Universidad de las Ciencias Informáticas

Facultad 1

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas.

Herramienta para la detección de arritmias cardíacas en electrocardiogramas

Autor:

Javier Alejandro Reyes Góngora

Tutores:

DrC. Arturo Orellana García

Ing. Emilio Enrique Cardero Álvarez

La Habana, septiembre de 2020

Ser culto es el único modo de ser libre.

José Martí

Declaración de autoría

Declaro ser el único autor del trabajo de diploma “Herramienta para la detección de arritmias cardíacas en electrocardiogramas”, concedo a la Universidad de las Ciencias Informáticas y en especial al Centro de Informática Médica la autorización a hacer uso del mismo en su beneficio.

Para que conste firmo el presente documento a los 12 días del mes de octubre del año 2020.

Javier Alejandro Reyes Góngora

_____

Firma del autor

DrC. Arturo Orellana García

_____

Firma del tutor

Ing. Emilio Enrique Cardero Álvarez

_____

Firma del tutor

Datos de contacto:

Dr.C. Arturo Orellana García: aorellana@uci.cu graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en el año 2012. Se desempeña como líder del Grupo de Investigación de Minería de procesos y Asesor de Capacitación, Desarrollo e Investigación del Centro de Soluciones de Informática Médica. Ha liderado proyectos I+D+i de desarrollo de componentes de *software* a partir de minería de procesos para el análisis de procesos de negocio del entorno hospitalario. Investiga la Ingeniería de comportamiento, la medicina de precisión y el procesamiento de imágenes médicas. Tutora varias tesis de grado, maestrías y doctorados enfocados al análisis de procesos de negocio, la informática médica y otras áreas del conocimiento. Obtuvo el grado de Máster en Informática Aplicada en 2015 desarrollando una herramienta informática basada en técnicas de minería de procesos para identificar problemas en la ejecución de procesos de negocio. Doctor en Ciencias Técnicas desde 2016 presentando un modelo computacional para la detección de variabilidad en procesos de negocio del entorno sanitario aplicando minería de procesos.

Ing. Emilio Enrique Cardero Álvarez: eecardero@uci.cu

Javier Alejandro Reyes Góngora: jagongora@estudiantes.uci.cu

Resumen

La detección automatizada de arritmias en electrocardiogramas se ha vuelto un tema popular en los últimos 5 años, debido a los avances en la teoría que posibilita hacerlo y en las capacidades de hardware existentes. Estos programas analizan datos provenientes de un electrocardiograma y son capaces de interpretarla, detectar posibles arritmias y así ayudar con el diagnostico de estas. La presente investigación tuvo como objetivo desarrollar una herramienta que permita la detección de latidos anómalos en un electrocardiograma, posibilitando así la detección de arritmias en este. Para el desarrollo de la investigación se emplearon los métodos científicos: teóricos y empíricos. Se seleccionó un algoritmo, luego de hacer una búsqueda en trabajos actuales sobre el tema, que permitiera la detección de latidos anómalos en un electrocardiograma, y se adaptó para la implementación de la solución. Para la implementación de la propuesta de solución se utilizaron los lenguajes de programación C# y Python, así como los frameworks Tensorflow.Net, .Net Framework y la herramienta Visual Studio 2019. Además, para el diseño de la propuesta se utilizó Enterprise Arquitect y el lenguaje UML. La metodología Variación de AUP para la UCI, guió el proceso de desarrollo de software. Se desarrolló una herramienta que facilita la detección de arritmias cardiacas en un electrocardiograma. La herramienta clasifica electrocardiogramas en normales o anormales. Para evaluar el objetivo planteado se realizaron pruebas de software validando el correcto funcionamiento de la propuesta de solución.

Palabras claves: arritmias, clasificación, detección automatizada de arritmias en electrocardiogramas, electrocardiograma, diagnostico

Abstract

The automated detection of arrhythmias in electrocardiograms has become a popular topic in the last 5 years, due to advances in the theory that makes it possible and in the existing hardware capabilities. These programs analyze data from an electrocardiogram and are capable of interpreting it, detecting possible arrhythmias and thus helping with their diagnosis. The present research aimed to develop a tool that allows the detection of abnormal beats in an electrocardiogram, thus enabling the detection of arrhythmias in it. For the development of the research, scientific methods were used: theoretical and empirical. An algorithm was selected, after doing a search in current works on the subject, that would allow the detection of abnormal beats in an electrocardiogram, and it was adapted for the implementation of the solution. For the implementation of the solution proposal, the programming languages C # and Python were used, as well as the frameworks Tensorflow.Net, .Net Framework and the Visual Studio 2019 tool. In addition, for the design of the proposal, Enterprise Architect and the UML language. The AUP Variation methodology for the ICU guided the software development process. A tool was developed that facilitates the detection of cardiac arrhythmias on an electrocardiogram. The tool classifies EKGs as normal or abnormal. To evaluate the proposed objective, software tests were carried out to validate the correct operation of the proposed solution.

Key words: arrhythmias, classification, automated detection of arrhythmias in electrocardiograms, electrocardiogram, diagnosis

Introducción

Las enfermedades cardiovasculares (ECV) son un grupo de trastornos del corazón y los vasos sanguíneos e incluyen enfermedad coronaria, -enfermedad de los vasos sanguíneos que irrigan el músculo cardíaco; enfermedad cardíaca reumática - daño en el músculo cardíaco y las válvulas cardíacas debido a la fiebre reumática, causada por bacterias estreptocócicas; cardiopatía congénita - malformaciones de la estructura cardíaca existente al nacer, y más.

Las ECV son la causa número 1 de muerte a nivel mundial: mueren más personas anualmente por ECV que por cualquier otra causa, y se estima que 17.9 millones de vidas cada año representan el 31% de todas muertes mundiales De estas muertes, el 85% se deben a ataques cardíacos y accidentes cerebrovasculares. Las personas con enfermedades cardiovasculares o con alto riesgo cardiovascular (debido a la presencia de uno o más factores de riesgo, como hipertensión, diabetes, hiperlipidemia o enfermedad ya establecida) necesitan detección y tratamiento precoces utilizando asesoramiento y medicamentos, según corresponda. (1)

En Cuba, las enfermedades cardíacas han sido históricamente una de las principales causas de muerte, generalmente tienen el mayor número.(2) A pesar de tener la mayor densidad de médicos en todo el mundo , la muerte por enfermedades del corazón sigue siendo muy significativa en la población cubana, como se aprecia en la Figura 1.(3)

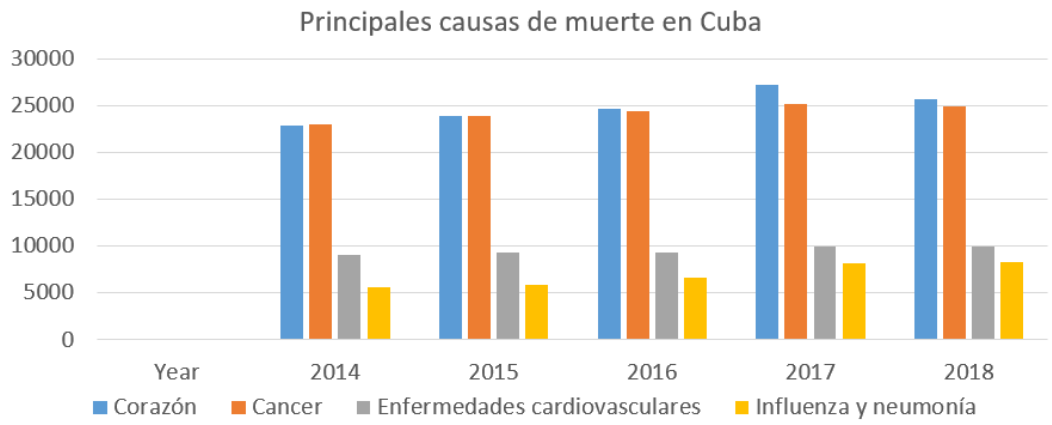


Figura 1. Principales causas de muerte en Cuba

La cobertura debe ir acompañada de una atención eficiente al paciente, que incluye tiempos de espera cortos. Desafortunadamente, las listas de espera aún pueden ser largas en Cuba, lo que lleva a la insatisfacción del paciente y a la disminución de la eficiencia. Esos tiempos de espera pueden reducirse si se disminuye el tiempo que un médico tiene que pasar con cada paciente, por ejemplo, dándole mejores herramientas para aliviar la carga del proceso de diagnóstico.

El electrocardiograma (ECG) ha sido, desde su creación, una herramienta predeterminada utilizada para diagnosticar una gran mayoría de anomalías cardíacas. Los avances recientes han extendido la importancia del ECG. Es una herramienta de diagnóstico no invasiva, económica y bien establecida. Representa los cambios de la actividad eléctrica del corazón a lo largo del tiempo y contiene información fisiológica esencial que se utiliza ampliamente para analizar la función cardíaca. Las señales de ECG son periódicas, porque están compuestas por una secuencia de ondas que se repiten periódicamente en el tiempo: una onda P, luego ondas Q, R y S (que forman el complejo QRS) y una onda T, como se muestra en la Figura 2.

Es una prueba vital para determinar la presencia y la gravedad de la isquemia miocárdica aguda, localizar sitios de origen y vías de taquiarritmias, evaluar las opciones terapéuticas para pacientes con insuficiencia cardíaca e identificar y evaluar pacientes con enfermedades genéticas propensas a arritmias. Los logros en fisiología y tecnología han ampliado las posibilidades de extraer más información sobre la actividad eléctrica del corazón del ECG que extenderá aún más estas aplicaciones clínicas.(4)

Los recientes avances en informática y un mejor hardware han hecho posible que los ECG sean interpretados no solo por humanos, sino también por máquinas. Muchas veces, los ECG son muy largos y el médico tiene que leerlos latido por latido, un proceso largo y agotador, y también bastante complicado para los médicos sin experiencia. Este proceso puede ser asistido, e incluso en

un futuro cercano, reemplazado por sistemas capaces de interpretar la información del ECG y clasificarla correctamente. Varios estudios han explorado esta posibilidad, clasificando los ECG en numerosas clases utilizando técnicas de Machine Learning, como redes neuronales y kNN.(5)

La mayor parte de estos son de autores extranjeros, y aunque existen algunos cubanos dentro de esta área, no se enfocan en la automatización del proceso de clasificación de electrocardiogramas, ni son utilizados regularmente en las áreas de atención de salud. A pesar de existir estos estudios, muy pocos culminan en una herramienta utilizable para el uso de un profesional de la salud.

Al aplicar estas técnicas, el trabajo de un médico se puede hacer más rápido, ya sea diagnosticando correctamente al paciente más rápido o simplemente pasando menos tiempo en un paciente sano. Sin embargo, ninguno de estos avances se utiliza regularmente en el sector de la salud de Cuba, ya sea en hospitales o por médicos generales en las comunidades.

