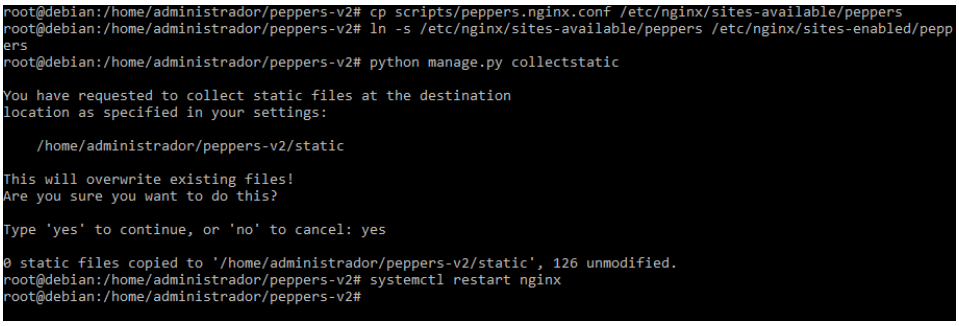


Ilustración 128 - Configuración Nginx II

Al visitar el servidor del siadtacc usando un explorador web en el Puerto 80 deberia aparecer asi:

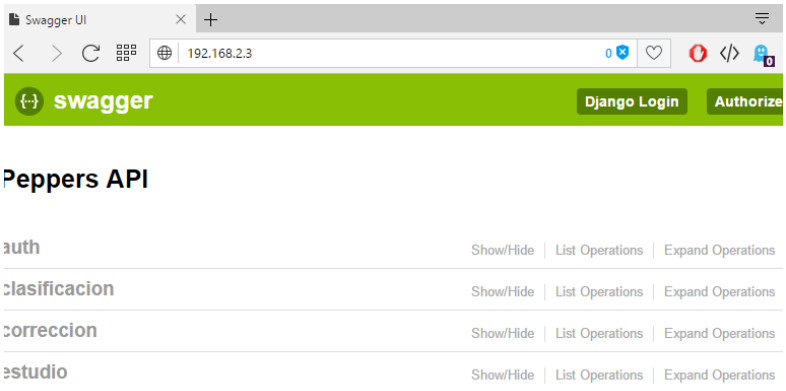


Ilustración 129 - Prueba Nginx

6.3.2 Instalacion y configuración del cliente.

El plugin desarrollado se integra al visor de archivos dicom WEASIS, el cual a su vez se instala en el PACS DCM4CHEE, dicho PACS es una aplicación que corre sobre un servidor JBOSS.

Para instalar un plugin personalizado en el WEASIS es necesario empaquetarlo en un archivo de extensión *war; este* archivo debe poseer cierta estructura y configuraciones específicas para interactuar adecuadamente con el visor, además de llamarse *weasis-ext.war*

6.3.2.1 Configuración del plugin

Es necesario que el plugin este configurado para consumir el web service, esta configuración se realiza en el proyecto del plugin antes de ser compilado, en la carpeta *src > main > resources* se encuentra el archivo *conf.properties* (1); en este archivo se deben agregar las URL que apunten al web service (2). Después de realizadas estas configuraciones se debe seguir el proceso de compilación y empaquetado como se describe en el *Manual Técnico del SIADTACC*.

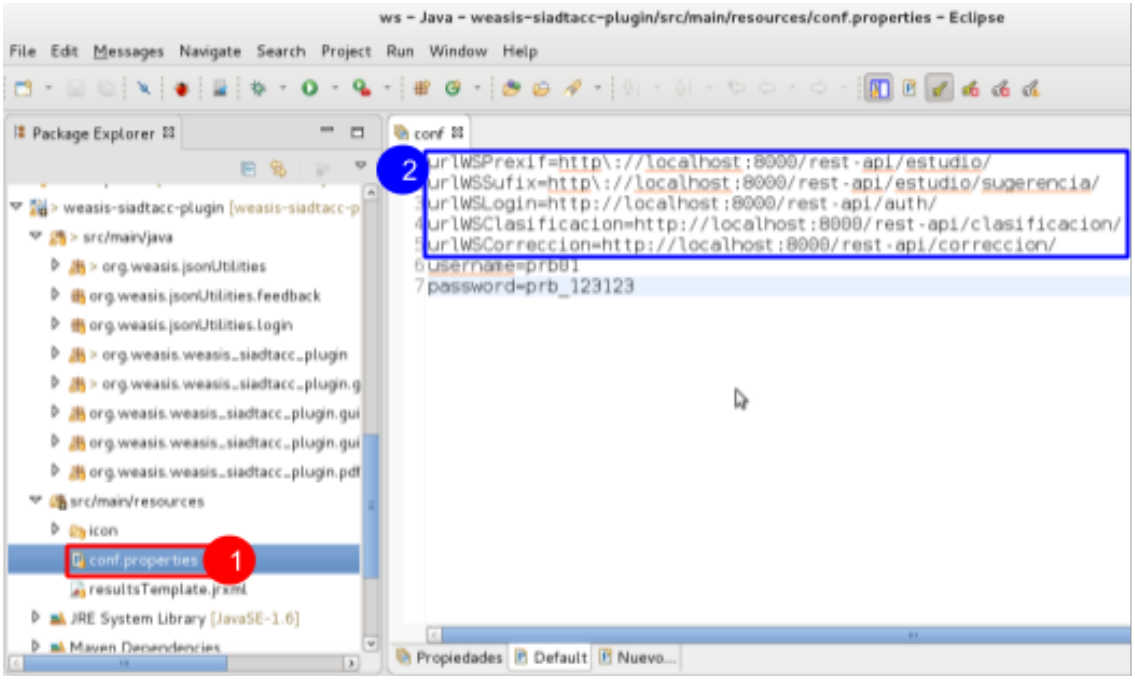


Ilustración 130 - Configuración de la URL del web service

6.3.2.2 Instalación del weasis-ext

La instalación del archivo weasis-ext.war consiste en colocarlo en una carpeta especifica del servidor, en la carpeta */dcm4chee > server > default > deploy*, como se muestra en la siguiente imagen.

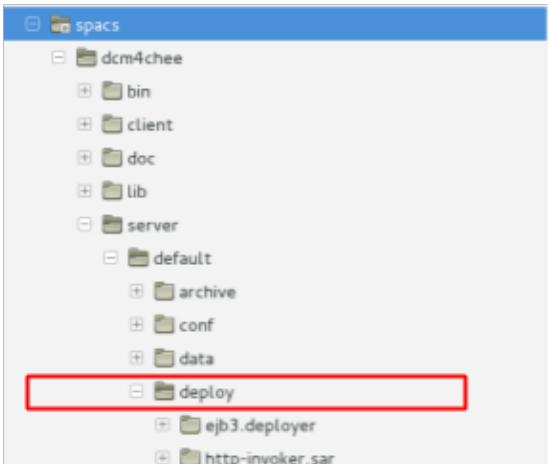


Ilustración 131 - Ubicación de la carpeta deploy

6.3.2.3 Instalación del weasis-pacs-connector

Adicionalmente, cuando se instalan plugins personalizados es necesario instalar el componente *weasis-pacs-connector.war* en su versión 5.0.1 (o superior) para que el visor sea capaz de procesar todas las peticiones del plugin; este componente es distribuido por el equipo de desarrollo de WEASIS y se encuentra disponible en el siguiente enlace: https://sourceforge.net/projects/dcm4che/files/Weasis/weasis-pacs-connector/5.0.1/

El proceso de instalación es igual al del *weasis-ext.war,* basta con colocarlo dentro de la carpeta deploy.

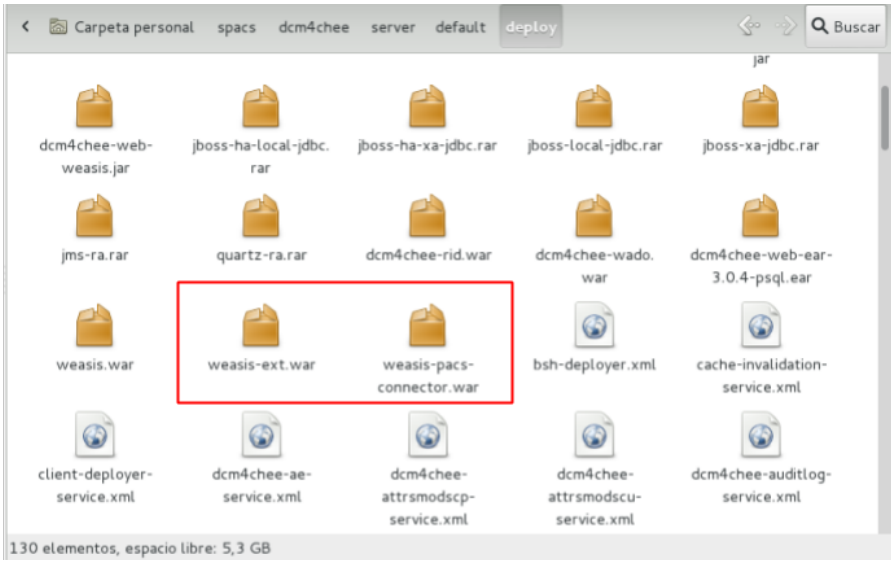


Ilustración 132 - Ubicación de los archivos en el servidor

El proceso detallado sobre la creación del archivo weasis-ext.war se encuentra en el Manual Técnico del SIADTACC.

6.4. Manual de entrenamiento

Como fue solicitado por la contraparte técnica, se provee de un script que automatiza el proceso de entrenamiento, el cual consiste de las siguientes partes:

- Obtencion de imágenes del servidor DCM4CHEE en una carpeta local.
- Codificación de las imágenes en formato LMDB.
- Entrenamiento de la red neuronal.

Se creó un script llamado train.sh dentro de la carpeta scripts del proyecto que contiene los comandos necesarios para iniciar un entrenamiento de la red neuronal.

```
cd /carpeta donde se clono el proyecto/

cd scripts

chmod +x train.sh

ls
```

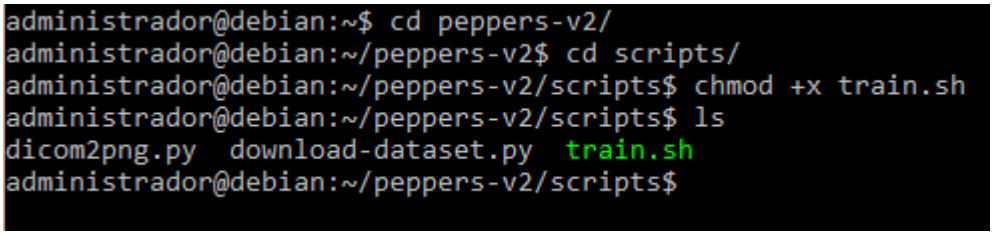


Ilustración 133 - Manual de entrenamiento, visión de scripts

Editar el archivo “train.sh” y reemplazar las variables en mayúsculas (al inicio del script) con los valores solicitados.

```
nano train.sh
```

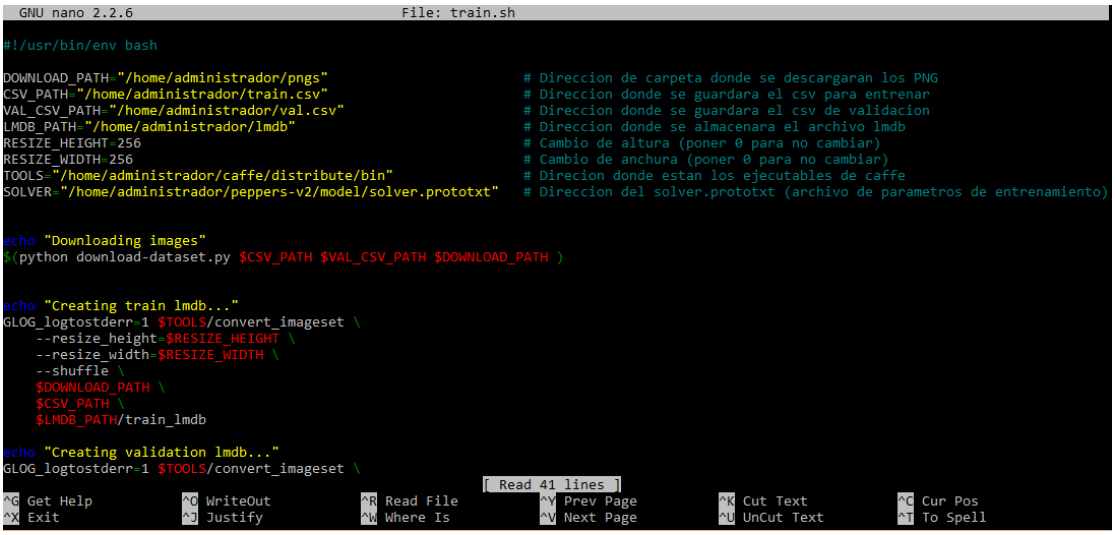


Ilustración 134 - Manual de entrenamiento, edición de script

Editar el archivo “solver.prototxt” estableciendo las variables convenientes (en algunos casos se debe cambiar el valor de solver_mode de GPU a CPU puesto que no todas las computadoras tienen acceso a las instrucciones CUDA propias de las tarjetas Nvidia).

```
nano (dirección de clonado de siadtacc)/model/solver.prototxt
```

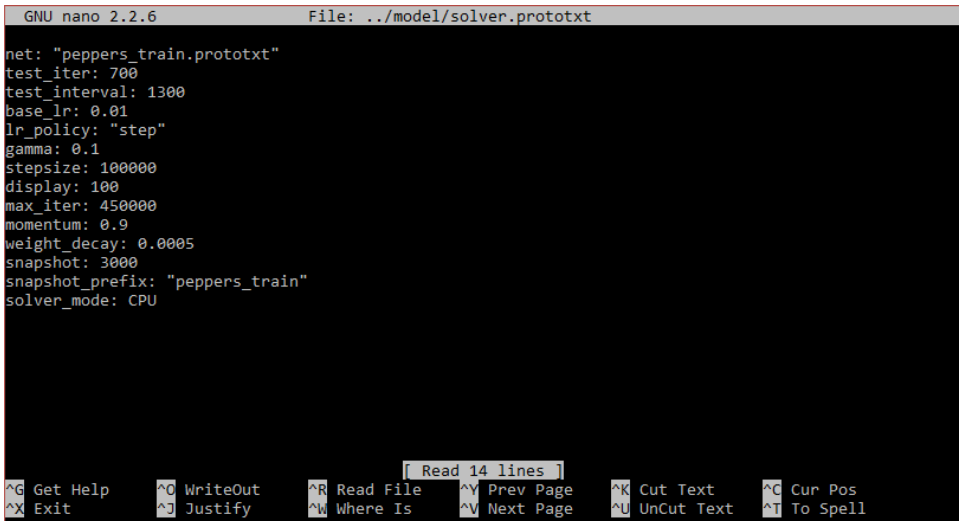


Ilustración 135 - Manual de entrenamiento, edición de solver

Ejecutar el script de entrenamiento

```
./train.sh
```

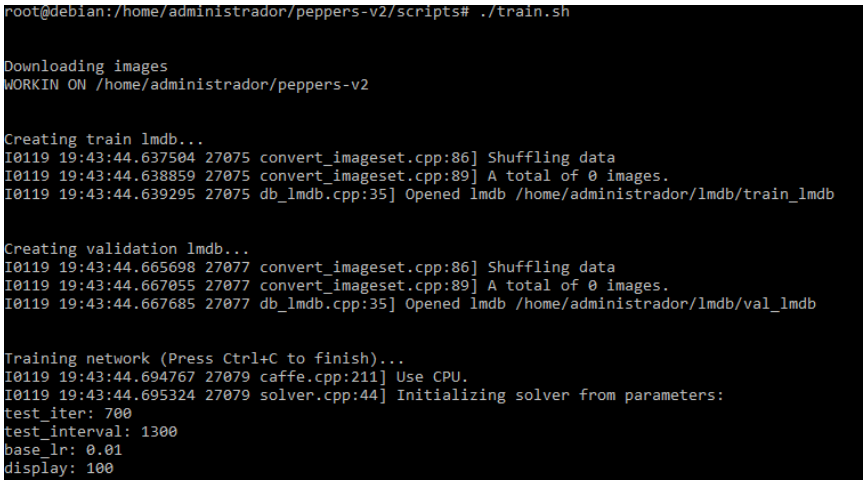


Ilustración 136 - Manual de entrenamiento, entremaniento

Luego de pasar un tiempo valido de entrenamiento pulse la combinación Ctrl+C para finalizar en entrenamiento y copiar los archivos a la carpeta de modelos dentro del proyecto de siadtacc (en el servidor). Revisar que el nombre del archivo debe ser: peppers.caffemodel para que funcione correctamente.

CAPÍTULO VII: PLAN DE IMPLEMENTACION

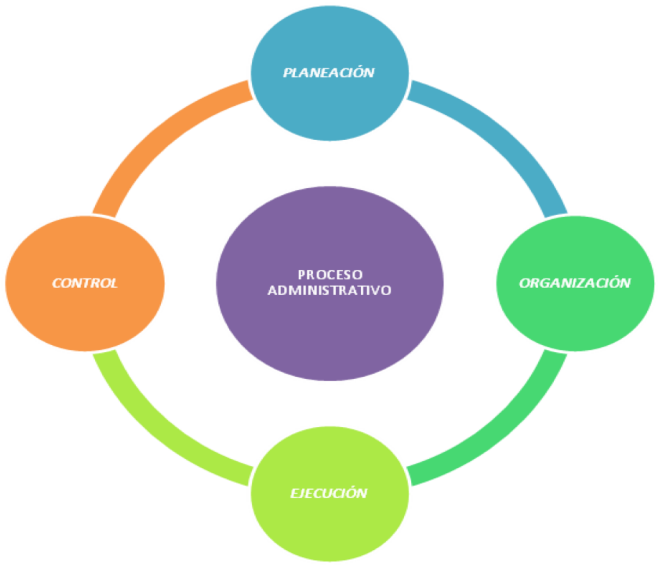
7.1. Plan de implementación

Dentro de la metodología RUP las actividades de implementación se encuentran comprendidas en las fases de construcción y transición, las principales actividades a desarrollar son las siguientes: planificar la integración, implementar cada componente del sistema, integrar cada subsistema e integrar el sistema general, etc. Todo esto con el objetivo de poner en funcionamiento el sistema desarrollado garantizando su interrelación con los demás sistemas existentes.

