REVISTA INCAING
ISSN 2448 9131

Propuesta de método de estimación ABAP

Cristian Camilo Carvajal Parra, Cesar Yesid Barahona Rodríguez

Universidad de Cundinamarca - ingeniería de sistemas

Facatativá

cccarvajal@ucundinamarca.edu.co

cbarahona@ucundinamarca.edu.co

Este documento busca plantear una propuesta de método de estimación para proyectos desarrollados en leguaje de programación ABAP, esto debido a que actualmente los modelos tradicionales están ajustados pensando en funcionalidades y dificultades que se presentan a lenguajes distintos a ABAP, por ende, se da una pequeña explicación de algunos de los modelos de estimación más populares y se presenta una posible conclusión como propuesta a esta necesidad.

Índice de Términos – ABAP, ERP, Estimación, SAP.

I. INTRODUCCIÓN

En este informe, se realizó una actividad de investigación y análisis experimental respecto a un posible modelo de estimación en leguaje de programación ABAP con el objetivo de proponer un método que se ajuste a medida que se obtenga información, esto debido a que actualmente no se encuentra un método de estimación ABAP y las técnicas actuales están enfocadas a otros lenguajes de programación por lo que se infiere mediante la investigación que la mejor forma de estimar en lenguaje de programación ABAP es mediante la combinación de juicio experto y un método algorítmico de aprendizaje.

II. DESARROLLO DE CONTENIDO

La estimación es una de las etapas más importantes en el ciclo de vida de un proyecto de software donde se afecta directamente la calidad del proyecto debido a que se tiene una relación entre tiempo y dinero, generalmente la mayoría de empresas prefiere usar juicio experto para sus estimaciones y dan poco uso a modelos algorítmicos [1].

Para poder realizar una propuesta de un modelo de estimación que sea posible aterrizar al lenguaje de programación ABAP, es necesario tener conceptos previos o la estructura de otros modelos y así poder llegar a una conclusión que esté por un camino similar. Por ende, tendremos en cuenta algunos métodos de estimación conocidos y similares en su estructura:

2.1 Puntos de función:

Para la explicación de método de estimación de puntos de función se toma como referencia principal un artículo de investigación sobre una “Técnica híbrida de estimación basada en el análisis de puntos de función y puntos de casos de uso” [2], además, con el fin de tener mayor compresión del tema se toma en cuenta un instructivo para la cuenta de puntos de función con el fin de aclarar términos [3] y un documento de planificación y gestión de sistemas de información donde se explican fórmulas y términos tenidos en cuenta para la estimación por puntos de función [4].

Los puntos de función se basan en las especificaciones técnicas recogidas en el formato de especificación de requisitos funcionales de un software, este método de estimación tiene una desviación de ±10%, para el análisis por puntos de función se tienen en cuenta dos etapas principales denominadas:

1. Puntos de función sin ajustar

En esta etapa se evalúan las funcionalidades clasificándolas en 5 tipos que se clasifican en rango de complejidad baja, media, alta con una puntuación específica predefinida para cada tipo:

- EI es una entrada externa del sistema o el ingreso de datos por parte del usuario.
- EO es una salida externa del sistema como lo podrían ser reportes o salidas de datos.
- EQ representa las consultas externas del sistema como la búsqueda de datos que realizan los usuarios.
- ILF como archivo lógico interno o un grupo de datos relacionados lógicamente mantenido dentro de la frontera de la aplicación

- ELF como un archivo de interfaz externo el cual es similar a ILF a diferencia que este archivo es tratado por otra aplicación

Ahora bien, con la puntuación específica de los diferentes tipos y con base en la cantidad de elementos que se tengan en cuenta en los requerimientos según su complejidad debe estar dada por un formato similar al siguiente:

Parámetro	Complejidad	Peso	Cantidad	Total = cantidad * peso
Ficheros Lógicos Internos	Alta	15		
	Media	10		
	Baja	7		
Ficheros Lógicos Externos	Alta	10		
	Media	7		
	Baja	5		
Entradas	Alta	6		
	Media	4		
	Baja	3		
Salidas	Alta	7		
	Media	5		
	Baja	4		
Consultas	Alta	6		
	Media	4		
	Baja	3		
			Total	Puntos Función = Suma de Totales

Fig. 1. Puntos sin ajustar [3].

2. Puntos ajustados

En esta etapa se ajustan los valores técnicos de los 14 factores de las características generales del sistema con un valor de 0 a 5 dado por la siguiente escala:

- Factores
- Comunicación de datos
- Procesamiento de datos distribuido
- Rendimiento
- Uso del hardware existente
- Transacciones
- Entrada de datos interactiva
- Eficiencia
- Actualizaciones on-line
- Complejidad de procesamiento
- Reusabilidad
- Facilidad de conversión e instalación
- Facilidad de operación
- Múltiples instalaciones
- Facilidad de mantenimiento

Escala:

Valor:	**0**	**1**	**2**	**3**	**4**	**5**
Significado:	Sin influencia	Incidental	Moderado	Medio	Significativo	Esencial

Fig. 1. Grados de relevancia de las características generales del sistema [4].

i	Factor	Valor F_i 0-5
1	Comunicación de datos	
2	Procesamiento de datos distribuido	
3	Rendimiento	
4	Uso del hardware existente	
5	Transacciones	
6	Entrada de datos interactiva	
7	Eficiencia	
8	Actualizaciones on-line	
9	Complejidad de procesamiento	
10	Reusabilidad	
11	Facilidad de conversión e instalación	
12	Facilidad de operación	
13	Múltiples instalaciones	
14	Facilidad de mantenimiento	

Tabla 1. Propuesta valor de ajuste.

y se presenta la siguiente fórmula la cual busca obtener el cálculo de valor de ajuste (VAF)

$$VAF = 0.65 + 0.01 * \sum_{i=1}^{14} F_i$$

Fórmula 1. Valor de ajuste.

Siendo F_i el valor de cada característica general del sistema cuya visibilidad se ve más clara en la Tabla 1 propuesta valor de ajuste; El número 0.65 que se observa en la ilustración puede variar entre 0.65 (en caso tal que el valor de F_i sea de 0) y 1.35 (en caso tal que el valor de F_i sea de 5).

Una vez obtenido valor de ajuste la forma de obtener el número total de puntos de función ajustados es la multiplicación del cálculo del valor de ajuste por el valor

total de los puntos de función ajustados:

$$PFA = VAF * PFsA$$

Fórmula 2. Puntos de función ajustados.