A partir de lo antes planteados se identifica como problema a resolver: ¿Cómo contribuir a la gestión de información relacionada con la detección de arritmias cardíacas en electrocardiogramas?

Se define como objeto de estudio el proceso de clasificación de señales de electrocardiograma, enmarca en el campo de acción la clasificación de latidos en electrocardiogramas para la detección de arritmias en Cuba.

Para solucionar a la problemática planteada se define como objetivo general del trabajo de diploma: Desarrollar una herramienta informática para la gestión de información relacionada con la detección de arritmias cardíacas en electrocardiogramas.

Para dar cumplimiento al Objetivo General de la Investigación se plantean las tareas siguientes:

- Determinación de las principales referencias teóricas de la clasificación de latidos basados en ECG.

- Diseñar la herramienta para clasificar los ECG en normales o anormales.
- Implementar la herramienta para clasificar los ECG en normales o anormales.
- Validación del correcto funcionamiento de la herramienta media la definición y ejecución de pruebas.
- Análisis de los resultados del clasificador de latidos, teniendo en cuenta la especificidad, sensibilidad y precisión logradas.

Con la creación de la herramienta presentada en este trabajo, se esperan los siguientes beneficios:

- Ayudar a los médicos experimentados en el proceso de detección de una arritmia en un ECG, y ayudar a los inexpertos a lograr mejores resultados al clasificar.
- Mejorar la atención al paciente al reducir los tiempos de diagnóstico y, por lo tanto, poder tratar o despedir al paciente más rápido

Durante este trabajo se utilizaron varios métodos científicos, tales como:

- Teórico:
  - ✓ Lógico-histórico: Utilizado para determinar los antecedentes, tendencias y regularidades de la detección de arritmia basada en ECG
  - ✓ Sintético-Analítico: Utilizado para determinar las generalidades y la selección de las técnicas de procesamiento que se utilizarán en la elaboración del algoritmo
  - ✓ Modelado: Utilizado para representar gráficamente los diferentes elementos que componen el diseño del algoritmo, además de hacer los modelos y diagramas asociados con su desarrollo.

- Empírico:
  - ✓ Observación: se utiliza como un medio para adquirir conocimiento.

Este trabajo se divide en tres capítulos:

En el capítulo 1 se realiza una visión general del estado del arte, junto con la presentación de las principales referencias teóricas utilizadas en este trabajo. También se presentan brevemente varios métodos utilizados para clasificar los latidos del corazón basados en ECG.

En el capítulo 2 se presenta el diseño y la implementación del algoritmo. Se propone una arquitectura, se modelan los diagramas de clase y se realiza una propuesta de solución.

En el capítulo 3, los resultados se discuten y analizan, evaluando el rendimiento de la aplicación de acuerdo con varias métricas.

Capítulo 1. Fundamentación teórica de la investigación

1.1 Conceptos relacionados con el campo de acción

1.1.1 Electrocardiograma

El ECG es el resultado final de una serie compleja de procesos fisiológicos y tecnológicos. Primero, las corrientes iónicas transmembranales son generadas por flujos de iones a través de las membranas celulares y entre las células adyacentes. Estas corrientes se sincronizan mediante secuencias de activación y recuperación cardíacas para generar un campo eléctrico cardíaco dentro y alrededor del corazón que varía con el tiempo durante el ciclo cardíaco.

Este campo eléctrico pasa a través de muchas otras estructuras, incluidos los pulmones, la sangre y el músculo esquelético, que perturban el campo eléctrico cardíaco. Las corrientes que alcanzan la piel son detectadas por electrodos colocados en lugares específicos en las extremidades y el torso que están configurados para producir derivaciones. (4) Las salidas de estas derivaciones son amplificadas, filtradas y mostradas por una variedad de dispositivos para producir una grabación electrocardiográfica, apreciable en la Figura 2.

La máquina de ECG procesa las señales captadas de la piel por los electrodos y produce una representación gráfica de la actividad eléctrica del corazón del paciente. El patrón básico del ECG es:

- La actividad eléctrica hacia una derivación provoca una desviación hacia arriba.
- La actividad eléctrica alejada de una derivación provoca una desviación hacia abajo.
- Las deflexiones de despolarización y repolarización ocurren en direcciones opuestas.

El patrón básico de esta actividad eléctrica se descubrió por primera vez hace más de cien años. Consta de tres ondas, que se han denominado P, QRS (un complejo de ondas) y T, apreciables en la Figura 2.

La onda P es una pequeña onda de desviación que representa la despolarización auricular.

El intervalo PR es el tiempo entre la primera desviación de la onda P y la primera desviación del complejo QRS.

Las tres ondas del complejo QRS representan la despolarización ventricular. Para los inexpertos, uno de los aspectos más confusos de la lectura de ECG es el etiquetado de estas ondas. La regla es: si la onda inmediatamente después de la onda P es una desviación hacia arriba, es una onda R; si es una desviación hacia abajo, es una onda Q:

- las pequeñas ondas Q corresponden a la despolarización del tabique interventricular. Las ondas Q también pueden relacionarse con la respiración y generalmente son pequeñas y delgadas. También pueden indicar un infarto de miocardio antiguo (en cuyo caso son grandes y anchos)
- la onda R refleja la despolarización de la masa principal de los ventrículos, de ahí que sea la onda más grande
- la onda S significa la despolarización final de los ventrículos, en la base del corazón

El segmento ST, que también se conoce como intervalo ST, es el tiempo entre el final del complejo QRS y el inicio de la onda T. Refleja el período de potencial cero entre la despolarización y la repolarización ventricular.

Las ondas T representan la repolarización ventricular (la repolarización auricular está oscurecida por el gran complejo QRS). (6)

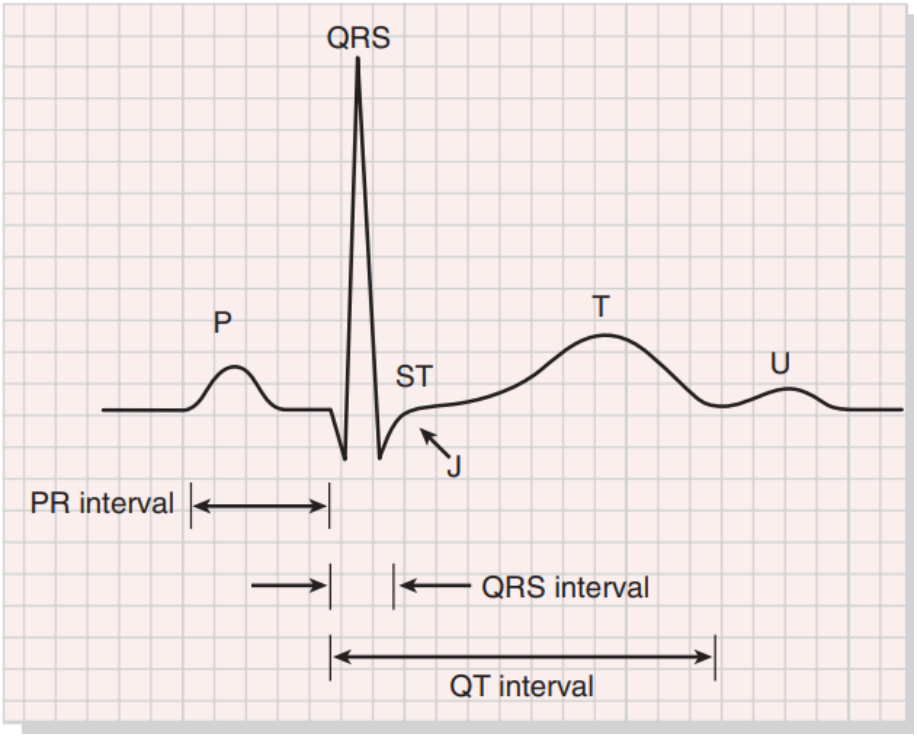


Figura 2. Electrocardiograma normal

En sistemas computarizados, estas señales son digitalizadas, almacenadas y procesadas por el software de reconocimiento de patrones. Los criterios de diagnóstico se aplican, ya sea manualmente o con la ayuda de una computadora, para producir una interpretación.(4)

1.1.2 Arritmia

El término "arritmia" se refiere a cualquier cambio de la secuencia normal de impulsos eléctricos. Los impulsos eléctricos pueden ocurrir demasiado rápido, demasiado lento o erráticamente, haciendo que el corazón lata demasiado rápido, demasiado lento o erráticamente. Cuando el corazón no late correctamente, no puede bombear sangre de manera efectiva. Cuando el corazón no bombea sangre de manera efectiva, los pulmones, el cerebro y todos los demás órganos no pueden funcionar correctamente y pueden cerrarse o dañarse.

Algunas arritmias son tan breves (por ejemplo, una pausa temporal o latidos prematuros) que la frecuencia o el ritmo cardíaco general no se ven muy afectados. Pero si las arritmias duran más, pueden hacer que la frecuencia cardíaca sea demasiado lenta o demasiado rápida o que el ritmo cardíaco sea errático, por lo que el corazón bombea con menos eficacia. La mayoría de las personas con un ritmo cardíaco anormal pueden llevar una vida normal si se diagnostica adecuadamente.(7) (8)

1.1.3 Red neuronal

Una red neuronal es un algoritmo de aprendizaje automático basado en el modelo de una neurona humana. El cerebro humano consta de millones de neuronas. Envía y procesa señales en forma de señales eléctricas y químicas. Estas neuronas están conectadas con una estructura especial conocida como sinapsis. Las sinapsis permiten que las neuronas pasen señales. Una red neuronal artificial consta de miles o incluso millones de nodos de procesamiento simples que están densamente interconectados.

La mayoría de las redes neuronales de hoy en día están organizadas en capas de nodos, y están "alimentadas", lo que significa que los datos se mueven a través de ellas en una sola dirección. Un nodo individual puede estar conectado a varios nodos en la capa debajo de él, desde la cual recibe datos, y a varios nodos en la capa por encima, a los cuales envía datos.

Las redes neuronales son un medio de aprendizaje automático, en el que una computadora aprende a realizar alguna tarea mediante el análisis de ejemplos de capacitación. Por lo general, los ejemplos se han etiquetado a mano de antemano. Un sistema de reconocimiento de objetos, por ejemplo, podría alimentarse con miles de imágenes etiquetadas de automóviles, casas, tazas de café, etc., y encontraría patrones visuales en las imágenes que se correlacionan consistentemente con etiquetas particulares.(9) (10)

1.1.4 Aprendizaje profundo

El aprendizaje profundo es un superconjunto de algoritmos que tienen como modelos base de aprendizaje automático, solo que su estructura es lo suficientemente compleja como para aprender altas abstracciones de los datos procesados. Esto casi siempre se logra agregando capas a las redes para que se llamen “profundas”. El aprendizaje profundo es una tecnología clave detrás de los autos sin conductor, que les permite reconocer una señal de alto o distinguir a un peatón de una farola. Es la clave para el control por voz en dispositivos de consumo como teléfonos, tabletas, televisores y altavoces manos libres.