7.1.1. Etapas de la implementación

La implementación de un nuevo sistema informático dentro de una organización es todo un proceso que incluye diversas etapas que van desde las actividades técnicas como la configuración e instalación de los componentes hasta la capacitación de los usuarios, además dependiendo de la magnitud del sistema el proceso se puede extender durante meses o años, por ello es necesario abordar la implementación como un proyecto en sí mismo el cual tendrá un plazo, recursos y objetivos propios claramente definidos.

Por lo tanto, la implementación estará basada en el proceso administrativo, el cual define una serie de etapas establecidas en el que una etapa se desarrolla inmediatamente después de la anterior, tomando como punto de partida la planeación.



137 - Etapas del proceso administrativo

7.1.2. Planeación

Robbins y De Cenzo (p.6) afirman que planificar abarca la definición de las metas de la organización, el establecimiento de una estrategia general para alcanzar esas metas y el desarrollo de una jerarquía minuciosa de los planes para integrar y coordinar las actividades. Establecer metas sirve para no perder de vista el trabajo que se hará y para que los miembros de la organización fijen su atención en las cosas más importantes.

A continuación, se presenta la descripción de cada uno de los elementos que comprenden la etapa de planeación.

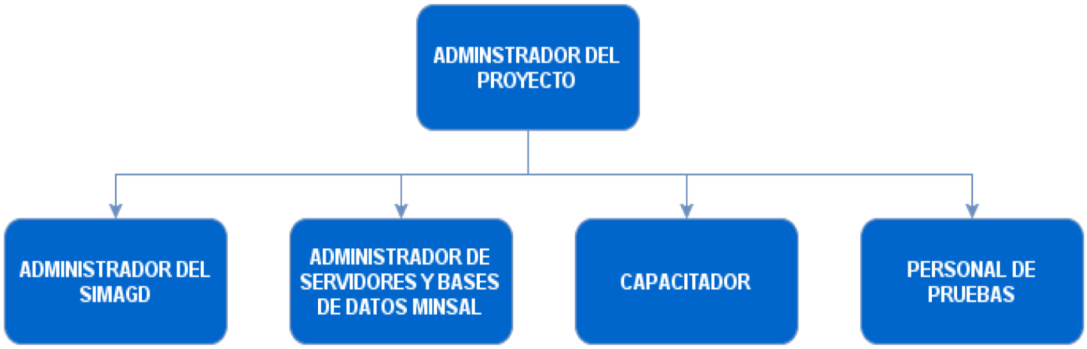
7.1.2.1. Definición de recursos

Los recursos necesarios para la llevar a cabo las actividades de implementación del SIADTACC se dividen de la siguiente manera.

7.1.2.1.1. Recurso de hardware
Las características de hardware necesarias para el correcto funcionamiento de los componentes del SIADTACC fueron descritas en la seccion 6.2.3.1 Requerimientos de hardware del Manual técnico.

7.1.2.1.2. Recurso de software
Las características de software necesarias para el correcto funcionamiento de los componentes del SIADTACC fueron descritas en la seccion 6.2.3.2 Requerimientos de software del Manual técnico.

7.1.2.1.4. Recurso humano
En este apartado se identifica el recurso humano responsable de la implementación del sistema, en las tablas posteriores se describen los perfiles necesarios para cubrir todas las actividades de manera satisfactoria y en la siguiente ilustración se muestra la estructura organizativa.



138 - Administrador de proyecto

Título del puesto:	Administrador del proyecto
Descripción del puesto:	Responsable de supervisar cada una de las actividades del proceso de implementación del sistema, distribuir los recursos disponibles y verificar el cumplimiento de los plazos de las actividades.
Funciones:	• Coordinar y controlar la ejecución de las diversas actividades del plan de implementación. • Supervisar la instalación y configuración de los componentes del SIADTACC. • Supervisar la integración del SIADTACC con los demás sistemas existentes. • Realizar actividades de control que permitan evaluar los avances en la implementación. • Coordinar las capacitaciones de los usuarios finales. • Supervisar la realización de las respectivas pruebas y evaluaciones al sistema.
Requerimientos:	• Experiencia en la implementación de proyectos informáticos. • Conocimiento sobre desarrollo y operación de aplicaciones web. • Capacidades de liderazgo y coordinación de personal.

Tabla 94 - Descripción del puesto Administrador del proyecto

Título del puesto:	Administrador del SIMAGD
Descripción del puesto:	Persona designada por el MINSAL y el Hospital en el que se haga la instalación para administrar el proceso de implementación y funcionamiento del SIMAGD.
Funciones:	Este perfil ya esta definido por el MINSAL, pero las funciones a desarrollar para este proyecto son las siguientes: • Realizar las configuraciones necesarias para la instalación de los componentes del SIADTACC (plugin de Weasis, Web Service y Red Neuronal Artificial), esto incluye configuraciones en el servidor de almacenamiento (NAS), infraestructura de red, en el servidor PACS y en el SIMAGD. • Supervisar la instalación y configuración de los componentes del SIADTACC que interactúan con el SIMAGD. • Realizar actividades de control que permitan evaluar los avances en la implementación. • Coordinar junto con el administrador del proyecto las capacitaciones de los usuarios finales.
Requisitos:	Este perfil ya esta definido por el MINSAL, pero las competencias necesarias para este proyecto son las siguientes: • Experiencia en la implementación de proyectos informáticos. • Conocimiento sobre la configuración del SIMAGD. • Capacidades de liderazgo y coordinación de personal. • Conocimientos sobre la infraestructura de red del Hospital San Rafael.

Tabla 95 - Descripción del puesto Administrador del SIMAGD

Título del puesto:	Administrador de servidores y bases de datos del MINSAL
Descripción del puesto:	Responsable de administrar los servidores del SIADTACC, bases de datos y red neuronal así como otras aplicaciones con las que el SIADTACC interactúa (SIAP y SIMAGD).
Funciones:	Las funciones a desarrollar para este proyecto son las siguientes: • Realizar las configuraciones necesarias para la instalación de los componentes de la RNA en los servidores del MINSAL. • Crear y configurar la base de datos que utilizará el SIADTACC. • Instalar y configurar el componente de la RNA. • Realizar actividades de control que permitan evaluar los avances en la implementación.
Requisitos:	Este perfil ya esta definido por el MINSAL, pero las competencias necesarias para este proyecto son las siguientes: • Experiencia en la implementación de proyectos informáticos. • Conocimientos sobre administración de bases de datos. • Conocimientos sobre administración de servidores. • Capacidades de liderazgo y coordinación de personal. • Conocimientos sobre la arquitectura del SIMAGD.

Tabla 96 - Descripción del Administrador de servidores y bases de datos del MINSAL

Título del puesto:	Capacitador
Descripción del puesto:	Responsable de brindar la capacitación a los usuarios que harán uso del sistema. Ademas de coordinar junto con el administrador del proyecto el alcance, metodología y plazos de la capacitación.
Funciones:	• Coordinar junto al administrador del proyecto las capacitaciones de los usuarios finales. • Impartir las capacitaciones definidas en el plan de implementación a los usuarios finales del sistema. • Realizar las pruebas y validaciones necesarias al sistema.
Requisitos:	• Conocimiento sobre el sistema SIMAGD y SIADTACC. • Capacidad para trabajar en equipo. • Facilidad de expresión. • Capacidad de enseñanza.

Tabla 97 - Descripción del puesto Capacitador

Título del puesto:	Personal de pruebas
Descripción del puesto:	Usuarios finales del sistema informático, dada la naturaleza del proyecto dichos perfiles corresponden con médicos radiologos y técnicos en radiología. Cabe aclarar que las siguientes funciones se refieren al rol a desempeñar dentro del proceso de implementación y no así a las demás funciones inherentes a su cargo.
Funciones:	• Participar en las capacitaciones impartidas por el capacitador. • Participar en las actividades de pruebas realizadas al sistema informático. • Informar los resultados de las pruebas al sistema así como cualquier incidencia que se pueda presentar.
Requisitos:	• Pertenecer al equipo de radiología del Hospital San Rafael. • Conocimiento sobre el uso del SIMAGD.

Tabla 98 - Descripción del puesto Personal del pruebas

7.1.2.2. Instalación del sistema

Esta etapa del proceso de implementación contempla las actividades relacionadas a la instalación de todos los elementos del SIADTACC; el sistema está compuesto de dos grandes componentes, a continuación, se describen los pasos a seguir para la instalación y configuración de cada uno de ellos:

7.1.2.2.1. Instalación y configuración de la rna

Para más detalles sobre este proceso, consultar el Manual Técnico (Sección 6.2) y el Manual de Instalación (Sección 6.3) del SIADTACC.

7.1.2.2.2. Instalación y configuración del plugin del weasis

Para más detalles sobre este proceso, consultar el Manual Técnico (Sección 6.2) y el Manual de Instalación (Sección 6.3) del SIADTACC.

Para la instalación de este componente es necesario que el plugin este empaquetado en un archivo con extensión .war, también se necesita un archivo adicional que contiene las configuraciones que el PACS requiere para resolver las peticiones; ambos archivos deben ser publicados en una ubicación especifica dentro del PACS, el proceso general contiene los siguientes pasos:

- Generar los archivos a instalar.
- Instalación del weasis-ext
- Instalación del weasis-pacs-connector

7.1.2.3. Preparación de estaciones de trabajo (clientes)

Los equipos utilizados en las estaciones de trabajo corresponden a las computadoras asignadas a los técnicos y médicos radiologos, para el correcto funcionamiento del SIADTACC es necesario que las computadoras cuenten con:

- Navegador web.
- Acceso a la red interna de la institución.
- Acceso a internet.
- JRE (Java Runtime Environment).

Las características específicas de hardware (Sección 6.2.3.1) y software (Sección 6.2.3.2) se encuentran detalladas en el *Manual Técnico del SIADTACC.*

7.1.2.4. Capacitación de los usuarios

Se puede definir a la capacitación como un conjunto de actividades didácticas orientadas a suplir las necesidades de la institución y que se orientan hacia una ampliación de los conocimientos, habilidades y aptitudes de los empleados, con lo cual adquieren las competencias necesarias para desarrollar sus actividades de manera eficiente, las capacitaciones pueden ser de manera individual o de manera grupal.

Los detalles respecto a la metodología, propósito y alcance de las capacitaciones se encuentran definido en el apartado de plan de capacitación.

7.1.2.5. Puesta en marcha del sistema

Esta etapa es la última del proceso de implementación, por ende, debe realizarse cuando hayan finalizado de manera satisfactoria las actividades de instalación y configuración del sistema, plugin e infraestructura de red, así como el proceso de capacitación del personal.

Consiste en poner en funcionamiento el sistema para el ambiente de producción, para ello se deben eliminar los datos de pruebas utilizados en las capacitaciones e implementar las medidas de control con el fin de mantener en óptimas condiciones de funcionamiento todos los componentes del sistema.

7.1.2.6 monitoreo del sistema

Luego de haber instalado el sistema y éste se encuentre en funcionamiento, se recomienda esté en constante monitoreo durante un mes, con ello se pueden detectar cuellos de botella en la configuración, puesto que se requiere una alta disponibilidad de red por la alta demanda de imágenes, así como altas velocidades de escritura y lectura en el disco duro del servidor puesto que estas imágenes necesitan ser almacenadas antes de ser procesadas.

Se debe tomar en cuenta que se tiene que hacer una constante medición de la velocidad de respuesta del sistema, por ello se recomienda que se deben revisar tanto la percepción del usuario sobre la rapidez de las respuestas del sistema a sus peticiones, como otros elementos más técnicos referentes a la velocidad de red, de base de datos, etc. Para hacer las configuraciones necesarias para mejorar la experiencia de usuario.

7.1.3. Organización

7.1.3.1. Matriz de responsabilidades

Notación:

- **P:** Planificación
- **O:** Organización
- **D:** Dirigir
- **C:** Controlar

Fases	Encargados				
	Coordinador de Proyecto	Administrador del SIMAGD	Administrador de servidores y bases de datos	Capacitador	Personal de pruebas
Definición de recursos	PODC	POC	POC		
Instalacion y configuración de sistema	PODC	POC	POC		
Carga de datos	PODC	POC	POC		

Capacitación de usuarios	PODC	C	C	PODC	OD
Puesta en marcha	PODC	OC	OC		

Tabla 99 - Matriz de responsabilidades de las actividades del plan de implementación

7.1.3.2. Diagrama de gantt

La adquisición del hardware necesario no está contemplado, puesto que el MINSAL ya cuenta con hardware que puede utilizarse para la implementación, cabe mencionar que el rendimiento de la red neuronal aumentaría con el uso de una tarjeta gráfica nvidia con CUDA (familia Titán o GTX). Es importante mencionar que no se consideran los tiempos necesarios para hacer un entrenamiento de la red neuronal desde cero debido a que ya se provee de una red neuronal inicializada con la muestra de entrenamiento que fue proveída por el MINSAL.

ID	Nombre de tarea	Duración	Predecesoras
1	Plan de implementación de SIADTACC	22 días	
2	Preparación de entorno	2 días	
3	Validacion de requerimientos de hardware	1 día	
4	Validacion de requerimientos de software	2 días	
5	Organización de personal	5 días	2
6	Gestión de Recurso Humano	3 días	
7	Asignación de responsabilidades	2 días	6
8	Ejecución de implementación	6 días	5
9	Instalación de hardware necesario	4 días	
10	Instalación de software	2 días	
11	Carga inicial de datos	2 días	9;10
12	Capacitaciones	7 días	8
13	Planificación de capacitaciones	2 días	
14	Ejecución de capacitaciones	5 días	13
15	Puesta en marcha	2 días	12

Tabla 100 - Diagrama de GANTT del plan de implementación

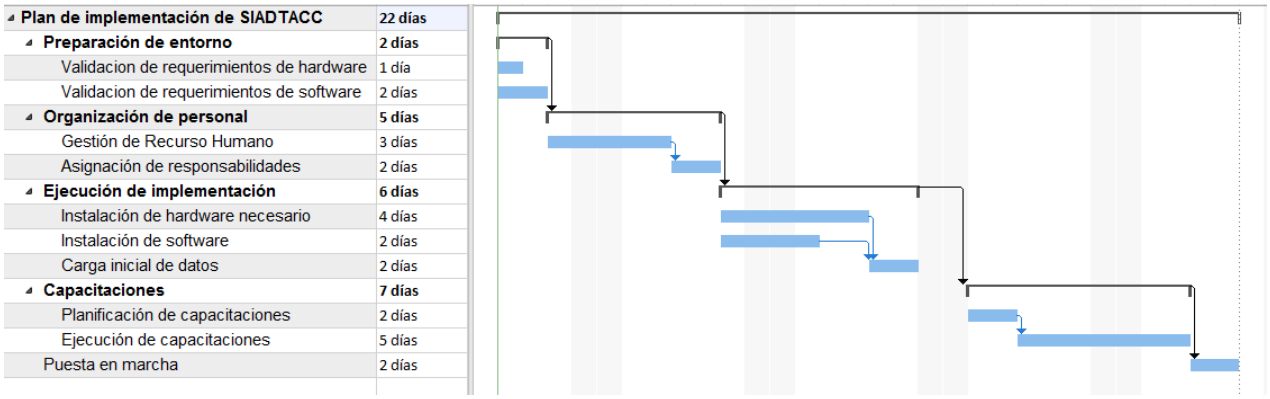


Ilustración 139 - Diagrama Gantt

7.1.4. Dirección

En este apartado se proponen las acciones que deberá realizar el director del proyecto durante la implementación del mismo.

7.1.4.1 Principios de la ejecución del plan de implementación

- **Jerarquía:** Enlaza entidades verticalmente, establece canales de comunicación entre estas.
- **Interés:** Coordinar los intereses del proyecto con los del grupo y los intereses individuales de quienes participan enfocándolos en finalizar correctamente el plan de implementación.
- **Disipar conflictos de forma oportuna:** Los conflictos se deben de resolver de la forma más rápida posible, de modo que estos no entorpezcan el desarrollo normal de las actividades.

7.1.4.2 Costos de implementación

- **Costos de recursos tecnológicos**: la DTIC del MINSAL cuenta con todo el equipo necesario para la implementación, por lo que no será necesario la adquisición de otros elementos de este tipo.
- **Costos de RRHH:** La DTIC del MINSAL ya cuenta con el personal encargado para la administración de servidores, red y el equipo de radiólogos necesario para la implementación.
- **Costos de capacitación:** El total de costos estimado de la capacitación en la sección de presupuesto de capacitación es de $160.00 como se define en la sección 7.1.6.6 Presupuesto de capacitación del Plan de Capacitación.

7.1.5. Control

En este apartado se buscan establecer mecanismos y estrategias para la medición del avance en la implementación del SIADTACC.

7.1.5.1. Puntos de control y monitoreo

Con la finalidad de evitar desvíos innecesarios y de monitorear el avance de cada actividad detallada en el plan de implementación, se establecen los siguientes puntos de control, en los cuales se deberán realizar las estrategias de monitoreo y control detalladas en el próximo apartado.

- Al finalizar la fase de preparación de entorno.
- Al finalizar la fase de organización de personal.
- Al finalizar la instalación de hardware necesario.
- Al finalizar la instalación de software y carga inicial de datos.

- Al iniciar y finalizar la fase de capacitaciones.
- Al finalizar la puesta en marcha.

Las actividades mencionadas anteriormente se detallan en la sección de organización de la implementación.