2.2 Casos de uso:

Para la explicación de método de estimación de casos de uso se toma como referencia principal el artículo de investigación sobre una "Técnica híbrida de estimación basada en el análisis de puntos de función y puntos de casos de uso" [2], además, con el fin de tener mayor compresión del tema se consultan diferentes fuentes que complementan una visión más clara de términos y fórmulas a tener en cuenta para la estimación por puntos de casos de uso, tomando así en cuenta la página WEB el laboratorio de las TI [5] junto con la página WEB de es-academic.com [6] y un documento de la universidad veracruzana de México de estimación de casos de uso [7].

La estimación por casos de uso se basa en el modelado UML de casos de uso de un sistema evaluando funcionalidades y componentes de software a desarrollar, este método no se desvía del más del 30% de la realidad.

1. Evaluación de funcionalidades

En esta fase etapa se obtienen los puntos de casos de uso sin ajustar (UUCP) basándose en el peso de los actores sin ajustar (UAW) que es la cantidad de actores presentes en el sistema con base en una complejidad asignada. Se presenta la siguiente tabla para una mayor visión de la forma de obtener peso de los actores sin ajustar:

i	Tipo de actor	Descripción	Cantidad	Factor	Total = cantidad * factor
1	Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API).			
2	Medio	Otro sistema interactuando a través de un protocolo (ej. TCP/IP) o una persona interactuando a través de una interfaz en modo texto.			
3	Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica (GUI).			

Fig. 3. Peso de actores sin ajustar.

Se plantea la siguiente fórmula con base en la tabla anterior para obtener los puntos de casos de uso sin ajustar (UAW).

$$UAW = \sum_{i=1}^{3} Total_i$$

Fórmula 3. Puntos de caso de casos de uso sin ajustar.

De igual manera, en esta etapa se obtiene el factor de peso de los casos de uso sin ajustar (UUCW) analizando la cantidad de casos de uso presentes en el sistema y la complejidad de los mismos, pueden obtenerse con base en clases o en las transacciones (actividades o flujos de eventos) que tiene un caso de uso con un valor de factor constante. Se presenta la siguiente tabla para mayor visión de la forma de obtener el factor de peso de los casos de uso sin ajustar:

i	Tipo de caso de uso	Descripción	Número de casos de uso	Factor	Total = número de casos de uso * factor
1	Simple	• 3 o menos transacciones. • Menos de 5 clases.		5	
2	Medio	• 4 o 7 transacciones. • De 5 a 10 clases		10	
3	Complejo	• Mas de 10 transacciones. • Mas de 10 clases		15	

Fig. 2. Factor de peso de los casos de uso sin ajustar.

Se plantea la siguiente fórmula para obtener el valor del factor de peso de los casos de uso sin ajustar (UUCW)

$$UUCW = \sum_{i=1}^{3} Total_i$$

Fórmula 5. Factor de peso de los casos de uso sin ajustar.

Ahora bien, con base en los resultados obtenidos del valor peso de los actores sin ajustar (UAW) y del factor de peso de los casos de uso sin ajustar (UUCW) para obtener el cálculo de los puntos de casos de uso sin ajustar, se tiene la siguiente fórmula:

$$UUCP = UAW + UUCW$$

Fórmula 6. Puntos de caso de uso sin ajustar.

2. Factores técnicos de ajuste

De una manera similar a la estimación por puntos de función, en este apartado también se tienen en cuenta unos factores para obtener el cálculo de valor de ajuste con un rango de valoración de peso de 0 a 5 siendo 0 a 2 una influencia irrelevante, 3 a 4 una influencia media y 5 una influencia esencial.

i	Factor	Valor F_i 0-5
1	Sistema Distribuido	
2	Objetivos de rendimiento	

3	Eficiencia respecto al usuario final	
4	Procesamiento complejo	
5	Código reutilizable	
6	Instalación sencilla	
7	Fácil utilización	
8	Portabilidad	
9	Fácil de cambiar	
10	Uso Concurrente	
11	Características de seguridad	
12	Accesible por terceros	
13	Se requiere formación especial	

Tabla 2. Cálculo de valor de ajuste casos de uso.

Con base en los resultados obtenidos se presenta la siguiente fórmula para el cálculo del factor de complejidad técnica (TCF):

$$TCF = 0.6 + 0.01 * \sum_{i=1}^{13} F_i$$

Fórmula 7. Factor de complejidad técnica.

Siendo F_i el valor de cada característica general del sistema cuya visibilidad se ve más clara en la Tabla 4 Cálculo de ajuste casos de uso; siendo 0.6 y 0.01 valores constantes para esta fórmula y F_i la valoración del peso dado para cada factor.

Una vez obtenido valor de ajuste es necesario evaluar los factores ambientales del proyecto que están relacionados con las habilidades y la experiencia de los involucrados en el desarrollo con un rango de peso igual al que se tuvo en cuenta para el factor de complejidad técnica.

Tabla 3. Factores ambientales.

Con base en los resultados obtenidos se presenta la siguiente fórmula para el cálculo del factor de complejidad ambiental (ECF):

$$ECF = 1.4 - 0.03 * \sum_{i=1}^{8} F_i$$

Fórmula 8. Factor de complejidad ambiental.

Siendo F_i el valor de cada factor ambiental, la visibilidad se ve más clara en la Tabla 5 Factores ambientales; siendo 1.4 y -0.03 valores constantes para esta fórmula y F_i la valoración del peso dado para cada factor.

Ahora bien, para obtener el número de casos de uso ajustados (UCP) se logran obtener con la siguiente fórmula:

$$UCP = UUCP * TCF * ECF$$

Fórmula 9. Casos de uso ajustados.

2.3 COCOMMO II:

Para la explicación de método de estimación COCOMO II mediante el modelo de diseño temprano se toma como referencia principal el Libro de ingeniería de software novena edición de Ian Sommerville [8], además, con el fin de tener mayor compresión del tema se consultan diferentes fuentes que complementan una visión más clara de términos y fórmulas a tener en cuenta para la estimación por el método COCOMO II mediante el modelo de diseño temprano, tomando así en cuenta la página WEB un poco de Java [9] junto con el documento de Darko Milic "Applying COCOMO II- A case study" [10], de igual manera se tuvo en cuenta el documento " "COCOMOII" MODELODECONSTRUCCIÓNDECOSTOS" [11] y el documento de administración de proyectos de la universidad veracruzana de México [12], de tal forma que estas referencias permitieron complementar el entendimiento del método de estimación COCOMO II mediante el modelo de diseño temprano que se explica en esta sección.