El aprendizaje profundo está recibiendo mucha atención últimamente. Está logrando resultados que antes no eran posibles. Con él, un modelo de computadora aprende a realizar tareas de clasificación directamente a partir de imágenes, texto o sonido. Dichos modelos pueden lograr una precisión de vanguardia, que a veces supera el rendimiento a nivel humano. Los modelos se entrenan mediante el uso de un gran conjunto de datos etiquetados y arquitecturas de redes neuronales que contienen muchas capas.(11)

1.2 Análisis de las principales herramientas y algoritmos para la detección de anormalidades en el electrocardiograma

La detección de arritmia en los ECG se ha convertido en un tema popular en el transcurso de la última década, y muchos autores han intentado numerosos enfoques para abordar el problema. Sin embargo, la mayor parte de los trabajos publicados se centra en el problema de detectar la arritmia, pero se detienen allí, sin proporcionar una aplicación adicional al conocimiento presentado. Los enfoques encontrados en los algoritmos para la detección de arritmia analizando latidos comparten un grupo de características generales, por ejemplo, el proceso se puede dividir aproximadamente en los siguientes pasos: preprocesamiento de datos, detección de complejos QRS, extracción de características y selección de clasificadores.

En (12), en la etapa de preprocesamiento, se aplica un filtro de media para eliminar el ruido de corriente directa presente en la señal de ECG. Después de eso un filtro de media móvil de 10 puntos es usado para eliminar el ruido de alta frecuencia. Luego un filtro basado en derivadas es usado para eliminar el ruido de baja frecuencia. También se usa un filtro de peine para eliminar la interferencia de la línea de alimentación presente en los datos. Para detector el complejo QRS se usa un algoritmo de Pan-Tompkins. Para la extracción de características, se usa *transfer learning*, (13) adaptando AlexNet (14) al conjunto de datos actual. Después de aplicar un análisis de componentes principales al resultado del paso anterior, utilizan tres redes neuronales profundas diferentes como clasificadores, cada uno con diferentes tamaños de entrada y capas ocultas. Alcanzan más del 85% de precisión en los conjuntos de prueba en los tres casos.

Siguiendo un enfoque muy diferente, (15) hace prácticamente cero el preprocesamiento y utiliza las funciones de base hermitiana para extraer características que luego se utilizarán como entradas para un clasificador kNN, logrando una sensibilidad del 99% y una especificidad del 99.84%.

En (16) se usaron reducción de componentes constantes y reducción de ganancia en la etapa de preprocesamiento, aunque sin detección del complejo QRS, ya que analizan fragmentos de 10 segundos del ECG en lugar de latidos individuales Se utilizó una red neuronal convolucional de 16 capas como clasificador para 17 tipos diferentes de arritmias, obteniendo una precisión general del 95,2%.

(17) Utiliza 2 filtros de mediana y un filtro FIR para la eliminación de ruido, y luego seleccionan manualmente un conjunto de rasgos de cada latido en el electrocardiograma, creando una base de datos nueva a partir de la original. Entrenan una red neuronal de 7 capas ocultas con las características seleccionadas y obtienen una precisión del 99.68% cuando la prueban sobre el conjunto de datos de prueba. Además de técnicas de preprocesamiento similares a las mencionadas hasta ahora, (18) utiliza una red neuronal profunda

como un clasificador con una precisión del 88,6% y llega al punto de desarrollar una GUI basada en Matlab para el sistema.

Para el desarrollo de la herramienta se selecciona parcialmente la estrategia del algoritmo usado por (17) debido a su facilidad de implementación, excelentes resultados y ser de los que menos recursos computacionales requiere a la hora de utilizarse.

Autores cubanos también han abordado temas similares. En (19) se realiza la implementación de varios algoritmos para analizar la variabilidad del ritmo cardíaco para ser usado por la herramienta HRV Station 2.7, descrita en la misma fuente.

Detectan el complejo QRS utilizando los métodos descritos en (20) (21)

Luego mediante una gran variedad de métodos, estadísticos y geométricos, se realiza el cálculo de la variabilidad de la frecuencia cardíaca, todos ellos son incorporados como opciones al sistema informático mencionado anteriormente.

El análisis de este factor está incluido en los algoritmos más avanzados de detección de anomalías en un electrocardiograma usando redes neuronales, aunque es uno de muchos.

En (22) se realiza la detección de apnea del sueño mediante el análisis de un ECG. Realizan un filtrado de mediana deslizante con ventanas de 200 ms y 600 ms en el preprocesamiento.

Se utilizan como rasgos para describir el problema en primer lugar, el área bajo la onda R, el área bajo la onda QR, la variabilidad del ritmo cardiaco (VFC) y la Trasformada Wavelet.

Se obtiene la serie de tiempo de los intervalos RR, de donde se obtienen características estadísticas como la media y la desviación típica, así como rasgos basados en el uso de la Trasformada Wavelet Discreta y un análisis espectral basado en el método de Welch.

Con todos estos rasgos se obtiene un vector de características que se utiliza en el proceso de clasificación, para el cual usan 3 clasificadores distintos (RNA, SVM y Análisis del Discriminante Lineal) y comparan sus resultados, todos teniendo una eficacia cercana al 90%.

Además de los estudios anteriormente mencionados, existen aplicaciones comerciales capaces de realizar la tarea que se propone este trabajo, aunque todas presentan severas limitantes para su uso por los profesionales de la salud cubanos.

1.3 Lenguajes de programación, herramientas y metodologías.

1.3.1 Metodología

El proceso unificado ágil es un enfoque de modelado híbrido creado por Scott Ambler cuando combina el Proceso unificado racional (RUP) con los métodos ágiles (AM). Es un proceso iterativo-incremental que consta de cuatro subprocesos o flujos de trabajo: modelado, implementación, prueba y despliegue. En la Universidad de Ciencias Informáticas, se utiliza una versión de AUP, llamada AUP-UCI, y esta es la metodología seleccionada que se utilizará. Utiliza el modelo CMMI-DEV v1.3 y, a diferencia de AUP, define 3 fases: Inicio, Ejecución, Cierre. La fase de Inicio es el equivalente de la fase de Inicio de la AUP, y Ejecución envuelve los 3 restantes. En Cierre, se discuten y analizan los resultados y se realizan algunos trámites relacionados con el cierre del proyecto.(23) A partir de que la disciplina de Modelado de negocio propone tres variantes a utilizar en los proyectos: casos de uso del negocio, descripción de proceso de negocio y modelo conceptual, y existen tres formas de encapsular los requisitos: casos de uso del sistema, historias de usuario y descripción de requisitos por proceso, surgen cuatro escenarios para modelar el sistema en los proyectos, quedando de la siguiente forma:

Escenario 1: proyectos que modelen el negocio con casos de uso del negocio solo pueden modelar el sistema con casos de uso del sistema.

Escenario 2: proyectos que modelen el negocio con modelo conceptual solo pueden modelar el sistema con casos de uso del sistema.

Escenario 3: proyectos que modelen el negocio con descripción de proceso de negocio solo pueden modelar el sistema con descripción de requisitos por proceso.

Escenario 4: proyectos que no modelen negocio solo pueden modelar el sistema con HU.

Para este trabajo se seleccionó el escenario 2.

1.3.2 Lenguajes

C#

C# es un lenguaje de programación multi-paradigma de propósito general que abarca disciplinas de programación de alcance léxico, imperativo, declarativo, funcional, genérico y orientado a componentes. Fue desarrollado y actualmente mantenido por Microsoft. Es un lenguaje orientado a objetos de tipo seguro que permite a los desarrolladores crear una variedad de aplicaciones seguras y robustas que se ejecutan en .NET Framework. C# se puede usar para crear aplicaciones cliente de Windows, servicios web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de bases de datos. C# admite métodos y tipos genéricos, que proporcionan mayor seguridad y rendimiento de tipo, e iteradores, que permiten a los implementadores de las clases de colección definir comportamientos de iteración personalizados que son fáciles de usar por el código del cliente. Es ampliamente utilizado junto con .Net Framework para desarrollar aplicaciones de Windows (24).

Python

Como se indica en (25) Python es un lenguaje de programación de alto nivel interpretado, orientado a objetos y con semántica dinámica. Sus estructuras de

datos integradas de alto nivel, combinadas con tipeo dinámico y enlace dinámico, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como para usarlo como un lenguaje de scripting o pegamento para conectar los componentes existentes. La sintaxis simple y fácil de aprender de Python enfatiza la legibilidad y, por lo tanto, reduce el costo del mantenimiento del programa. Python admite módulos y paquetes, lo que fomenta la modularidad del programa y la reutilización de código. El intérprete de Python y la extensa biblioteca estándar están disponibles en formato fuente o binario sin cargo para todas las plataformas principales, y se pueden distribuir libremente.

UML

Como se indica en (26) UML, abreviatura de Unified Modeling Language, es un lenguaje de modelado estandarizado que consiste en un conjunto integrado de diagramas, desarrollado para ayudar a los desarrolladores de sistemas y software a especificar, visualizar, construir y documentar los artefactos de los sistemas de software, así como para el modelado de negocios y otros sistemas sin software. El UML representa una colección de mejores prácticas de ingeniería que han demostrado ser exitosas en el modelado de sistemas grandes y complejos. El UML es una parte muy importante del desarrollo de software orientado a objetos y el proceso de desarrollo de software. El UML utiliza principalmente notaciones gráficas para expresar el diseño de proyectos de software. El uso de UML ayuda a los equipos de proyecto a comunicarse, explorar diseños potenciales y validar el diseño arquitectónico del software.

1.3.3 Herramientas

Tensorflow.NET

Según (27) TensorFlow.NET (TF.NET) proporciona un enlace estándar .NET para TensorFlow. (28) Su objetivo es implementar la API completa de Tensorflow en C # que permite a los desarrolladores de .NET desarrollar, entrenar e implementar modelos de Machine Learning con el marco estándar

.NET multiplataforma. TensorFlow es una plataforma de código abierto de extremo a extremo para el aprendizaje automático. Cuenta con un ecosistema integral y flexible de herramientas, bibliotecas y recursos comunitarios que permite a los investigadores impulsar el estado del arte en ML y los desarrolladores crean y despliegan fácilmente aplicaciones basadas en ML.

TensorFlow ofrece múltiples niveles de abstracción, por ejemplo, la API de Keras permite construir y entrenar modelos fácilmente sobre tensorflow. Permite entrenar e implementar fácilmente modelos en la nube, en las instalaciones, en el navegador o en el dispositivo con una amplia variedad de idiomas. Permite a los usuarios expresar cálculos arbitrarios como un gráfico de flujos de datos. Los nodos en este gráfico representan operaciones matemáticas, mientras que los bordes representan datos que se comunican de un nodo a otro. Los datos en TensorFlow se representan como tensores, que son matrices multidimensionales. Aunque este framework para pensar en la computación es valioso en muchos campos diferentes, TensorFlow se utiliza principalmente para el aprendizaje profundo en la práctica y la investigación.