7.1.5.2. Estrategias de monitoreo y control

De acuerdo a Biafore, B (2006), para mejorar la medición de los avances se proponen 2 métodos:

- Indicadores de gestión cuantitativos.
- Encuesta cualitativa de avances.

7.1.5.2.1 Indicadores de gestión cuantitativos.

Realizan una comparación de lo realizado con lo planificado, para este proyecto se sugieren las siguientes razones:

- **Indicador de duración de actividades:** (IDA) Grado de desviación entre el tiempo real de una actividad y el tiempo programado para la misma. Si se obtiene un valor <=1 es aceptable puesto que indica que la duración real de la actividad es acorde a lo programado. Si se obtiene un valor >1, indica que el tiempo que dura una actividad es mayor al programado y se deben tomar medidas correctivas como la planificación de las actividades con tiempos más holgados, reducir el tiempo de ejecución de las próximas tareas o asignar más recursos a las actividades.
  IDA = Duración real de actividad / Tiempo programado para la actividad
- **Indicador de actividades planeadas ejecutadas:** (IAPE) Muestra el grado de avance del proyecto, y se espera que el valor sea el mayor posible entre 0 y 1.
  IAPE = Total de duración de actividades ejecutadas / Total de actividades planificadas

7.1.5.2.2 Encuesta cualitativa de avances.

Se propone que al finalizar cualquier fase de la implementación el administrador del proyecto responda sinceramente las siguientes preguntas:

- ¿Se finalizó correctamente la fase?
- ¿Debió hacerse algo de forma diferente?
- ¿El tiempo planeado fue suficiente para finalizar?
- ¿Se puede considerar el producto final como “de alta calidad”?

7.1.6. Plan de capacitación

7.1.6.1. Propósito

Establecer los lineamientos y consideraciones generales para formular los elementos del plan de capacitación que permitan la formación de los usuarios finales en el correcto uso del SIADTACC, así como el personal técnico del MINSAL y Hospital Nacional San Rafael en cuanto a las configuraciones del SIADTACC y el proceso de entrenamiento de la RNA descrito en la sección 6.4 Manual de entrenamiento.

7.1.6.2. Alcance

El presente plan de capacitación es de aplicación para el personal de la Unidad de Radiología, el administrador del SIMAGD del Hospital Nacional San Rafael y al administrador de servidores y bases de datos del MINSAL. La capacitación está divida en dos partes:

1. **Capacitación técnica:** Enfocada hacia a los usuarios técnicos, tiene por objetivo preparar al personal sobre la instalación y configuración de los diferentes componentes del SIADTACC haciendo un énfasis importante en el proceso de entrenamiento de la RNA.
2. **Capacitación de uso:** Tiene por objetivo instruir al personal de radiología en el uso de las nuevas funciones disponibles en el visor WEASIS mediante el plugin del SIADTACC.

7.1.6.3. Planificación de la capacitación

El proceso de capacitación será desarrollado siguiendo las siguientes etapas:

1. **Definir el temario y las guías de capacitación:** Los manuales de usuario, técnico e instalación son la fuente principal de información para desarrollar las sesiones de capacitación, pero es necesario distribuir el contenido según la duración de cada sesión y el tipo de usuarios que la reciba. Para reforzar el proceso de aprendizaje durante las sesiones se realizarán ejercicios prácticos.
2. **Preparación de un escenario real para el desarrollo de la capacitación:** Se debe preparar un entorno adecuado para realizar las sesiones de capacitación, esto implica utilizar las maquinas reales donde utilizarán el sistema de manera habitual, un espacio físico con las condiciones mínimas de comodidad y contar con los recursos necesarios como conexión a Internet, cañón proyector, pizarra y papelería.
3. **Carga de datos:** Previo al proceso de capacitación se debe preparar la base de datos con datos de prueba del SIADTACC esto incluye los listados con las clasificaciones y las configuraciones de la RNA, adicionalmente es necesario contar con un banco de estudios de pruebas provistos por el SIMAGD.

7.1.6.4. Metodología de capacitación

Para desarrollar las capacitaciones de manera óptima y alcanzar los resultados deseados se deben tomar en cuenta las siguientes consideraciones:

- **Mobiliario y equipo:** Se debe contar con una sala con capacidad para albergar a por lo menos 10 personas, ademas debe poseer cañón proyector, pizarra y aire acondicionado o algún sistema de ventilación que garantice la comodidad de los presentes. Ademas el lugar debe estar equipado con una computadora con conexión internet para cada uno de los asistentes.
- **Duración:** La capacitación será desarrollada en 2 jordanas de 4 horas cada una, una jornada para el personal técnico y otra para los usuarios finales, a su vez cada jornada estará dividida en 2 sesiones y 1 receso, la primera sesión de 2 horas y la

segunda de 1 hora 45 minutos, dejando 15 minutos para el receso entre ambas sesiones. Estas sesiones de capacitación estarán distribuidas a lo largo de una semana, debido a que no se pueden realizar de manera continua para no afectar la atención a los pacientes de los radiólogos.

- Metodología de la sesión:

Ponencias magistrales

El expositor desarrolla el tema sobre el que trata la sesión de capacitación, aportando los aspectos teóricos necesarios y realizando las demostraciones pertinentes, los asistentes pueden realizar preguntas aclaratorias en cualquier momento durante la ponencia.

Prácticas interactivas

Se exponen y presentan casos prácticos sencillos y graduales para ser desarrollados por los asistentes a fin de afianzar los conceptos explicados, ademas los asistentes pueden realizar las consultas que consideren convenientes con el fin de esclarecer cualquier duda.

7.1.6.5. Carta didáctica

N.º	Temática	Contenido	Personal	Recurso	Duración
1	Instalación y configuración del SIADTACC.	- Configuración de la base de datos. - Proceso de compilación y empaquetado del plugin. - Instalación del plugin del WEASIS. - Configuraciones al PACS para integración con el plugin.	Capacitador, Administrador del SIMAGD, Administrador de servidores y bases de datos del MISAL.	Proyector, pizarra, computadora por cada usuario con acceso a internet, manual técnico, manual de usuario y manual de instalación.	2 horas
2	Instalación y configuración del componente de RNA del SIADTACC.	- Instalación y configuración del web service. - Instalación y configuración de la RNA. - Proceso de entrenamiento de la RNA.	Capacitador, Administrador del SIMAGD, Administrador de servidores y bases de datos del MISAL.		
3	Acceso al plugin y opción de solicitud de análisis.	- Proceso para accesar al plugin. - Presentación de las funciones del plugin.	Capacitador, Administrador del SIMAGD, Personal de pruebas		

		- Proceso de solicitud de análisis: Por corte individual y por serie. - Generación de reporte de análisis por serie.	(usuarios finales).		
4	Opción del plugin para realización de sugerencias (feedback).	- Proceso para accesar a las funciones de envió de sugerencias. - Envío de sugerencias: mediante el panel lateral y como feedback sobre los resultados del análisis.	Capacitador, Administrador del SIMAGD, Personal de pruebas (usuarios finales).		

Tabla 101 - Carta didáctica del plan de implementación

7.1.6.6. Presupuesto de capacitación

Para determinar los costos aproximados del proceso de capacitación se han considerado los siguientes rubros:

- **Recuso humano:** Dado que el personal a capacitar trabaja para el MINSAL o el Hospital Nacional San Rafael, asistir a las capacitaciones forma parte de sus funciones y no genera costos adicionales, por lo tanto, solo se tomará en cuenta el costo generado por el capacitador, considerando un costo de $10.00 por hora clase impartida y que las capacitaciones tendrán una duración de 8 horas en total, distribuidas a lo largo de una semana, los costos de recursos humanos son los siguientes:

  $10.00/h * 8 h = **$80.00**

- **Material de apoyo:** Se ha determinado que el material a utilizar estará disponible en formato digital aprovechando que cada asistente contará con una computadora, de esta manera se reducen los costos al prescindir del material impreso.

- **Otros costos:** En este rubro se han considerado costos estimados por papelería (hojas de asistencia, plumones, etc), alimentación y cualquier otro costo imprevisto, estimando que cada hora genere costos por $10.00 y dado que la capacitación dura 8 horas los costos adicionales son los siguientes:

  $10.00/h * 8 h = **$80.00**

El total de costos estimado de la capacitación es: $80.00 + $80.00 = **$160.00.** Cabe destacar que es el único costo para la implementación es el de capacitaciónes puesto que los usuarios ya tienen un salario fijo y el recurso de hardware y software no genera costos nuevos, por que ya cuentan con dichos recursos.

8. CONCLUSIONES

- La situación actual del Ministerio de Salud de El Salvador, referente al área de sistemas informáticos, es apropiada para la implementación de un sistema informático de asistencia al diagnóstico radiológico, considerando los esfuerzos que la institución ha realizado en conjunto con estudiantes de la Universidad de El Salvador para implementar el SIMAGD el cual sirvió como punto de inicio del sistema propuesto.

- El sistema desarrollado, ayudará a focalizar el uso de personal de radiología en casos de emergencia puesto que clasificará los TACS en anormales y normales. Además reduce el margen de error de diagnóstico, disminuyendo posibles causas del mismo por ejemplo exceso de trabajo del especialista al reducir la cantidad de estudios a analizar.

- La investigación teórica de las RNAs sirvió para descartar modelos, que no eran adecuados para el problema de clasificación.

- Para el entrenamiento de las RNAs es necesario tener hardware de alto procesamiento, para poder tener batch más grandes y arquitecturas de red más complejas.

- La cantidad y características de las imágenes influyen en el porcentaje de precisión dado en las pruebas de la RNA.

- La documentación presentada del sistema servirá para la implementación y puesta en marcha en la infraestructura del MINSAL, así como la expansión del sistema.

9. RECOMENDACIONES

- Para garantizar la fiabilidad y funcionalidad del sistema se recomienda realizar el entrenamiento de la RNA con un banco de imágenes de TACs cerebrales diagnosticadas.

- Futuros entrenamientos de la red neuronal necesitaran de una muestra de mayor tamaño (de mínimo 45,000 imágenes), cuyo tiempo de entrenamiento depedera directamente de las características del hardware que posea el equipo en el que se realizará .

- Se recomienda a la Escuela de Ingeniería de Sistemas Informáticos, estandarizar los entregables de los proyectos que dependan de una investigación científica para el desarrollo de los mismos.

- Es necesario implementar un plan de comunicación efectivo entre las entidades a las que se les desarrolla los sistemas informáticos y el equipo de trabajo.

- Crear un directorio de experiencia, donde se coordinen proyectos previamente implementados con fuentes abiertas en las Administraciones Públicas para brindar información relevante y experiencias exitosas.

10. GLOSARIO DE TÉRMINOS

Término básico	Definición
API	Interfaz de Programación de Aplicaciones (del Ingles Application Programming Interface), es el conjunto de subrutinas, funciones y procedimientos o métodos, que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.
BLAS	Basic Linear Algebra Subprograms. Conjunto de núcleos (rutinas) computacionales escritos en Fortran 77 para realizar operaciones básicas de álgebra lineal.
CAD	Sistema de diagnóstico asistido por computadora (Computer Aided Diagnosis). Un sistema de CAD analiza la imagen médica y trata de detectar zonas sospechosas para llamar la atención del profesional médico para tomar decisiones sobre al tratamiento ante estos hallazgos.
CAFFE	Plataforma para la creación, formación, evaluación y despliegue de redes neuronales profundas.
Consulta médica	La consulta médica es el acto, encuentro de dos personas, una de ellas enferma en alguna de sus dimensiones, y el otro el profesional de la salud que en este caso es un médico radiólogo y que se encarga de brindar una lectura y diagnóstico de los estudios TAC.
CUDA	Arquitectura Unificada de Dispositivos de Cómputo (Compute Unified Device Architecture). Desarrollada por NVIDIA y lanzada en 2006, es una arquitectura de cálculo paralelo de NVIDIA que aprovecha la gran potencia de la GPU (unidad de procesamiento gráfico) para proporcionar un incremento extraordinario del rendimiento del sistema.
DCM4CHEE	Implementación de código abierto del estándar DICOM de alto rendimiento. Está desarrollado en el lenguaje de programación Java. Es un gestor de imagen médica (PACS) conforme al estándar IHE. Su principal responsabilidad es establecer un servicio de almacenamiento y recuperación de pruebas médicas almacenadas en formato DICOM.
Deep Learning	Es un conjunto de algoritmos en aprendizaje automático (machine learning) que intenta modelar abstracciones de alto nivel en datos usando arquitecturas compuestas de transformaciones no-lineales múltiples.
Diagnóstico	Es el resultado del análisis de los síntomas o pruebas realizadas a un paciente con la finalidad de determinar terapias o tratamientos adecuados para su recuperación.
DICOM	Estándar en Imagen Digital y Comunicaciones en Medicina (Digital Imaging and Comunications in Medicine). Estándar global de tecnología e información diseñado para asegurar la interoperabilidad de los sistemas utilizados para producir, almacenar, visualizar, enviar, recuperar, consulta o imprimir imágenes médicas.
DTIC	Dirección de Tecnologías de Información y Comunicaciones, DTIC, tiene a su cargo la administración de los recursos informáticos del MINSAL, facilitando el ordenamiento de los sistemas de información y el soporte técnico del equipamiento de cómputo, así como los enlaces y servicios de telecomunicaciones. Dada dicha responsabilidad, la DTIC debe contar con una estructura y funciones acordes a tal encomienda.
Fuerza sináptica	En una red neuronal, es el valor que determina la fuerza de conexión entre 2 neuronas.

Ginkgo CADx	Visor DICOM que puede conectarse a un PAC. Ha sido diseñado para satisfacer las expectativas de varios sistemas de información clínica y su evolución futura con respecto a las imágenes médicas.
HIS	Sistema de Información Hospitalaria (de sus siglas en inglés Hospital Information System), es un elemento de la informática de la salud que se centra principalmente en las necesidades de administración de los hospitales. En muchas implementaciones, un HIS es un sistema de información global, integrado, diseñado para gestionar todos los aspectos del funcionamiento de un hospital, tales como problemas médicos, administrativos, financieros y legales además del correspondiente procesamiento de los servicios.
IHE	Integración de la Empresa/Organización de salud (Integrating The Healthcare Enterprise). Iniciativa de profesionales de la salud (incluyendo colegios profesionales de médicos) y empresas proveedoras cuyo objetivo es mejorar la comunicación entre los sistemas de información que se utilizan en la atención al paciente.
Inteligencia Artificial	Considerada una rama de la computación que relaciona un fenómeno natural con una analogía artificial a través de programas de computadora. La inteligencia artificial puede ser tomada como ciencia si se enfoca hacia la elaboración de programas basados en comparaciones con la eficiencia del hombre, contribuyendo a un mayor entendimiento del conocimiento humano.
ISO 8859	Conjunto ISO que plantea un estándar de 8 bits para codificación de caracteres para su uso en computadoras
Machine Learning	La capacidad de un sistema para aprender de grandes conjuntos de datos en lugar de seguir las reglas preestablecidas tambien conocido como aprendizaje automático.
Medios de Contraste Radiológicos	Son sustancias o elementos químicos que permiten mejorar la visualización de algunas estructuras corporales por medio del contraste en imágenes médicas.
Metodología	Metodología es un vocablo generado a partir de tres palabras de origen griego:metà (“más allá”), odòs (“camino”) y logos (“estudio”). El concepto hace referencia al plan de investigación que permite cumplir ciertos objetivos en el marco de una ciencia. Cabe resaltar que la metodología también puede ser aplicada en el ámbito artístico, cuando se lleva a cabo una observación rigurosa. Por lo tanto, puede entenderse a la metodología como el conjunto de procedimientos que determinan una investigación de tipo científico o marcan el rumbo de una exposición doctrinal.
Ministerio de Salud (MINSAL)	Es la instancia del Estado rectora en materia de salud, que garantiza a los habitantes de la República de El Salvador la cobertura de servicios oportunos e integrales, con equidad, calidad y calidez, en corresponsabilidad con la comunidad, incluyendo todos los sectores y actores sociales, para contribuir a lograr una mejor calidad de vida.
MYCIN	Desarrollado a principios de los años 70 por Edward Shortliffe, fue uno de los primeros sistemas expertos que se usaron para diagnosticar enfermedades en medicina. El sistema podía identificar bacterias que causaban severas infecciones, tales como la bacteremia y la meningitis. Igualmente, podía recomendar antibióticos dosificados, basándose en el peso del paciente.

Neurona artificial	La unidad básica de una RNA es la neurona. Aunque hay varios tipos de neuronas diferentes, la más común es la de tipo McCulloch-Pitts.
Números Hounsfield	La escala de Hounsfield Es la asignación numérica que se realiza a los datos de absorción de los rayos X que se realizan con el TAC; van desde -1000 para el aire hasta +1.000 para la densidad metálica, pasando por el valor 0 que corresponde al agua. A este valor numérico se le asigna una escala de grises en la imagen.
PACS	Sistema de archivado y transmisión de imágenes (Picture Archiving and Communication System). Es un sistema computarizado para el archivar imágenes de estudios médicos y también para la transmisión de éstas a visores o estaciones de visualización dedicadas a través de una red informática.
PostgreSql	PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.
Proceso Racional Unificado	Proceso Racional Unificado o RUP (Rational Unified Process). Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta y de mayor calidad para satisfacer las necesidades de los usuario Se caracteriza por ser iterativo e incremental.
Pruebas	Ensayos aplicados a los datos, comprende todo el proceso desde la selección y transformación de los datos de entrada hasta el análisis de los resultados obtenidos por cada modelo de RNA probado.
PyCharm	PyCharm es un IDE o entorno de desarrollo integrado multiplataforma utilizado para desarrollar en el lenguaje de programación Python. Proporciona análisis de código, depuración gráfica y soporte para el desarrollo web con Django, entre otras bondades. PyCharm es desarrollado por la empresa JetBrains y debido a la naturaleza de sus licencias tiene dos versiones, la Community que es gratuita y orientada a la educación y al desarrollo puro en Python y la Professional, que incluye más características.
Python	Python es un lenguaje de programación, que se puede usar en desarrollo web, para escribir interfaces gráficas de usuario (GUI) de escritorio, crear juegos, y mucho más. Es de alto nivel, interpretado y orientado a los objetos.
Rayos X	Los rayos X son un tipo de radiación llamada ondas electromagnéticas. Las imágenes resultantes de la aplicación de rayos X muestran el interior de su cuerpo en diferentes tonos de blanco y negro. Esto es debido a que diferentes tejidos absorben diferentes cantidades de radiación. El calcio en los huesos absorbe la mayoría de los rayos X, por lo que los huesos se ven blancos. La grasa y otros tejidos blandos absorben menos, y se ven de color gris.
Red Neuronal Artificial	Las Redes Neuronales son un campo de la Inteligencia Artificial. Inspirándose en el comportamiento conocido de las neuronas en el cerebro humano, tratan de crear modelos artificiales que solucionen problemas difíciles de resolver mediante técnicas algorítmicas convencionales.