La estimación con el método COCOMO II tiene una desviación que disminuye entre más exactos sean los datos de entrada, mediante el modelo de diseño temprano para conocer el esfuerzo estimado ($PM_{estimado}$) se da la siguiente fórmula:

i	Factor	Valor F_i 0-5
1	Familiaridad con el modelo del proyecto usado	
2	Experiencia en la aplicación	
3	Experiencia en POO	
4	Capacidad del analista líder	
5	Motivación	
6	Estabilidad de los requerimientos	
7	Personal de media jornada	
8	Dificultad en lenguaje de programación	

$$PM_{estimado} = PM_{nominal} * \prod_{i=1}^{7} EM_i$$

Fórmula 10. Esfuerzo estimado.

Siendo $PM_{estimado}$ el valor del esfuerzo estimado, EM_i los multiplicadores de esfuerzo nominal que se basan en 7 factores de costo:

- fiabilidad y complejidad del producto (RCPX),
- reutilización requerida (RUSE),
- dificultad de plataforma (PDIF),
- habilidad personal (PERS),
- experiencia personal (PREX),
- calendario (SCED)
- facilidades de soporte (FCIL).

EL esfuerzo nominal tiene una escala de estimación de seis puntos donde 1 corresponde a muy bajo y seis a muy alto. Mediante la siguiente tabla se busca dar una visión más detallada de la forma en que se obtiene el esfuerzo nominal:

i	Factor	Valor EM_i 1-6
1	fiabilidad y complejidad del producto (RCPX),	
2	reutilización requerida (RUSE),	
3	dificultad de plataforma (PDIF),	
4	habilidad personal (PERS),	
5	experiencia personal (PREX),	
6	calendario (SCED)	

Tabla 4. Esfuerzo nominal COCOMO II.

Ahora bien, $PM_{nominal}$ está dado por la fórmula

$$PM_{nominal} = A * (KSLOC)^B$$

Fórmula 11. Esfuerzo nominal.

Donde A una constante equivalente a 2.94, $KSLOC$ se puede definir como el tamaño del sistema y se calcula al estimar los puntos de función no ajustados en el software o en tamaño expresado en miles de líneas de código fuente y el exponente B es el tamaño requerido conforme aumenta el tamaño del proyecto cuyo valor varía entre 1.1 a 1.24 dependiendo de diferentes variables que pueden afectar el proyecto otros autores definen la fórmula de $PM_{estimado}$ como:

$$Esfuerzo = A * Tamaño^B * M$$

Fórmula 12. Esfuerzo.

Siendo M se los multiplicadores de esfuerzo ($M = PM_{estimado}$) que se basan en los siete atributos o factores de esfuerzo del proyecto que se enseñaron en la Tabla 6 Esfuerzo nominal COCOMO II.

2.4 Método hibrido de estimación

Primero el resultado del valor sin ajustar del resultado ponderado de la estimación (RPSA) que es la suma de los puntos de función sin ajustar (PFSA) más los puntos de caso de uso sin ajustar (UUCP).

$$RPSA = PFSA + UUCP$$

Fórmula 13. Resultado ponderado de estimación.

Luego el valor del resultado ponderado ajustado (RPA) que es igual a los puntos de función ajustado (PFA) más los puntos de caso de uso (UCP).

$$RPA = PFA + UCP$$

Fórmula 14. Resultado ponderado ajustado.

Con estas dos fórmulas se puede obtener el resultado del porcentaje ponderado (% ponderado) cuya fórmula es igual a la diferencia que hay entre el resultado ponderado ajustado (RPA) y el resultado ponderado sin ajustar de la estimación (RPSA), este resultado se divide entre el resultado ponderado sin ajustar de la estimación (RPSA).

$$RPA = PFA + UCP$$

Fórmula 15. Resultado ponderado ajustado.

Y así se puede obtener los puntos ajustados del hibrido que será la suma entre los puntos sin ajustar del hibrido más la multiplicación del porcentaje ponderado por los puntos sin ajustar del hibrido.

$$Puntos\ ajustados\ del\ hibrido = puntos\ \text{sin}\ ajustar\ del\ hibrido + (puntos\ \text{sin}\ ajustar\ del\ hibrido * \%Ponderado)$$

Fórmula 16. Puntos ajustados del hibrido [2].

Se puede decir que los modelos de estimación se basan en técnicas de regresión donde la hipótesis se obtiene con base en información histórica mediante regresiones [13].

2.5 Regresiones

Una regresión es la relación entre una variable dependiente y por lo menos una variable independiente con el fin de encontrar una relación y así encontrar el valor

futuro de una variable de estudio[14].

Para aplicar un modelo de regresión podemos encontrar que es necesario tener en cuenta "Al momento de evaluar la relación entre una variable que suscita especial interés (variable dependiente que suele denominarse Y) respecto a un conjunto de variables (variables independientes, que se denominan X1 , X2 , …, Xn) las pruebas de contraste de hipótesis mostradas hasta ahora no nos aportan suficiente información sobre la relación en conjunto de todas ellas, dado que los contrastes de hipótesis que conocemos hasta ahora se basan en probar relaciones bivariantes (2 variables), en las que no se tiene en cuenta la posibilidad de que haya otras variables de interés y en las que el sentido de la relación es bidireccional" [15].

Se puede decir que existen dos tipos de regresión que son regresión lineal y la regresión logística

Regresión logística es una técnica multivariante predictiva de regresión. Concretamente, es un modelo que permite asignar a los individuos en una opción de respuesta según los coeficientes estimados para cada una de las variables independientes y la probabilidad de estos en la variable dependiente[16].

Podemos decir que la logística técnica explica un comportamiento, en función de los valores que tomen otras variables, en el modelo de regresión logística binaria la variable dependiente debe tomar exactamente dos valores (Sí-No, 0-1, Verdadero-Falso, etc.). Entonces podemos decir que este modelo debe estar más reducido y debe ser congruente e interpretable [16].

Regresión Múltiple se entiende como regresión lineal cuando se pretende buscar un modelo donde se busque una correcta relación entre la variable dependiente (binaria) y un conjunto de covariables, donde no se necesita ser variables poder llegar a esta relación. Existen diferentes métodos para poder evaluar esta regresión entre los cuales están:

Mínimos cuadrados: consiste en calcular la suma de las distancias al cuadrado entre los puntos reales y los puntos definidos por la recta estimada a partir de las variables introducidas en el modelo, de forma que la mejor estimación será la que minimice estas distancias. Para poder decidir qué modelo es el que mejor se adecua a los datos de los que disponemos en el modelo de regresión lineal se comparan la F parcial obtenida en cada uno de los modelos de regresión construidos [17].