NET Framework

Según la documentación oficial de Microsoft, .NET Framework es una tecnología que admite la creación y ejecución de la próxima generación de aplicaciones y servicios web XML. .NET Framework consta de Common Language Runtime (CLR) y la biblioteca de clases de .NET Framework. Common Language Runtime es la base de .NET Framework. El tiempo de ejecución puede considerarse como un agente que administra el código en el momento de la ejecución, brindando servicios básicos como administración de memoria, administración de subprocesos y comunicación remota, a la vez que impone seguridad de tipo estricta y otras formas de precisión de código que promueven la seguridad y la solidez.

El código que se dirige al tiempo de ejecución se conoce como código administrado, mientras que el código que no se dirige al tiempo de ejecución se

conoce como código no administrado. La biblioteca de clases es una colección completa y orientada a objetos de tipos reutilizables que utiliza para desarrollar aplicaciones que van desde aplicaciones tradicionales de línea de comandos o interfaz gráfica de usuario (GUI) hasta aplicaciones basadas en las últimas innovaciones proporcionadas por ASP.NET, como Web Formularios y servicios web XML. El marco .Net se puede usar para crear aplicaciones de consola de Windows, aplicaciones GUI de Windows, aplicaciones de Windows Presentation Foundation, aplicaciones ASP.NET, servicios de Windows y más.(29)

Visual Studio

El entorno de desarrollo integrado de Visual Studio es una plataforma de lanzamiento creativa que se puede utilizar para editar, depurar y crear código, y luego publicar una aplicación. Un entorno de desarrollo integrado (IDE) es un programa rico en funciones que se puede utilizar para muchos aspectos del desarrollo de software.

Además del editor y depurador estándar que proporcionan la mayoría de los IDE, Visual Studio incluye compiladores, herramientas de finalización de código, diseñadores gráficos y muchas más funciones para facilitar el proceso de desarrollo de software. Ofrece soporte e integración para una amplia variedad de idiomas y herramientas.

Se utiliza para desarrollar programas informáticos, así como sitios web, aplicaciones web, servicios web y aplicaciones móviles. Visual Studio utiliza plataformas de desarrollo de software de Microsoft como Windows API, Windows Forms, Windows Presentation Foundation, Windows Store y Microsoft Silverlight. Puede producir tanto código nativo y administrado código.(30)

Enterprise Architect

Sparx Systems Enterprise Architect es una herramienta de diseño y modelado visual basada en OMG UML. La plataforma admite: el diseño y la construcción de sistemas de software; modelado de procesos comerciales; y modelado de

dominios basados en la industria. Las empresas y organizaciones lo utilizan no solo para modelar la arquitectura de sus sistemas, sino también para procesar la implementación de estos modelos en todo el ciclo de vida de desarrollo de aplicaciones.(31)

1.4 Conclusiones del capítulo

Con la realización de este capítulo se plantearon los conceptos principales relacionados con la investigación para lograr una mejor comprensión de la problemática, además de las definiciones necesarias para lograr un mejor entendimiento del campo de acción en el que se está investigando.

Luego de analizar las técnicas usadas para la clasificación de electrocardiogramas se seleccionó como base la descrita en (17) por razones anteriormente expuestas, y luego de analizar las herramientas disponibles en el momento de la realización del trabajo se decidió la creación de una nueva que permita clasificar electrocardiogramas en normales o anormales, teniendo en cuenta que no se encontró ninguna que clasificase electrocardiogramas, tuviese un alto grado de precisión y su código fuese accesible para ser modificado y usado.

Capítulo 2. Análisis y diseño de la herramienta para la detección de arritmias cardiacas en electrocardiogramas

2.1 Propuesta de solución

Se desarrolla una herramienta para que permite clasificar electrocardiogramas en normales o anormales. Se utilizó una red neuronal como clasificador, la cual tiene como entrada datos de los electrocardiogramas en un formato específico. Luego de ser entrenada, esta red será utilizada para clasificar los electrocardiogramas.

La figura 6 muestra un mapa conceptual del procedimiento:

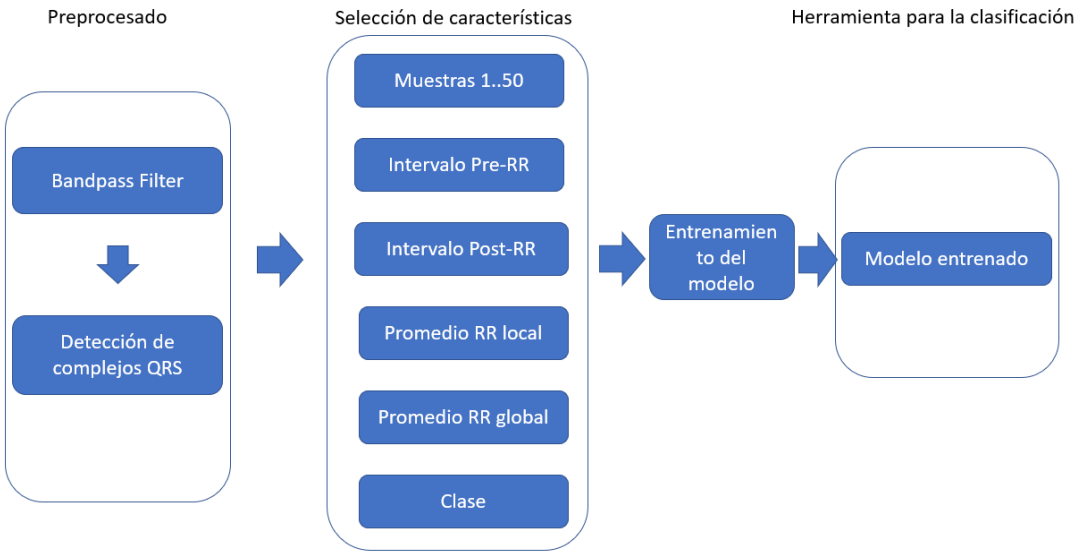


Figura 6: Diagrama conceptual de los pasos necesarios para la obtención de la herramienta para clasificar electrocardiogramas

Como base de datos se utilizó la MIT-BIH, que contiene 48 extractos de media hora de registros de ECG ambulatorios de dos canales, obtenidos de 47 sujetos estudiados por el Laboratorio de arritmia de BIH entre 1975 y 1979.

Las grabaciones se digitalizaron a 360 muestras por segundo por canal con una resolución de 11 bits en un rango de 10 mV(milivoltios). Dos o más cardiólogos anotaron independientemente cada registro; Los desacuerdos se resolvieron

para obtener las anotaciones de referencia legibles por computadora para cada latido (aproximadamente 110,000 anotaciones en total) incluidas con la base de datos.(32) (33) (34) Para el entrenamiento de la red se utilizó solo un canal, generalmente el V1 o V5.

Para el correcto funcionamiento de la herramienta, se transita por 3 fases o etapas: preprocesamiento de los datos, entrenamiento de la red neuronal y utilización de esta para clasificar los electrocardiogramas.

Para el preprocesamiento de los datos y el diseño de la red neuronal se utilizó la base teórica ya creada por varios autores, descrita en (17) y (35).Cada una de estas etapas tiene sus especificidades.

Durante la etapa de preprocesamiento de los datos, se realizan varias transformaciones a la señal original para que sea posible utilizarla en el entrenamiento de la red neuronal. Estas transformaciones pueden clasificarse como eliminación del ruido en la señal original, y extracción de las características relevantes para el entrenamiento de la red neuronal.

La señal de un electrocardiograma puede ser contaminada por señales que no pertenecen a este, estas señales son conocidas como ruido. Los 2 tipos de ruido más comunes en un electrocardiograma son la variación de la línea base y la interferencia de la línea eléctrica. La primera es causada por la respiración del paciente o los movimientos de este, mientras que la segunda es causada por la interferencia que generan los equipos eléctricos cercanos al dispositivo que graba el electrocardiograma, así como electrodos sueltos o flojos.(17)

Para la eliminación del ruido se decidió utilizar un filtro pasa banda entre 5 y 20Hz, paso que está implementado en el módulo wfdb.processing para Python, es parte del algoritmo de detección de complejos QRS, y coincide con los métodos usados en la literatura. (36)

Luego de haber eliminado el ruido de la señal original, se procede a detectar todos los picos de interés en el electrocardiograma, utilizando el detector

implementado en wfdb.processing.XQRS (36), así como los complejos QRS, que servirán para entrenar a la red neuronal.

Con la información previa se procede a crear los datos que serán usados como entrada por la red neuronal. Para ello, de cada latido se seleccionan 50 muestras (valores) que denotan aproximadamente cada latido, tomando como referencia el pico R. Además, se selecciona lo siguiente:

El intervalo Pre-RR, el cual es el intervalo entre el pico R actual y el pico R anterior.

El intervalo Post-RR, el cual es el intervalo entre el pico R actual y el pico R posterior.

El promedio local RR, definido como el promedio de los 10 intervalos RR comprendidos entre los últimos 10 segundos.

El promedio global RR, definido como el promedio de los 10 intervalos RR comprendidos entre los últimos 5 minutos.

Además, a cada electrocardiograma se le asigna un id y una clase, la cual será 1 en caso de ser normal y 2 en caso de ser anormal.

En resumen, de cada electrocardiograma se selecciona siguiendo a (17):

- Id del electrocardiograma
- 50 muestras comprendidas entre el pico P y el pico T
- Intervalo Pre-RR
- Intervalo Post-RR
- Promedio local RR
- Promedio global RR

La red neuronal creada utilizará este formato como entrada, sin el Id. La misma está diseñada siguiendo el trabajo de (17). Esta tiene 54 neuronas de entrada, una para cada uno de los datos seleccionados anteriormente. Tiene 7 capas ocultas, con 5,10,30,50,30,10 y 5 neuronas respectivamente. La función de activación usada en estas capas es la *Rectified Linear Unit* (ReLU) (37) , la

función de la neurona de salida es la función *Softmax* y la función de costo es la Entropía Cruzada. La etapa de entrenamiento de la red neuronal, que es la más costosa computacionalmente, solo se realiza una vez para obtener el modelo entrenado, que luego será usado tantas veces como se necesite.

Para crear la red neuronal se utilizó Tensorflow.NET (27), módulo creada para poder usar Tensorflow dentro del ecosistema .NET, usando C#.

La herramienta desarrollada espera como entrada un archivo en formato json, teniendo 2 llaves: 'id' y 'signal'. El valor de id será una cadena, mientras que el de 'signal' será un arreglo de valores que representan los valores de voltaje en cada momento. Ya que los electrocardiogramas utilizados para entrenar el modelo de clasificación fueron digitalizados a una frecuencia de 360Hz, se espera que 360 valores del arreglo asociado a la llave 'signal' representen 1 segundo del electrocardiograma.