RIS	Sistema de información de radiología o RIS (Radiology Information System) es un sistema informatizado de base de datos utilizado por los Departamentos de Radiología para almacenar, manipular y distribuir datos radiológicos de pacientes e imágenes. El sistema consiste generalmente en el seguimiento del paciente y la programación de citas, presentación de informes de resultados y capacidades de seguimiento de imagen.
SIADTACC	Sistema informático para el análisis digital de tomografías axiales computarizadas cerebrales. Este sistema dará un apoyo en el diagnóstico diferencial a los radiólogos del Ministerio de Salud de la República de El Salvador.
SIAP	Sistema Integral de Atención al Paciente. Es un supra sistema informático que pretende dar soporte para la atención de los pacientes en el área de registro de pacientes, citas médicas, consulta externa, farmacia, imagenología, laboratorio clínico.
SIMAGD	Sistema Informático de Imagenología Digital que permite vincular al expediente clínico de cada paciente registrado en el SIAP con la captura, almacenamiento y recuperación de imágenes médicas.
TAC	Tomografía Axial Computarizada. Procedimiento computarizado de imágenes por rayos X en el que se proyecta un haz angosto de rayos X a un paciente y se gira rápidamente alrededor del cuerpo, produciendo señales que son procesadas por computadora. Estas imágenes permiten la identificación y ubicación de las estructuras básicas, así como de posibles tumores o anormalidades.
Test	Comprende la realización de pruebas de validación a la RNA entrenada, pueden ser realizadas por lotes (batch) o individuales.
UML	El Lenguaje de Modelamiento Unificado (Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software.
Voxel	Unidad de volumen de la imagen. Es la unidad de volumen que representa el píxel en el monitor de la TAC (píxel por sección de corte) que es representada en la imagen plana por el píxel. Dentro de cada voxel se considera constante el coeficiente de atenuación del objeto. Un corte de TAC tiene un grosor definido y se compone de una matriz de unidades cúbicas (voxels) de idéntico tamaño.
Weasis	Visor DICOM que puede conectarse a un PAC. Ha sido diseñado para satisfacer las expectativas de varios sistemas de información clínica y su evolución futura con respecto a las imágenes médicas.

Tabla 102 - Glosario de términos

11. REFERENCIAS BIBLIOGRÁFICAS

- Alex Krizhevsky, Ilya Sutskever & Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Network.
- Alfaro M. (2012). *Utilización de metodologías de Inteligencia Artificial y sus aplicaciones en El Salvador.* Ing-novación, Revista semestral de ingeniería e innovación de la Facultad de Ingeniería, Universidad Don Bosco. 2(3), 57-68.
- Arias F.G. (1999). *El proyecto de investigación.* (3ra ed.). Caracas: Editorial Episteme.
- Asamblea Legislativa - Republica de El Salvador, Ley Especial Contra los Delitos Informáticos y Conexos. [En línea], [Fecha de consulta: 10 de abril de 2016]. Disponible en: <http://www.asamblea.gob.sv/eparlamento/indice-legislativo/buscador-de-documentos-legislativos/ley-especial-contra-los-delitos-informaticos-y-conexos>
- Asamblea Legislativa - Republica de El Salvador, Ley de Deberes y Derechos de los Pacientes y Prestadores de Servicios de Salud. [En línea], [Fecha de consulta: 10 de abril de 2016]. Disponible en: <http://www.asamblea.gob.sv/eparlamento/indice-legislativo/buscador-de-documentos-legislativos/ley-de-deberes-y-derechos-de-los-pacientes-y-prestadores-de-servicios-de-salud>
- Barberis, L. S., (2009). Aplicación de un sistema automático de procesamiento de imágenes médicas basada en estándares. Instituto Balseiro. Universidad de Cuyo, Argentina.
- Bernal, C. (2006). *Metodología de la investigación. México*: Pearson-Prentice Hall.
- Bertona, L.F., (2005). Entrenamiento de redes neuronales basado en algoritmos evolutivos. Argentina.
- Biafore, B (2006). Gestión de proyectos con MS Project, España: Anaya.
- Biblioteca Universidad de Palermo [En línea], *Las citas bibliográficas y otras normas de estilo.* [Fecha de consulta: 25 de marzo de 2016]. Disponible en: <http://www.palermo.edu/biblioteca/Archivos/biblioteca_guia_de_citas.doc>
- Bui A.,Taira R. K. (2010). *The Image Processing Handbook*. Springer, USA.
- Bunge, M. (2008). En busca de la filosofía en las Ciencias Sociales. México: Siglo XXI.
- CATS. WageIndicator Foundation, Comparador de salarios en El Salvador [En línea], [Fecha de consulta: 11 de abril de 2016]. Base de datos disponible en: <http://www.tusalario.org/elsalvador/Portada/salario/comparador-salarial>
- Corbo D. N. (2004). *Tomografía Axial Computada.* Ponencia presentada en el XIII Seminario de Ingeniería biomédica, Universidad de la República Oriental del Uruguay. Uruguay.
- DCM4CHEE Org., [En línea], *documentación del software dcm4che,* [Fecha de consulta: 25 de marzo de 2016]. Disponible en: <https://dcm4che.atlassian.net/wiki/>
- Delgado A. (1999). *Aplicación de las Redes Neuronales en Medicina.* Revista de la Facultad de Medicina, Universidad Nacional de Colombia, 47(4), 221-223.

- Departamento de Tecnologías de la Información y las Comunicaciones Universidade da Coruña [En línea], *Introducción a las Redes Neuronales Artificiales,* [Fecha de consulta: 05 de abril de 2016]. Disponible en: <http://sabia.tic.udc.es/mgestal/cv/RNAtutorial/index.html>
- De la Cerda A. (2009). *Equipos de Tomografía computarizada (TAC).* Revista digital para profesionales de la enseñanza, 1(5).
- Díez R.P., Gómez A.G., Martínez A. (2001). Introducción a la inteligencia artificial: sistemas expertos, redes neuronales artificiales y computación evolutiva.
- Escuela Superior de Ingeniería de Bilbao, EHU [En línea]. *Redes Neuronales Artificiales y sus aplicaciones.* [Fecha de consulta: 05 de abril de 2016]. Disponible en: <http://www.ciberesquina.una.edu.ve:8080/2014_2/350_E.pdf>
- Facultad de Ciencias Económicas y de Administración [En línea], *Material de apoyo sobre administración, redacción e investigación.* [Fecha de consulta: 25 de marzo de 2016]. Disponible en: <http://www.ccee.edu.uy/ensenian/catmetinvcont/material/>
- García C. (2003). *Anatomía del error en radiología.* Revista Chilena de Radiología, 9(3), 144-150.
- Giger M., Chan H. y Boone J. (2008). History and status of CAD and quantitative image analysis: the role of Medical Physics and AAPM.
- Giger M. y Suzuki K. (2008). *Computer-Aided Diagnosis*. USA: David Dagan Feng. Academic Press.
- Google Inc. [En Linea], Aplicación de la librería de IA TensorFlow. [Fecha de consulta: 03 de mayo de 2016]. Disponible en: <https://www.tensorflow.org/>
- Google Inc. [En Linea], Google Commandline Flags. [Fecha de consulta: 06 de marzo de 2016]. Disponible en: <*https://github.com/google/glog*>
- Google Inc. [En Linea], Protocol Buffers. [Fecha de consulta: 06 de marzo de 2016]. Disponible en: <*https://developers.google.com/protocol-buffers/*>
- Hernández, R., Fernández, C. & Baptista, L. (2008). *Metodología de la investigación*. México: Mc Graw Hill.
- Herrera A., (2006). Sistema PACS mínimo basado en el estándar DICOM (Tesis de maestria). Universidad autónoma metropolitana. México, D.F.
- Jia, Yangqing S., Donahue E., Karayev J., Sergey L., Girshick J., Ross G, Darrell S., Trevor, Caffe: Convolutional Architecture for Fast Feature Embedding.
- Kruchten P. (1995). *The 4+1 View Model of Architecture.* IEEE Software, 6 (12), 45–50.
- Kunio, D. (2007). Computer-aided diagnosis in medical imaging: Historical review, current status and future potential. *Comput Med Imaging Graph.* 31(4-5). 198–211.
- Lackes R, Mack D, Ziola J, Ahern K. (1998). Neuronal Networks: Basics and Applications. CBT (Computer Based Training) Springer, Alemania.
- Lizadra, J., Aranda C., Orallo J. Síntesis de imágenes en imagen médica. Universidad Politécnica de Valencia, Valencia, España.
- López R.F., Fernández J.M.F., (2008). Las redes neuronales artificiales.
- Malhotra, M., Grover V., (1998). An assessment of survey research in production and operations management: From constructs to theory. Journal of Operations Management, 16(4), 407-425.

- Matich, Damian Jorge. (2001). Redes Neuronales: conceptos Básicos y Aplicaciones. Argentina.
- Matthew D., Zeiler R. F., Visualizing and Understanding Convolutional Networks.
- McCulloch W, Pitts W. (1943). A logical calculus of the ideas immanent in nervous activity. Bull Mayhen Biophys 1943; 7:115 – 133. United States.
- Méndez, C. (2006). Metodología, diseño y desarrollo del proceso de investigación con énfasis en Ciencias Empresariales. Bogotá: Limusa, Noriega Editores.
- MINSAL, *Informe de Traumatismo Intracraneal.* El Salvador: Sistema Informatico de MorbiMortalidad (SIMMOW). Programa Computacional.
- Molina J. L., Catarrero J. (2011). *Estudios craneoencefálicos a través del TAC.* Málaga: Fesitess Andalucía.
- National Center for Biotechnology Information, U.S. National Library of Medicine [En línea], *Benefits of Using the DCM4CHE DICOM Archive,* [Fecha de consulta: 25 de marzo de 2016]. Disponible en: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2039778/>
- NVIDIA Corporation. [En Linea], Procesamiento Paralelo CUDA. [Fecha de consulta: 03 de marzo de 2016]. Disponible en: <*http://www.nvidia.es/object/cuda-parallel-computing-es.html*>
- Orellana P. (2003). *Errores neurorradiologicos frecuentes en TC y RM***.** Revista Chilena de Radiología, 9(2), 93-103.
- Pons L. C., (1997). *Comparación entre la TAC y la RM*. FMC; 4(3), 174-84.
- Ríos J., Pazos A., (1991). Estructura dinámica y aplicaciones de las redes de neuronas artificiales. Madrid.
- Sierra, R. (2008). Técnicas de investigación social. Teoría y ejercicios. Madrid: Thompson.
- Simonyan K., Zisserman A., Very Deep Convolutional Networks for Large-Scale Image Recognition.
- Sociedad Andaluza de Enseñanza Matemáticas Thales, *Fundamento de las Redes Neuronales* [En línea], [Fecha de consulta: 05 de abril de 2016]. Disponible en: <http://thales.cica.es/rd/Recursos/rd98/TecInfo/07/capitulo2.html>
- The Berkeley Artificial Intelligence Research. Blobs, Layers, and Nets: anatomy of a Caffe model. [En línea], [Fecha de consulta: 25 de abril de 2016]. Disponible en: <http://caffe.berkeleyvision.org/tutorial/net_layer_blob.html>
- Torres Z., Navarro J. (2007). *Conceptos y principios fundamentales de Epistemología y de Metodología*. México: IIEE, Universidad Michoacana de San Nicolás de Hidalgo.
- University of Tennessee System. [En Linea], Collection of mathematical software. [Fecha de consulta: 06 de marzo de 2016]. Disponible en: <*http://www.netlib.org/blas/*>
- Universidad Tecnológica de Pereira. (2010). Descripción del estándar DICOM para un acceso confiable a la información de las imágenes médicas. Scientia et Technica, 17(45).
- Van Gigch, J. (2008). *Teoría general de sistemas*. México: Trillas.
- Viruez J., Vera O., Torrez K. y Bailey F. (2011). *Errores diagnósticos en el Accidente Vascular Cerebral.* Revista Médica La Paz, 17(1), 16-21.

- Vilarrasa, A. (2006). Sistema inteligente para la detección y diagnóstico de patología mamaria (Tesis doctoral). Universidad Complutense de Madrid, Madrid, España.

12. ANEXOS

ANEXO 1: Evaluación de metodologías de investigación y desarrollo de software.

Para la selección de las diferentes metodologías a utilizar en la realización del proyecto se realizó una evaluación entre varias alternativas, a continuación se presentan los criterios de selección y los resultados de la evaluación:

Criterios para la selección de la metodología de investigación.

Aspecto	Tipo de investigación	Descripción	Cumple con objetivos de la investigación
Por su finalidad	Básica	Busca enriquecer el conocimiento teórico	×
	Aplicada	Busca conocimientos específicos, la resolución de problemas prácticos	✔
Alcance temporal	Transversal	La investigación considera un momento único	✔
	Longitudinal	Permite estudiar la evolución del fenómeno en un momento determinado	×
Por su profundidad	Exploratoria	Cuando el objetivo consiste en examinar un tema poco estudiado	✔
	Descriptiva	Cuando busca especificar propiedades, características y rasgos importantes de cualquier fenómeno que se analice.	×
	Explicativa	Pretende establecer causas y eventos, sucesos o fenómenos que se estudian.	×
Por su carácter o enfoque	Cuantitativa	Aquella en la que los datos adoptan forma numérica.	✔
	Cualitativa	No busca cuantificar sino comprender, profundizar en el entendimiento y las	✔

		interioridades de los fenómenos.	
Por su naturaleza	Documental	Se basa en documentos	✔
	Empírica	Aquella que implica trabajar con hecho de experiencia directa.	×
	Experimental	Observa fenómenos producidos en laboratorios.	×
	Encuestas	Los datos provienen de sujetos observados	×

Tabla 103 - Criterios para la selección de la metodología de desarrollo de software

Criterios de evaluación para la selección de la metodología de desarrollo de software

En la siguiente tabla se aprecian los criterios utilizados (con su respectiva ponderación) para evaluar cada una de las metodologías propuestas, dichos criterios permiten determinar el dominio y experiencia del equipo de trabajo sobre la metodología, las necesidades y características del proyecto que se deben satisfacer.

N°	Criterio	Descripción	Ponderación
1	Conocimiento previo	Los miembros del equipo poseen conocimiento previo de la existencia de esta metodología	5 %
2	Posee documentación	La metodología cuenta con un robusto soporte documental para referencias futuras	10 %
3	Experiencia de uso de la metodología	Los miembros del equipo tienen experiencia previa en la aplicación de la metodología	35 %
4	Herramientas de Software relacionadas	Los miembros del equipo tienen conocimiento y experiencia de uso sobre herramientas que permitan gestionar la metodología	5 %
5	Desarrollo por prototipos	La metodología permite gestionar el desarrollo de proyectos que involucren entrega de prototipos	5 %
6	Aseguramiento de la calidad	La metodología garantiza el aseguramiento de la calidad en cada una de sus etapas	15 %
7	Administración de riesgos	La metodología posee mecanismos para la gestión de riesgos	10 %

8	Medición de avance del proyecto	La metodología posee mecanismos para la gestión eficiente del avance del proyecto y la asignación y uso de recursos	15 %

Tabla 104 - Criterios para la evaluación de la metodología de desarrollo de software

Las metodologías sometidas a evaluación son las siguientes:

- **SCRUM**: Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

- **Proceso Racional Unificado (RUP):** Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta y de mayor calidad para satisfacer las necesidades de los usuarios que tienen un cumplimiento al final dentro de un límite de tiempo y presupuesto previsible. Es una metodología de desarrollo iterativo que es enfocada hacia “diagramas de los casos de uso, y manejo de los riesgos y el manejo de la arquitectura” como tal.

- **Programación Extrema (XP):** Es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck, es de las metodologías más destacadas de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

Evaluación individual

La manera de realizar la evaluación es la siguiente: los miembros del equipo asignarán una nota comprendida entre 0 y 10 a cada criterio de evaluación basados en su conocimiento y experiencia previa, dicha nota será multiplicada por su respectiva ponderación, estos valores se sumarán dando como resultado la nota final obtenida por cada metodología. Se han considerado criterios generales los cuales comparten ponderación con la evaluación realizada por cada uno de los miembros del equipo, como se observa en la tabla 13.