Mínimos cuadrados ponderados: es una extensión del método de mínimos cuadrados donde se busca ampliar la información de un cierto grupo de datos, pero es necesario que los pesos para calcular este método sean exactos y se debe usar preferiblemente cuando las estimaciones de los pesos son muy precisas [18].

Podemos encontrar que cada método tiene un cierto peso que se multiplica por una cierta entrada del sistema, luego de esto hay un cierto ajuste aritmético para estos pesos, por ende, se propone que el ajuste de los pesos sea dado mediante un algoritmo de aprendizaje. Esto con el fin de que un desarrollador no tenga que intervenir el código para mejorar la estimación, sino que el algoritmo tenga la capacidad de aprender para dar una respuesta más aproximada [19]:

Redes monocapa: Son aquellas que se componen de una sola capa de neuronas donde se proyectan entradas y salidas con diferentes cálculos de una forma simple [20].

Redes multicapa: contiene un conjunto de capas de neuronas donde se tiene una capa de entrada y otra de salidas donde se relacionan mediante diferentes capas donde se realizan diferentes cálculos [21].

2.6 Perceptrón

Un perceptrón es la forma abstracta más simple de una neurona y es una de las bases principales del aprendizaje automático [22], este busca basarse en el funcionamiento de una neurona biológica donde se obtienen un cierto número de entradas y un resultado, siendo así una referencia a lo que serían las dendritas y el axón de una neurona siendo esto elemental para una red neuronal artificial, es posible considerar a un perceptrón como una red neuronal con una única capa que contiene valores de entrada, pesos, Bias, suma neta y función de activación [23].

Un perceptrón consiste en una salida que depende de un peso y una entrada con el fin de ajustar el peso y disminuir el error de exactitud de las salidas del perceptrón.

Inicialmente consta de la multiplicación del producto punto de entradas versus pesos

$$Entrada = \sum_{i=1}^{n} (Entradas \cdot Pesos)$$

Fórmula 17. Producto punto entradas.

i	Entrada	Peso	Total
1	$Entrada_i$	$Peso_i$	$Entrada_i * Peso_i$
2	$Entrada_i$	$Peso_i$	$Entrada_i * Peso_i$
n	$Entrada_i$	$Peso_i$	$Entrada_i * Peso_i$
Total			$\sum Entrada_i * Peso_i$

Tabla 5. Producto punto perceptrón.

Posteriormente a obtener el producto punto entre

entradas y pesos es necesario sumar el Bias, valor que permite encontrar la separación entre posibilidades de salida de la red.

$$(\sum Entrada_i * Peso_i) + Bias$$

Fórmula 18. Producto punto más Bias.

Posteriormente el resultado se enfrenta a una validación mediante una función de activación (ReLu, Softmax, tangente hiperbólica, sigmoide) y se calcula el de error de la salida que se obtiene mediante la siguiente fórmula:

$$Error = Error\ obtenido - Error\ esperado$$

Fórmula 19. Error.

Ajuste de pesos:

$$Peso_i = Peso_i + Error * Entrada_i$$

Fórmula 20. Ajuste de Pesos.

Ajuste del Bias

$$Bias = BiasActual + Error$$

Fórmula 21. Bias.

Luego se repite nuevamente el producto punto entre entradas y pesos hasta que el error sea el mínimo posible o sea igual a cero [24].

Funciones de activación

Una función de activación busca transformar una señal de entrada a salida particular [25], de igual manera se puede decir que una función de activación tiene un papel importante en una red neuronal cuyo fin es transmitir información y dependiendo de cada caso se verá qué tan conveniente es usar cada función [26]. Podemos encontrar diferentes tipos de funciones de activación como, por ejemplo:

Función de activación de umbral: principalmente para redes de una sola capa

Función de activación lineal: preferiblemente para múltiples neuronas

Función de activación sigmoidea: permiten entradas concierto rango de 0 a 1 o de -1 a 1

Función de activación avanzadas: funciones útiles para diferentes áreas de aplicación [27]

Dentro de las más comunes encontramos:

Función sigmoidal	**Función tangente hiperbólica**	**Función ReLu**
Rango de salida entre 0 y 1	Rango de salida entre -1 y 1	Mapea cualquier entrada entre 0 y N

$f(x) = \frac{1}{1 - e^{-x}}$	$f(x) = \frac{2}{1 + e^{-2x}} - 1$	$f(x) = max(0, x)$ $f(x) = \begin{cases} 0\ para\ x < 0 \\ x\ para\ x \geq 0 \end{cases}$

Tabla 6. Comparación de funciones de activación [28].

La función de activación ReLU es fácil de calcular, alivia problemas de desaparición del gradiente y formula una propuesta más general con el fin de evitar un rango de salida [29].

III. MÉTODO DE ESTIMACIÓN PROPUESTO

Basando una estructura en los métodos de estimación investigados podemos observar que inicialmente se obtiene una evaluación de funcionalidades, posteriormente un factor de ajuste y finalmente el resultado de la estimación. Con base en esta estructura se plantea la siguiente forma de estimación ABAP:

Actualmente un proyecto ABAP en SAP no cuenta con un estándar de valores de pesos de dificultad para sus funcionalidades motivo por el cual no se puede seguir un método de estimación al pie de la letra y por ende se propone que las variables independientes necesarias para poder realizar la estimación sean brindadas en un inicio mediante el juicio experto del equipo encargado debido a que los pesos de las funcionalidades de los métodos de estimación elegidos están pensados para otros lenguajes de programación como se menciona en la investigación. Para poder obtener las variables independientes de la estimación se propone establecer un rango de cinco dificultades:

- Muy fácil (MF)
- Fácil (F)
- Medio (M)
- Difícil (D)
- Muy difícil (MD)

Ingreso de variables independientes y estimación sin ajustar

Las dificultades planteadas evaluarán funcionalidades u objetos de desarrollo del proyecto (OD) de tal forma que inicialmente se obtendrán como variables establecidas por juicio experto los pesos de dificultad (PD) de cada OD tenido en cuenta para el proyecto:

Objeto de desarrollo	Peso dificultad muy fácil	Peso dificultad fácil	Peso dificultad medio	Peso dificultad difícil	Peso dificultad muy difícil
OD1	PDMF	PDF	PDM	PDD	PDMD

Tabla 7. Ingreso de los pesos de las dificultades.