2.2 Modelo del Dominio

En el Modelo de Dominio se muestran las clases del dominio y las relaciones entre ellas. En la figura 3 se representa el Diagrama de Clase de Dominio realizado, correspondiente a la clasificación de electrocardiogramas, en el cual un especialista interactúa con una estación de trabajo que contiene la herramienta para clasificar, esta última utiliza el modelo entrenado para clasificar el electrocardiograma y se obtiene el electrocardiograma clasificado.

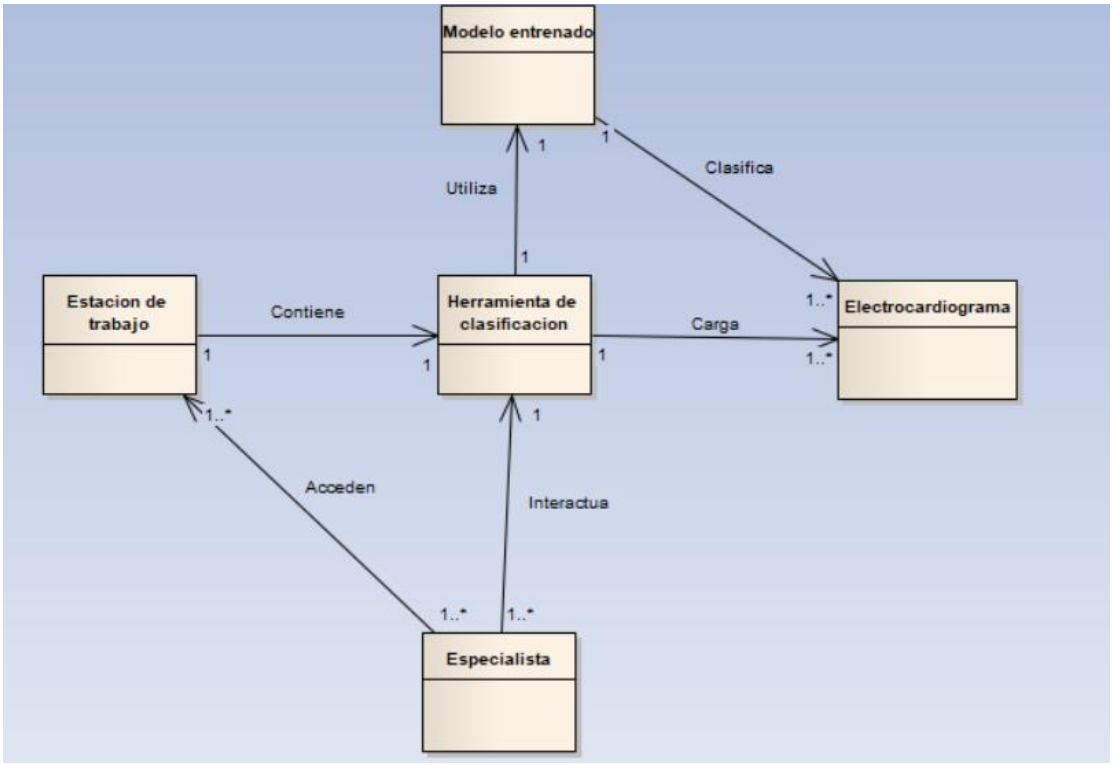


Figura 3: Modelo del Dominio de la Herramienta para clasificar electrocardiogramas. (fuente: elaboración propia)

Para un mejor entendimiento del modelo de dominio en la Tabla 1 se describen brevemente los conceptos que lo conforman.

Tabla 1: Clases del Modelo de Dominio de la herramienta para clasificar electrocardiogramas. (fuente: elaboración propia)

Clases	Descripción
Especialista	Especialista de la salud que interactúa con la estación de trabajo para obtener un electrocardiograma clasificado
Estación de trabajo	Computadora que contiene la herramienta de clasificación
Electrocardiograma	Archivo que contiene la variación del voltaje en el tiempo medida en un canal de un electrocardiograma
Herramienta de clasificación	Programa que permite la clasificación de un electrocardiograma en normal o anormal
Modelo entrenado	Archivo que contiene la red neuronal entrenada para clasificar electrocardiogramas

2.2 Modelo del Negocio

El modelo del negocio describe cada proceso del negocio, especificando sus datos, actividades, roles y las reglas del mismo. Facilita un entendimiento entre clientes y desarrolladores y se enfoca en comprender los problemas actuales de la organización e identifica mejoras potenciales.

2.2.1 Identificación de roles del entorno del negocio

Una vez identificados los procesos de negocio, es preciso encontrar los involucrados en su realización. A continuación, se muestran los roles que se identificaron en la solución propuesta:

Tabla 2. Actores del negocio. Fuente: Elaboración propia.

Actores del negocio	Descripción
Paciente	Personas egresadas a una institución hospitalaria.
Computadora	Se encarga de cargar los datos de la memoria del Holter y permitir su posterior análisis.
Cardiólogo	Se encargado de realizar una impresión diagnóstica y la solicitud de estudios cardiológicos.

2.2.2 Diagrama del proceso de negocio

Para la presente investigación se adopta el modelado de proceso de negocio “Diagnosticar paciente”.

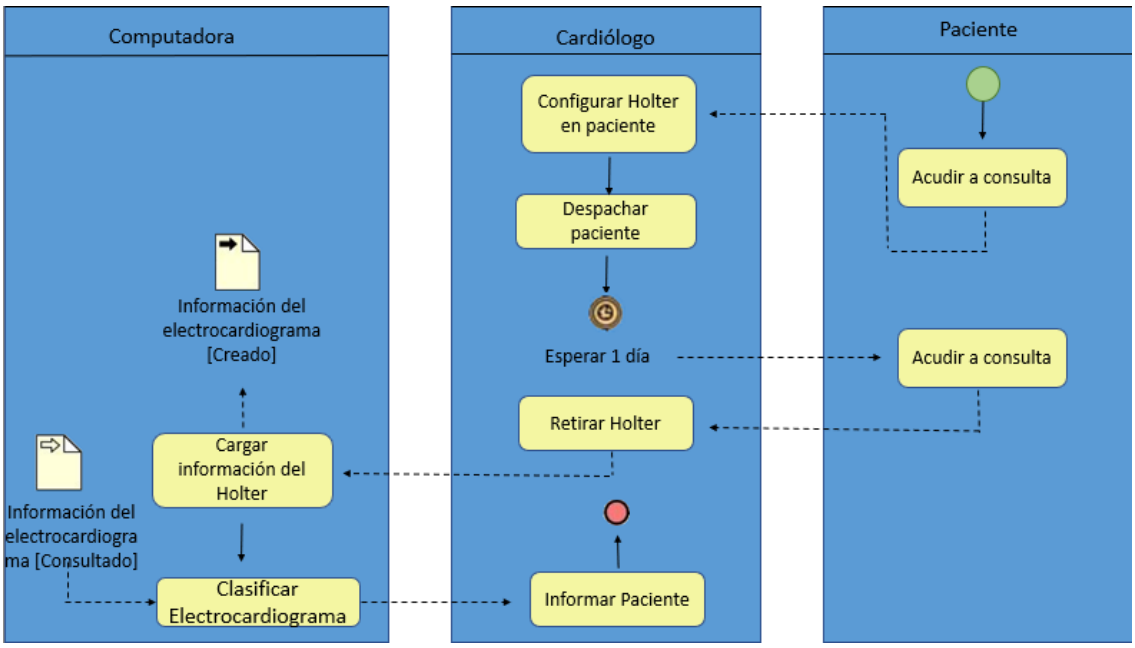


Figura 4: Diagrama del proceso Atender paciente en Electrocardiograma

2.3 Análisis de los requisitos funcionales

Un requisito funcional define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas.(38) Los requisitos funcionales de la herramienta se muestran en la tabla:

Tabla 3: Requisitos funcionales de la herramienta para clasificar electrocardiogramas. (fuente: elaboración propia)

Requisitos	Descripción
RF 1 Cargar el electrocardiograma	Se carga el electrocardiograma, o el directorio de electrocardiogramas
RF 2 Eliminar la variación de la línea base	Se aplica un filtro pasa banda entre 5 y 20Hz para eliminar el ruido de la señal

RF 3 Eliminar la interferencia de la línea de alimentación	Se aplica un filtro pasa banda entre 5 y 20Hz para eliminar el ruido de la señal
RF 4 Detectar picos en la señal	Se detectan todos los picos existentes en la señal
RF 5 Detectar complejos QRS	Se detectan los picos R en la señal
RF 6 Detectar ondas T y P	Se detectan los picos T y P
RF 7 Seleccionar datos de entrada al modelo	Se seleccionan los datos teniendo en cuenta el formato especificado para la entrada a la red neuronal
RF 8 Creación del modelo	Se crea el modelo de la red neuronal utilizando TensorFlow.Net
RF 9 Entrenamiento del modelo	Se entrena el modelo con los datos de la base de datos seleccionada
RF 10 Clasificación de electrocardiograma	El sistema clasifica un electrocardiograma en normal o anormal
RF 11 Renombrar electrocardiograma	Se renombra el electrocardiograma añadiendo la cadena “- Normal” en caso de que sea normal o “- Anormal” en caso de q sea anormal.

En la figura 5 se muestra el diagrama de requisitos funcionales.