Evaluación por Carlos Hernández							
Criterio	Ponderacion (%)	SCRUM		RUP		XP	
		Nota	%	Nota	%	Nota	%
Conocimiento previo	5	8,00	0,40	7,00	0,35	6,00	0,30
Posee documentacion	10	8,00	0,80	9,00	0,90	7,00	0,70
Experiencia de uso de la metodología	35	8,00	2,80	7,00	2,45	6,00	2,10
Herramientas de Software relacionadas	5	7,00	0,35	6,00	0,30	5,00	0,25
Desarrollo por prototipos	5	8,00	0,40	9,00	0,45	8,00	0,40
Aseguramiento de la calidad	15	8,00	1,20	9,00	1,35	7,00	1,05
Administración de riesgos	10	7,00	0,70	9,00	0,90	7,00	0,70
Medicion de avance del proyecto	15	7,00	1,05	9,00	1,35	7,00	1,05
TOTAL			**7,70**		**8,05**		**6,55**

Tabla 105 - Evaluación de la metodología por Carlos Hernández

Evaluación por Alvaro Orrego							
Criterio	Ponderacion (%)	SCRUM		RUP		XP	
		Nota	%	Nota	%	Nota	%
Conocimiento previo	5	6,00	0,30	7,00	0,35	6,00	0,30
Posee documentacion	10	8,00	0,80	9,00	0,90	6,00	0,60
Experiencia de uso de la metodología	35	6,00	2,10	7,00	2,45	5,00	1,75
Herramientas de Software relacionadas	5	7,00	0,35	6,00	0,30	5,00	0,25
Desarrollo por prototipos	5	8,00	0,40	8,00	0,40	7,00	0,35
Aseguramiento de la calidad	15	7,00	1,05	9,00	1,35	6,00	0,90
Administración de riesgos	10	7,00	0,70	8,00	0,80	7,00	0,70
Medicion de avance del proyecto	15	7,00	1,05	9,00	1,35	7,00	1,05
TOTAL			**6,75**		**7,90**		**5,90**

Tabla 106 - Evaluación de la metodología por Alvaro Orrego

Evaluación por Santiago Colato							
Criterio	Ponderacion (%)	SCRUM		RUP		XP	
		Nota	%	Nota	%	Nota	%
Conocimiento previo	5	6,00	0,30	7,00	0,35	5,00	0,25
Posee documentacion	10	5,00	0,50	8,00	0,80	6,00	0,60
Experiencia de uso de la metodología	35	5,00	1,75	7,00	2,45	5,00	1,75

Herramientas de Software relacionadas	5	5,00	0,25	9,00	0,45	6,00	0,30
Desarrollo por prototipos	5	7,00	0,35	9,00	0,45	6,00	0,30
Aseguramiento de la calidad	15	7,00	1,05	8,00	1,20	5,00	0,75
Administración de riesgos	10	8,00	0,80	8,00	0,80	5,00	0,50
Medicion de avance del proyecto	15	8,00	1,20	8,00	1,20	6,00	0,90
TOTAL			6,20		7,70		5,35

Tabla 107 - Evaluación de la metodología por Santiago Colato

Evaluación por Marlon Menjívar							
Criterio	Ponderacion (%)	SCRUM		RUP		XP	
		Nota	%	Nota	%	Nota	%
Conocimiento previo	5	5,00	0,25	8,00	0,40	4,00	0,20
Posee documentacion	10	9,00	0,90	9,00	0,90	5,00	0,50
Experiencia de uso de la metodología	35	3,00	1,05	8,00	2,80	3,00	1,05
Herramientas de Software relacionadas	5	8,00	0,40	8,00	0,40	5,00	0,25
Desarrollo por prototipos	5	10,00	0,50	10,00	0,50	7,00	0,35
Aseguramiento de la calidad	15	8,00	1,20	9,00	1,35	6,00	0,90
Administración de riesgos	10	9,00	0,90	8,00	0,80	7,00	0,70
Medicion de avance del proyecto	15	8,00	1,20	8,00	1,20	6,00	0,90
TOTAL			6,40		8,35		4,85

Tabla 108 - Evaluación de la metodología por Marlon Menjívar

Evaluación por Lennin Hernández							
Criterio	Ponderacion (%)	SCRUM		RUP		XP	
		Nota	%	Nota	%	Nota	%
Conocimiento previo	5	7,00	0,35	8,00	0,40	7,00	0,35
Posee documentacion	10	9,00	0,90	9,00	0,90	7,00	0,70
Experiencia de uso de la metodología	35	7,00	2,45	8,00	2,80	7,00	2,45
Herramientas de Software relacionadas	5	8,00	0,40	8,00	0,40	7,00	0,35
Desarrollo por prototipos	5	8,00	0,40	9,00	0,45	8,00	0,40

Aseguramiento de la calidad	15	8,00	1,20	8,00	1,20	8,00	1,20
Administración de riesgos	10	8,00	0,80	8,00	0,80	8,00	0,80
Medicion de avance del proyecto	15	8,00	1,20	9,00	1,35	8,00	1,20
TOTAL			**7,70**		**8,30**		**7,45**

Tabla 109 - Evaluación de la metodología por Lennin Hernández

Resultado de la evaluación

Evaluador / Criterios generales	Ponderacion (%)	SCRUM		RUP		XP	
		Nota	%	Nota	%	Nota	%
Colato Mártir, Felipe Santiago	18	6,20	1,12	7,70	1,38	5,35	0.96
Hernández, Carlos Antonio	18	7,70	1,38	8,05	1,45	6,55	1,18
Hernández Nieto, Lennin David	18	7,70	1,38	8,30	1,49	7,45	1,34
Hernández Orrego, Alvaro Ernesto	18	6,75	1,22	7,90	1,42	5,90	1,06
Menjívar Martínez, Marlon Armando	18	6,4	1,15	8,35	1,50	4,85	0,87
Menor riesgo de aplicación de la metodología	10	5,0	0,50	7,0	0,70	4,0	0,40
TOTAL			**6,83**		**7,94**		**5,81**

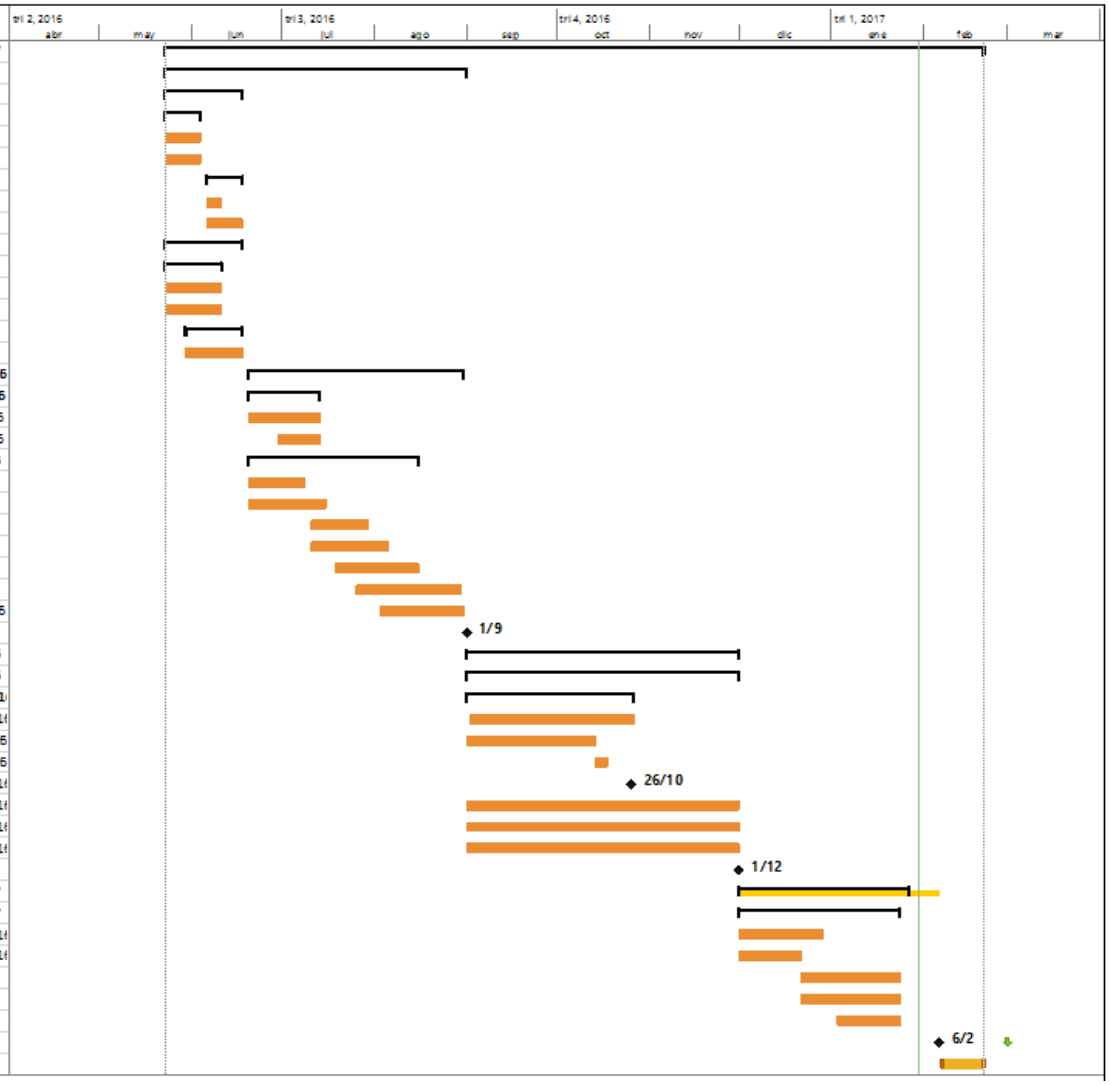
Tabla 110 - Resultados de la evaluación de la metodología de desarrollo de software

Conclusión

Se selecciona la metodología RUP por ser la que obtuvo mayor calificación por parte del equipo evaluador, lo que indica que cumple con las necesidades del proyecto y existe mayor dominio por parte de los miembros del equipo.

ANEXO 2: Diagrama Gantt del proyecto.

Id	Nombre de tarea	Duración	Comienzo	Fin
1	**Trabajo de graduación**	**196 días**	**lun 23/5/16**	**lun 20/2/17**
2	**Primera Entrega**	**73 días**	**lun 23/5/16**	**jue 1/9/16**
3	**Investigación previa al desarrollo**	**20 días**	**lun 23/5/16**	**vie 17/6/16**
4	**Fase Cualitativa**	**10 días**	**lun 23/5/16**	**vie 3/6/16**
5	Recolección de información	10 días	lun 23/5/16	vie 3/6/16
6	Análisis de información	10 días	lun 23/5/16	vie 3/6/16
7	**Fase Cuantitativa**	**10 días**	**lun 6/6/16**	**vie 17/6/16**
8	Experimentación y pruebas con datos obtenidos	5 días	lun 6/6/16	vie 10/6/16
9	Elaboración de informe de resultados	10 días	lun 6/6/16	vie 17/6/16
10	**Fase de Iniciación**	**20 días**	**lun 23/5/16**	**vie 17/6/16**
11	**Análisis**	**15 días**	**lun 23/5/16**	**vie 10/6/16**
12	Obtención de requerimientos de arquitectura del sistema	15 días	lun 23/5/16	vie 10/6/16
13	Obtención de macro requerimientos	15 días	lun 23/5/16	vie 10/6/16
14	**Diseño**	**15 días**	**lun 30/5/16**	**vie 17/6/16**
15	Diseño de arquitectura	15 días	lun 30/5/16	vie 17/6/16
16	**Fase de Elaboración**	**52 días**	**lun 20/6/16**	**mar 30/8/16**
17	**Análisis**	**18 días**	**lun 20/6/16**	**mié 13/7/16**
18	Obtención de requerimientos	18 días	lun 20/6/16	mié 13/7/16
19	Validación de requerimientos	10 días	jue 30/6/16	mié 13/7/16
20	**Diseño**	**41 días**	**lun 20/6/16**	**lun 15/8/16**
21	Diseño de arquitectura	15 días	lun 20/6/16	vie 8/7/16
22	Especificaciones de diseño (estándares de programación)	20 días	lun 20/6/16	vie 15/7/16
23	Diseño de interfaces	15 días	lun 11/7/16	vie 29/7/16
24	Diseño de datos	20 días	lun 11/7/16	vie 5/8/16
25	Diseño de algoritmos de la red neuronal	20 días	mar 19/7/16	lun 15/8/16
26	Desarrollo (Prototipo 1)	25 días	mar 26/7/16	lun 29/8/16
27	Pruebas	20 días	mié 3/8/16	mar 30/8/16
28	Segunda Defensa	0 días	jue 1/9/16	jue 1/9/16
29	**Segunda Entrega**	**65 días**	**jue 1/9/16**	**jue 1/12/16**
30	**Fase de Construcción**	**65 días**	**jue 1/9/16**	**jue 1/12/16**
31	**Análisis y Diseño**	**40 días**	**jue 1/9/16**	**mié 26/10/1**
32	Correcciones de primera Entrega	39 días	vie 2/9/16	mié 26/10/1
33	Nuevos requerimientos	31 días	jue 1/9/16	jue 13/10/16
34	Validación de requerimientos	2 días	vie 14/10/16	lun 17/10/16
35	Entrega documentacion de análisis y diseño	0 días	mié 26/10/1	mié 26/10/1
36	Desarrollo	65 días	jue 1/9/16	mié 30/11/1
37	Pruebas Unitarias	65 días	jue 1/9/16	mié 30/11/1
38	Pruebas de integración de componentes	65 días	jue 1/9/16	mié 30/11/1
39	Tercera defensa	0 días	jue 1/12/16	jue 1/12/16
40	**Tercera Entrega**	**41 días**	**jue 1/12/16**	**jue 26/1/17**
41	**Fase de Transición**	**38 días**	**jue 1/12/16**	**lun 23/1/17**
42	Correcciones de segunda entrega	20 días	jue 1/12/16	mié 28/12/1
43	Análisis y Diseño	15 días	jue 1/12/16	mié 21/12/1
44	Desarrollo	23 días	jue 22/12/16	lun 23/1/17
45	Pruebas	23 días	jue 22/12/16	lun 23/1/17
46	Despliegue en entorno de pruebas	15 días	mar 3/1/17	lun 23/1/17
47	Defensa pública	0 días	lun 6/2/17	lun 6/2/17
48	**Preparación de tomos finales**	10 días	mar 7/2/17	lun 20/2/17



ANEXO 3: Selección de framework Caffe

Herramientas para el desarrollo de aplicaciones de Inteligencia Artificial.

En la actualidad existen muchas herramientas disponibles para el desarrollo de aplicaciones que implemente algún modelo de inteligencia artificial, este avance se debe en gran medida a empresas que están constantemente invirtiendo en la investigación y desarrollo de este tipo de tecnología con el fin de incorporarlas en sus procesos de negocios. Entre estas empresas podemos mencionar a Google, Microsoft, IBM, Amazon (AWS), Facebook, entre otras.

A continuación se presenta un listado de algunas de las APIs de *machine learning* y frameworks de *deep learning* disponibles para el desarrollo de componentes de inteligencia artificial:

- Amazon's Deep Scalable Sparse Tensor Network Engine (DSSTNE)

Librería de deep learning de código abierto, que se pronuncia "destiny" (destino en inglés), permite a los científicos de datos entrenar y desplegar redes neuronales profundas utilizando las GPU. DSSTNE fue construido para alimentar el motor de recomendaciones de Amazon, el cual hace sugerencias de productos a los cientos de millones de clientes en sus sitios web cada día.

- Amazon Web Services Machine Learning API

Tiene por objetivo hacer facilitar a los desarrolladores el acceso a algoritmos complejos, está basado en la tecnología usada en sus servicios de búsqueda y predicción de Amazon.

- Google APIs

Google tiene una gran cantidad de herramientas de machine learning en su servicio Google Cloud Platform. Esto incluye su API de predicción que permite a los usuarios aprovechar los algoritmos disponibles para analizar los datos y predecir resultados futuros. Utiliza procesamiento por GPU y permite crear nuevos modelos e integrarlos a otros servicios como Google Data Flow y Big Query.

- Google TensorFlow

Librería de código abierto capaz de producir gráfos en C++ o Python, los cuales se pueden procesar en las CPU o GPU; estos gráfos representan el flujo de datos en tiempo de ejecución en un sistema.

- Microsoft Distributed Machine Learning Toolkit (DMLT)

Tiene como objetivo optimizar el trabajo con clusters de machine learning lo que permite que sea más fácil ejecutar múltiples y diferentes aplicaciones al mismo tiempo.

- Microsoft Computational Network Toolkit (CNTK)

Permite crear RNAs representadas en gráfos dirigidos. Desarrollada inicialmente para la tecnología de reconocimiento de voz, desde abril de 2015 se ha convertido en un conjunto de herramientas más general dando soporte a imágenes, texto y entrenamiento de RNN (red neuronal recurrente).

- IBM Watson Analytics

IBM Watson es una plataforma tecnológica que utiliza el procesamiento del lenguaje natural y de machine learning para revelar diferentes puntos de vista sobre grandes cantidades de datos no estructurados. IBM ofrece acceso a la API como un servicio en su plataforma Bluemix, lo que permite crear aplicaciones basadas en Watson.

- Apache Spark MLlib and Singa

Apache Spark MLlib: framework para el procesamiento de datos en memoria. Spark ofrece una amplia (y creciente) libreria de algoritmos y utilidades para clasificación, regresión, clustering, filtrado colaborativo y demas útilidades para el procesamiento de datos en memoria.

Singa: Framework que proporciona una herramienta de programación para RNAs profundas distribuidas.

- Samsung VELES

Plataforma para el desarrollo de apliaciones de deep learning, esta escrito en C++ y utliza Python para gestionar los diferentes nodos, ofrece una API que permite a los desarrolladores usar modelos entrenados que pueden ser utilizados para el analisis de datos.

- Alibaba’s Aliyun

Aliyun ofrece un servicio de machine learning para apoyar a las empresas en el desarrollo de aplicaciones para el analisis de datos. Esta basado en la tecnologia de Open Data Processing Service (ODPS) el cual es capaz de procesar 100 petabytes de datos en 6 horas.

- Caffe

Es un framework de deep learning desarrollado en C++, creado inicialmente para usos de visión artificial (una inspección automática basada en la imagen). Es desarrollado por el Berkeley Vision and Learning Center (BVLC) y la comunidad de desarrolladores. Es utilizado en proyectos de investigación academica, prototipos y aplicaciones de escala industrial en areas de visión, habla y multimedia.