De igual manera es necesario obtener el número de objetos (NO) estimados que se desarrollarán para un OD categorizándolos con base en las dificultades planteadas previamente:

Objeto de desarrollo	Número de objetos dificultad muy fácil	Número de objetos dificultad fácil	Número de objetos dificultad medio	Número de objetos dificultad difícil	Número de objetos dificultad muy difícil
OD1	NODMF	NODF	NODM	NODD	NODMD

Tabla 8. Ingreso del número de objetos para cada OD y Dificultad.

Para obtener el valor de la evaluación de las funcionalidades sin ajustar podemos interpretar cada objeto de desarrollo (OD) como un vector y podemos realizar la sumatoria de todos los escalares obtenidos con el producto punto entre el peso de las dificultades (PD) y el número de objetos de desarrollo (NO), hay que tener en cuenta que las funcionalidades sin ajustar tendrán en cuenta solo lo que abarca la programación ABAP:

$$PD = < \text{PDMF} + \text{PDF} + \text{PDM} + \text{PDD} + \text{PDMD} >$$

Fórmula 22. Producto dificultades.

$$NO = < \text{NODMF} + \text{NODF} + \text{NODM} + \text{NODD} + \text{NODMD} >$$

Fórmula 22. Número de objetos de desarrollo.

$$Funcionalidades\ sin\ ajustar\ = \sum_{i=1}^{n} (PD\ \cdot\ NO)$$

Fórmula 23. Funcionalidades sin ajustar.

Ajuste del método

Para poder realizar un ajuste del modelo de estimación es necesario obtener las horas reales obtenidas en cada objeto de desarrollo del proyecto con el fin que cada peso de cada objeto de desarrollo sea ajustado mediante un perceptrón:

Objeto de desarrollo	Número de horas estimadas (NHE)	Número de horas reales (NHR)
OD1	NHE1	NHR1
OD2	NHE2	NHR2

Tabla 9. Ingreso de horas reales obtenidas.

Posteriormente se podrá realizar la lógica de un perceptrón para el ajuste correspondiente del método ya que se tienen todos los aspectos necesarios para realizarlo como lo son:

- Entradas
- Pesos
- Bias
- Función de activación ReLu
- Resultado esperado o número de horas reales
- Resultado obtenido o número de horas estimadas

Se puede observar que con base en los métodos de estimación utilizados se tiene una estructura similar, evaluando pesos de funcionalidades por número de veces que se ven en el proyecto, pero estos son pesos ajustados específicamente para otros lenguajes que no pertenecen a ABAP por lo que asimilar la estructura que se maneja con la de un perceptrón permite hacer que el modelo tenga un factor de ajuste que genere cada vez más valor por cada uso que se le dé, motivo por el cual se determina establecer la duración real de cada objeto de desarrollo de la estimación y con base en esto ajustar pesos gracias a la ayuda de un perceptrón de tal forma que el último paso a realizar de este artículo es la comprobación del modelo.

IV. RESULTADOS

Se tiene en cuenta el modelado aritmético propuesto para la estimación ABAP y se hacen diferentes pruebas con la estimación de diferentes objetos de desarrollo de los cuales se observan los siguientes resultados con el fin de comprobar la efectividad del método:

Para el diseño de los casos de prueba se ha buscado realizarlos de la manera mas simple posible mediante la herramienta de Excel generando valores al azar dentro de ciertos rangos establecidos donde se han propuesto 25 proyectos de los cuales 17 proyectos fueron usados para realizar el entrenamiento del perceptrón de ajuste del método y 8 se han tomado para comprobar el funcionamiento del ajuste del método.

Cada proyecto cuenta con 5 tipos de objetos de desarrollo para el lenguaje ABAP apuntando cada uno a una dificultad en específica que cuenta con un peso brindado con juicio experto:

Pesos

OBJETO DE DESARROLLO	DIFICULTAD	PESO INICIAL	PESO AJUSTADO
Interfaces de entrada (Usualmente usando BDCs y IDOCS)	MUY FACIL	7	1.12952885950356
Interfaces de salida (Usualmente usando SQL)	FACIL	10	8.66940982223453
Reportes Interactivos	MEDIO	20	18.335539053799970
Reportes ALV	DIFICIL	41	24.397328036395523
Reportes de Lista	MUY DIFICIL	41	24.867976715648887

Tabla 30. Pesos juicio experto - pesos ajustados.

Ahora bien, cada proyecto tiene un número de objetos de desarrollo al azar de los cuales se obtuvo el resultado y a ese resultado se le adicionó un margen de error al azar entre -0.4% a 0.4% obteniendo así el resultado real para cada objeto de desarrollo de los 25 proyectos.

V. GRÁFICAS

Se indica mediante una línea morada que posterior a esta los proyectos no fueron tomados en cuenta para el entrenamiento del perceptrón.

Objeto de desarrollo Interfaces de entrada (Usualmente usando BDCs y IDOCS)