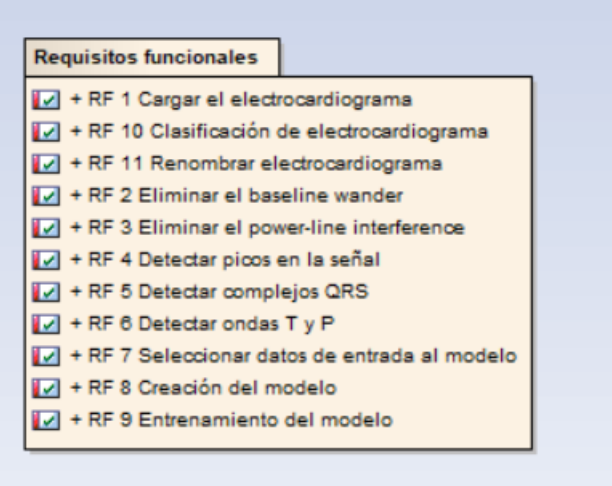


Figura 5: Diagrama de requisitos funcionales de la herramienta para clasificar electrocardiogramas. (fuente: elaboración propia)

2.4 Análisis de los requisitos no funcionales

Los requisitos no funcionales no están directamente relacionados con los servicios específicos prestados por el sistema a sus usuarios, sino que especifican o restringen las características del sistema en su conjunto. Pueden relacionarse con propiedades emergentes del sistema tales como fiabilidad y tiempo de respuesta y definir restricciones en la implementación del sistema, tales como la capacidad de almacenamiento.(39)

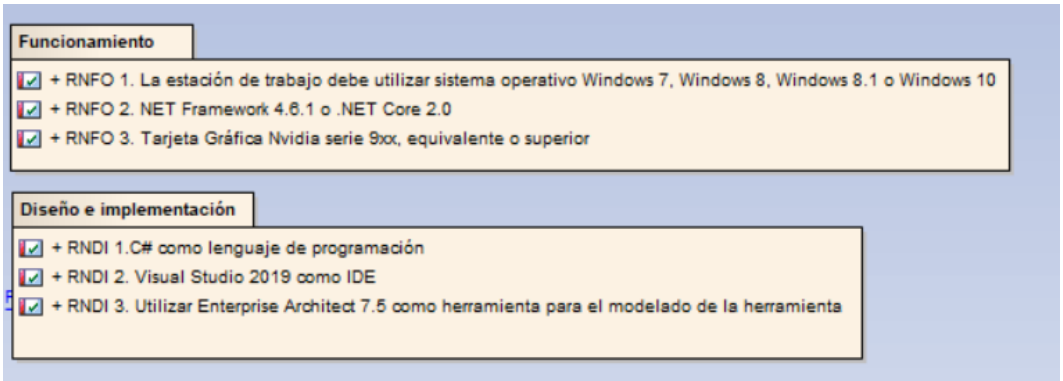
Diseño e implementación

- C# como lenguaje de programación.
- Visual Studio 2015 como IDE.
- Utilizar Enterprise Architect 7.5 como herramienta para el modelado de la herramienta.

Funcionamiento

- La estación de trabajo debe utilizar sistema operativo Windows 7, Windows 8, Windows 8.1 o Windows 10.
- .NET Framework 4.6.1 o .NET Core 2.0
- Tarjeta Gráfica Nvidia serie 9xx, equivalente o superior

En la figura 6 se muestra el diagrama de requisitos no funcionales.



2.5 Definición de actores

Un actor representa a cualquier ente externo que interactúa con el sistema, pueden ser humanos u otros sistemas.(39) El actor asociado al presente trabajo se muestra en la tabla 4:

Tabla 4: Actor asociado al sistema. (fuente: elaboración propia)

Actor	Descripción
Especialista	Médico de cualquier especialidad que interactúa con la estación de trabajo que contiene la herramienta para

2.6 Diagrama de casos de uso

Los diagramas de casos de uso representan las interacciones entre el sistema y sus usuarios. Determinan las funciones que el sistema puede ejecutar (27). Este diagrama representa el comportamiento y la interacción de los usuarios con la herramienta.

La figura 7 muestra el diagrama de caso de uso del sistema referente a la herramienta para clasificar electrocardiogramas.

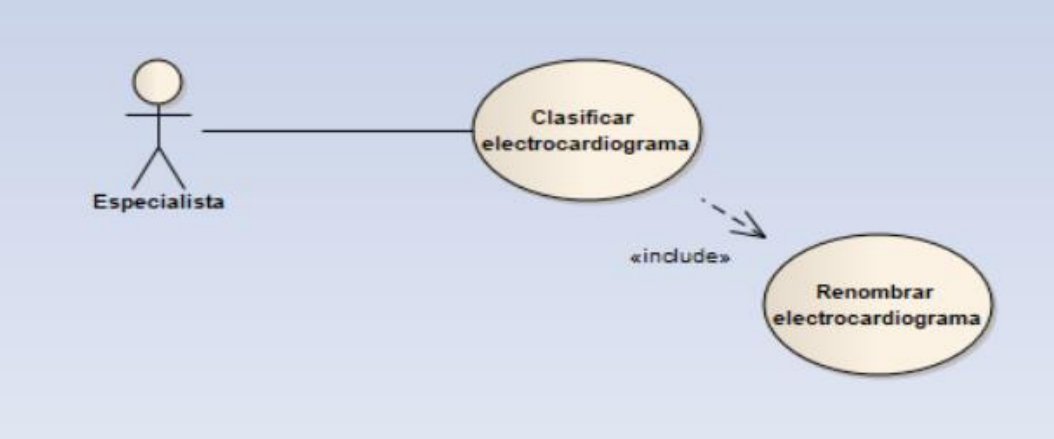


Figura 7: Diagrama de caso de uso del sistema referente a la herramienta para clasificar electrocardiogramas. (fuente: elaboración propia)

2.7 Descripción de casos de uso

En la tabla 5 se muestra la descripción del caso de uso Clasificar Electrocardiograma. La descripción se realiza con el objetivo de especificar cada uno de los elementos que componen el caso de uso, así como los flujos de eventos por los que está compuesto. Aparece reflejada la valoración de la complejidad y la prioridad del caso de uso.

Tabla 5: Descripción del caso de uso Clasificar Electrocardiograma. (fuente: elaboración propia)

Objetivo	Clasificar electrocardiograma
Actor	Especialista de la salud
Resumen	El caso de uso comienza cuando el especialista abre la herramienta. Como resultado se obtendrá el electrocardiograma clasificado y renombrado
Complejidad	Alta
Prioridad	Crítica

Precondiciones	El electrocardiograma debe estar en el formato especificado
Flujo de eventos	
Flujo básico Clasificar electrocardiograma	
1.	El especialista ejecuta la herramienta
2.	El especialista introduce la dirección del electrocardiograma en el sistema
3.	El sistema clasifica el electrocardiograma
4.	El sistema renombra el electrocardiograma
5.	Termina el caso de uso
Flujo alterno 1 El electrocardiograma no tiene el formato correcto	
1.	El especialista ejecuta la herramienta
2.	El especialista introduce la dirección de un electrocardiograma con un formato incorrecto
3.	La herramienta renombra el electrocardiograma añadiendo la cadena “-No soportado”
Sección 1: Cargar el electrocardiograma	
Flujo básico	
1.	El sistema muestra una ventana para que se indique la dirección de el/los electrocardiograma y lo carga
Flujo alterno	
No aplica	
Sección 2: Eliminar la variación de la línea base	

Flujo básico	
1.	Se aplica un filtro pasa banda entre 5 y 20Hz para eliminar el ruido de la señal
Flujo alterno	
No aplica	
Sección 3: Eliminar la interferencia de la línea de alimentación	
Flujo básico	
1.	Se aplica un filtro pasa banda entre 5 y 20Hz para eliminar el ruido de la señal
Flujo alterno	
No aplica	
Sección 4: Detectar picos en la señal	
Flujo básico	
1.	Se detectan todos los picos existentes en la señal
Flujo alterno	
No aplica	
Sección 5: Detectar complejos QRS	
Flujo básico	
1.	Se detectan los picos R en la señal
Flujo alterno	
No aplica	
Sección 6: Detectar ondas T y P	

Flujo básico	
1.	Se detectan los picos T y P
Flujo alterno	
No aplica	
Sección 7: Seleccionar datos de entrada al modelo	
Flujo básico	
1.	Se seleccionan los datos teniendo en cuenta el formato especificado para la entrada a la red neuronal
Flujo alterno	
No aplica	
Sección 8: Creación del modelo	
Flujo básico	
1.	Se crea el modelo de la red neuronal utilizando TensorFlow.Net
Flujo alterno	
No aplica	
Sección 9: Entrenamiento del modelo	
Flujo básico	
1.	Se entrena el modelo con los datos de la base de datos seleccionada
Flujo alterno	
No aplica	
Sección 10: Clasificación de electrocardiograma	
Flujo básico	

1.	El sistema clasifica un electrocardiograma en normal o anormal utilizando el modelo entrenado
Flujo alterno	
No aplica	
Sección 11: Renombrar electrocardiograma	
Flujo básico	
1.	Se renombra el electrocardiograma añadiendo la cadena “- Normal” en caso de que sea normal o “- Anormal” en caso de q sea anormal.
Flujo alterno	
No aplica	

Relaciones	CU incluidos	Renombrar electro
	CU extendidos	No aplica
Requisitos no funcionales	RNDI 1, RNFO 2, RNFO 3	
Asuntos pendientes	No aplicable	

2.8 Conclusiones del capítulo

En este capítulo se realizó la propuesta de solución, En la propuesta de solución se definió como clasificador una red neuronal cuya arquitectura esta descrita teóricamente; se definieron las acciones a tomar en cada paso del preprocesamiento de los datos teniendo en cuenta la literatura analizada pero realizando pequeños cambios en la detección de los complejos QRS y la eliminación del ruido, para lo cual se hace uso del algoritmo implementado en wfdb.processing.XQRS y se definió la entrada esperada por el programa, así como su salida.

Se modelo el dominio de la herramienta haciendo uso de un Diagrama de Clase de Dominio compuesto por 5 clases, su composición y sus relaciones. Dichas clases fueran descritas con detalle posteriormente.

Se definieron 11 requisitos funcionales para el sistema y 6 no funcionales, de estos últimos 3 de diseño e implementación y 3 de funcionamiento.

Se definió como actor del sistema a 'Especialista', con su respectiva descripción.

Se describieron los casos de uso mediante diagramas de casos de uso, mostrando el caso de uso referente a la herramienta para clasificar electrocardiogramas.

Capítulo 3. Validación de la propuesta de solución

En el presente capítulo se plantea la estrategia para la validación de la propuesta de solución. Además, se describen los estándares de codificación que se emplean en el desarrollo de la herramienta y se realiza la validación de la propuesta de solución a partir de los métodos y técnicas definidos.

3.1 Modelo Arquitectónico

El diseño arquitectónico se ocupa de comprender cómo debe organizarse un sistema y diseñar la estructura general de ese sistema. Es el vínculo crítico entre el diseño y la ingeniería de requisitos, ya que identifica los principales componentes estructurales de un sistema y las relaciones entre ellos. La salida del proceso de diseño arquitectónico es un modelo arquitectónico que describe cómo el sistema está organizado como un conjunto de componentes que se comunican. (39) Como modelo arquitectónico de la herramienta para la detección de arritmias cardíacas en electrocardiogramas se utilizó Tuberías y Filtros (Pipes and Filter).

La arquitectura basada en tuberías y filtros es utilizada cuando se requiere el procesamiento de un objeto mediante un conjunto de transformaciones comprendidas en fases secuenciales, donde los datos fluyen de uno a otro y se transforman a medida que se mueven a través de la secuencia. Cada paso de procesamiento se implementa como una transformación. Los datos de entrada fluyen a través de estas transformaciones hasta convertirlas en salida. Las transformaciones pueden ejecutarse secuencialmente o en paralelo.(39) Esto se evidencia en el conjunto de transformaciones que se le deben aplicar a la señal de un electrocardiograma con el objetivo de interpretarlo y clasificarlo. En la Figura 8 se muestra la secuencia de técnicas aplicadas a un electrocardiograma para poder clasificarlo.