La empresa Yahoo creo el proyecto CaffeOnSpark el cual combina las capacidades de deep learning de Caffe con el motor de procesamiento de datos Apache Spark. Tambien es utilizado por Google y Pinterest en algunas de sus aplicaciones.

Selección de la herramienta

A continuación, se muestra una tabla comparativa con las herramientas más populares para el desarrollo de aplicaciones con inteligencia artificial, los criterios de comparación se basan en la estabilidad y confiabilidad de estos; así como los alcances y limitantes del proyecto actual.

Herramienta	Desarrollador	Versión estable	Fecha de lanzamiento	Licencia	Open source	Tipo	Interface	Plataforma	Almacenamiento máximo	Ejecución como servicio	Procesamiento por GPU	Optimizado para imágenes	Modelos preentrenados	Integración con otras herramientas
DSSTNE	Amazon Labs	1.8.12	10/05/2016	Apache 2.0	Si	Bibliotecas de software	C++	AWS/Linux	--	No	Si	No	Si	Si
Amazon web services Machine Learning API	Amazon Web Services	2014-12-12	09/04/2015	Comercial	No	Servicio en la nube	Java/.NET/Python/PHP/JavaScript/Ruby	Cloud con Windows/ Mac/Linux UNIX	100GB de datos	Si	--	No	Si	Si
Google Cloud Prediction	Google Cloud Platform	V1.6	01/09/2010	Comercial	No	Servicio en la nube	Python/JavaScript/.NET	Cloud con Linux/Mac OS X/Windows	5MB por día	Si	--	No	Si	Si
Google Cloud Machine Learning	Google Cloud Platform	v1beta1	29/09/2016	Comercial	No	Servicio en la nube	Python/JavaScript/.NET	Cloud con Linux/Mac OS X/Windows	5MB por día	Si	--	No	No	Si
Google Tensor Flow	Google Brain Team y Google Machine Intelligence research organization	r0.11	08/12/2015	Apache 2.0	Si	Biblioteca de software	Python/C/C++	Linux/Mac OS	--	Si	Si	Si	Si	Si

DMLT	Artificial Intelligence Group of Microsoft Research Asia	v0.1	12/11/2015	MIT	Si	Kit de herramientas	Python/Lua	Linux/Windows	--	Si	Si	No	Si	Si
CNTK	Microsoft Research	v 1.7.1	25/04/2015	MIT	Si	Kit de herramientas	BrainScript/ Python/C++/ .NET	Linux/Windows	--	Si	Si	Si	Si	Si
Microsoft Azure Machine Learning API	Microsoft	2	18/02/2015	Comercial	No	API	R/Python	Cloud	--	Si	--	Si	--	Si
IBM Watson Analytics	IBM	--	16/09/2014	Comercial	No	Servicio en la nube	--	Cloud	100GB de datos	No	--	No	--	--
Apache Spark MLlib	Apache Software Foundation, originalmente desarrollado por AMPLab de la University of California, Berkeley	2.0.1	07/10/2012	Apache 2.0	Si	Biblioteca de software	Scala/Java/ Python/R	Windows/ Linux/Mac OS	--	Si	Si	No	Si	Si
Apache Singa	Apache Incubator	1.0.0	08/10/2015	Apache 2.0	Si	Biblioteca de software	C++/Python/ Java	Linux/Mac OS X /Windows	--	Si	Si	Si	Si	Si
SAMSUNG VELES	SAMSUNG	0.10.0	01/07/2015	Apache 2.0	Si	Plataforma distribuida	Python/Flow-Based Programming	Windows/ Mac OS X	--	Si	Si	Si	Si	Si
Alibaba's Aliyun	Alibaba	--	25/08/2015	Comercial	No	Servicio en la nube	Python	Cloud	5000TB de datos	Si	--	Si	Si	Si

Caffe	Berkeley Vision and Learning Center, community contribu	1.0	09/10/2013	BSD 2-Clause license	Sí	Framework	C++, Python, MATLAB	Linux/Windows	--	Si	Si	Si	Si	Si

Tabla 111 - Comparativa de las herramientas de desarrollo de Redes Neuronales Artificiales

Debido a la naturaleza del proyecto, el cual requiere el uso exclusivo de software libre, se tuvieron que descartar potentes herramientas como las de Google, Amazon y Alibaba. Además, se eligieron las herramientas que usan procesamiento por GPU para optimización de tiempo, que se ejecutará como servicio y se pudiera integrar con otras herramientas, debido a que será un componente del sistema SIMAGD. Y se le dio mucha importancia al procesamiento de imágenes y por último, pero de gran importancia, la fecha de lanzamiento, debido a que esta garantiza que la herramienta sea estable, este documentada y tenga apoyo de la comunidad en Internet. Estas fueron las razones principales por las que se eligió el framework caffe desarrollado por Berkeley Vision and Learning Center y con contribuciones de la comunidad.

ANEXO 4: Memorándum autorización de uso de TACs

MINISTERIO DE SALUD
GOBIERNO DE
EL SALVADOR
UNÁMONOS PARA CRECER

2016-6014-295

MEMORÁNDUM

PARA: **Dr. Douglas Salvador Martí Panameño-Director Hospital Nacional Zacamil**
Dr. Mauricio Ventura Centeno-Director Hospital Nacional Rosales
Dr. Raúl Robverto Castillo Durán-Hospital Nacional Neumológico
Dr. Yeerles Luis Ramírez-Director Hosdpital Nacional San Rafael

DE: **Ing. Carlos Juan Martín Pérez**
Director de Tecnologías de Información y Comunicaciones

A TRAVÉS DE: **Dr. Eduardo Espinoza Fiallos**
Viceministro de Políticas de Salud

Dr. Luis Enrique Fuentes
Director Nacional de Hospitales

FECHA: **1 de Junio de 2016**

Muy estimados Dres.

Por este medio le solicitamos su amable apoyo para el desarrollo del trabajo de tesis de Ingeniería en Sistemas Informáticos de la Universidad de El Salvador denominado "Herramienta informática de apoyo al diagnóstico de tomografías axiales computarizadas".

En el marco de este esfuerzo, complementario al módulo de imagenología digital del Sistema Integral de Atención al Paciente, se requiere el apoyo del departamento de imagenología en coordinación con la Unidad de Informática del Hospital en cuanto a las siguientes necesidades:

- Provisión de estudios en formato DICOM con su respectivo diagnóstico. Se aclara que los estudios y diagnósticos serán oportunamente anonimizados antes de ser entregados al equipo y que el medio de almacenamiento que se utilizará con este fin será un disco duro de 500 Gb que esta Dirección estará poniendo a disposición de este trabajo para cada uno de sus establecimientos.

MINISTERIO DE SALUD
GOBIERNO DE
EL SALVADOR
UNÁMONOS PARA CRECER

2016-6014-295

- Participación eventual en reuniones del personal de radiología con el fin de obtener insumos para el análisis de la herramienta informática y la realización de pruebas una vez esté desarrollada. Rogamos de su parte la designación de la persona idónea de contacto.

Esta labor estará siendo acompañada por personal de la Unidad de Sistemas de Información de esta Dirección.

Agradeciendo siempre su inestimable ayuda y esperando que los resultados de este trabajo sean fructíferos, les saludamos muy cordialmente.

CJM*ceg

ANEXO 5: Hojas de presentación en MINSAL
Observaciones hechas por Dra. Jacqueline Escobar Chávez

PRESENTACIÓN PRELIMINAR

SISTEMA INFORMÁTICO PARA EL ANÁLISIS DIGITAL DE TOMOGRAFÍAS AXIALES COMPUTARIZADAS CEREBRALES UTILIZANDO REDES NEURONALES ARTIFICIALES COMO APOYO EN EL DIAGNÓSTICO DIFERENCIAL A LOS RADIÓLOGOS DEL MINISTERIO DE SALUD DE LA REPÚBLICA DE EL SALVADOR

SIADTACC

DOCUMENTO DE OBSERVACIONES

Datos del observador.

Nombre de observador	Dra. Roxana Jaquelin Escobar Chávez.
Unidad	Radiología
Cargo:	Jefe de Depto de Radiología e Imágenes Diagnosticas

El presente documento tiene el objetivo de dar a conocer la interfaz de SIADTACC a la cual tendrá acceso el usuario. Además se busca la recolección de observaciones que posteriormente serán evaluadas y solventadas de acuerdo a la factibilidad de cada caso.

Solicitamos que indique sus respectivas observaciones a cada interfaz por medio de texto o anotaciones sobre las mismas.

Hospital Nacional San Rafael, Santa Tecla
21 de Diciembre de 2016

INTEGRACIÓN DEL PLUGIN

OBSERVACIONES:

OPCIONES DE ANÁLISIS

OBSERVACIONES:

ANALIZAR SERIE

Resultado del análisis

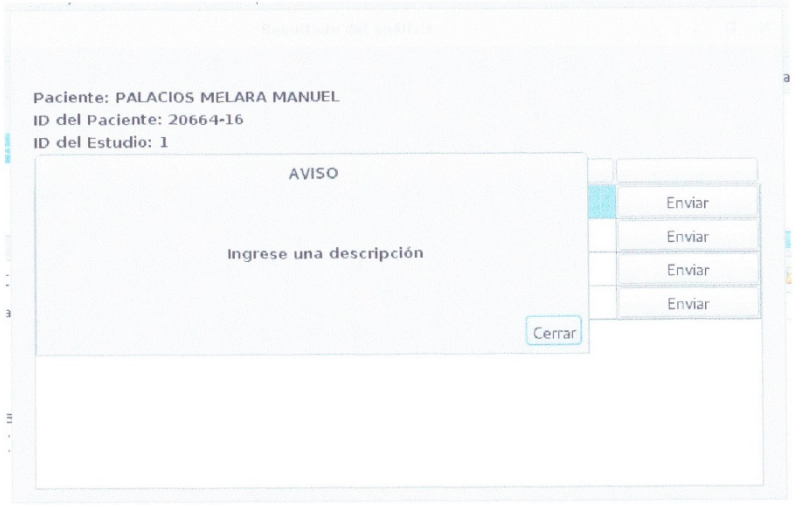
Paciente: PALACIOS MELARA MANUEL
ID del Paciente: 20664-16
ID del Estudio: 1

Series Nb	Frame	Categoría	Descripción	
501	1/1	Click para elegir ✓		Enviar
602	1/1	Click para elegir		Enviar
1	1/1	Click para elegir		Enviar
1	1/1	Click para elegir		Enviar

OBSERVACIONES:

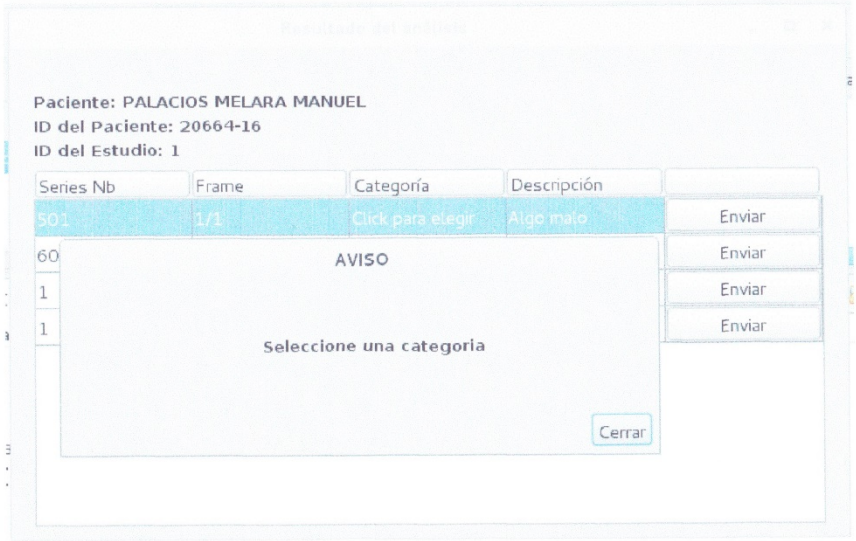
identificar con una "alarma", los casos que sean positivos !!! cerebrales, ejemplo: Hemorragia cerebral moderada !!.

AVISOS



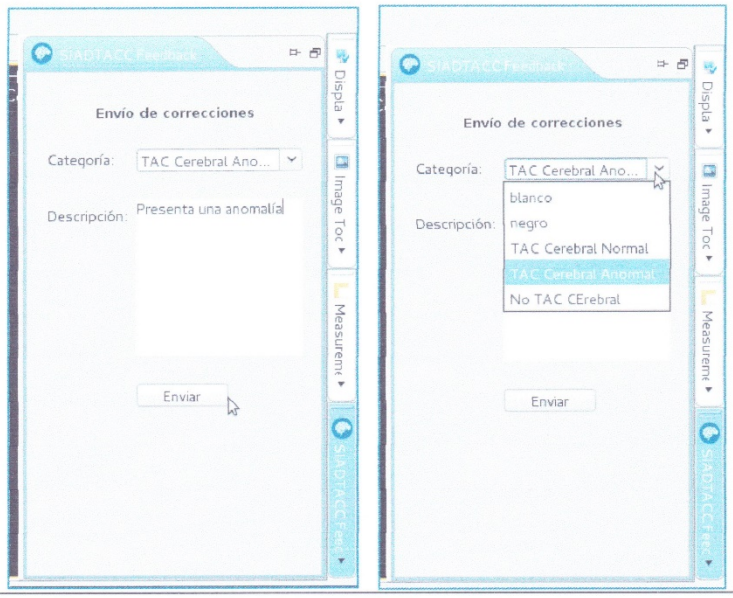
OBSERVACIONES:

AVISOS



OBSERVACIONES:

ENVIO DE SUGERENCIAS



OBSERVACIONES:

Observaciones hechas por Dra. Jasmín Orquídea Aquino Flores

PRESENTACIÓN PRELIMINAR

SISTEMA INFORMÁTICO PARA EL ANÁLISIS DIGITAL DE TOMOGRAFÍAS AXIALES COMPUTARIZADAS CEREBRALES UTILIZANDO REDES NEURONALES ARTIFICIALES COMO APOYO EN EL DIAGNÓSTICO DIFERENCIAL A LOS RADIÓLOGOS DEL MINISTERIO DE SALUD DE LA REPÚBLICA DE EL SALVADOR

SIADTACC

DOCUMENTO DE OBSERVACIONES

Datos del observador.

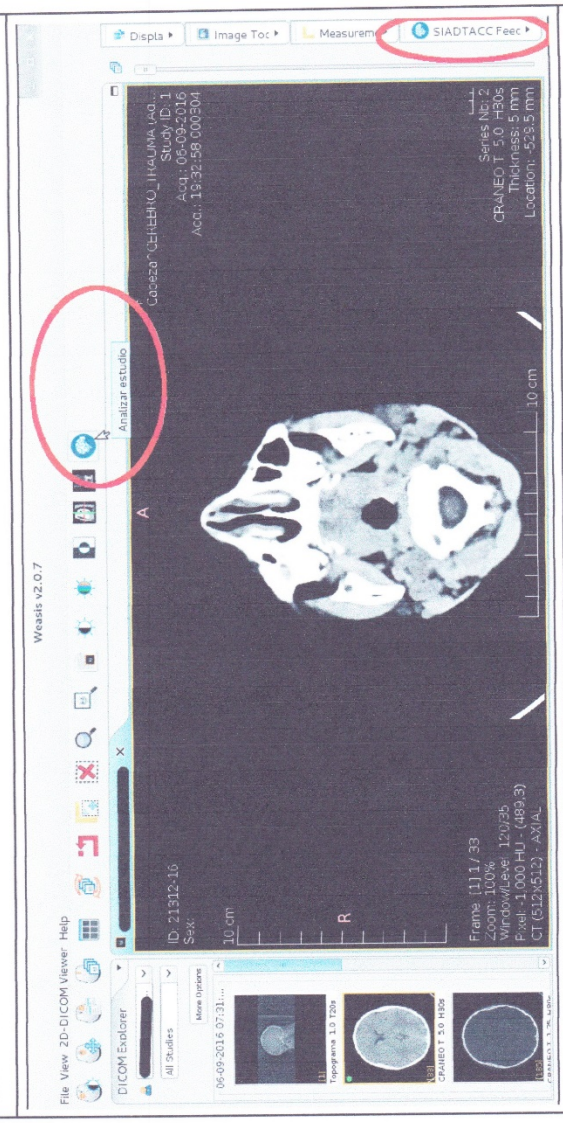
Nombre de observador	Jasmín Orquídea Aquino Flores.
Unidad	Rayos X.
Cargo:	Médico Radiólogo.

El presente documento tiene el objetivo de dar a conocer la interfaz de SIADTACC a la cual tendrá acceso el usuario. Además se busca la recolección de observaciones que posteriormente serán evaluadas y solventadas de acuerdo a la factibilidad de cada caso.

Solicitamos que indique sus respectivas observaciones a cada interfaz por medio de texto o anotaciones sobre las mismas.