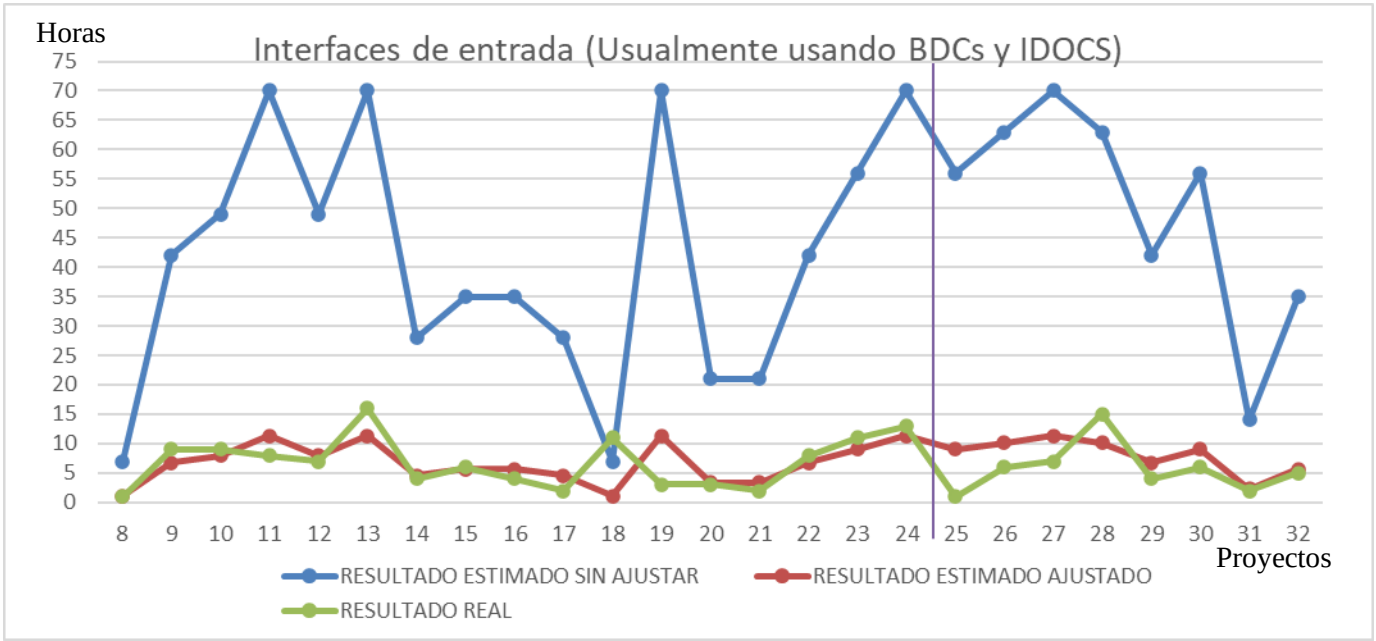

Fig. 4. Estimación de objeto de desarrollo interfaces de entrada.

Objeto de desarrollo Interfaces de salida (Usualmente usando SQL)

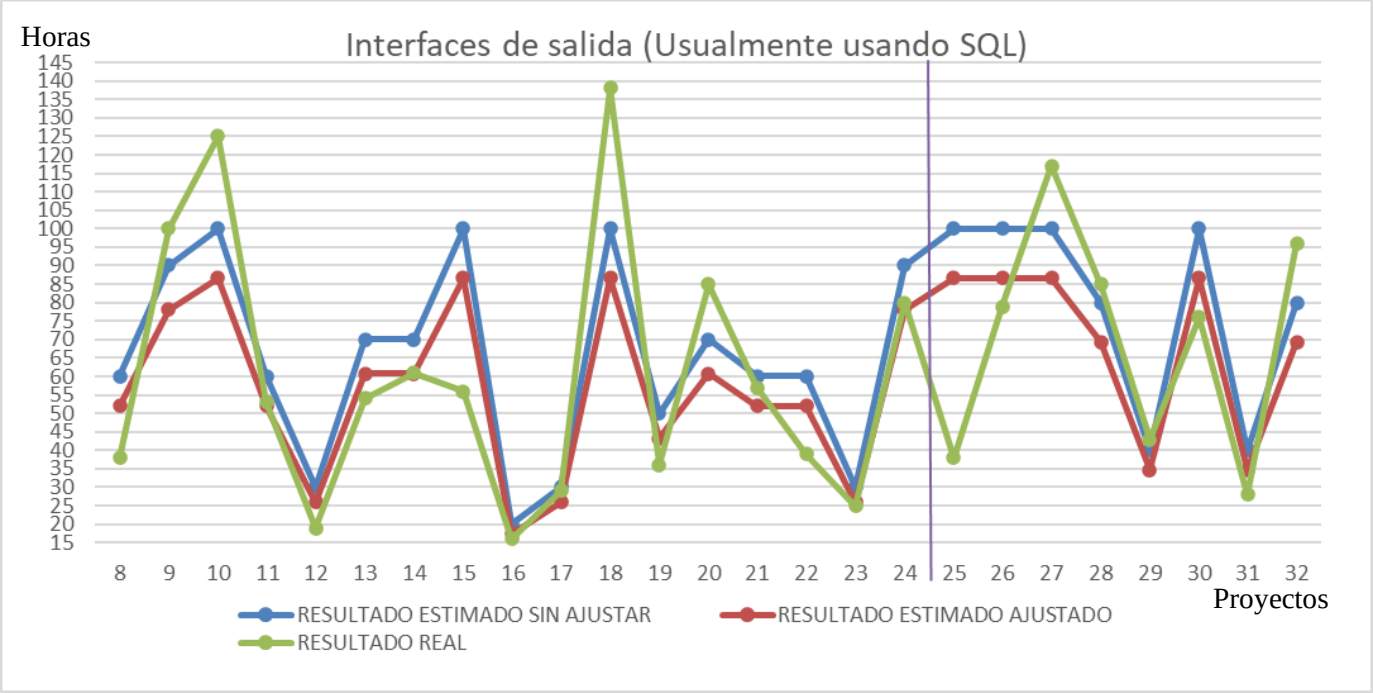

Fig. 5. Estimación de objeto de desarrollo interfaces de salida.