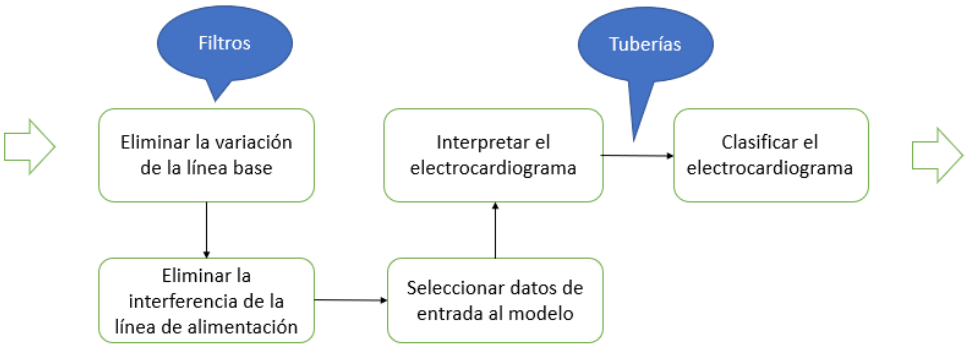


Figura 8. Arquitectura basada en Tuberías y Filtros de la herramienta para la detección de arritmias cardíacas en electrocardiogramas (fuente: elaboración propia).

3.2 Implementación

El objetivo de esta disciplina es construir el sistema informático, a partir de los resultados del Análisis y Diseño. Una implementación es la realización de una especificación técnica o algoritmos con un programa. Muchas implementaciones son realizadas según una especificación o un estándar.(39)

El uso de estos estándares o guías es tiene beneficios tangibles y su uso se recomienda por varias causas, entre las cuales se encuentran:

- El 80% del costo de por vida de un producto de software se destina al mantenimiento.
- Casi ningún software es mantenido durante toda su vida por el autor original.
- Las convenciones de código mejoran la legibilidad del software, lo que permite a los ingenieros comprender el código nuevo de forma más rápida y completa.(40)

3.2.1 Estrategia de codificación. Estándares y estilos usados

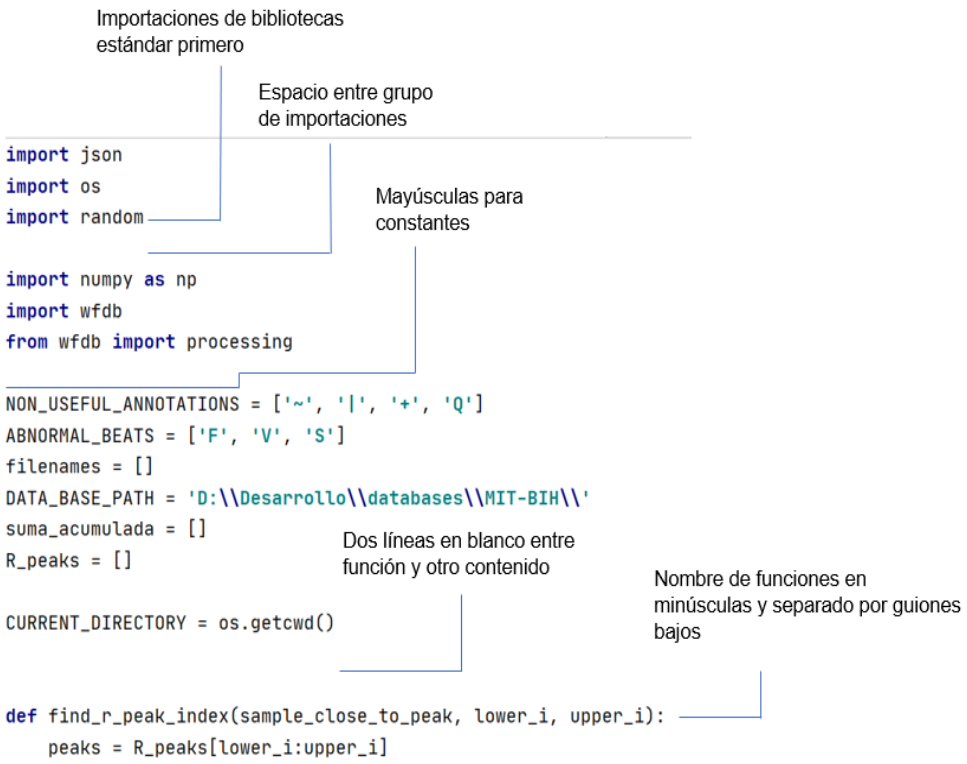
A continuación, se mencionan los estándares de codificación de Python utilizados para la implementación de la herramienta:

- Siempre se debe rodear estos operadores binarios con un solo espacio a cada lado: asignación (=), asignación aumentada (+ =, - = etc.), comparaciones (==, <,>,! =, <>, <=,> = , en, no en, es, no es), booleanos (y, o no).
- Las importaciones siempre se colocan en la parte superior del archivo, justo después de los comentarios y cadenas de documentación del módulo, y antes de las constantes y globales del módulo.
- Las importaciones generalmente deben estar en líneas separadas.
- Las importaciones deben agruparse en el siguiente orden:

1. Importaciones de bibliotecas estándar.
2. Importaciones de terceros relacionadas.
3. Importaciones específicas de aplicaciones / bibliotecas locales.

- Se debe poner una línea en blanco entre cada grupo de importaciones.
- Se rodean las definiciones de clases y funciones de nivel superior con dos líneas en blanco.
- Los nombres de las funciones están en minúsculas, con palabras separadas por guiones bajos según sea necesario para mejorar la legibilidad.
- Los nombres de variables siguen la misma convención que los nombres de funciones.
- Las constantes generalmente se definen a nivel de módulo y se escriben en letras mayúsculas con guiones bajos que separan las palabras.

Un ejemplo de la implementación de este estándar en el código de la aplicación se puede apreciar en la siguiente imagen:



3.3 Pruebas de software

Las pruebas de software consisten en la verificación del comportamiento de un programa en un conjunto finito de casos de prueba. El objetivo de las pruebas es descubrir posibles errores de implementación y calidad de un programa informático.(39) A continuación, se describen los tipos de prueba de software, métodos y técnicas aplicados en la evaluación de la solución desarrollada:

3.3.1 Tipos de pruebas de software

Pruebas Unitarias

Las pruebas unitarias o de componente consisten en la ejecución de actividades que le permitan verificar la funcionalidad y la estructura (lógica interna) de cada elemento individualmente, una vez que ha sido codificado. Estas pruebas se realizan sobre cada sección del software, de manera independiente, con el objetivo de comprobar que está correctamente codificada. (41)

Pruebas Funcionales

Las pruebas funcionales se centran en comprobar que el sistema funcione acorde a los requisitos funcionales y no funcionales, detectando posibles defectos derivados de errores en la fase de desarrollo. Mediante estas pruebas se demuestra que las funciones del software son operativas, la entrada se acepta de forma adecuada y se produce una salida correcta. (41)

3.3.2 Métodos de prueba

Método de caja blanca

Las pruebas de caja blanca se centran en los detalles procedimentales del software (código fuente), para obtener los casos de prueba. Mediante este método, el ingeniero del software puede examinar el estado de la aplicación, en varios puntos, para determinar si el estado real coincide con el estado deseado o esperado. (41)

Método de caja negra

Las pruebas de caja negra o pruebas funcionales se aplican sobre la interfaz del software. Mediante este método, el ingeniero del software puede validar que las salidas sean las esperadas. Además, se centra en encontrar las circunstancias en las que el sistema no se comporte conforme a las especificaciones establecidas. (41)

3.4 Resultados alcanzados por la herramienta para la detección de arritmias cardíacas en electrocardiogramas

3.4.1 Indicadores para evaluar el desempeño del procedimiento de diagnóstico.

Para verificar el adecuado funcionamiento del presente algoritmo, se hace necesario el cálculo de los indicadores para evaluar el rendimiento del proceso de diagnóstico. El resultado de una prueba diagnóstica puede ser positivo o

negativo, pero estos pueden ser correctos e incorrectos, dando lugar a cuatro tipos de resultados como se evidencia en la Tabla 6.

Tabla 6. Relación entre los resultados de una prueba diagnóstica. (42)

Resultados de la	Electrocardiograma	Electrocardiograma
El algoritmo clasifica el electrocardiograma	verdadero negativo (VN)	falso negativo (FN)
El algoritmo clasifica el electrocardiograma	falso positivo (FP)	verdadero positivo (VP)

- VP: como resultado de la clasificación el algoritmo detecta una anormalidad en presencia de una anormalidad clínica.
- VN: como resultado de la clasificación el algoritmo no detecta una anormalidad en ausencia de la anormalidad clínica.
- FP: como resultado de la clasificación el algoritmo detecta una anormalidad en ausencia de la anormalidad clínica.
- FN: como resultado de la clasificación el algoritmo no detecta una anormalidad en presencia de la anormalidad clínica.

El promedio de falsos positivos (PFP), representa la cantidad de falsos positivos encontrados en total en una prueba, dividido por la cantidad de imágenes (CI) analizadas en dicha prueba.

$$PFP = \frac{FP}{CI}$$

La validez se define como la capacidad de un instrumento para medir lo que intenta medir. Esta sólo puede determinarse si existe un procedimiento de referencia, también conocido como estándar de oro, el cual, es considerado como un procedimiento definitivo para establecer si alguien tiene la característica de interés. Los parámetros que miden la validez de una prueba diagnóstica son la sensibilidad y la especificidad. (42)

La sensibilidad (S) es la proporción de los individuos que tienen la enfermedad y el resultado de la prueba realizada da positivo, es decir, la proporción de verdaderos positivos, o de enfermos diagnosticados, respecto del total de enfermos en la población de estudio. Cuanto más alto es el valor de S, hay una mejor capacidad en la detección de enfermos por medio de la prueba:

$$S = \frac{VP}{VP + FN}$$

La especificidad (E) es la habilidad para identificar aquellos individuos que no tienen cáncer. Es la proporción de los verdaderos negativos, respecto al total de individuos sanos en la población de estudio. Mientras mayor sea el valor de E, hay una mejor capacidad en la detección de individuos sanos:

$$E = \frac{VN}{VN + FP}$$

Para evaluar la efectividad de un CAD, se utiliza el indicador de precisión (P), el cual se calcula a partir de los indicadores sensibilidad y especificidad:

$$P = \frac{VP + VN}{VP + VN + FP + FN}$$

En la Tabla 7 se evidencia una relación de valores de precisión, que especifica qué tan buenos son los resultados alcanzados por un CAD, a partir de los resultados obtenidos al medir este indicador.