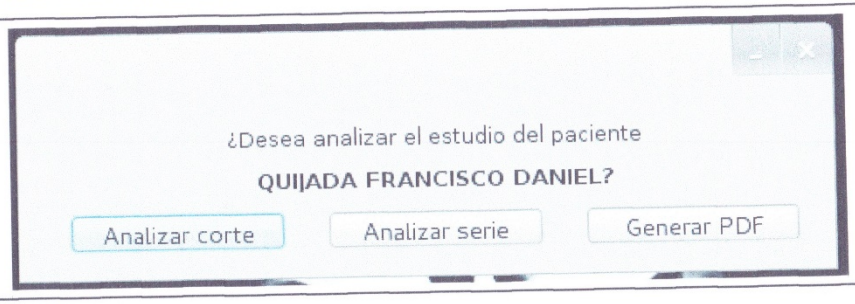
Hospital Nacional San Rafael, Santa Tecla
21 de Diciembre de 2016

INTEGRACIÓN DEL PLUGIN



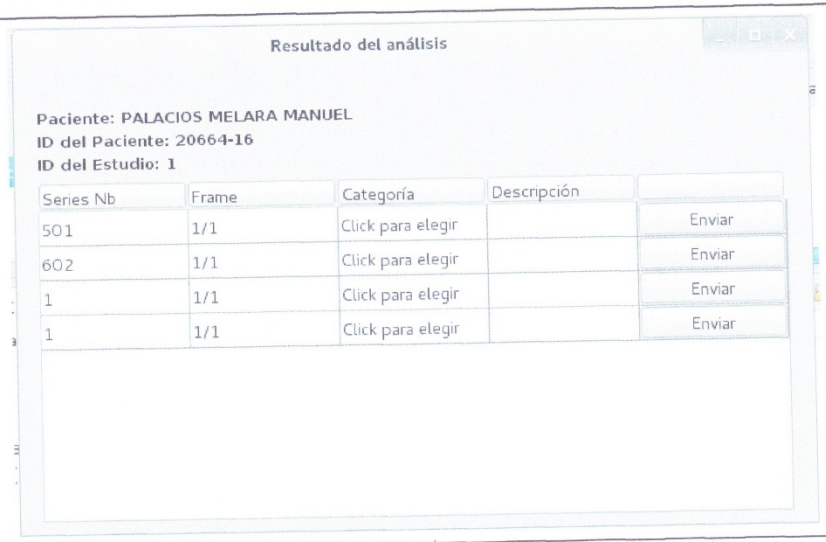
OBSERVACIONES:

OPCIONES DE ANÁLISIS



OBSERVACIONES:

ANALIZAR SERIE

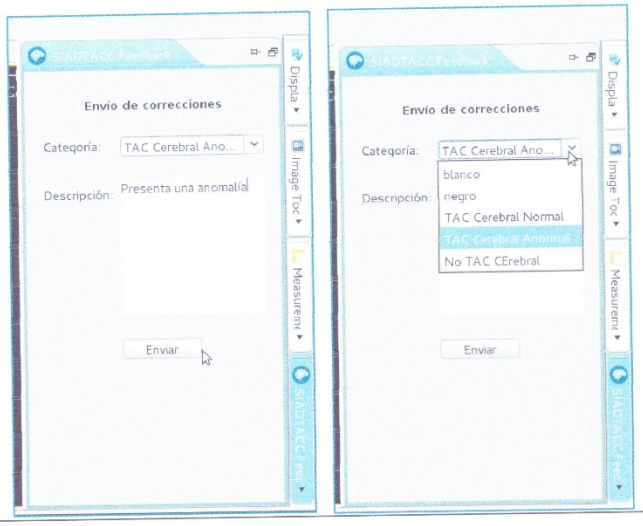


OBSERVACIONES: Me gustaría que pudiá dar un anuncio de alarma en la serie patológica para que su uso sea de utilidad.

AVISOS

OBSERVACIONES:

ENVIO DE SUGERENCIAS



OBSERVACIONES:

Observaciones hechas por el Ing. Carlos Juan Martín Pérez

PRESENTACIÓN PRELIMINAR

SISTEMA INFORMÁTICO PARA EL ANÁLISIS DIGITAL DE TOMOGRAFÍAS AXIALES COMPUTARIZADAS CEREBRALES UTILIZANDO REDES NEURONALES ARTIFICIALES COMO APOYO EN EL DIAGNÓSTICO DIFERENCIAL A LOS RADIÓLOGOS DEL MINISTERIO DE SALUD DE LA REPÚBLICA DE EL SALVADOR

SIADTACC

DOCUMENTO DE OBSERVACIONES

Datos del observador.

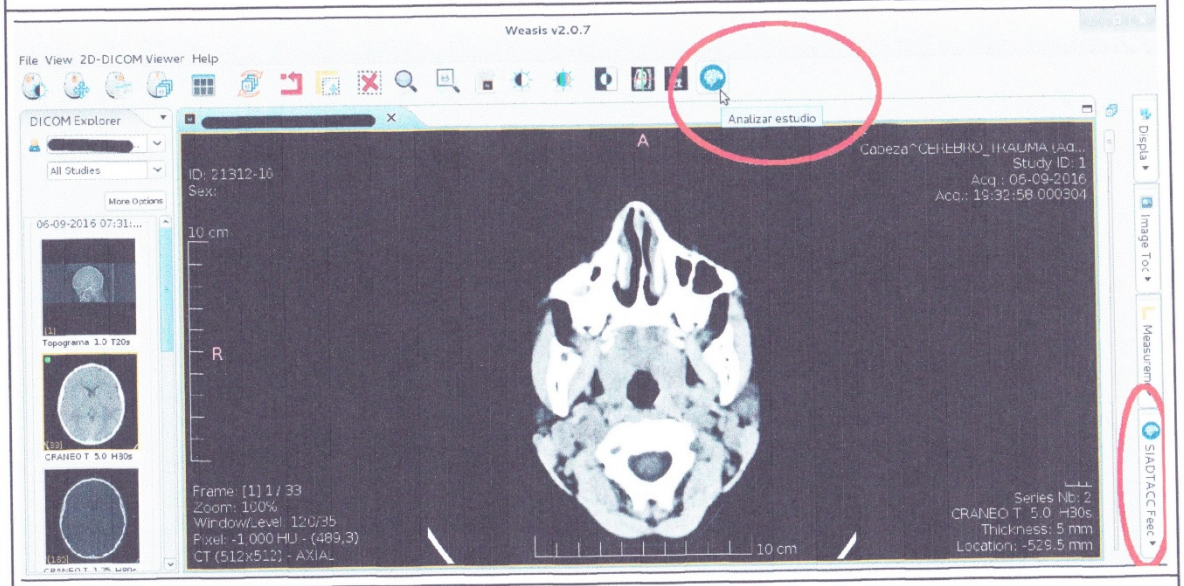
Nombre de observador	CARLOS JUAN MARTÍN PÉREZ
Unidad	DIRECCIÓN DE TIC / MINSAL
Cargo:	Director

El presente documento tiene el objetivo de dar a conocer la interfaz de SIADTACC a la cual tendrá acceso el usuario. Además se busca la recolección de observaciones que posteriormente serán evaluadas y solventadas de acuerdo a la factibilidad de cada caso.

Solicitamos que indique sus respectivas observaciones a cada interfaz por medio de texto o anotaciones sobre las mismas.

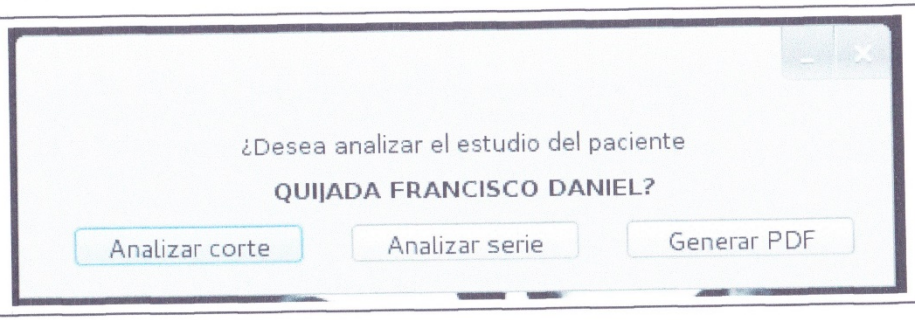
Hospital Nacional San Rafael, Santa Tecla
21 de Diciembre de 2016

INTEGRACIÓN DEL PLUGIN



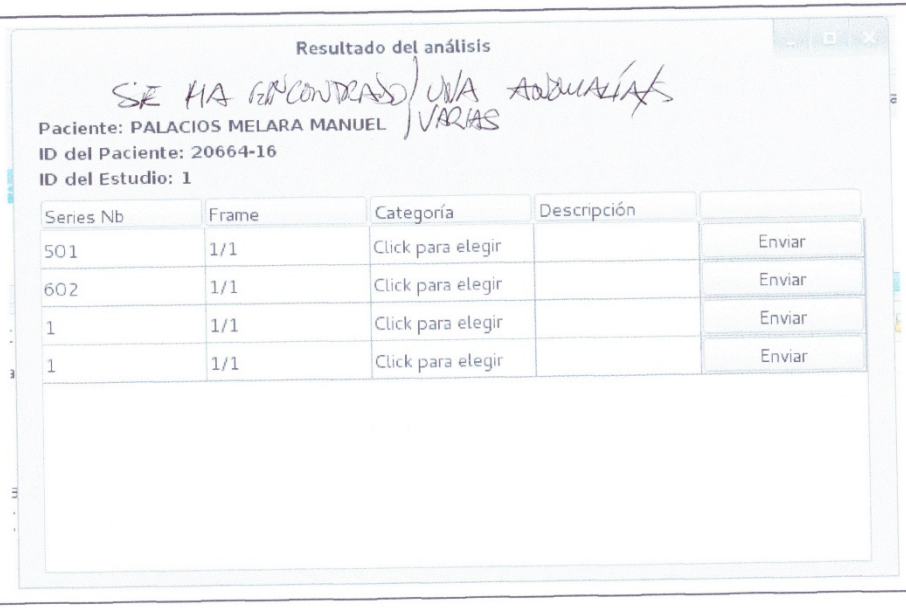
OBSERVACIONES:

OPCIONES DE ANÁLISIS



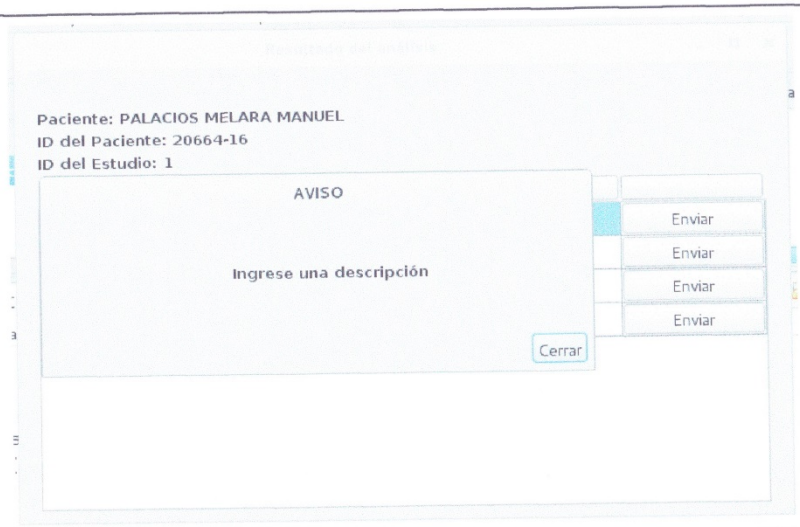
OBSERVACIONES:

ANALIZAR SERIE



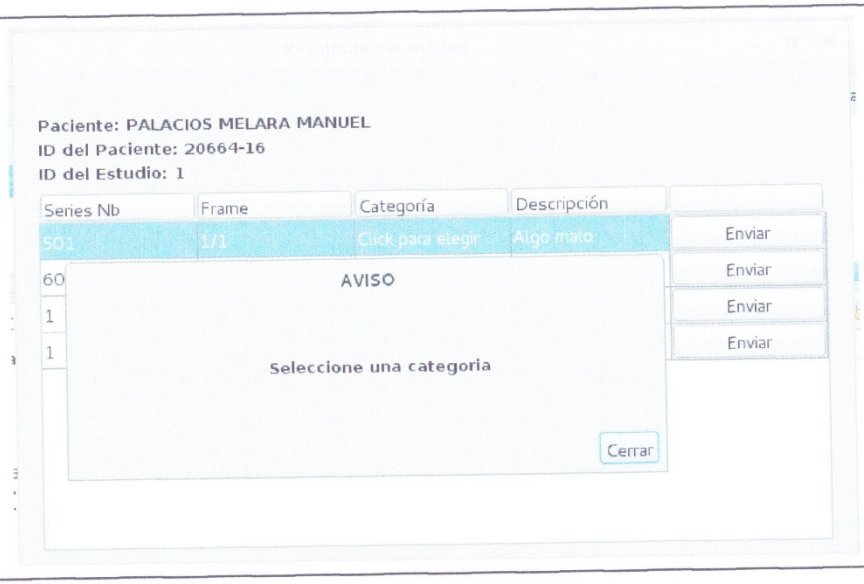
OBSERVACIONES:

AVISOS



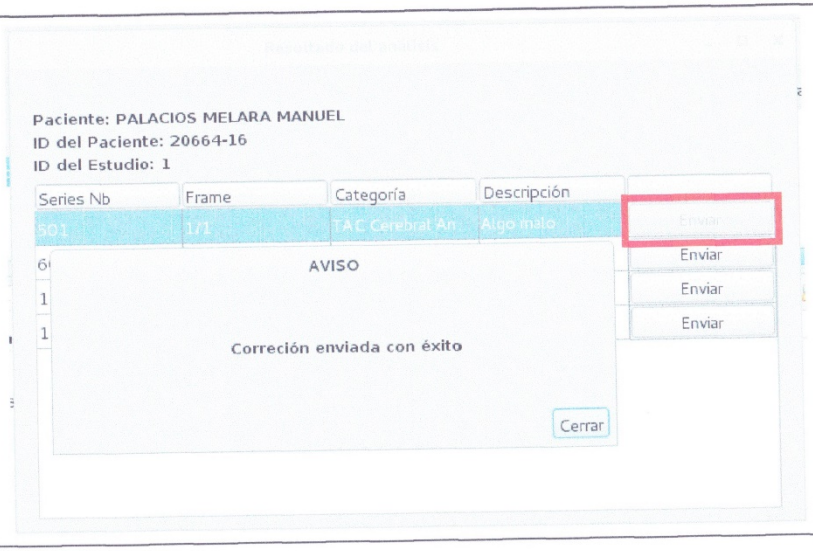
OBSERVACIONES:

AVISOS



OBSERVACIONES:

AVISOS



OBSERVACIONES:

ANEXO 6: Artículo Científico.

APLICACIÓN DE REDES NEURONALES CONVOLUCIONALES PARA CLASIFICACIÓN DE IMÁGENES MÉDICAS

Santiago Colato
Carlos Hernández
Lennin Hernández
Alvaro Hernández
Marlon Menjívar

Escuela de Ingeniería de Sistemas Informáticos
Facultad de Ingeniería y Arquitectura
Universidad de El Salvador

Se llevó a cabo una investigación sobre aplicación de Redes Neuronales Artificiales en Imágenes Médicas cuyos resultados satisfactorios colaboraron al desarrollo de un sistema informático que apoya al diagnóstico médico de Tomografías Axiales Computarizadas cerebrales en la red hospitalaria pública de El Salvador.

Introducción

Desde los inicios de la informática, se busca la automatización de procesos tediosos, delicados y peligrosos para el ser humano. Esta búsqueda encaminó al desarrollo de estudios y posteriores aplicaciones de Inteligencia Artificial. Las Redes Neuronales Artificiales son una emulación del funcionamiento del cerebro humano que permiten a una máquina aprender de una forma precisa.

Desde su introducción las Redes Neuronales Artificiales han contribuido en la resolución de problemas de distintos caracteres desde inteligencias artificiales de videojuegos hasta inteligencias capaces de conducir vehículos sin asistencia.

Este proyecto busca utilizar las redes neuronales convolucionales, como la herramienta para realizar clasificaciones de tomografías axiales computarizadas cerebrales, para mejorar tiempos de atención a pacientes, y de manera indirecta mejorar calidad de los servicios a éstos en el área de radiología de la red hospitalaria del Ministerio de Salud de El Salvador (MINSAL).

Para cumplir con ese objetivo se ha decidido utilizar como base la Red neuronal desarrollada por Alex Krizhevsky, Ilya Sustskever y Geoffrey E. Hinton descrita en el artículo “ImageNet Classification with Deep Convolutional Neural Networks” popularmente conocida por AlexNet, red que será entrenada con un conjunto de datos brindados por el MINSAL.

El conjunto de datos utilizados para el entrenamiento inicial de la red neuronal consiste en un aproximado de 50,000 imágenes de tomografías cerebrales. Y el framework escogido para la creación de la red neuronal es caffe.

Redes neuronales artificiales (RNA)

A grandes rasgos, las redes neuronales son sistemas artificiales basados en el funcionamiento del sistema nervioso humano. La premisa básica consiste en emular la capacidad de aprendizaje de tal forma que la estructura que forma la RNA sea capaz de aprender a identificar patrones de asociación entre los valores de un conjunto de variables que conforman las entradas y un grupo de conclusiones que se consideran como resultado de estas variables (salidas).

Desde un punto de vista de implementación, la RNA consiste en un grupo de elementos procesales llamados nodos, que simulan a las neuronas, estos están interconectados por medio de un entramado de relaciones con pesos, análogas al concepto de conexiones sinápticas en el sistema nervioso. Esta concepción puede ser asimilada como un grafo, estructura integrada por un conjunto de nodos (vértices) y de conexiones (links) entre los mismos.

La forma más habitual de representación de un grafo caracteriza a todos los nodos a través de círculos y a las conexiones como líneas o flechas, de esta manera una buena representación de una RNA sería la siguiente:

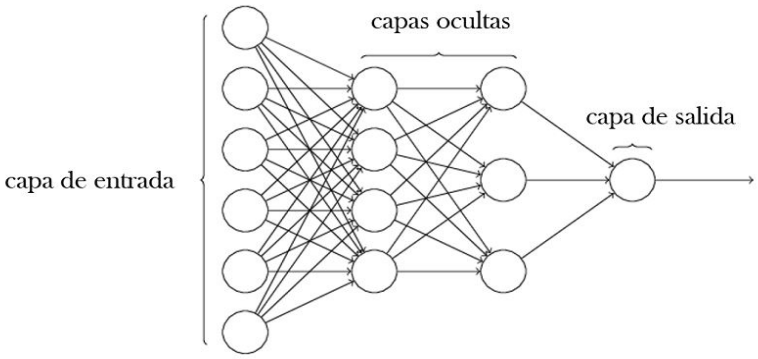


Fig. 1. Representación de una RNA

La capa de más a la izquierda es llamada capa de entrada y cada vértice (neurona) de esta capa se conoce como neurona de entrada. La capa de más a la derecha conocida como capa de salida, contiene a las neuronas de salida, en este caso única. Las capas del medio son conocidas como capas ocultas generalmente siendo más de una.