Objeto de desarrollo Reportes ALV

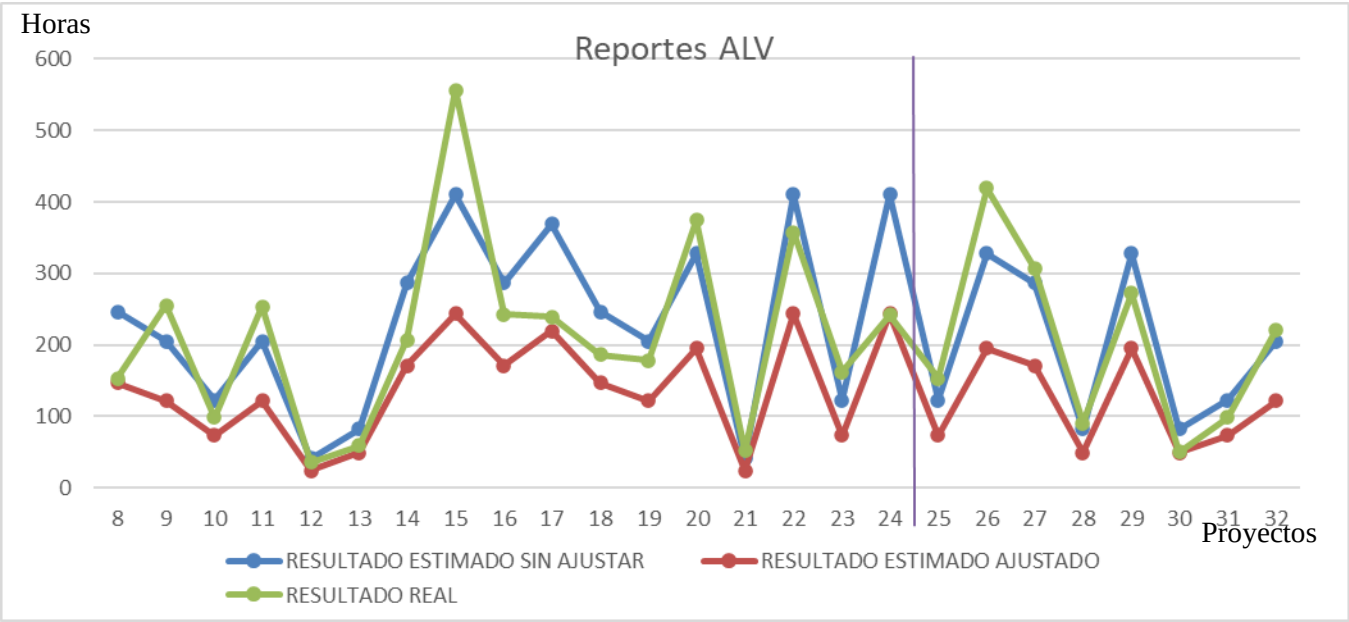

Fig. 6. Estimación de objeto de desarrollo reportes ALV.

Objeto de desarrollo Reportes de Lista

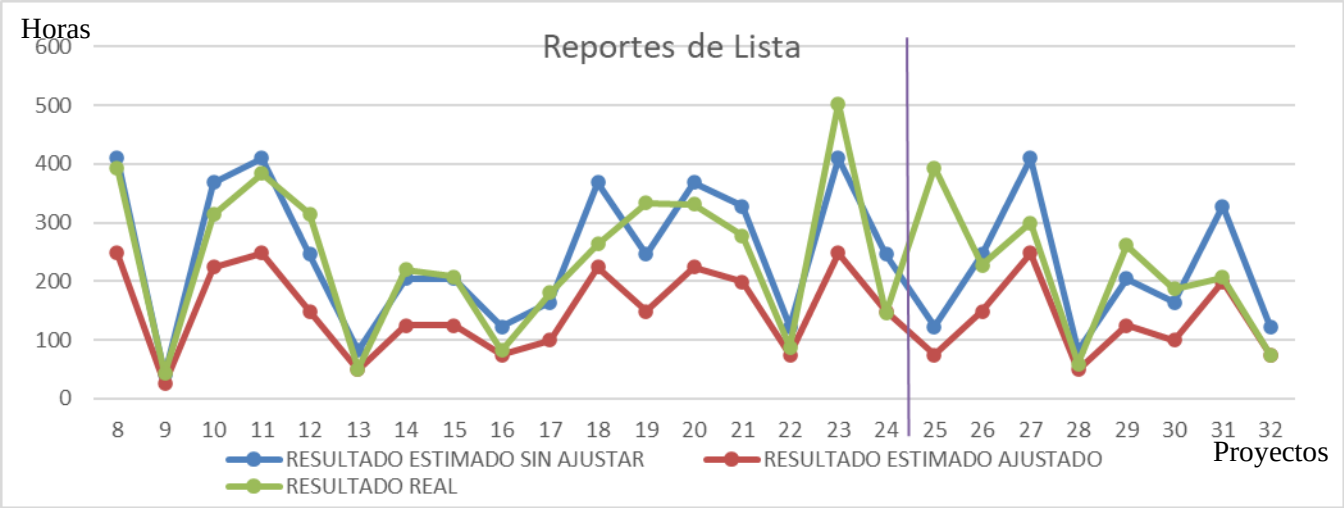

Fig. 7. Estimación de objeto de desarrollo reportes de lista.

Objeto de desarrollo Reportes Interactivos

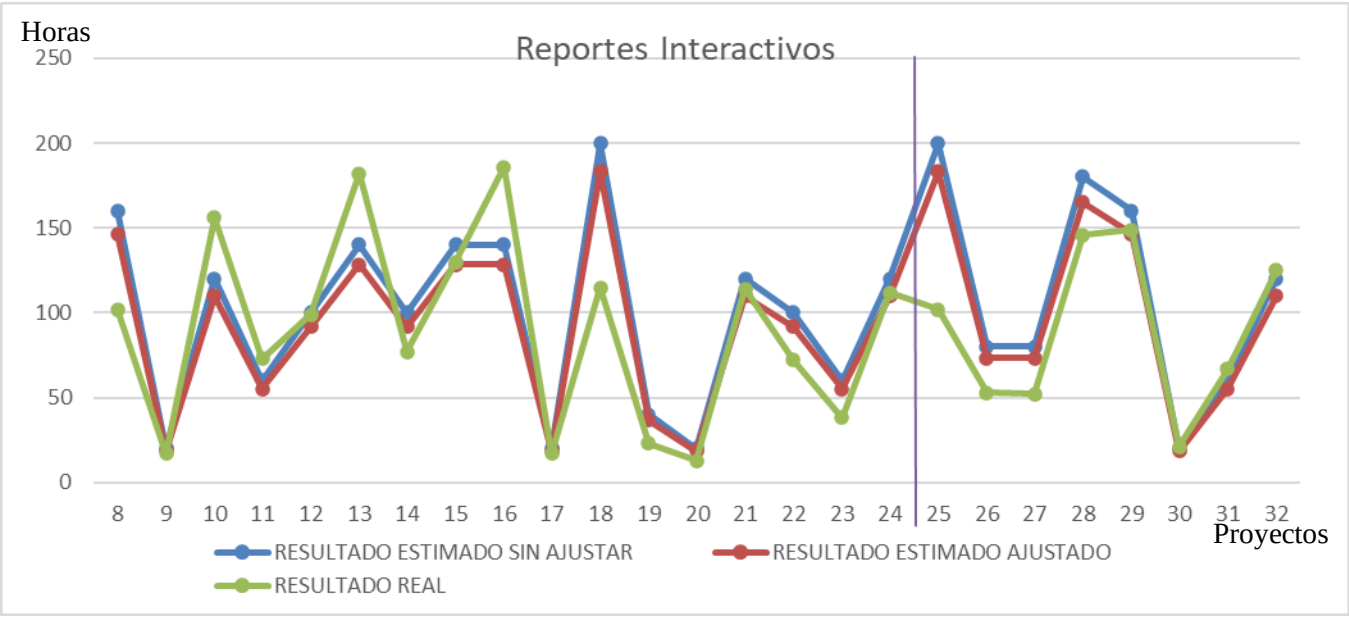

Fig. 8. Estimación de objeto de desarrollo reportes interactivos.