Tabla 7. Relación entre los valores de precisión y los conjuntos que definen. (43)

Precisión	Etiqueta
90%-100%	Excelente
80%-90%	fMuy Bien

70%-80%	Bien
60%-70%	Suficiente
50%-60%	Malo
< 50%	No es de utilidad

El algoritmo utilizado en la herramienta presenta un promedio de precisión de un 99% tras entrenarlo y probarlo sobre el conjunto de datos 20 veces. En los videos anexos al presente trabajo se demuestra el procedimiento de entrenamiento y prueba.

3.5 Conclusiones del capítulo

- El uso de los estándares de codificación y el tratamiento de errores fortalecen la construcción del código proporcionándole legibilidad, independencia y mantenibilidad.
- Las pruebas de software permitieron constatar la calidad en la implementación realizada.

Conclusiones

- El análisis del marco teórico sobre el proceso de detección de arritmias en un electrocardiograma usando medios computarizados, así como el análisis de las diferentes herramientas y algoritmos desarrollados hasta ahora con ese fin, demostraron la necesidad de desarrollar una herramienta que fuese capaz de alcanzar resultados similares a las actuales y a la vez ser utilizada y auditada por profesionales cubanos.
- Los 11 requisitos funcionales y los 6 no funcionales definidos permitieron cumplir con las necesidades del cliente. El análisis y diseño propició la elaboración de una herramienta que permite detectar electrocardiogramas anómalos.
- La aplicación de las pruebas de software permitió evaluar la solución desarrollada y garantizar el correcto funcionamiento de la herramienta implementada.
- El desarrollo de una herramienta para detectar electrocardiogramas anómalos permitió cumplir con las exigencias del cliente y las restricciones del diseño elaborado.

Recomendaciones

Para futuras investigaciones se recomiendan las siguientes acciones:

- Ampliar el conjunto de clasificaciones posibles por la herramienta.
- Implementar una interfaz más amigable para el usuario

Referencias bibliográficas

1. Cardiovascular diseases (CVDs). [online]. [Accedido 5 September 2020]. Disponible en: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)

2. Anuario Estadistico de Cuba. [online]. [Accedido 5 September 2020]. Disponible en: http://www.one.cu/aec2018/19 Salud Publica.pdf

3. • Highest physicians density worldwide by country 2016 | Statista. [online]. [Accedido 5 September 2020]. Disponible en: https://www.statista.com/statistics/280152/countries-with-the-highest-physicians-density-worldwide/

4. BONOW, Robert O., MANN, Douglas L., ZIPES, Douglas P. and LIBBY, Peter. *Braunwald's Heart Disease E-Book: A Textbook of Cardiovascular Medicine*. Elsevier Health Sciences, 2011. ISBN 978-1-4377-2770-8. Google-Books-ID: b5wADkB9oDoC

5. VELIC, Marko, PADAVIC, Ivan and CAR, Sinisa. Computer aided ECG analysis — State of the art and upcoming challenges. In : *Eurocon 2013*. July 2013. p. 1778–1784.

6. ASHLEY, Euan A. and NIEBAUER, Josef. *Conquering the ECG* [online]. Remedica, 2004. [Accedido 5 September 2020]. Disponible en: https://www.ncbi.nlm.nih.gov/books/NBK2214/

7. About Arrhythmia | American Heart Association. [online]. [Accedido 5 September 2020]. Disponible en: https://www.heart.org/en/health-topics/arrhythmia/about-arrhythmia

8. Arrhythmia. *nhs.uk* [online]. 19 February 2018. [Accedido 5 September 2020]. Disponible en: https://www.nhs.uk/conditions/arrhythmia/

9. Explained: Neural networks. *MIT News | Massachusetts Institute of Technology* [online]. [Accedido 5 September 2020]. Disponible en: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414

10. Artificial Neural Network (ANN) in Machine Learning - Data Science Central. [online]. [Accedido 5 September 2020]. Disponible en: https://www.datasciencecentral.com/profiles/blogs/artificial-neural-network-ann-in-machine-learning

11. What Is Deep Learning? | How It Works, Techniques & Applications - MATLAB & Simulink. [online]. [Accedido 5 September 2020]. Disponible en: https://www.mathworks.com/discovery/deep-learning.html

12. ISIN, Ali and OZDALILI, Selen. Cardiac arrhythmia detection using deep learning. *Procedia Computer Science* [online]. 1 January 2017. Vol. 120, p. 268–275. [Accedido 5 September 2020]. DOI 10.1016/j.procs.2017.11.238. Disponible en: http://www.sciencedirect.com/science/article/pii/S187705091732450X

13. VENTURA, Dan and WARNICK, Sean. A theoretical foundation for inductive transfer. *Brigham Young University, College of Physical and Mathematical Sciences*. 2007.

14. KRIZHEVSKY, Alex, SUTSKEVER, Ilya and HINTON, Geoffrey E. Imagenet classification with deep convolutional neural networks. In : *Advances in neural information processing systems*. 2012. p. 1097–1105.

15. KARIMIFARD, S., AHMADIAN, A., KHOSHNEVISAN, M. and NAMBAKHSH, M. S. Morphological Heart Arrhythmia Detection Using Hermitian Basis Functions and kNN Classifier. In : *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*. August 2006. p. 1367–1370.

16. YILDIRIM, Özal, P\LAWIAK, Pawe\l, TAN, Ru-San and ACHARYA, U. Rajendra. Arrhythmia detection using deep convolutional neural network with long duration ECG signals. *Computers in biology and medicine*. 2018. Vol. 102, p. 411–420.

17. SANNINO, Giovanna and DE PIETRO, Giuseppe. A deep learning approach for ECG-based heartbeat classification for arrhythmia detection. *Future Generation Computer Systems*. 2018. Vol. 86, p. 446–455.

18. PONNLE, Akinlolu A. and OGUNDEPO, Oludare Y. Development of a computer-aided application for analyzing ECG signals and detection of cardiac arrhythmia using back propagation neural network-part i: Model development. *Development*. 2015. Vol. 9, no. 3, p. 29–36.

19. HERNÁNDEZ CASTRO, Dayán. *Desarrollo del software HRV Station 2.7 para el análisis de la variabilidad de la frecuencia cardíaca (VFC)*. TRABAJO DE DIPLOMA. Santa Clara : Universidad Central “Marta Abreu” de Las Villas, 2012.

20. FRIESEN, Gary M., JANNETT, Thomas C., JADALLAH, Manal Afify, YATES, Stanford L., QUINT, Stephen R. and NAGLE, H. Troy. A comparison of the noise sensitivity of nine QRS detection algorithms. *IEEE Transactions on biomedical engineering*. 1990. Vol. 37, no. 1, p. 85–98.

21. DEL TORO ALMENARES, A., LORENZO GINORI, J.V. and TABOADA CRISPÍ, A. *Experiencias en el Procesamiento Digital de la Señal ECG*. Santa Clara : Universidad Central “Marta Abreu” De Las Villas, Cuba, 2008.

22. HERNÁNDEZ HERRERA, Lisbel. *Detección de apnea del sueño a partir de la señal electrocardiográfica.* TRABAJO DE DIPLOMA. Santa Clara : Universidad Central “Marta Abreu” de Las Villas, 2013.

23. The Agile Unified Process. [online]. [Accedido 5 September 2020]. Disponible en: http://www.cs.sjsu.edu/~pearce/modules/lectures/se/aup.htm

24. Introduction to the C# Language and .NET | Microsoft Docs. [online]. [Accedido 5 September 2020]. Disponible en: https://docs.microsoft.com/en-us/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework

25. What is Python? Executive Summary. *Python.org* [online]. [Accedido 5 September 2020]. Disponible en:

https://www.python.org/doc/essays/blurb/

26. What is Unified Modeling Language (UML)? [online]. [Accedido 5 September 2020]. Disponible en: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/

27. *SciSharp/TensorFlow.NET* [online]. C#. SciSharp STACK, 2020. [Accedido 5 September 2020]. Disponible en: https://github.com/SciSharp/TensorFlow.NET

28. *tensorflow/tensorflow* [online]. C++. tensorflow, 2020. [Accedido 12 September 2020]. Disponible en: https://github.com/tensorflow/tensorflow

29. GEWARREN. Overview of the .NET Framework. [online]. [Accedido 5 September 2020]. Disponible en: https://docs.microsoft.com/en-us/dotnet/framework/get-started/overview

30. TERRYGLEE. Overview of Visual Studio. [online]. [Accedido 5 September 2020]. Disponible en: https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide

31. Full Lifecycle Modeling for Business, Software and Systems | Sparx Systems. [online]. [Accedido 5 September 2020]. Disponible en: https://sparxsystems.com/products/ea/index.html

32. GOLDBERGER, Ary L., AMARAL, Luis AN, GLASS, Leon, HAUSDORFF, Jeffrey M., IVANOV, Plamen Ch, MARK, Roger G., MIETUS, Joseph E., MOODY, George B., PENG, Chung-Kang and STANLEY, H. Eugene. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *circulation*. 2000. Vol. 101, no. 23, p. e215–e220.

33. MOODY, George B. and MARK, Roger G. The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*. 2001. Vol. 20, no. 3, p. 45–50.

34. MIT-BIH Arrhythmia Database v1.0.0. [online].

[Accedido 5 September 2020]. Disponible en: https://physionet.org/content/mitdb/1.0.0/

35. DE CHAZAL, Philip and REILLY, Richard B. A patient-adapting heartbeat classifier using ECG morphology and heartbeat interval features. *IEEE transactions on biomedical engineering*. 2006. Vol. 53, no. 12, p. 2535–2543.

36. processing — wfdb 3.1.0 documentation. [online]. [Accedido 5 September 2020]. Disponible en: https://wfdb.readthedocs.io/en/latest/processing.html

37. NAIR, Vinod and HINTON, Geoffrey E. Rectified linear units improve restricted boltzmann machines. In : *ICML*. 2010.

38. Requisito functional. [online]. [Accedido 5 September 2020]. Disponible en: http://umh2818.edu.umh.es/wp-content/uploads/sites/884/2016/02/Requisito-funcional.pdf

39. SOMMERVILLE, Ian. *Ingeniería del software*. Pearson Educación, 2005. ISBN 978-84-7829-074-1. Google-Books-ID: gQWd49zSut4C

40. Code Conventions for the Java Programming Language: 1. Introduction. [online]. [Accedido 13 September 2020]. Disponible en: https://www.oracle.com/java/technologies/javase/codeconventions-introduction.html#16712

41. PRESSMAN, Roger S. *Software Engineering-A Practitioner's Approach*. McGraw-Hill, 2010.

42. ALEJO MARTÍNEZ, AMALIA and CUEVAS RENAUD, CORINA. Sensibilidad y Especificidad de una prueba [online]. Disponible en: http://www.psicol.unam.mx/Investigacion2/pdf/SENSIBILIDAD%20Y%20ESPECIFICIDAD.pdf

43. ŠIMUNDIĆ, Ana-Maria. Measures of diagnostic accuracy: basic definitions. Ejifcc. 2009. Vol. 19, no. 4, p. 203.