A este grafo se le añaden las siguientes propiedades importantes que lo vuelven una RNA:

- A cada nodo se le asocia una variable de estado.
- A cada conexión entre dos nodos se le asocia un peso.
- A cada nodo se le asocia un umbral de disparo.
- Para cada nodo se define una función que, dependiendo de los pesos de sus conexiones, del umbral y de las entradas respectivas proporciona el nuevo estado del nodo.

Finalmente, y basándose en la representación en modo de grafo se concluye:

- Un nodo representa una neurona.
- Una conexión representa una sinapsis.
- Una neurona de entrada es aquella sin conexiones de entrada.
- Una neurona de salida es aquella sin conexiones de salida.
- Las neuronas que no son de entrada ni de salida se conocen como neuronas ocultas.
- Una RNA es unidireccional si no presenta bucles cerrados de conexiones.
- Una RNA es recurrente si el flujo de entradas y salidas puede encontrar un bucle de atrás hacia adelante.

Características

A pesar de su simplicidad, en comparación a los modelos biológicos reales, las RNAs conservan algunas propiedades características del cerebro biológico entre las que destacan:

- Habilidad para aprender.
- Capacidad de continuar funcionando aceptablemente a pesar de que se produzcan deterioros o fallos en sus componentes.

Ventajas

- Aprendizaje adaptativo: Las RNAs aprenden a llevar a cabo ciertas tareas mediante un entrenamiento con ejemplos ilustrativos, tales como diferenciar patrones. Estas son adaptables debido a la capacidad de autoajuste de los elementos procesales que componen el sistema y son dinámicas pues son capaces de estar constantemente cambiando para adaptarse a las nuevas condiciones.
- Auto-organización: Las RNAs emplean su capacidad de aprendizaje adaptativo para auto organizar la información que reciben durante el aprendizaje o la operación. Mientras que el aprendizaje es la modificación de cada elemento procesal, la auto organización consiste en la modificación de la red neuronal completa para llevar a cabo un objetivo específico.
- Tolerancia a fallos: La razón por la que las redes neuronales son tolerantes a los fallos es que tienen su información distribuida en las conexiones entre neuronas, existiendo cierto grado de redundancia en este tipo de almacenamiento. Las RNAs pueden aprender a reconocer patrones con ruido, distorsionados o incompletos. Esta es una tolerancia a fallos respecto a los datos
- Operación en tiempo real: La mayoría de las RNAs pueden operar en un entorno de tiempo real, la necesidad de cambio en los pesos de las conexiones o entrenamiento, luego de la estabilización, es mínimo

Aplicaciones de las RNAs

Las RNAs proveen soluciones innovadoras a problemas que los sistemas informáticos convencionales o bien no pueden dar solución o la solución que proporcionan no es aceptablemente eficiente. La mayoría de estos problemas están relacionados con reconocimiento de patrones en una serie de datos, reconstrucción de patrones partiendo de datos incompletos o datos distorsionados.

- Clasificación: Un clasificador debe ser entrenado para que cuando reciba como entrada una variación ligera del patrón, pueda ser reconocida correctamente sin problemas.
- Regeneración de patrones: Consiste en que, a partir de un conjunto de datos incompletos, la RNA pueda obtener una conclusión a la que se llegaría si el conjunto de datos estuviese completo, en otras palabras, consiste en completar el patrón original a partir de un conjunto de datos parcial.

Redes Neuronales Convolucionales

Estas redes utilizan como base las redes neuronales artificiales, puesto que están compuestas de neuronas con pesos aprendidos y todo su funcionamiento general es el mismo. El cambio de estas redes es debido a que se asume que las entradas son imágenes y lo que nos permite realizar algunas operaciones a estas para enriquecer la arquitectura de la red.

El problema de utilizar redes neuronales con imágenes es que no es tan eficiente, puesto que cada pixel en la imagen se traduce a una neurona de entradas, es decir que para una imagen normal de 3 canales (RGB) de 512 x 512 pixeles se tendría que programar un total de 786,432 neuronas de entrada, por lo que en la capa posterior, cada neurona se deberá

conectar con las 786,432 neuronas anteriores, es decir; si la segunda capa tuviera 20 neuronas, tendría que controlarse un total de 15,728,640 pesos diferentes, las redes neuronales no tienen un funcionamiento eficiente con las imágenes.

Las redes neuronales convolucionales se aprovechan de la realidad que las entradas son imágenes, obligando que su arquitectura trabaje de forma más eficiente; para ello utiliza funciones de convolución para obtener imágenes más pequeñas y con información más valiosa para la red, como pueden ser bordes, círculos, cambios de luz, etc. Se puede ver en la siguiente imagen la aplicación de un filtro convolucional para obtener ejes o líneas.

Fig. 2. Aplicación de filtro en una convolución.

La aplicación de estas funciones le permite a la red neuronal la búsqueda de características especiales en una imagen, para facilitar la clasificación de la misma, es decir; si en una imagen hay un circulo es probable que sea una llanta, y un rectángulo es posible que sea un vidrio; la combinación de estas dos características genera la posibilidad que la imagen sea de un vehículo.

Tipos de capas utilizados

- **Capa de entrada:** Contiene la imagen.
- **Capa convolucional:** Utiliza filtros pequeños (3x3, 4x4 o 5x5) que al convolucionales con la imagen, generan imágenes con información especial.
- **Capa de pooling**: Se encargan de disminuir el tamaño de la imagen, se pueden encontrar de 2 tipos:
    - Máximos: disminuye la cantidad de pixeles, generando un pixel con el valor del máximo pixel local.
    - Promedios: disminuye la cantidad de pixeles, generando un pixel con el valor del promedio de los pixeles locales.

Arquitecturas convolucionales conocidas

- **LeNet:** Primera aplicacion de redes neuronales convolucionales, realizada por Yann LeCUnn, utilizada para reconocimiento de dígitos escritos a mano.
- **AlexNet:** Basada en la arquitectura LeNet, pero con mayor profundidad y utilizando multiples GPU para mejorar la velocidad de entrenamiento, utilizada para la clasificación de imágenes generales.
- **GoogleNet:** Mejora realizada sobre AlexNet, disminuye la cantidad de parámetros aplicando pooling promedio al inicio de la red.

Framework caffe

Caffe es un framework de aprendizaje profundo, desarrollado para ser expresivo, rápido y modular. Fue desarrollado por el Centro de Visión y Aprendizaje de Berkeley (BVLC) y por contribuyentes de la comunidad.

Caffe provee de un grupo de herramientas para el entrenamiento, pruebas, mejoras y despliegue de modelos con documentación bien realizadas para todas esas tareas; es además, ideal para desarrolladores que desean aprender sobre inteligencia artificial. Sus principales características son:

- **Modular:** Es fácilmente extensible, permite crear nuevos tipos de formatos de datos, capas de red, funciones de costos, etc.
- **Lenguaje declarativo:** Las redes se definen en un archivo de texto, el cual contiene especificaciones de la red. No es necesario implementar ni codificar para generar una red neuronal desde cero.
- **Soporte de pruebas:** Cada módulo de café tiene una batería de pruebas.
- **Soporte a python y Matlab:** Se pueden crear aplicaciones y extensiones usando código de python y Matlab.
- **Modelos de referencia pre-entrenados:** Existen varios modelos de red neuronal que se pueden utilizar sin hacer configuración.

Clasificación de imágenes médicas

Formato DICOM

DICOM (Digital Imaging and Communications in Medicine) es el estándar que define los métodos para la transferencia de imágenes médicas para diagnóstico y la información asociada a ellas, entre equipos de imagenología y sistemas de fabricantes distintos.

El contenido del estándar DICOM va más allá de la definición del formato de intercambio de imágenes, sus alcances principales son:

- Estructuras de datos (formatos) para imágenes médicas y datos relacionados.
- Servicios orientados a red, como: Transmisión de imágenes, búsqueda de imágenes, impresión y modalidades de integración entre un sistema PACS y un sistema general de información de un hospital (RIS).
- Formatos para intercambio entre medios de almacenamiento.
- Requerimientos de conformidad de los equipos y aplicaciones.

DICOM cubre todas las necesidades de un PACS, las cuales involucran el almacenamiento, transmisión e impresión de imágenes médicas. De esta forma se integran todas las máquinas que forman un PACS, desde los equipos médicos encargados de la obtención de

imágenes, hasta las computadoras usados por el personal clínico para visualizar las imágenes.

El formato DICOM define cuatro elementos (Jiménez, 2006):

- Preámbulo y prefijo identificativo del archivo.
- Meta-cabecera.
- Cabecera.
- Imagen.

El preámbulo tiene un tamaño fijo de 128 bytes y contiene información sobre el nombre de la aplicación usada para crear el archivo, o información que permita a aplicaciones acceder directamente a los datos de la imagen almacenada en éste. El prefijo consiste en cuatro bytes que contienen una cadena de caracteres DICOM. Esta cadena debe estar codificada siempre con las letras en mayúscula y usando el conjunto de caracteres ISO 8859 G0. El propósito de este prefijo es permitir a las implementaciones diferenciar si un archivo es DICOM o no.

La cabecera y la meta-cabecera de un archivo DICOM consisten en una serie de campos con toda la información necesaria sobre la imágen en cuestión, incluyendo la propia imagen. Al conjunto de toda la información codificada sobre un campo se le conoce con el nombre de Elemento de Datos. Así, tanto la cabecera como la meta-cabecera de un archivo DICOM consisten en una sucesión de elementos de datos.

Un elemento de datos está constituido por los campos:

- **Etiqueta del Elemento de Datos:** Sirve para identificar cada elemento de datos de forma unívoca. Esta etiqueta está constituida por un número de grupo y un número de elemento. El elemento de datos se suele representar como un vector de dos posiciones, siendo la primera el número de grupo, y la segunda el número de elemento, en hexadecimal con cuatro dígitos. Por ejemplo, si el número de grupo es ocho y el número de elemento es doce, la etiqueta será (0008,000C).
- **Representación del Valor:** Indica la forma en que se codifica el valor del elemento. Por ejemplo, puede estar codificado como una cadena de caracteres o un entero sin signo.
- **Longitud del Valor:** Como su nombre indica, es la longitud del campo valor.
- **Valor:** Codificado según la Representación y Longitud del Valor.

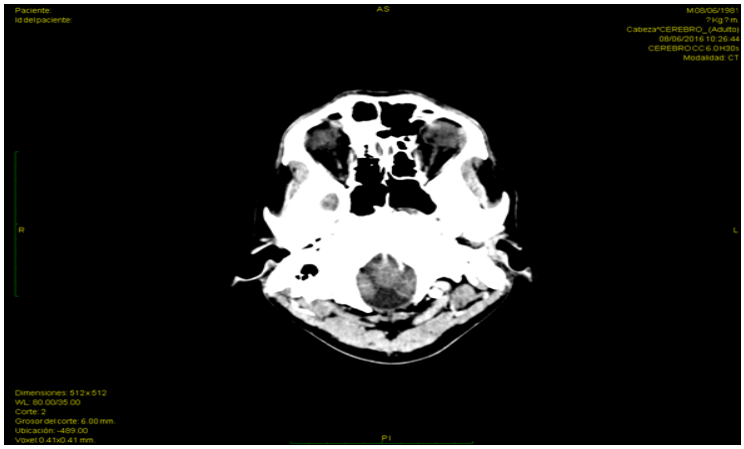


Fig. 3. Imagen almacenada en un archivo DICOM

Diagnóstico asistido por computadora

El diagnóstico asistido por computadora (CAD) se define como el proceso de diagnóstico que realiza un radiólogo, teniendo en consideración el resultado de un análisis cuantitativo que hace una computadora sobre imágenes médicas. Según esta definición el sistema informático que denominamos sistema CAD facilita una segunda opinión que permite enfocar la atención del radiólogo sobre ciertas zonas sospechosas de la imagen, mejorando así el rendimiento y la consistencia del diagnóstico y reduciendo el tiempo de lectura de dichas imágenes.

Los conceptos de diagnóstico automatizado por computadora y el diagnóstico asistido por computadora están claramente presentes en la actualidad. Una diferencia importante entre ellos es la forma en que se utiliza la salida provista por la computadora en el proceso de diagnóstico. Con CAD, el radiólogo utilizará el resultado como una segunda opinión, son los radiólogos quienes toman las decisiones finales. Por lo tanto, para algunos casos los radiólogos pueden estar de acuerdo o no, con los resultados. Sin embargo, para otros casos en los que el radiólogo tiene alguna duda, se espera que la decisión final se pueda mejorar mediante el uso de estos resultados.

El nivel de rendimiento del equipo no tiene que ser igual o mayor que el de los radiólogos. Con CAD, la ganancia potencial es debido al efecto sinérgico obtenido mediante la combinación de la capacidad y experiencia del radiólogo y la potencia de la computadora. Debido a estos beneficios multiplicativos, el CAD actualmente ha llegado a ser ampliamente utilizada en situaciones clínicas prácticas. (Kunio, 2007)

Con el diagnóstico automatizado el nivel de rendimiento de los resultados de la computadora está obligado a ser muy alto. Si la precisión para la detección de lesiones es inferior a la precisión media de los médicos, sería difícil justificar el uso de un equipo de diagnóstico automatizado. Además, está presente la dificultad de implementar el diagnóstico automatizado si el número de detecciones de falsos positivos es grande, por ejemplo, uno o más falsos positivos por caso, dicha situación implicaría que cada caso debe ser estudiado por un radiólogo y esto ya no cumple con el objetivo del diagnóstico automatizado. (Kunio, 2007)

Aplicación de las RNA en medicina

Las redes neuronales artificiales tienen una amplia aplicación en el ámbito médico en general, existen antecedentes de implementaciones exitosas en áreas específicas de la medicina, tales como:

- **Los sistemas de diagnóstico:** Las RNAs se utilizan ampliamente en sistemas de diagnóstico, por lo general para detectar cáncer y problemas cardíacos. Los beneficios del uso de redes neuronales es que no se ven afectados por factores tales como la fatiga, las condiciones de trabajo y el estado emocional.
- **Análisis bioquímico:** Existe una amplia variedad de aplicaciones de química analítica. Por ejemplo, se han utilizado para analizar muestras de sangre y orina, con el fin de determinar niveles de glucosa y detectar condiciones patológicas tales como la tuberculosis.
- **Análisis de imagen:** Las aplicaciones de las RNAs en esta área incluyen la detección de tumores en ultrasonografías, la clasificación de las radiografías de tórax, identificación del tejido y la clasificación imágenes de resonancia magnética (MRI), la determinación de la edad ósea a partir de imágenes de rayos X, etc.
- **Desarrollo de fármacos:** Las RNAs son utilizadas como herramientas en el desarrollo de fármacos para tratar el cáncer, el SIDA y en el proceso de modelado biomolecular.

Solución al problema de clasificación de imágenes.

Arquitectura RNA

Para resolver el problema se utilizó la red neuronal descrita en [3] “ImageNet Classification with Deep Convolutional Neural Network” popularmente conocido como AlexNet, que utiliza 5 capas convolucionales y algunas capas de pooling para disminución de tamaños. Este modelo fue escogido por que presento una buena profundidad en las convoluciones y por permitir un gran número de clasificaciones (+4000) clasificaciones.

Luego de 360,000 iteraciones se obtuvo una precisión de 65% de aciertos en la muestra de prueba compuesta de 15,000 tomografías. Para alcanzar un mayor porcentaje de precisión se necesitan de más clasificaciones, más muestras por clasificación y un entrenamiento con más iteraciones.

Arquitectura Sistema

El sistema se compone de 2 partes:

- **El Backend:** Se encarga de obtener las imágenes, analizarlas y clasificarlas a petición del cliente. Se encarga de exponer la red neuronal utilizando servicios web para disparar el análisis de una o más imágenes, así como provee de un servicio web para almacenar correcciones hechas por el usuario. Estas correcciones son muy importantes porque son utilizadas cuando se hace un entrenamiento de la red neuronal.

- **El Frontend:** Esta parte del sistema se comunica con el usuario, permite ver y explorar las tomografías, y facilita la comunicación del usuario con la red neuronal

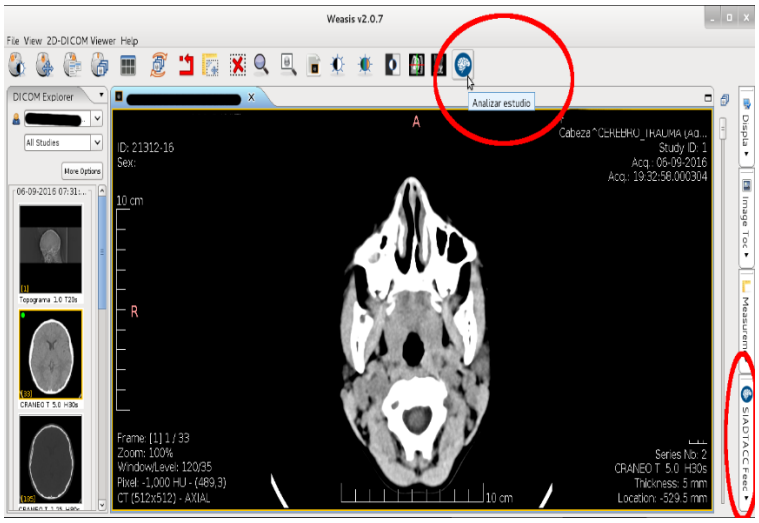


Fig. 4. Visor de tomografías con el cliente de la RNA