VI. CONCLUSIONES

Se puede observar que el método de estimación brinda un resultado que busca acercarse a los resultados reales. El perceptrón permite tener una retroalimentación de estimaciones previas con el fin de ser más exacto en estimaciones futuras para cada objeto de desarrollo; Como se observa en los resultados del ajuste a un solo objeto de desarrollo la estimación por juicio experto puede tener un margen de error pero al aplicar el uso del perceptrón los resultados de estimación tienden a marcar una diferencia demostrando que el perceptrón está buscando una línea de tendencia en los datos siendo así un método que integra juicio experto y un algoritmo de aprendizaje para mejorar la exactitud de la estimación del desarrollo de programas ABAP.

VII. REFERENCIAS

[1] A. B. Lencina, Y. Medina, and G. Dapozo, “Aplicación para estimar costos en proyectos de software,” 2016. [Online]. Available: http://www.corrientes.gov.ar/

[2] D. S. ARIAS ROJAS and P. A. CHIA RODRIGUEZ, “Técnica híbrida de estimación basada en el análisis de puntos de función y puntos de casos de uso,” 2017.

[3] Universidad de la república de Uruguay, “Instructivo para la Cuenta de Puntos Función,” 2004.

[4] F. Sánchez Rodríguez, “Planificación y Gestión de Sistemas de Información,” 1999.

[5] El laboratorio de las TI, “Método de Estimación Puntos Casos de Uso (Use Case Points),” 2013. https://www.laboratorioti.com/2013/02/14/metodo-de-estimacion-puntos-casos-de-uso-use-case-points/ (accessed May 14, 2022).

[6] es-academic.com, “Puntos de caso de uso,” 2021. https://es-academic.com/dic.nsf/eswiki/973053 (accessed May 14, 2022).

[7] R. K. Clemmons, “PUNTOS DE CASOS DE USO,” 2006.

[8] I. Sommerville, *Software engineering*. Pearson, 2011.

[9] Un poco de Java, “Modelos de estimación: un poco sobre COCOMO II,” 2012. https://unpocodejava.com/2012/02/07/modelos-de-estimacion-un-poco-sobre-cocomo-ii/ (accessed May 14, 2022).

[10] D. Milicic, “Applying COCOMO II - A case study,” 2004. [Online]. Available: www.bth.se/tek

[11] E. E. Cuevas Aparicio, E. D. Diaz Jaimes, L. Hernandez Solorio, and O. Reyes Castellanos, “‘COCOMO II’ MODELO DE CONSTRUCCIÓN DE COSTOS ADMINISTRACIÓN DE PROYECTOS DE SOFTWARE,” 2009.

[12] Universidad Veracruzana, “ADMINISTRACIÓN DE PROYECTOS,” 2017.

[13] M. N. Moreno García and F. J. García Peñalvo, “Modelos de estimación del software basados en técnicas de aprendizaje automático,” 2014.

[14] J. Sánchez Galán, “Análisis de regresión,” Jul. 01, 2020. https://economipedia.com/definiciones/analisis-de-regresion.html (accessed Apr. 08, 2023).

[15] I. M. Peláez, “Modelos de regresión: lineal simple y regresión logística,” 2014.

[16] V. Berlanga-Silvente and R. Vilà-Baños, “Cómo obtener un Modelo de Regresión Logística Binaria con SPSS,” *REIRE. Revista d’Innovació i Recerca en Educació*, no. 8(2), 2014, doi: 10.1344/reire2014.7.2727.

[17] I. M. Peláez, “Modelos de regresión: lineal simple y regresión logística,” 2014.

[18] luis Benites, “Mínimos cuadrados ponderados: definición simple, ventajas y desventajas,” Feb. 20, 2022. https://statologos.com/minimos-cuadrados-ponderados/ (accessed Apr. 08, 2023).

[19] L. Judith Sandoval, “ALGORITMOS DE APRENDIZAJE AUTOMÁTICO PARA ANÁLISIS Y PREDICCIÓN DE DATOS,” 2018.

[20] L. Angel, A. Marica, E. Wanser Herrera Villa, J. Mejía Huayhua, and L. Q. Flores, “Clasificador de estrellas de Neutrones con una red neuronal multicapa utilizando R,” *Revista Innovación y Software*, vol. 2, no. 1, pp. 35–36, 2021.

[21] A. Nacelle, “Redes neuronales artificiales,” 2009.

[22] İ. K. Bozkır *et al.*, “Classiication of a Two-Class ECG Dataset Based on Perceptron Learning in a Cortical Pyramidal Neuron Model Classification of a Two-Class ECG Dataset Based on Perceptron Learning in a Cortical Pyramidal Neuron Model,” pp. 1–2, 2022, doi: 10.21203/rs.3.rs-2149891/v1.

[23] M. Banoula, “What is Perceptron: A Beginners Guide for Perceptron,” *Dec 9, 2022*, 2022. https://www.simplilearn.com/tutorials/deep-learning-tutorial/perceptron (accessed Jan. 02, 2023).

[24] D. Espitia, “Entrenamiento del perceptron,” 2017. https://platzi.com/contributions/entrenamiento-del-perceptron/ (accessed Jan. 02, 2023).

[25] S. Sharma, S. Sharma, and A. Athaiya, “ACTIVATION FUNCTIONS IN NEURAL NETWORKS,” 2020. [Online]. Available: http://www.ijeast.com

[26] L. Vanessa Sosa Jerez Laura Camila Zamora Alvarado Dirigido por and L. D. Alejandro Másmela Caita Bogotá Julio de, “‘ESTRUCTURA DE REDES NEURONALES (MLP) Y SU APLICACIÓN COMO APROXIMADOR UNIVERSAL,’” 2022.

[27] D. B. Mehta, P. A. Barot, and S. G. Langhnoja, “Effect of Different Activation Functions on EEG Signal Classification based on Neural Networks,” in *Proceedings of the 4th International Conference on Computing Methodologies and Communication, ICCMC 2020*, Institute of Electrical and Electronics Engineers Inc., Mar. 2020, pp. 132–135. doi: 10.1109/ICCMC48092.2020.ICCMC-00027.

[28] D. Calvo, “Función de activación – Redes neuronales,” 2018. https://www.diegocalvo.es/funcion-de-activacion-redes-neuronales/ (accessed Jan. 03, 2023).

[29] A. Lomuscio and L. Maganti, “An approach to reachability analysis for feed-forward ReLU neural networks,” pp. 1–2, Jun. 2017, [Online]. Available: http://arxiv.org/abs/1706.07351